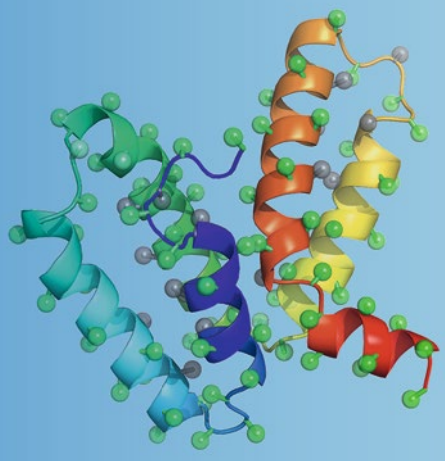
Methods in Molecular Biology 1526

Springer Protocols



Jonathan M. Keith *Editor*

# Bioinformatics

## Volume II: Structure, Function, and Applications

*Second Edition*

EXTRAS ONLINE

Humana Press

# METHODS IN MOLECULAR BIOLOGY

*Series Editor*
**John M. Walker**
**School of Life and Medical Sciences**
**University of Hertfordshire**
**Hatfield, Hertfordshire, AL10 9AB, UK**

For further volumes:
http://www.springer.com/series/7651

# Bioinformatics

## Volume II: Structure, Function, and Applications

### Second Edition

Edited by

**Jonathan M. Keith**

*Monash University*
*Melbourne, VIC, Australia*

Humana Press

*Editor*
Jonathan M. Keith
Monash University
Melbourne, VIC, Australia

ISSN 1064-3745 ISSN 1940-6029 (electronic)
Methods in Molecular Biology
ISBN 978-1-4939-6611-0 ISBN 978-1-4939-6613-4 (eBook)
DOI 10.1007/978-1-4939-6613-4

Library of Congress Control Number: 2016955824



Printed on acid-free paper

This Humana Press imprint is published by Springer Nature
The registered company is Springer Science+Business Media LLC
The registered company address is: 233 Spring Street, New York, NY 10013, U.S.A.

# Preface

Bioinformatics sits at the intersection of four major scientific disciplines: biology, mathematics, statistics, and computer science. That's a very busy intersection, and many volumes would be required to provide a comprehensive review of the state-of-the-art methodologies used in bioinformatics today. That is not what this concise two-volume work of contributed chapters attempts to do; rather, it provides a broad sampling of some of the most useful and interesting current methods in this rapidly developing and expanding field.

As with other volumes in Methods in Molecular Biology, the focus is on providing practical guidance for implementing methods, using the kinds of tricks and tips that are rarely documented in textbooks or journal articles, but are nevertheless widely known and used by practitioners, and important for getting the most out of a method. The sharing of such expertise within the community of bioinformatics users and developers is an important part of the growth and maturation of the subject. These volumes are therefore aimed principally at graduate students, early career researchers, and others who are in the process of integrating new bioinformatics methods into their research.

Much has happened in bioinformatics since the first edition of this work appeared in 2008, yet much of the methodology and practical advice contained in that edition remains useful and current. This second edition therefore aims to complement, rather than supersede, the first. Some of the chapters are revised and expanded versions of chapters from the first edition, but most are entirely new, and all are intended to focus on more recent developments.

Volume 1 is comprised of three parts: Data and Databases; Sequence Analysis; and Phylogenetics and Evolution. The first part looks at bioinformatics methodologies of crucial importance in the generation of sequence and structural data, and its organization into conceptual categories and databases to facilitate further analyses. The Sequence Analysis part describes some of the fundamental methodologies for processing the sequences of biological molecules: techniques that are used in almost every pipeline of bioinformatics analysis, particularly in the preliminary stages of such pipelines. Phylogenetics and Evolution deals with methodologies that compare biological sequences for the purpose of understanding how they evolved. This is a fundamental and interesting endeavor in its own right but is also a crucial step towards understanding the functions of biological molecules and the nature of their interactions, since those functions and interactions are essentially products of their history.

Volume 2 is also comprised of three parts: Structure, Function, Pathways and Networks; Applications; and Computational Methods. The first of these parts looks at methodologies for understanding biological molecules as systems of interacting elements. This is a core task of bioinformatics and is the aspect of the field that attempts to bridge the vast gap between genotype and phenotype. The Applications part can only hope to cover a small number of the numerous applications of bioinformatics. It includes chapters on the analysis of

genome-wide association data, computational diagnostics, and drug discovery. The final part describes four broadly applicable computational methods, the scope of which far exceeds that of bioinformatics, but which have nevertheless been crucial to this field. These are modeling and inference, clustering, parameterized algorithmics, and visualization.

*Melbourne, VIC, Australia* *Jonathan M. Keith*

# Contents

# Contributors


Gökmen Altay • *Biomedical Engineering, Bahcesehir University, Istanbul, Turkey*

A.K.M. Azad • *School of Mathematical Sciences, Monash University, Clayton, VIC, Australia*

Jürgen Bajorath • *Department of Life Science Informatics, B-IT, LIMES Program Unit, Chemical Biology and Medicinal Chemistry, Rheinische Friedrich-Wilhelms-Universität, Bonn, Germany*

Richard W. Bean • *Department of Health, The University of Queensland, Brisbane, QLD, Australia*

Jan M. Czarnecki • *Institute of Structural and Molecular Biology, Department of Biological Sciences, Birkbeck, University of London, Malet Street, Bloomsbury, London, WC1E 7HX, UK*

Tobias Czauderna • *Faculty of Information Technology, Monash University, Clayton, VIC, Australia*

David L. Duffy • *Genetic Epidemiology Laboratory, QIMR Berghofer Medical Research Institute, Brisbane, QLD, Australia*

Richard D. Emes • *School of Veterinary Medicine and Science, University of Nottingham, Leicestershire, UK; Advanced Data Analysis Centre, University of Nottingham, Leicestershire, UK*

Julia C. Engelmann • *Institute of Functional Genomics, University of Regensburg, Regensburg, Germany*

Tom C. Giles • *School of Veterinary Medicine and Science, University of Nottingham, Leicestershire, UK; Advanced Data Analysis Centre, University of Nottingham, Leicestershire, UK*

Wolfram Gronwald • *Institute of Functional Genomics, University of Regensburg, Regensburg, Germany*

Thomas Huber • *Research School of Chemistry, Australian National University, Canberra, Australia*

Falk Hüffner • *Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany*

Mario Inostroza-Ponta • *Departamento de Ingeniería Informática, Facultad de Ingeniería, Universidad de Santiago de Chile, Santiago, Chile*

Christian Komusiewicz • *Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany*

Insuk Lee • *Department of Biotechnology, College of Life Science and Biotechnology, Yonsei University, Seoul, South Korea*

William D. Lees • *Department of Biological Sciences and Institute of Structural and Molecular Biology, Birkbeck, University of London, London, UK*

Claudio Lottaz • *Institute of Functional Genomics, University of Regensburg, Regensburg, Germany*

John Marsden • *Centre for Bioinformatics, Biomarker Discovery and Information-Based Medicine (CIBM), Faculty of Engineering and Built Environment, The University of Newcastle, Callaghan, NSW, Australia*

Luke Mathieson • *Centre for Bioinformatics, Biomarker Discovery and Information-Based Medicine (CIBM), Faculty of Engineering and Built Environment, The University of Newcastle, Callaghan, NSW, Australia*
John S. Mattick • *RNA Biology and Plasticity Laboratory, Garvan Institute of Medical Research, Darlinghurst, NSW, Australia; St-Vincent's Clinical School, Faculty of Medicine, UNSW Australia, Sydney, NSW, Australia*
Geoffrey J. McLachlan • *Department of Mathematics, The University of Queensland, Brisbane, QLD, Australia*
Allan F. McRae • *Centre for Neurogenetics and Statistical Genomics, Queensland Brain Institute, The University of Queensland, St. Lucia, QLD, Australia*
Rebekah McWhirter • *Menzies Institute for Medical Research, University of Tasmania, Hobart, TAS, Australia*
Alexandre Mendes • *Centre for Bioinformatics, Biomarker Discovery and Information-Based Medicine (CIBM), Faculty of Engineering and Built Environment, The University of Newcastle, Callaghan, NSW, Australia*
Onur Mendi • *Software Engineering, Bahcesehir University, Istanbul, Turkey; Faculty of Medicine, Istanbul Bilim University, Istanbul, Turkey*
Gabriel Moreno-Hagelsieb • *Department of Biology, Wilfrid Laurier University, Waterloo, ON, Canada*
Pablo Moscato • *Centre for Bioinformatics, Biomarker Discovery and Information-Based Medicine (CIBM), Faculty of Engineering and Built Environment, The University of Newcastle, Callaghan, NSW, Australia*
Shu-Kay Ng • *School of Medicine, Griffith Health Institute, Griffith University, Brisbane, QLD, Australia*
Rolf Niedermeier • *Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany*
Gottfried Otting • *Research School of Chemistry, Australian National University, Canberra, Australia*
Kala Bharath Pilla • *Research School of Chemistry, Australian National University, Canberra, Australia*
Jeffrey Pond • *Centre for Bioinformatics, Biomarker Discovery and Information-Based Medicine (CIBM), Faculty of Engineering and Built Environment, The University of Newcastle, Callaghan, NSW, Australia*
Falk Schreiber • *Faculty of Information Technology, Monash University, Clayton, VIC, Australia; Department of Computer and Information Science, University of Konstanz, Konstanz, Germany*
Adrian J. Shepherd • *Department of Biological Sciences and Institute of Structural and Molecular Biology, Birkbeck, University of London, London, UK*
Junha Shin • *Department of Biotechnology, College of Life Science and Biotechnology, Yonsei University, Seoul, South Korea*
Martin A. Smith • *RNA Biology and Plasticity Laboratory, Garvan Institute of Medical Research, Darlinghurst, NSW, Australia; St-Vincent's Clinical School, Faculty of Medicine, UNSW Australia, Sydney, NSW, Australia*
Jiangning Song • *Centre for Research in Intelligent Systems, Faculty of Information Technology, Monash University, Clayton, VIC, Australia; Department of Biochemistry and Molecular Biology, Faculty of Medicine, Monash University, Clayton, VIC, Australia*
Rainer Spang • *Institute of Functional Genomics, University of Regensburg, Regensburg, Germany*

Russell Thomson • *Centre for Research in Mathematics, School of Computing, Engineering and Mathematics, Western Sydney University, Parramatta, Australia*
Tianhai Tian • *School of Mathematical Sciences, Faculty of Science, Monash University, Clayton, VIC, Australia*
Chloe Warren • *Centre for Bioinformatics, Biomarker Discovery and Information-Based Medicine (CIBM), Faculty of Engineering and Built Environment, The University of Newcastle, Callaghan, NSW, Australia*
Sebastian Wernicke • *Seven Bridges Genomics, Cambridge, MA, USA*

# Part I

## Structure, Function, Pathways and Networks

# Chapter 1

# 3D Computational Modeling of Proteins Using Sparse Paramagnetic NMR Data

**Kala Bharath Pilla, Gottfried Otting, and Thomas Huber**

## Abstract

Computational modeling of proteins using evolutionary or de novo approaches offers rapid structural characterization, but often suffers from low success rates in generating high quality models comparable to the accuracy of structures observed in X-ray crystallography or nuclear magnetic resonance (NMR) spectroscopy. A computational/experimental hybrid approach incorporating sparse experimental restraints in computational modeling algorithms drastically improves reliability and accuracy of 3D models. This chapter discusses the use of structural information obtained from various paramagnetic NMR measurements and demonstrates computational algorithms implementing pseudocontact shifts as restraints to determine the structure of proteins at atomic resolution.

**Key words** Pseudocontact shifts, PCS, Paramagnetic NMR, Rosetta, GPS-Rosetta, Sparse restraints, 3D structure determination

## 1 Introduction

Nuclear magnetic resonance (NMR) spectroscopy has for decades facilitated structure determination in solution or solid-state. NMR exploits the nuclear spin properties in strong constant magnetic fields. The nuclear spins are manipulated by radiofrequency pulses and their free induction decay is recorded. These are then Fourier transformed to produce a frequency spectrum of the NMR experiment. Two spins that are close in space have a direct magnetic interaction between them, referred as dipole–dipole coupling. When these two spins are aligned, the interaction energy becomes minimal resulting in nuclear Overhauser effect (NOE). Intermolecular and intramolecular NOEs are observed for spins that are typically separated by 3–6 Å. By resolving a dense network of NOEs [1], the 3D structures of proteins and nucleic acids can be determined. This conventional method is relied upon in structure determination of a large number of proteins; however, assigning spin resonances of all spins in the system typically requires various 3D or

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_1, 

4D NMR experiments to be applied. In addition, with increasing molecular weight of proteins, they tend to produce poor spectra and determining 3D structures becomes increasingly difficult.

As an alternative to short range restraints using NOEs, paramagnetic NMR generates versatile structural restraints. Proteins carrying paramagnetic metal ions induce significant effects in NMR experiments. These effects arise from the unpaired electrons of the paramagnetic metals, as electrons have a magnetic moment that is three orders of magnitude larger than that of a proton. Metalloproteins, which make up to 25 % of proteins in any organism's proteome [2], offer natural metal centers that potentially can be directly exploited in paramagnetic NMR experiments. Further, $Mn^{2+}$, $Fe2^{+}$, $Cu^{2+}$, and $Co^{2+}$ are naturally paramagnetic and found in native biological samples.

Lanthanide ions are highly useful for paramagnetic NMR experiments, as their paramagnetism varies greatly while their physicochemical properties are highly similar. This makes it possible for different lanthanides to be used interchangeably in different NMR experiments [3]. Proteins that lack a natural metal center can be engineered to carry lanthanides. Figure 1 illustrates different ways to introduce metal ions into proteins. Small peptides, containing 12–18 residues, are designed to bind lanthanide ion to their side chain atoms and these peptides are attached to either a thiol-reactive cysteine or at an N- or C-terminus of a protein [4]. The most popular means of attaching lanthanide ions is through metal chelating chemical tags. These chemical tags are site specifically attached either through cysteine ligation or more recently using unnatural amino acids which can be reacted via bio-orthogonal click chemistry [5]. Several reviews [6–9] provide a comprehensive overview of the chemistries to functionalise proteins with lanthanide tags.

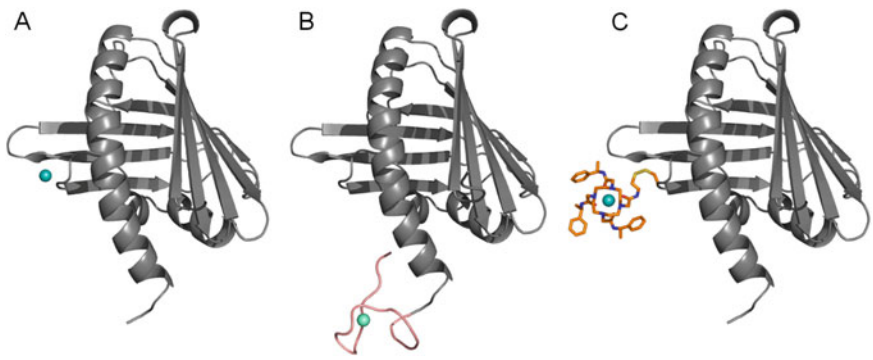



**Fig. 1** Illustration of various modes to introduce metal ions into proteins. (**a**) Replacing a native metal with paramagnetic lanthanide ion in metalloproteins. (**b**) Lanthanide binding peptides attached at C-terminus of a protein. (**c**) Lanthanide carrying chemical tag site specifically attached to a cysteine

### 1.1 Paramagnetic Effects in NMR

The unpaired electrons in a paramagnetic metal ion strongly interact with nuclear spins and the NMR spectrum changes due to induced paramagnetic effects. These paramagnetic effects are quantified by comparing with a diamagnetic (reference) spectra and then translated into structural restraints. The resulting structural restraints can be either distance dependent or orientation dependent or both. One can measure four distinct paramagnetic observables from NMR experiments, namely:

#### 1.1.1 Pseudocontact Shift (PCS)

PCS is a contribution to the chemical shift experienced by a spin caused by the presence of centers of unpaired electrons. PCS of a nucleus influenced by a paramagnetic center can be calculated from a $\Delta\chi$-tensor, shown in Fig. 2a, given by:

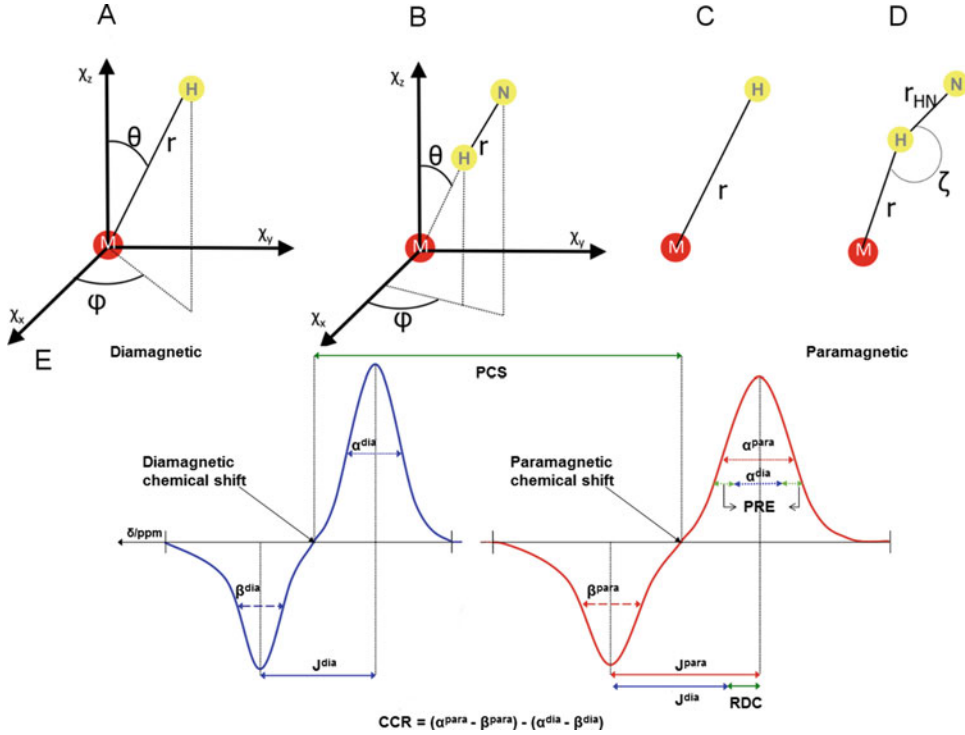



**Fig. 2** The four distinct paramagnetic effects represented geometrically. (**a**) The pseudocontact shift (PCS) between metal center (M) and amide hydrogen (H). (**b**) The residual dipolar coupling (RDC) between two spins H and N. (**c**) The Paramagnetic relaxation enhancement (PRE) between m and H. (**d**) The cross correlation between Curie spin and dipole–dipole relaxation (CCR) between m and H. (**e**) Measurement of the four different paramagnetic effects, illustrated with two 1D undecoupled spectra, showing the diamagnetic and paramagnetic antiphase doublets. PCS is measured as the change in chemical shift between paramagnetic and diamagnetic states. RDC is measured as the difference in line splitting. PRE and CCR can be determined from the differential line broadening. Adapted from Schmitz (2009) [49]

$$\mathrm{PCS}_i^{\mathrm{calc}} = \frac{1}{12\pi r_{\mathrm{MH}}^3}\left[\Delta\chi_{\mathrm{ax}}\left(3\cos^2\theta_{\mathrm{MH}} - 1\right) + \frac{3}{2}\Delta\chi_{\mathrm{rh}}\sin^2\theta_{\mathrm{MH}}\cos 2\varphi_{\mathrm{MH}}\right] \tag{1}$$

where, $r$, $\theta$, $\varphi$ define the polar coordinates of the nuclear spin with respect to principal axis of the $\Delta\chi$-tensor (centered on the paramagnetic ion) and $\Delta\chi_{\mathrm{ax}}$, $\Delta\chi_{\mathrm{rh}}$ define the axial and rhombic component of the magnetic susceptibility tensor $\chi$ and $\Delta\chi$-tensor is defined as $\chi$-tensor minus its isotropic component [10]. PCS is measured as change in the chemical shift of a spin's paramagnetic and diamagnetic states, illustrated in Fig. 2e.

#### 1.1.2 Residual Dipolar Coupling (RDC)

Presence of paramagnetic metal weakly aligns the protein to an external magnetic field resulting in observable RDCs, which are manifested as an increase or decrease in magnitude of multiplet of splits that can be observed in undecoupled spectra, illustrated in Fig. 2e. The RDC is given by Eq. (2) shown in Fig. 2b:

$$D_{\mathrm{NH}} = -\frac{B_0^2}{15kT}\cdot\frac{\gamma_{\mathrm{H}}\gamma_{\mathrm{N}}\hbar}{8\pi^2 r_{\mathrm{NH}}^3} \left[\Delta\chi_{\mathrm{ax}}(3\cos^2\theta_{\mathrm{NH}} - 1) + \frac{3}{2},\Delta\chi_{\mathrm{rh}}\sin^2\theta_{\mathrm{NH}},\cos 2\varphi_{\mathrm{NH}}\right] \tag{2}$$

where $B_0$ is the magnetic field strength, $\gamma_{\mathrm{H}}$ and $\gamma_{\mathrm{N}}$ are the gyromagnetic ratios of the proton and nitrogen spin, $\hbar = h/2\pi$ with $h$ being Planck's constant, $r_{\mathrm{NH}}$ is the distance between the nitrogen and proton nuclei [11].

#### 1.1.3 Paramagnetic Relaxation Enhancement (PRE)

PREs give distance restraints between the paramagnetic lanthanide and spin of interest from peak intensity ratios between paramagnetic and diamagnetic states (Fig. 2e). The PRE is given by Eq. (1) shown in Fig. 2c.

$$\lambda^{\mathrm{PRE}} = \frac{K}{r^6}\left(4\tau_{\mathrm{r}} + \frac{3\tau_{\mathrm{r}}}{1+\omega_{\mathrm{H}}^2\tau_{\mathrm{r}}^2}\right) \tag{3}$$

with,

$$K = \frac{1}{5}\left(\frac{\mu_0}{4\pi}\right)^2\frac{B_0^2\gamma_{\mathrm{H}}^2\left(g_j\mu_{\mathrm{B}}\right)^4 J^2(J+1)^2}{\left(3k_{\mathrm{B}}T\right)^2} \tag{4}$$

where $\tau_{\mathrm{r}}$ is the rotational correlation time, $\omega_{\mathrm{H}}$ is the Larmor frequency of the proton, $\mu_0$ is the vacuum permeability, $g_j$ the g-factor, $\mu_{\mathrm{B}}$ the Bohr magneton, and $J$ the total spin moment [11].

#### 1.1.4 Cross Correlated Relaxation (CCR)

This effect is measured by comparing the line width between the two components of the antiphase doublet (Fig. 2e) [11]. This effect combines distance and angle dependence given by Eq. (3) shown in Fig. 2d.

$$\eta^{\mathrm{CCR}} = K\frac{3\cos^2\eta - 1}{r^3}\left(4\tau_{\mathrm{r}} + \frac{3\tau_{\mathrm{r}}}{1+\omega_{\mathrm{H}}^2\tau_{\mathrm{r}}^2}\right) \tag{5}$$

with,

$$K = \frac{1}{30}\left(\frac{\mu_0}{4\pi}\right)^2 \frac{B_0^2\gamma_{\mathrm{H}}^2\left(g_j\mu_{\mathrm{B}}\right)^4 J^2(J+1)^2}{(3k_{\mathrm{B}}T)^2} \tag{6}$$

### 1.2 Structural Information from Paramagnetic Effects

RDCs, which are defined from the molecular alignment tensor (Eq. 2, Fig. 2b), give the orientation of spin pairs relative to the external magnetic field in a distance independent fashion. RDCs by themselves can be directly used to determine the structure of small proteins only when a large number of experimental RDCs are available. Measurement of heteronuclear RDCs becomes difficult for proteins that exhibit limited solubility or produce broad NMR line widths due to tag mobility.

PREs on the other hand give distance information from the paramagnetic center (Eq. 1, Fig. 2c). PREs induced by lanthanide ions range up to 20 Å, but the effect is heavily influenced by the motion of the metal carrying tag [12]. Direct usage of PREs in structure determination is limited but chemically inert paramagnetic probes when added as co-solvents can be quantitatively used to characterize interfaces in protein–protein complexes.

#### 1.2.1 Uniqueness of PCS

In comparison to RDCs and PREs, PCSs are the most potent structural restraints. A PCS defined by the $\Delta\chi$-tensor is both orientation and distance dependent (Eq. 1, Fig. 2a). The PCS effect has the longest range among all the paramagnetic effects and extends up to 80 Å (40 Å from the paramagnetic center) and can be precisely measured even at low protein concentrations (<20 μM) [13]. It can be easily seen from the $\Delta\chi$-tensor defined in Eq. (1) that PCS influenced by a spin is proportional to $r^{-3}$ from the metal center, which decays slower with distance than PRE with $r^{-6}$ dependence (Eq. 1). RDCs, in stark contrast to PCSs and PREs, are only orientation dependent (Eq. 2) brought about by the weak alignment from the inserted paramagnetic metal [8].

Experimentally PCSs are easy to measure in proteins by taking the difference in chemical shifts of a protein's paramagnetic and diamagnetic states from simple 2D NMR spectra (shown in Fig. 3a). PCS can also be measured with higher accuracy and sensitivity compared to other paramagnetic effects, such as measuring coupling constants between nuclei for RDCs and measuring peak intensities for PREs. The induced PCS described within the $\Delta\chi$-tensor can be visualized as isosurfaces of constant PCS (shown in Fig. 3b). The $\Delta\chi$-tensor is fully defined by eight parameters, the origin of the tensor frame which coincides with the coordinates of

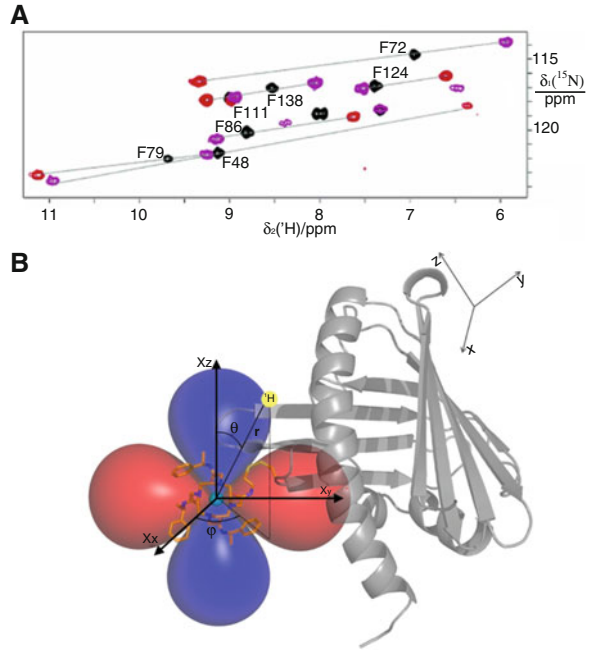



**Fig. 3** Measurement of pseudocontact shift (PCSs) and display of PCS as isosurfaces. (**a**) An illustration of three superimposed $^{15}$N-HSQC spectra, showing the chemical shift changes due to presence of paramagnetic metal ions in the protein. Black resonances come from the diamagnetic reference ($Y^{3+}$) sample, while red ($Dy^{3+}$) and magenta ($Er^{3+}$) resonances show chemical shift changes due to the paramagnetic lanthanide ions attached in the sample. (**b**) Visualization of induced PCS as isosurfaces calculated from the $\Delta\chi$-tensor

the metal $(x,y,z)$, orientation of $\Delta\chi$-tensor frame (three Euler angles $\alpha, \beta, \gamma$) with respect to the coordinate frame of the protein, and two components of the $\Delta\chi$-tensor, $\Delta\chi_{ax}$ (axial) and $\Delta\chi_{rh}$ (rhombic). To solve for the full mathematical description of a $\Delta\chi$-tensor one needs to measure a minimum of eight PCSs.

## 2 PCSs in Protein Structure Characterization

### *2.1 Paramagnetic NMR Spectrum Assignment*

Accurate assignment of resonances in the NMR spectrum is the essential first step in extracting restraints. Especially for large proteins (>20 kDa), assignment of multidimensional NMR spectra becomes increasingly difficult due to spectral overlap and increased transverse relaxation of spins. If 3D atomic coordinates of nuclear spins are known, the NMR resonance assignments of both paramagnetic and diamagnetic spectra can be assigned with software algorithms. Several software algorithms are available to assist with NMR assignments, including Numbat [14], Possum [15], Echidna [16], and PARAssign [17].

### 2.2 Protein–Ligand Interactions

PCSs can be measured not only on the protein's nuclear spins but also on the spins of the bound ligands. With the availability of a diverse range of metal binding chemical tags, the orientation and location of the ligand can be easily identified with the help of PCSs [3, 9]. This ability has major implications for rational drug design. John et al. [18] have demonstrated this concept using *E. coli*'s $\varepsilon 186/\theta$ (a natural lanthanide binding protein) in complex with the ligand thymidine, where the ligand affinity and its binding orientation was entirely determined using only PCSs. Saio et al. [19] showed that a combination of PCSs and PREs generated from two point anchored lanthanide binding peptide can be used to screen for ligands for protein Grb2. Guan et al. [20] showed that even in the absence of isotope labeled protein samples, the location of the ligand bound to the protein can be determined in low resolution with predicted $\Delta\chi$-tensor parameters.

### 2.3 Protein–Protein Complexes

Protein–protein complexes are fundamental to the function of cellular signaling and function. If 3D structures of the interacting protein partners are known, then the directionality and distance dependence of the $\Delta\chi$-tensor can be exploited in docking the interacting partners in the right orientation. Pintacuda et al. [21] reported the first demonstration of the use of PCSs to compute the structure of a protein complex, using the interacting partners of *E. coli* DNA polymerase complex's N-terminal domain of the subunits $\varepsilon$ and $\theta$. Recent studies involving a large PCS data set (446 PCSs) have been used to characterize cytochrome P450cam in complex with putidaredoxin using double cysteine anchored tag [22]. PCS restraints are incorporated into protein–protein docking program Haddock, where the orientation of interacting partners and $\Delta\chi$-tensors are simultaneously fitted for finding optimized interacting surfaces [23].

### 2.4 Protein Structure Refinement

If the coordinates of atoms in the protein are known, PCSs can be effectively used to refine protein structures. Allegrozzi et al. [24] showed that NOE derived structural models can be further refined using PCSs that are measured using three different lanthanides ($Ce^{3+}$, $Yb^{3+}$, and $Dy^{3+}$), which have different coverage range over the protein. Supplementing PCS restraints on the protein calbindin decreased the overall RMSD over NOE derived NMR structures. Gaponenko et al. [25] showed that using PCS data generated from three different lanthanide attachment sites extended the refinement approach to proteins larger than 30 kDa. PCS refined structures showed improvement over an Ångström RMSD when compared to NOE only structures, and this improved accuracy is also validated using RDCs. Other paramagnetic restraints also have been used in a similar manner. Sparse datasets of RDCs combined with sparse NOEs have been used to identify the best models from a pool of structures generated using homology modeling [26] and de novo methods [27].

To directly use PCSs for structure calculation is challenging as one needs to determine the eight parameters to describe the $\Delta\chi$-tensor, which are difficult to estimate as they depend on the chemical environment of the metal. Without the knowledge of 3D coordinates of the protein it is not possible to fit the $\Delta\chi$-tensor to reproduce the experimentally observed PCSs. However, one can use PCSs as restraints in de novo structure prediction methods such as Rosetta [28]. Rosetta's forcefield accurately describes the protein state and the software algorithms are designed to robustly search the conformational space accessible to the protein.

## 3 Protein Structure Determination Using PCS and Rosetta

Incomplete or sparse structural data generated from NMR experiments can be used as structural restraints in Rosetta calculations to facilitate structure determination. Unlike traditional methods where structure calculation is mainly determined by the completeness of experimental data which defines the position of atomic coordinates in a protein structure, the sparse NMR data is used to guide the conformational search which directs the sampling towards the global minimum. Different types of NMR measurements have been incorporated as additional scoring restraints in Rosetta. Chemical shift measurements combined with predicted backbone dihedrals and secondary structure elements can be used in picking fragments that match the prediction, a procedure known as CS-Rosetta [29, 30]. Backbone NOEs in combination with RDCs have also been included in protein structure determination [28] using an advanced genetic algorithm [31]. Incorporation of sparse NMR data in structure calculations has been shown to improve protein structure predictions.

### 3.1 Rosetta Structure Calculation Algorithm

Based on folding studies of small proteins, Rosetta's algorithms are built on the assumption that the ensemble of local structures sampled by a sequence fragment can be approximated by a small number of local structures that a similar fragment adopts in known protein structures [32]. For a given protein sequence whose structure is to be determined, the sequence is decomposed into overlapping windows of nine and three residues. The fragment libraries are constructed for each of the nine and three residue windows by searching through 3D structure databases for protein fragments whose sequences or secondary structures have high similarity to that of the query. The corresponding backbone dihedral angles of the matched protein fragments are bundled up into fragment libraries The search for the lowest energy structure is carried out by assembling the fragments into protein-like structures using Metropolis Monte-Carlo and simulated annealing algorithms [33]. Starting from a linear polypeptide, the search is carried out in

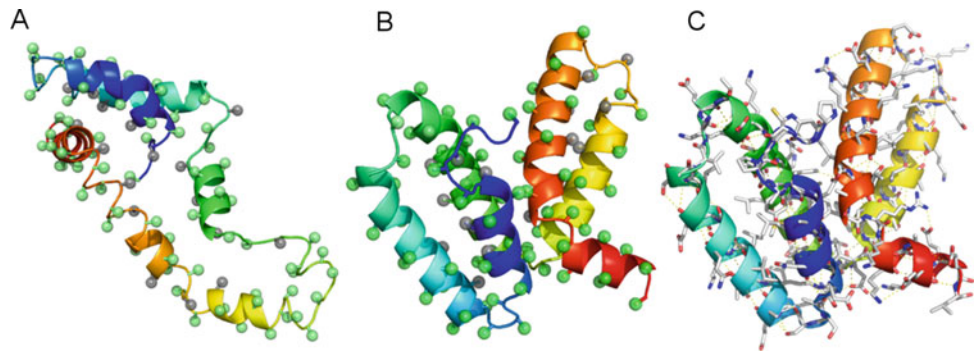



**Fig. 4** Illustration of Rosetta's ab initio fragment assembly. (**a**) A protein decoy in an intermittent state during fragment assembly. The backbone atoms are shown in a cartoon representation and the side chain atoms are represented as spheres attached to Cβ atoms. The hydrophobic residues represented in *grey* and the solvent accessible residues represented in *green*. (**b**) Final fold of the protein shown in (**a**) after the low resolution fragment assembly. (**c**) All-atom representation of the final fold of the protein with complete side-chain atoms

two distinct phases, a low resolution centroid mode and a high resolution all-atom mode [34].

#### 3.1.1 *Centroid Mode*

In this mode, the conformational search is carried out in a low-resolution phase, in which the amino acid residues are represented in a stripped down version that lacks complete side chain detail. The side chains are represented as spheres attached to the backbone (Cβ and beyond) at their centroid point as shown in Fig. 4a. The fragment assembly follows Monte-Carlo moves starting from an arbitrary position from a random nine residue fragment window. For every move, which replaces the coordinates of a protein segment from that of a fragment library, the energy of the resultant protein decoy is evaluated. The scoring function in the centroid phase is a coarse-grained description of probabilistic functions which favors the formation of globular compact structures. This scoring function explicitly scores for electrostatic and solvation effects among residues which are based on the observed distributions in known proteins. Formation of secondary structural elements in the folding pathway is encouraged with distinct function terms that favor helix–helix, helix–sheet, and sheet–sheet pairing. This low resolution centroid mode generates protein like decoy structures, in which the polar amino acids are exposed to the solvent while burying the hydrophobic residues in the core of the protein (shown in Fig. 4b). Multiple folding pathways are independently sampled, generating tens to hundreds of thousands of protein decoy structures to sample the vast conformational space.

#### 3.1.2 *All-Atom Mode*

This mode generates complete and optimized placement of side-chain coordinates (shown in Fig. 4c). Here side chains are modeled by searching through discrete combinations of amino acid rotamers

by simulated annealing. To further optimize the geometry, multistep Monte Carlo minimisation is enforced on each decoy; steps include torsion angle perturbations, one-at-a-time rotamer optimization and continuous gradient based minimisation of backbone torsion angles and side chain coordinates. The scoring function during this stage is more detailed, physically realistic, accurate to the atomic level and computationally expensive. Hydrogen bonding is explicitly included in the analysis. Hydrogen bonding terms are knowledge-based terms which are orientation and secondary structure dependent and were derived from high resolution protein structures. Typically, multiple independent trajectories are first clustered and atomic details are generated on the desired cluster [33].

#### 3.1.3 PCS Restraints in Rosetta

In the centroid mode, at each instance of a fragment move, $\Delta\chi$-tensors are fitted to the assembled structure and PCSs are back-calculated. The difference between the input and back-calculated PCSs are then used as a quality score to guide assembly to the right fold of the protein. It has been shown that using PCSs from a single metal center, 3D protein structures up to 150 amino acid residues can be determined at atomic resolution [35]. However, this method is limited in its application for proteins larger than 150 amino acids.

The primary limitation associated with the PCSs measured from a single metal center is the reduction in quality of PCS data. Lanthanide tags attached to a single metal center often fail to induce significantly large PCS for most of the spins in the protein. This loss of data is pronounced in large molecular weight proteins. Secondly, there is additional loss of data due to induced PRE effect by the lanthanide ions, where NMR signals of the spins near the vicinity of the lanthanides are broadened beyond detection.

### 3.2 Extending PCS Scoring to Multiple Metal Centers

To resolve the ambiguities associated with the PCS data generated from a single metal center and to achieve complete coverage, the approach has been extended from a single metal center to multiple metal centers. A second PCS measured for the same nucleus from a lanthanide attached at a different site restricts the spin to lie on intersecting isosurfaces. A third PCS measured from a lanthanide attached at a site different from the first two would further restrict the location of the spin in space. This technique, which is analogous to the method of finding a location on Earth from three or more GPS satellites, is incorporated into the Rosetta framework and was dubbed GPS-Rosetta [36].

### 3.2.1 Scoring PCS Data from Multiple Metal Centers in Rosetta

The $\Delta\chi$-tensor from Eq. (1) can be rewritten as

$$\mathrm{PCS}_i^{\mathrm{calc}} = \frac{1}{12\pi r_i^5} \cdot \mathrm{Trace}\left[ \begin{pmatrix} 3x_i^2 - r_i^2 & 3x_i y_i & 3x_i z_i \\ 3x_i y_i & 3y_i^2 - r_i^2 & 3y_i z_i \\ 3x_i z_i & 3y_i z_i & 3z_i^2 - r_i^2 \end{pmatrix} \begin{pmatrix} \Delta\chi_{xx} & \Delta\chi_{xy} & \Delta\chi_{xz} \\ \Delta\chi_{xy} & \Delta\chi_{yy} & \Delta\chi_{yz} \\ \Delta\chi_{xz} & \Delta\chi_{yz} & \Delta\chi_{zz} \end{pmatrix} \right] \tag{7}$$

where, $r_i$ is the distance between the spin $i$ and the paramagnetic center $M$; $x_i$, $y_i$, and $z_i$ are the Cartesian coordinates of the vector between the metal ion and the spin $i$ in an arbitrary frame $f$; and $\Delta\chi_{xx}$, $\Delta\chi_{yy}$, $\Delta\chi_{zz}$, $\Delta\chi_{xy}$, $\Delta\chi_{xz}$, and $\Delta\chi_{yz}$ are the $\Delta\chi$-tensor components in the frame $f$ (as $\Delta\chi_{zz} = -\Delta\chi_{xx} - \Delta\chi_{yy}$, there are only five independent parameters). The determination of $\mathrm{PCS}_i^{\mathrm{calc}}$ (Eq. 5) poses a nonlinear least-square fit problem, which can be divided into its linear and nonlinear parts. $\mathrm{PCS}_i^{\mathrm{calc}}$ is linear with respect to the five $\Delta\chi$-tensor components which can be optimized efficiently using singular value decomposition. With the knowledge of the location of the chemical tag used, search over the metal coordinates $x_{\mathrm{M}}$, $y_{\mathrm{M}}$, and $z_{\mathrm{M}}$ of the paramagnetic center can be carried out on a 3D grid. The 3D grid is defined with parameters which include center of the grid search (*cg*), step size between two nodes (*sg*), an outer cutoff radius (*co*) which limits the search to a minimal distance from *cg* and an inner cutoff radius (*ci*) to avoid a search too close to *cg* [35].

PCSs recorded from multiple lanthanide carrying chemical tags are given as input into Rosetta by constructing multiple 3D grids for individual tag site. For each PCS dataset per metal and chemical tag, the $\Delta\chi$-tensor components are fitted at each node of the 3D grid and the PCSs are back-calculated. The grid node with the lowest score obtained from Eq. (6) is then taken as the starting point to further optimize the metal position and the five components of the $\Delta\chi$-tensor to reach the minimum cost for all the metal centers.

$$s_k = \sum_{q=1}^{m} \sqrt{\sum_{p=1}^{n_{\mathrm{pcs}}} \left( \mathrm{PCS}_{\mathrm{calc}}^{pq} - \mathrm{PCS}_{\mathrm{exp}}^{pq} \right)^2} \tag{8}$$

where $m$ is the number of PCS data sets (one dataset per metal ion) per binding site $k$ and $n_{\mathrm{pcs}}$ is the number of PCSs in the dataset. A total weighted sum of square deviations are used as PCS scoring $S_{\mathrm{total}}$ and added to the low-resolution energy function of Rosetta:

$$S_{\mathrm{total}} = \sum_{k=1}^{n} s_k . w_k \tag{9}$$

where $n$ is the total number of metal binding centers and $w$ denotes the weighting factor relative to the Rosetta ab initio scoring function. The weighting factor $w$ for each of the $n$ centers was calculated independently by

$$w = \left(\frac{a_{\text{high}} - a_{\text{low}}}{c_{\text{high}} - c_{\text{low}}}\right)/n \tag{10}$$

where $a_{\text{high}}$ and $a_{\text{low}}$ are the averages of the highest and lowest 10 % of the values of the Rosetta ab initio score, and $c_{\text{high}}$ and $c_{\text{low}}$ are the averages of the highest and lowest 10 % of PCS score obtained by rescoring 1000 decoys with unity weighting factor.

## 4 The GPS-Rosetta Algorithm

The algorithm incorporating PCS scoring from multiple metal centers in Rosetta's structure determination protocol is described as a flow chart in Fig. 5. The $\Delta\chi$-tensors for each dataset from multiple sites are simultaneously optimized and the weighted PCS scores for individual metal sites are added to the centroid scoring function. Side chain atoms are then added to all the structural decoys and scored using Rosetta's all-atom scoring function. The PCS scoring is not used in this mode, because only minor changes in the backbone structure are generated. The side chain optimized structural models are rescored with PCS data from multiple metal centers with new weights generated using Eq. (8), except that they are now weighted against Rosetta's all-atom scoring function. The top structures are selected based on lowest combined scores of Rosetta's all-atom score and weighted PCS score from all the tag sites.

GPS-Rosetta protocol has been implemented in determining 3D structures from PCSs data generated from two different NMR experiments, solution state NMR and magic angle spinning (MAS) solid-state NMR experiments. C-terminal domain of endoplasmic reticulum protein 29, ERp29-C (106 residues) from rat, is determined from the PCS data generated at 4 different metal centers in solution state and Immunoglobulin Binding Domain of Protein G, GB1 (56 residues) from *Streptococcus* spp, is determined from PCS data generated at three different metal centers in microcrystalline state.

### 4.1 Fold Determination Using PCSs from Solution NMR Experiments

ERp29-C is a chaperone protein expressed in the endoplasmic reticulum of a mammalian cell, where it facilitates the folding and transport of other protein molecules. The 3D structure was first determined by solution NMR using a conventional NOE approach, and the result is referred to as the NOE structure [37]. However, the crystal structure of human ERp29-C [38] shows a significantly different fold with Cα root mean squared deviation (RMSD) of 4.5 Å when compared to the NOE structure. GPS-Rosetta protocol was employed to reassess the structure in solution [36]. Four different sites on the protein were chosen to bind two different lanthanide tags. The cysteine ligated, C1 tag [39] was chosen to

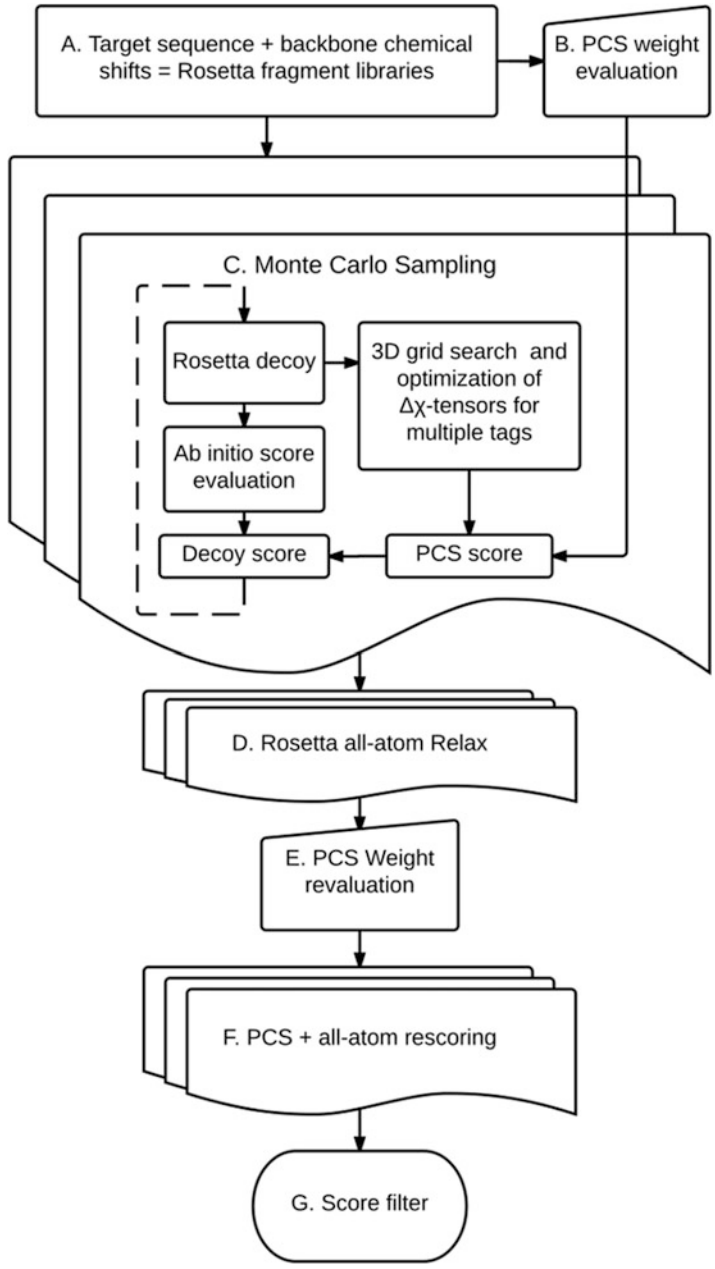



**Fig. 5** Flowchart illustrating series of steps involved in running GPS-Rosetta protocol. (**a**) Short nine and three residue fragments are generated based on target sequence and secondary structure prediction based on backbone chemical shifts. (**b**) PCS weights are calculated using Eq. (8). (**c**) Centroid models are generated by fragment assembly following Metropolis Monte-Carlo sampling algorithm. PCS scores for individual tag sites are independently optimized and the PCS scores are added to Rosetta's scoring function. (**d**) Side chain generation and optimization to centroid models. (**e**) PCS score for individual tag sites are reweighted from all-atom models. (**f**) Models are rescored with PCSs and Rosetta's all-atom scoring function. (**g**) Final structure is selected based on lowest combined score value

bind at the native cysteine (C157) and IDA-SH tag [40] was attached at double mutants S200C/K204D, A218C/A222D, and Q241C/N245D. All the double mutations were on α-helices and the aspartate residue at $(i + 4)$th position forming a specific lanthanide binding site. The side chain carboxyl-oxygen of the aspartate served as an additional coordination site to immobilize the lanthanide ion. The PCS dataset from eight paramagnetic samples is composed of a total of 212 PCSs measured using lanthanides $Tb^{3+}$, $Tm^{3+}$, and $Y^{3+}$, where $Y^{3+}$ served as diamagnetic reference.

The unique coordination feature of IDA-SH enabled determination of the position of the metal ion at 5.9 Å from the Cα of $(i + 4)$th residue, lying on a vector that joins the backbone amide nitrogen at $(i + 6)$ and Cα of $(i + 4)$th aspartate. The lanthanide position defined by C1 tag at C157 was dynamically optimized during the folding simulation. More than 100,000 all-atom models were generated using GPS-Rosetta protocol and multiple structures satisfying combined Rosetta and PCS score and experimental data were selected. The final structure was selected for the model that has the lowest Rosetta's all-atom and weighted PCS energy. The final selected structure, which is represented by the red point in Fig. 6a, has a backbone Cα RMSD of 2.4 Å to the crystal structure (Fig. 6b) [PDBID: 2QC7;[38]], and is referred to as the GPS-Rosetta model. The top five structures that are lowest in PCS RMSD are shown in blue points and the top five models with an arbitrary low combined score and low PCS RMSD are represented as green points (Fig. 6a). The GPS-Rosetta structure was compared against the crystal structure and top 10 selected structures. Superposition structures with low PCS RMSD are represented in shades of blue (Fig. 6b), and low scoring in PCS and Rosetta energy and low PCS RMSD are represented in shades of green (Fig. 6c). The Cα RMSD of all the selected structures lies in the range 2.0–2.9 Å to the crystal structure with the exception of small variations in the orientation of the C-terminal residues which were reported to be disordered [37]. The GPS-Rosetta structure, in red, (PDBID: 2M66) clearly resembles the crystal structure more closely (Fig. 6b, RMSD of 2.4 Å) than the NOE structure (Fig. 6d, RMSD 6 Å), effectively overruling it.

### 4.2 High Resolution Protein Structure Determination Using PCSs from MAS Solid-State NMR Experiments

MAS solid-state NMR spectroscopy has been routinely employed to determine structure of membrane biomolecules and proteins that are difficult to study by solution NMR or X-ray crystallography [41]. 3D structures are determined by resolving large number of dipolar couplings between $^{1}H$, $^{13}C$, and $^{15}N$ nuclei [42, 43]; however, the spectrum resolves in densely packed cross-peaks which are highly difficult to assign. Moreover, the peaks arising from long range correlations in dipolar couplings produce low signal-to-noise ratio and the time required to acquire a 2D spectra is several days [44, 45].

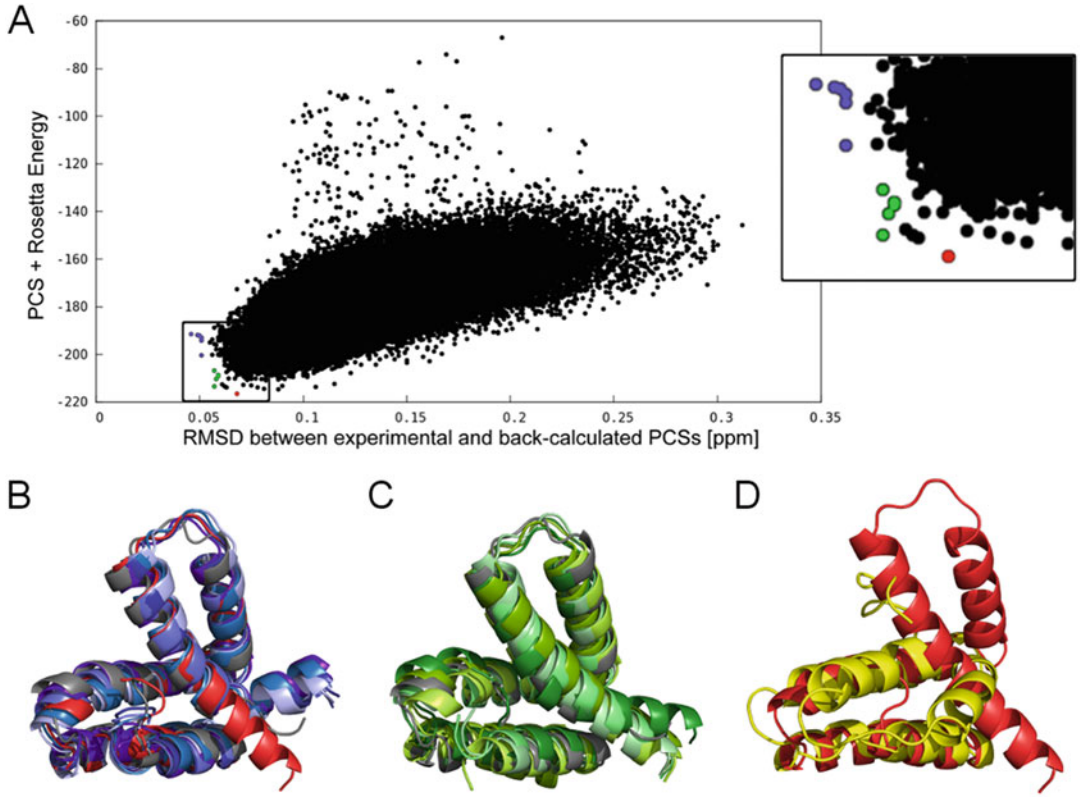



**Fig. 6** Structure determination using GPS-Rosetta protocol for ERp29-C. (**a**) Combined score of weighted PCS and Rosetta energy is plotted against the PCS RMSD for each of the 100,000 generated structures. The final selected structure is represented in *red* has the lowest combined score. Structures with lowest PCS RMSD are represented in *blue* and the models with an arbitrary low combined score and low PCS RMSD are represented in *green*. (**b**) Superimposed cartoon representations of top structures selected using the GPS-Rosetta protocol. The crystal structure [PDBID: 2QC7] is shown in grey and the GPS-Rosetta structure is represented in red has 2.4 Å Cα RMSD to the crystal structure (residues 158–228 and 230–244). Top five models with low PCS RMSD represented in shades of blue have a Cα RMSD range of 2.0–2.9 Å to the crystal structure (residues 158–228 and 230–244). (**c**) Top five models with low PCS and Rosetta energy and also low in PCS RMSD are represented in shades of *green* have a Cα RMSD range of 2.2–2.6 Å to the crystal structure (residues 158–228 and 230–244). (**d**) The NOE structure [PDBID: 1G7D] represented in *yellow* has Cα RMSD of 6 Å to the GPS-Rosetta structure (residues 158–244) represented in *red*

Here we demonstrate the implementation of PCSs recorded in solid state for structure calculation. GB1 protein (56 amino acids) served as a model system. GB1 was covalently ligated to 4-mercaptomethyl-dipicolinic acid (4MMDPA) tag [46] at three different sites by generating three cysteine mutants at K28C, D40C, and E42C. The tags were loaded with paramagnetic metal ions $Co^{2+}$, $Yb^{3+}$, and $Tm^{3+}$, while $Zn^{2+}$ and $Lu^{3+}$ served as diamagnetic references. A total of 244 PCSs were measured from five paramagnetic datasets [47]. GB1 being a small protein, a stripped down version of GPS-Rosetta protocol was employed. Three

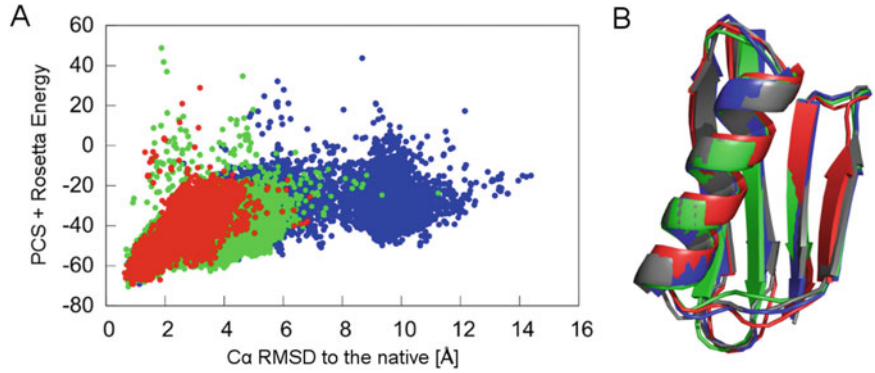



**Fig. 7** Structure determination using GPS-Rosetta protocol with MAS-NMR PCSs. (**a**) Combined score of PCS energy from three tags and Rosetta energy versus the RMSD to the crystal structure of GB1 [PDBID:1PGA, [48]]. Sampling from K28C is represented in red, D40C in green and E42C in *blue*. (**b**) 3D superpositions of calculated models using GPS-Rosetta. The crystal structure of GB1 is represented in *grey*, mutant K28C in *red*, D40C in *green*, and E42C in *blue*. The three lowest scored structures have an RMSD to the crystal structure of 0.9, 0.7, and 1.1 Å respectively

independent Rosetta simulations were carried out for each mutant with nonhomologous fragment libraries. Around 4500, 8400, and 10,000 all-atom models were generated for each of the three mutants. To take advantage of all three datasets for GB1, Rosetta's all-atom structures for each of the mutants were rescored using the GPS-Rosetta protocol and the final structures were selected based on low Rosetta energy and combined low PCS score from all three datasets (Fig. 7a). The lowest combined energy structure was found to have RMSD of 0.7 Å when superimposed over the crystal structure (Fig. 7b) [PDBID: 1PGA, [48]], at atomic resolution.

The GPS-Rosetta protocol along with demonstration tutorials is available for download with the current Rosetta release.

## 5 Conclusion

Here GPS-Rosetta protocol's success in determining 3D structures using PCS data from multiple tags from both solution and solid-state NMR experiments has been demonstrated. This method offers great promise in resolving structures of large proteins. PCSs are obtained from simple $^{15}$N-HSQC measurements which are highly accurate and sensitive compared to traditional NOE measurements and versatile PCS datasets can be generated by swapping a diverse range of available paramagnetic metals, metal carrying tags, and peptide sequences.

In computational modeling, incorporation of PCS data as structural restraints has enabled the computationally intractable conformational space to be explored in finite time. Inaccuracies in

molecular force fields always posed a challenge in identifying native protein fold from well-formed structural decoys and PCSs being long range in nature effectively discriminates the native from non-native folds. PCSs can be also complemented with other sparse restraints such as RDCs, PREs, and NOEs, enhancing the structural information which can be efficiently exploited in computational modeling. In conclusion, the hybrid approach of incorporating experimental PCSs with structure determination algorithms forms a more efficient alternative approach to solve protein structures than traditional methods.

## References


1. Wuthrich K (1986) NMR of proteins and nucleic acids. The George Fisher Baker Non-resident Lectureship in Chemistry at Cornell University
2. Andreini C, Bertini I, Rosato A (2009) Metalloproteomes: a bioinformatic approach. Acc Chem Res 42:1471–1479
3. Otting G (2010) Protein NMR using paramagnetic ions. Annu Rev Biophys 39:387–405
4. Su X-C, McAndrew K, Huber T, Otting G (2008) Lanthanide-binding peptides for NMR measurements of residual dipolar couplings and paramagnetic effects from multiple angles. J Am Chem Soc 130:1681–1687
5. Loh CT, Ozawa K, Tuck KL, Barlow N, Huber T, Otting G, Graham B (2013) Lanthanide tags for site-specific ligation to an unnatural amino acid and generation of pseudocontact shifts in proteins. Bioconjug Chem 24:260–268
6. Rodriguez-Castañeda, F., Haberz, P., Leonov, A., Griesinger, C. (2006) Paramagnetic tagging of diamagnetic proteins for solution NMR. *Magn Reson Chem* 44 Spec No, S10–S16.
7. Su X-C, Otting G (2010) Paramagnetic labelling of proteins and oligonucleotides for NMR. J Biomol NMR 46:101–112
8. Koehler J, Meiler J (2011) Expanding the utility of NMR restraints with paramagnetic compounds: background and practical aspects. Prog Nucl Magn Reson Spectrosc 59:360–389
9. Liu W-M, Overhand M, Ubbink M (2014) The application of paramagnetic lanthanoid ions in NMR spectroscopy on proteins. Coord Chem Rev 273–274:2–12
10. Bertini I, Luchinat C, Parigi G (2002) Magnetic susceptibility in paramagnetic NMR. Prog Nucl Magn Reson Spectrosc 40:249–273
11. Bertini I, Luchinat C, Parigi G, Pierattelli R (2008) Perspectives in paramagnetic NMR of metalloproteins. Dalton Trans 29:3782–3790
12. Iwahara J, Schwieters CD, Clore GM (2004) Ensemble approach for NMR structure refinement against (1)H paramagnetic relaxation enhancement data arising from a flexible paramagnetic group attached to a macromolecule. J Am Chem Soc 126:5879–5896
13. Keizers PHJ, Mersinli B, Reinle W, Donauer J, Hiruma Y, Hannemann F, Overhand M, Bernhardt R, Ubbink M (2010) A solution model of the complex formed by adrenodoxin and adrenodoxin reductase determined by paramagnetic NMR spectroscopy. Biochemistry 49:6846–6855
14. Schmitz C, Stanton-Cook M, Su X-C, Otting G, Huber T (2008) Numbat: an interactive software tool for fitting Deltachi-tensors to molecular coordinates using pseudocontact shifts. J Biomol NMR 41:179–189
15. John M, Schmitz C, Park AY, Dixon NE, Huber T, Otting G (2007) Sequence-specific and stereospecific assignment of methyl groups using paramagnetic lanthanides. J Am Chem Soc 129:13749–13757
16. Schmitz C, John M, Park AY, Dixon NE, Otting G, Pintacuda G, Huber T (2006) Efficient chi-tensor determination and NH assignment of paramagnetic proteins. J Biomol NMR 35:79–87
17. Skinner SP, Moshev M, Hass MAS, Keizers PHJ, Ubbink M (2013) PARAssign—paramagnetic NMR assignments of protein nuclei on the basis of pseudocontact shifts. J Biomol NMR 55:379–389
18. John M, Pintacuda G, Park AY, Dixon NE, Otting G (2006) Structure determination of protein-ligand complexes by transferred paramagnetic shifts. J Am Chem Soc 128:12910–12916
19. Saio T, Ogura K, Shimizu K, Yokochi M, Burke TR, Inagaki F (2011) An NMR strategy for fragment-based ligand screening utilizing a paramagnetic lanthanide probe. J Biomol NMR 51:395–408

20. Guan J-Y, Keizers PHJ, Liu W-M, Loehr F, Skinner SP, Heeneman EA, Schwalbe H, Ubbink M, Siegal GD, Löhr F, Skinner SP, Heeneman EA, Schwalbe H, Ubbink M, Siegal GD (2013) Small molecule binding sites on proteins established by paramagnetic NMR spectroscopy. J Am Chem Soc 135:5859–5868
21. Pintacuda G, Park AY, Keniry MA, Dixon NE, Otting G (2006) Lanthanide labeling offers fast NMR approach to 3D structure determinations of protein-protein complexes. J Am Chem Soc 128:3696–3702
22. Hiruma Y, Gupta A, Kloosterman A, Olijve C, Olmez B, Hass MA, Ubbink M (2014) Hot-spot residues in the cytochrome P450cam-putidaredoxin binding interface. Chembiochem 15:80–86
23. Schmitz C, Bonvin AMJJ (2011) Protein-protein HADDocking using exclusively pseudocontact shifts. J Biomol NMR 50:263–266
24. Allegrozzi M, Bertini I, Janik MBL, Lee Y, Liu G, Luchinat C (2000) Lanthanide-induced pseudocontact shifts for solution structure refinements of macromolecules in shells up to 40 Å from the metal ion. J Am Chem Soc 122:4154–4161
25. Gaponenko V, Sarma SP, Altieri AS, Horita DA, Li J, Byrd RA (2004) Improving the accuracy of NMR structures of large proteins using pseudocontact shifts as long-range restraints. J Biomol NMR 28:205–212
26. Song Y, Dimaio F, Wang RY-R, Kim D, Miles C, Brunette T, Thompson J, Baker D (2013) High-resolution comparative modeling with RosettaCM. Structure 21:1735–1742
27. Meiler J, Baker D (2003) Rapid protein fold determination using unassigned NMR data. Proc Natl Acad Sci U S A 100:15404–15409
28. Raman S, Lange OF, Rossi P, Tyka M, Wang X, Aramini JM, Liu G, Ramelot TA, Eletsky A, Szyperski T, Kennedy MA, Prestegard J, Montelione GT, Baker D (2010) NMR structure determination for larger proteins using backbone-only data. Science 327:1014–1018
29. Shen Y, Vernon R, Baker D, Bax A (2009) De novo protein structure generation from incomplete chemical shift assignments. J Biomol NMR 43:63–78
30. Shen Y, Lange O, Delaglio F, Rossi P, Aramini JM, Liu G, Eletsky A, Wu Y, Singarapu KK, Lemak A, Ignatchenko A, Arrowsmith CH, Szyperski T, Montelione GT, Baker D, Bax A (2008) Consistent blind protein structure generation from NMR chemical shift data. Proc Natl Acad Sci U S A 105:4685–4690
31. Lange OF, Baker D (2012) Resolution-adapted recombination of structural features significantly improves sampling in restraint-guided structure calculation. Proteins Struct Funct Bioinforma 80:884–895
32. Baker D (2014) Centenary award and Sir Frederick Gowland Hopkins Memorial Lecture. Protein folding, structure prediction and design. Biochem Soc Trans 42:225–229
33. Rohl CA, Strauss CEM, Misura KMS, Baker D (2004) Protein structure prediction using Rosetta. Methods Enzymol 383:66–93
34. Das R, Baker D (2008) Macromolecular modeling with Rosetta. Annu Rev Biochem 77:363–382
35. Schmitz C, Vernon R, Otting G, Baker D, Huber T (2012) Protein structure determination from pseudocontact shifts using ROSETTA. J Mol Biol 416:668–677
36. Yagi H, Pilla KB, Maleckis A, Graham B, Huber T, Otting G (2013) Three-dimensional protein fold determination from backbone amide pseudocontact shifts generated by lanthanide tags at multiple sites. Structure 21:883–890
37. Liepinsh E, Baryshev M, Sharipo A, Ingelman-Sundberg M, Otting G, Mkrtchian S (2001) Thioredoxin fold as homodimerization module in the putative chaperone ERp29: NMR structures of the domains and experimental model of the 51 kDa dimer. Structure 9:457–471
38. Barak NN, Neumann P, Sevvana M, Schutkowski M, Naumann K, Malesević M, Reichardt H, Fischer G, Stubbs MT, Ferrari DM (2009) Crystal structure and functional analysis of the protein disulfide isomerase-related protein ERp29. J Mol Biol 385:1630–1642
39. Graham B, Loh CT, Swarbrick JD, Ung P, Shin J, Yagi H, Jia X, Chhabra S, Barlow N, Pintacuda G, Huber T, Otting G (2011) DOTA-amide lanthanide tag for reliable generation of pseudocontact shifts in protein NMR spectra. Bioconjug Chem 22:2118–2125
40. Swarbrick JD, Ung P, Chhabra S, Graham B (2011) An iminodiacetic acid based lanthanide binding tag for paramagnetic exchange NMR spectroscopy. Angew Chem 123:4495–4498
41. Hong M, Zhang Y, Hu F (2012) Membrane protein structure and dynamics from NMR spectroscopy. Annu Rev Phys Chem 63:1–24
42. De Paepe G, Lewandowski JR, Loquet A, Bockmann A, Griffin RG, De Paëpe G, Lewandowski JR, Loquet A, Böckmann A, Griffin RG (2008) Proton assisted recoupling and protein structure determination. J Chem Phys 129:245101
43. Korukottu J, Schneider R, Vijayan V, Lange A, Pongs O, Becker S, Baldus M, Zweckstetter M (2008) High-resolution 3D structure

determination of kaliotoxin by solid-state NMR spectroscopy. PLoS One 3:e2359

44. Wasmer C, Lange A, Van Melckebeke H, Siemer AB, Riek R, Meier BH (2008) Amyloid fibrils of the HET-s(218-289) prion form a beta solenoid with a triangular hydrophobic core. Science 319:1523–1526
45. Loquet A, Lv G, Giller K, Becker S, Lange A (2011) 13C spin dilution for simplified and complete solid-state NMR resonance assignment of insoluble biological assemblies. J Am Chem Soc 133:4722–4725
46. Su X-C, Man B, Beeren S, Liang H, Simonsen S, Schmitz C, Huber T, Messerle BA, Otting G (2008) A dipicolinic acid tag for rigid lanthanide tagging of proteins and paramagnetic NMR spectroscopy. J Am Chem Soc 130:10486–10487
47. Li J, Pilla KB, Li Q, Zhang Z, Su X, Huber T, Yang J (2013) Magic angle spinning NMR structure determination of proteins from pseudocontact shifts. J Am Chem Soc 135:8294–8303
48. Gallagher T, Alexander P, Bryan P, Gilliland GL (1994) Two crystal structures of the B1 immunoglobulin-binding domain of streptococcal protein G and comparison with NMR. Biochemistry 33:4721–4729
49. Schmitz C (2009) Computational study of proteins with paramagnetic NMR: automatic assignments of spectral resonances, determination of protein-protein and protein-ligand complexes, and structure determination of proteins. Ph.D. thesis, University of Queensland

# Chapter 2

# Inferring Function from Homology

## Tom C. Giles and Richard D. Emes

## Abstract

Recent technological advances in sequencing and high-throughput DNA cloning have resulted in the generation of vast quantities of biological sequence data. Ideally the functions of individual genes and proteins predicted by these methods should be assessed experimentally within the context of a defined hypothesis. However, if no hypothesis is known *a priori*, or the number of sequences to be assessed is large, bioinformatics techniques may be useful in predicting function.

This chapter proposes a pipeline of freely available Web-based tools to analyze protein-coding DNA and peptide sequences of unknown function. Accumulated information obtained during each step of the pipeline is used to build a testable hypothesis of function.

The following methods are described in detail:

1. Annotation of gene function through Protein domain detection (SMART and Pfam).
2. Sequence similarity methods for homolog detection (BLAST and DELTA-BLAST).
3. Comparing sequences to whole genome data.

**Key words** Comparative genomics, Homology, Orthology, Paralogy, BLAST, Protein domain, Pfam, SMART, Ensembl, UCSC genome browser

## 1 Introduction

This chapter describes an analysis pipeline comprised of freely available bioinformatics sequence comparison tools that can be used to infer potential function from protein-coding DNA and peptide sequences (Fig. 1).

### 1.1 What Is Homology?

In a biological context, homology is defined as the existence of shared ancestry between a pair of structures in different species, either by descent or recombination. The central thesis for inferring related function from sequence data is that if two or more genes have evolved slowly enough to allow detection of statistically significant sequence similarity; common ancestry (homology) between the genes can be inferred. This follows from the assumption that the most parsimonious explanation of sequence similarity

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_2, 

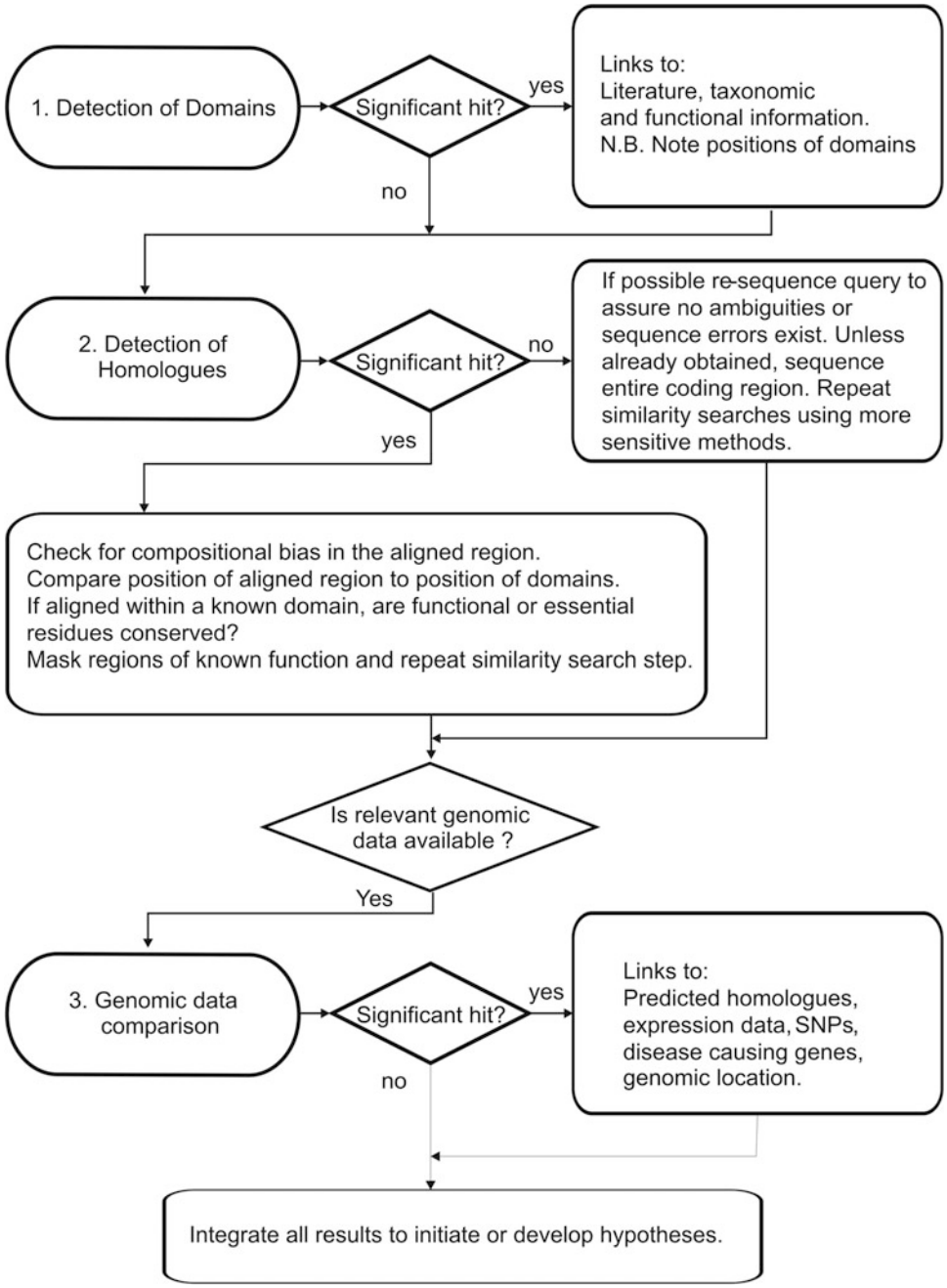



**Fig. 1** Analysis pipeline for inference of function. Schematic representation of analysis procedure for inference of function by similarity methods

is derived from conserved ancestry rather than convergent evolution [1, 2].

While the possession of sequence similarity is indicative of underlying structural similarity it may not always imply conserved

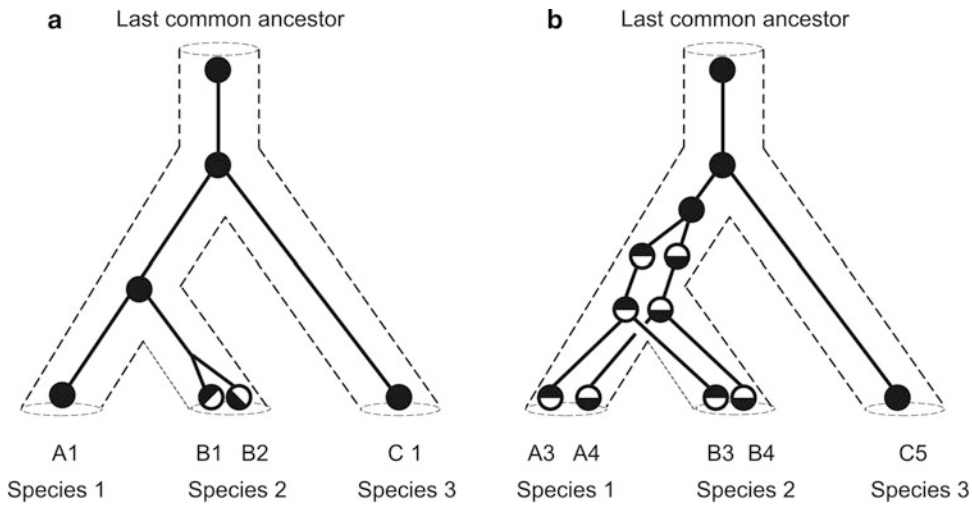



**Fig. 2** Homology and paralogy. *Dotted lines* represent the relationship of species 1, species 2, and species 3, separated by two speciation events. Genes are represented by *filled circles*. All genes represented are homologous because they have descended from a single gene in the last common ancestor. Definition of genes as orthologs or paralogs depends on their shared ancestry. If the genes in question are most recently related by a gene duplication event they are termed paralogs, whereas if the genes are most recently related by a speciation event, they are termed orthologs. If the gene duplication is an intra-genome event, occurring following speciation, the genes are further defined as in-paralogs. If the duplication is prior to a speciation event they are termed as out-paralogs [1, 5]. (**a**) The intra-genome duplication within species 2 has resulted in a pair of in-paralogous sequences B1 and B2. Both B1 and B2 are orthologous to A1 and all genes are orthologous to C1. (**b**) For a different set of genes, a gene duplication prior to the speciation of species 1 and 2 results in a single copy of each duplicated gene being retained in both species. As a result genes A3 and B4 are termed as out-paralogs, as are genes A4 and B3. Genes A3 and B3 share an orthologous relationship as do A4 and B4

function [2]. Homologous genes that are related by a speciation event are termed orthologs whereas those related by gene duplication events are termed paralogs (Fig. 2) [3–6]. The functions of orthologous genes tend to be fairly conserved; therefore, high sequence similarity between a gene of unknown function and a detected ortholog will often indicate conserved function [7]. In contrast, paralogous genes that have undergone a duplication event may either retain different but related roles (subfunctionalization) or rapidly diverge and undertake new roles (neofunctionalization) [5, 8].

Because the identification of sequence similarity between two or more paralogous genes may not be indicative of conserved function, tools that compare sequences in this manner should be used with caution (*see* **Note 1**). It is recommended that the results generated by these tools should be augmented with additional information when formulating hypotheses concerning function. Many proteins contain functional units known as domains. In comparative analysis, a domain constitutes a region of conserved

sequence between different proteins. These may equate to functional units of proteins, and often encompass a hydrophobic core [9, 10]. Thus, domains can be thought of as the building blocks of protein functionality, and hence the possession or indeed lack of a protein domain, and the architecture of domains within a given protein will aid in the assessment of homology predictions and reduce the chances of incorrectly assigning function [11, 12].

## 2 Materials

Tools that are described in the analysis pipeline (*see* Subheading 3.2.5 for additional methods of interest) (Table 1).

**Table 1**
**Pipeline tools**

| | | |
|---|---|---|
| **General** | | |
| EBI Toolbox | http://www.ebi.ac.uk/services | Links to tools and analysis packages |
| ExPASy Server | http://www.expasy.org/ | Links to tools and analysis packages |
| COGs | http://www.ncbi.nlm.nih.gov/COG/ | Clusters of orthologous genes from multiple archaeal, bacterial, and eukaryotic genomes [7, 8] |
| Ensembl | www.ensembl.org | Whole genome annotation [13, 14] |
| UCSC Genome Browser | genome.cse.ucsc.edu | Whole genome annotation [15] |
| **Domain Identification Tools** | | |
| CDD | http://www.ncbi.nlm.nih.gov/Structure/cdd/cdd.shtml | Conserved domain database. Options to search Pfam SMART and COG databases [16, 17] |
| Interpro | http://www.ebi.ac.uk/interpro/search/sequence-search | Multiple databases of protein families and domains [18, 19] (Includes Pfam, SMART, PRINTS, Prosite, etc.) |
| Pfam | http://pfam.xfam.org/ | Library of protein domain HMMs [20, 21] |
| SMART | http://smart.embl-heidelberg.de/ | Library of protein domain HMMs [22, 23] |
| **Similarity tools** | | |
| FASTA | http://www.ebi.ac.uk/Tools/sss/fasta/ | Local alignment search tool [24] |
| NCBI-BLAST | http://blast.ncbi.nlm.nih.gov/Blast.cgi | Local alignment search tool at the NCBI [25, 26] |

## 3 Methods

In this chapter, we propose a pipeline that can be used to accurately categorize the functions of biological sequences.

It is often assumed that the obligatory first step when investigating an unknown sequence is to perform a BLAST or FASTA search. If a single high significance hit to a closely related species is detected (and the alignment extends to the full length of both sequences) then it may be safe to assume that a true ortholog has been detected. However, if a partial alignment is reported, or if the results indicate similarity to more than one protein, the output may be more difficult to interpret. We therefore recommended that users conduct domain searches prior to sequence alignments.

### 3.1 Analysis Pipeline Step 1: Domain Identification

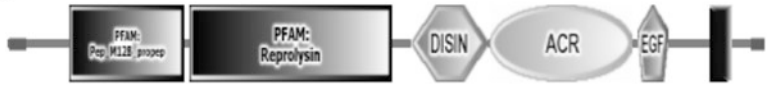
By definition, protein domains are conserved and although they can appear within genes in different combinations, they are rarely fragmented [10]. Traditionally members of domain families are compared using hidden Markov models (HMM) [27]. These HMMs predict the probability that specific residues will occur at each position within a domain based on the level of conservation across the domain family. This method has been widely used in investigations into gene families and in the annotation of whole genomes [28–31]. When developing hypotheses, a researcher should keep in mind that the presence or absence of proteins domains only partially defines protein function. Other biologically relevant caveats such as the co-occurrence of domains, domain–protein interactions, and protein localization (both cellular and subcellular) should be considered when formulating hypotheses [10, 12, 32].

#### 3.1.1 Tools for Identifying Protein Domains

Two tools that are widely used to compare query sequences to precomputed libraries of HMMs are Pfam [21, 33] and SMART [22, 23].

Pfam (release 30.0) contains profile-HMMs of 16,306 protein domains [21]. It links detected domains to a database describing their taxonomic abundance, potential evolutionary origins, and relationships (via Pfam clans) [34]. In contrast, the latest update to version 7.0 of the SMART database contains fewer HMMs (1,200 domains) but offers the option to include the Pfam HMM library within a search [22]. Like Pfam, the SMART database gives extensive details of function, evolution, and structure. It also provides links to relevant literature, information of proteins from 1133 completely sequenced genomes (choose "Use SMART in Genomic mode" at the start page), and highlights inactive domains if key functional residues differ between the query and target sequences. Both Pfam and SMART databases can be searched independently or via metasearch tools such as CDD [16, 17] or Interpro [18, 19].

a




| Name | Begin | End | E-value |
|---|---|---|---|
| Low complexity | 2 | 18 | - |
| Pfam:Pep_M12B_propep | 61 | 173 | 2.40e-49 |
| Pfam:Reprolysin | 178 | 375 | 4.70e-83 |
| DISIN | 393 | 468 | 1.50e-41 |
| ACR | 469 | 607 | 6.50e-53 |
| EGF | 613 | 643 | 2.80e+00 |
| Transmembrane | 683 | 705 | - |
| Low complexity | 722 | 735 | - |

b


| Domain | Start | End | Evalue |
|---|---|---|---|
| Pep_M12B_propep | 61 | 173 | 1.8e-50 |
| Reprolysin | 178 | 375 | 3.6e-84 |
| Disintegrin | 393 | 468 | 1.8e-37 |
| ADAM_CR | 470 | 587 | 2.3e-48 |

**Fig. 3** Domain detection by Pfam and SMART. Graphical and textual representations of domains detected in an ADAM 2 precursor from *Cavia porcellus* (swissprot accession Q60411). (**a**) Domains detected by SMART version 7.0 with the additional parameters, Pfam domains, signal peptides, and internal repeats selected [22, 23]. (**b**) Domains detected by Pfam version 27.0 [21, 33]. A global and fragment search was conducted with SEG low complexity filter on and an E-value cutoff = 1.0. Significant Pfam-B domains are not shown

Queries of Pfam and SMART with an ADAM 2 precursor from *Cavia porcellus* (swissprot accession Q60411) identify several domains with significant E-value (*see* Subheading 3.2.1 for a description of E-value statistics). Both tools indicate the positions and architecture of domains present within the query sequence as well as providing the user with information regarding domain function, co-occurrence, evolution, and residue conservation (Fig. 3). For example, SMART links the ACR (ADAM Cysteine-Rich Domain) to an Interpro abstract that informs the user of the function and domain architecture of ADAM proteins, while the evolution section displays the abundance of the ACR domains within the database (565 domains, 563 proteins, all metazoa). Similarly, the Pfam annotation of the Reprolysin domain provides information regarding domain function. Figure 3 highlights the

overlap of both these methods. Slight discrepancies are evident. These are reflective of the different length HMMs contained within each of the databases. Users are therefore encouraged to use multiple applications and consolidate results to achieve a consensus. Armed with this information, users can begin to build a hypothesis of sequence function (*see* **Note 2**).

### 3.2 Analysis Pipeline Step 2: Detection of Homologs

Sequence comparison tools aim to accurately infer homology from truly related biological sequences. Nucleotide alignment algorithms only look for direct similarities between sequences in base space and do not account for factors such as the class of amino acid or the relative abundance of amino acid types. Protein alignment algorithms integrate these factors to improve the accuracy of alignments based on the likelihood of amino acid substitutions. For example, a conservative substitution; such as an isoleucine for a valine (both possess aliphatic R groups) would be more heavily weighted than a substitution of rare amino acids such as tryptophan or cysteine. A number of schemes have been developed to weight all of the possible amino acid substitutions as matrices. The most commonly used examples are PAM (percent accepted mutation) [35] and BLOSUM (Blocks Substitution matrix) [36]. PAM matrices are based on an evolutionary model of point acceptable mutations per million years whereas BLOSUM matrices are based on empirical datasets of aligned sequences. The suffix of a BLOSUM matrix denotes the maximum percentage similarity of the alignment. Thus, the scores in BLOSUM45 and BLOSUM80 are generated from sequences of >45 % and >80 % similarity, respectively (*see* **Note 3**). Equipped with these substitution matrices, various algorithms are available to align sequences in such a way so as to maximize the overall alignment score. Algorithms that produce a guaranteed optimal local alignment include the Smith–Waterman algorithm [37]. Due to their computational requirements such methods are often impractical for large datasets. To accelerate identification of the most significant alignments, heuristic algorithms such as BLAST [25, 38] and FASTA [24] have been developed.

#### 3.2.1 Detection of Homologs by BLAST

The most widely used sequence comparison tool is BLAST [25, 38] and the NCBI version of BLAST is probably the most commonly used variation. It can be used online via the NCBI web interface (blast.ncbi.nlm.gov/Blast.cgi) or downloaded and run locally as a stand-alone tool (BLAST+). This tutorial focuses on the NCBI web interface for a more detailed description of BLAST+ see [39]. Many of the nuances of BLAST and detailed descriptions of the statistics will not be discussed here but are covered in detail elsewhere [25, 26, 40–44]. A particularly thorough explanation is given in [39]. Also

refer to the BLAST Help pages (http://blast.ncbi.nlm.nih.gov/Blast.cgi?CMD=Web&PAGE_TYPE=BlastDocs), tutorials (http://www.ncbi.nlm.nih.gov/books/NBK1734/), and *see* **Note 4**.

The basic options required for a BLAST search are a query sequence, a database to search, the type of search, and the search parameters. The query sequence can be entered either as plain text, a valid NCBI sequence identifier, or as a fasta formatted sequence file where the first line (containing identifier information) is demarked by an initial greater than symbol (>) followed by the sequence on subsequent lines. It is good practice to create fasta formatted sequence files as the identifiers are reported in the BLAST output, which helps when tracking multiple search results. The database searched will relate to the hypothesis of the user's experiment and may have implications for the test statistics (*see* **Note 5**). NCBI-blast has access to 7 protein and 16 nucleotide databases. For an initial search when identifying potential homologs it is best practice to search one of the nr databases. These contain nonredundant (nonidentical) entries from Gen-Bank translations, RefSeq Proteins, PDB, SwissProt, PIR, and PRF databases. If a species of interest is known, then the database can be filtered by organism using a taxon id code (http://www.ncbi.nlm.nih.gov/taxonomy/) or from predefined taxonomic groups, for example, primate or eukaryote.

It is recommended that protein or translated nucleotide sequences are used when conducting searches to infer function. If a protein sequence is available it is best to search in protein space using blastp. If a DNA sequence is available it is best to ignore blastn (which searches in nucleotide space) and use either blastx or tblastx. In the program selection section one can observe that multiple BLAST algorithms are available, these are described in Table 2. Additional search settings are shown in the "Algorithm parameters" tab. Of specific note are the expect score (E-value) and the low complexity filter. The E-value is the statistical significance threshold for reporting matches against the database. The default value is 10. This indicates that for each alignment ten similar matches are expected to be found merely by chance. The lower the E-value the more significant the alignment [25]. For example, a E-value of $1 \times 10^{-3}$ indicates that the likelihood of a match occurring by chance is 1 in 1000 [45]. In the filters and masking section there is a filter to exclude low complexity regions. As these are likely to result in alignments of statistical but not biological significance, unless the user is confident in their hypothesis, these should always be turned on. The default filtering algorithms are SEG masking for protein searches [46] and DUST (Tatusov and Lipman, unpublished) for translated nucleotide searches. Other parameters that can be modified in the Algorithm parameters section include the word size (the number of matching residues required to seed an alignment extension algorithm) and the scoring parameters (the

**Table 2**
**Search parameters and common uses of NCBI-BLAST variants**

| Program | Query | Database | Search type | Algorithms | Common uses |
|---|---|---|---|---|---|
| blastn | DNA | DNA | DNA-DNA | Highly similar, more dissimilar, somewhat similar | Search for near identical DNA sequences, confirmation of DNA sequencing experiment. Compare query to genomic DNA to identify splicing patterns |
| blastp | Protein | Protein | Protein-Protein | blastp, PSI-BLAST, PHI-BLAST, DELTA-BLAST | Search for homologous protein sequences. Annotation of genes of unknown function. Searches can be direct (blastp), use profile models (PSI-BLAST, PHI-BLAST), or database-assisted profile models (DELTA-BLAST) to improve sensitivity |
| blastx | Protein | DNA | Translated DNA-Protein | N/A | Gene finding within DNA sequences |
| tblastn | DNA | DNA | Translated DNA–Translated DNA | N/A | Identify protein coding structures in DNA sequences |

reward and penalty for matches, mismatches, and gaps in the alignment).

For our search we will keep these parameters at their default settings. Clicking the "BLAST" button will submit your search. A new status window will open that automatically updates until the search is complete and the results page appears. The blast results page consists of a header containing information of query and database searched, a graphical representation of the alignments, a summary of each significant hit, and a footer containing details of the search statistics. The graphical summary displays the significant hits (colored according to the degree of similarity) to the query sequence (at the top of the graphic). This view gives the user ready information regarding the region(s) of the query sequence that produce significant hits. The one-line output summary ranks each hit by E-value. Each hit is hyperlinked to a corresponding entrez database entry containing links to associated genes, structures, taxonomy, and publications. Scrolling down or clicking on an individual score will show the alignments. Each aligned region (known as a high scoring segment pair or hsp) has a header with gene identifier and score summary. The bit score "Score = x bits" is determined from the raw scores of the alignment as defined in the substitution matrix. From this the E-value "Expect = x" is

calculated. The alignment section also visualizes the hsp to show the specific similarities and differences between the query and subject sequences. Sandwiched between these sequences, identical matches are highlighted by corresponding amino acids and conserved matches by a plus sign. This section of the page also links to additional resources including gene information, Map Viewer (for genomic localization), and lists of known identical homologs.

### 3.2.2 Assessing the Results of a BLAST Search

Confidence can be placed in an homology assignment if all relevant domains are conserved, form the same architecture and key residues known to be important for function are shared in the correct spatial context. Thus, hsps should always be critically assessed using information determined from domain searches. When viewing an alignment, users should pose the questions; "do the start and end positions of the hsp correspond to a predicted domain?" If so, "do aligned residues correspond to critical residues described in the domain annotation?" If conserved functional residues are not aligned then caution should be exercised in assigning function. Alignments should also be checked for residue bias that has escaped the low-complexity filters. Certain proteins, for example, myosins (which are rich in lysine and glutamic acid) have inherent compositional biases that can affect alignment scores. When investigating such sequences users should assess corresponding residues to check whether the significance of the alignment is due to both protein types sharing a common bias rather than a common function.

### 3.2.3 Detection of More Distant Homologs

The BLAST method identifies homologs by comparing sequences directly. As a result it will undoubtedly miss align homologs with more divergent ancestry where greater sequence change is expected. In such cases, more powerful methods are required. When viewing the output of alignment, it can be observed that some regions are highly conserved whereas others accept greater numbers of substitutions. This information can be translated into profile-sequence models and used to guide alignments.

Profile-sequence models are weighted based upon the number of expected matches and mismatches within a given region of sequence. As a result mismatches in conserved areas are penalized to a greater extent compared with mismatches in regions of high variability. NCBI offers two profile-sequence model methods for predicting more distant homologs, PSI-BLAST (position-specific iterated BLAST) and DELTA-BLAST (Domain enhanced lookup time accelerated BLAST). Both methods utilize position-specific score matrix (PSSM) profile-sequence models [26, 40, 42]. A PSSM is an $L \times 20$ amino acid matrix of scores where $L$ is the length of the query sequence and 20 represents each possible amino acid. Each position is subsequently weighted according to its conservation within the multiple alignment. Conserved positions are

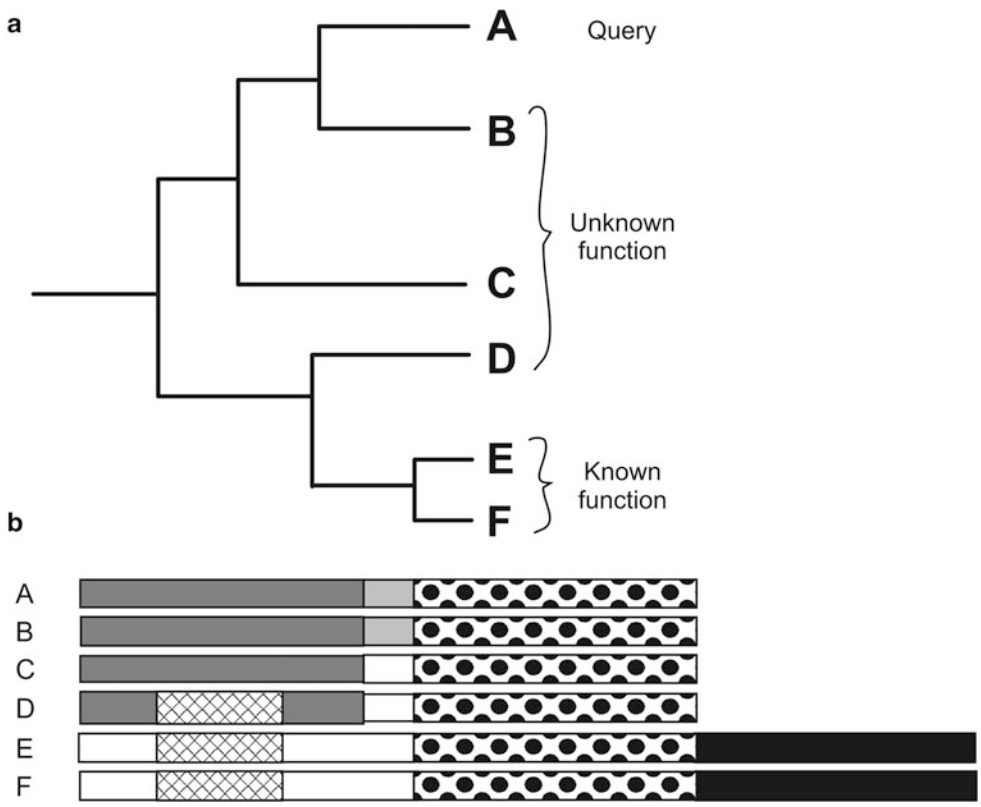



**Fig. 4** Detection of homologs by PSI-BLAST and DELTA-BLAST. (**a**) Hypothetical, sequences A–F are distantly related homologs. Their unknown relationship and similarity are represented by distance on the stylized phylogenetic tree. Initiating a standard gapped BLAST [25, 26] search with sequence of unknown function A would result in identification of similar sequences B and C. If no other sequences were identified we have no functional information with which to annotate sequence A. However, if a PSSM approach is used the additive information of sequences A, B, and C will allow for the detection of sequence D and subsequently functionally annotated sequences E and F in later iterations of the algorithm. The BLAST methodology means that if sequences A and D are homologous as are sequences D and E, it follows that A and E must also be homologous allowing annotation of the initial query sequence. (**b**) Schematic representation of an alignment of sequences A–F. Aligned domains share a color, whereas unaligned regions are represented by open boxes. To correctly annotate a gene with functional information, the alignments described must occur in the same region of the alignment. Therefore, while sequences E and F are related by the presence of the solid black domain, its function may not be reflected in sequences A–D as these sequences do not contain this domain

assigned high scores whereas regions of high variability are assigned scores close to zero [47, 48]. The basis of the PSSM approach is outlined in more detail in Fig. 4.

PSI-BLAST first compares the query sequence to a defined database using the standard gapped BLAST algorithm [26, 43]. From this initial search, significant matches (the NCBI default E-value is 0.005) are selected and a multiple sequence alignment generated (matches identical to the query sequence or >98 % identical to another match are purged to avoid redundancy). A

PSSM is generated and used to seed a new alignment. Any significant hits are added to the multiple sequence alignment and the process is repeated in an iterative manner until convergence occurs (no new sequences with significance below the E-value cutoff are detected) (*see* **Note 6**). Using PSSM's in this way results in a wider search of the sequence space, improves sensitivity, and incorporates more distant homologs. In contrast, DELTA-BLAST utilizes a precomputed database, the NCBI Conserved Domain Database (CDD), to guide the initial PSSM model [42]. This resource was developed to identify conserved domains within protein sequences and includes manually curated domain models (which have been refined using protein 3D structures), as well as models constructed from clusters of related sequences. After the initial alignment step DELTA-BLAST proceeds using the same iterative PSSM model as PSI-BLAST.

#### *3.2.4 Assessing the Results of PSI-BLAST and DELTA-BLAST Searches*

The user should be aware that the primary concern for false prediction of homology by PSI-BLAST is inclusion of a nonhomologous sequence into the PSSM, which can be particularly problematic if the profile is compositionally biased. For example, if a profile model includes a protein domain common to several of the target sequences but not shared with the query then the model may be incorrectly enriching for that domain. By seeding the PSSM model with domains known to be related to the query sequence DELTA-BLAST reduces the likelihood of these compound errors but instead can be subject to database bias if the query sequence contains predominantly uncommon domains. Therefore, as with standard BLAST searches, the user should exhibit caution when interpreting the results [26, 41, 43, 44, 49]. In both cases, the incorporation of a nonhomologous sequence can lead to the identification and subsequent profile inclusion of sequences with high similarity to the erroneous sequence rather than to the query sequence. As with any BLAST search the alignment should be inspected carefully. Due to the iterative nature of these methods, any sequences included when they should not be, usually leads to an amplification of problems that may go unnoticed if the user is not vigilant. The user should look for a similar conservation pattern in each of the alignments. If a sequence seems to be included erroneously, it can be excluded from the PSSM and subsequent searches by unchecking the relevant radio button in the BLAST output. If the sequence returns in later iterations seek corroboration of the finding by other means such as reciprocal searching (*see* **Note 7**).

#### *3.2.5 Additional Methods of Homolog Detection*

There are several other available methods that employ profile or HMM sequence comparison or combine multiple methods to infer function. Interested users should investigate these as they

potentially offer greater sensitivity for detection of distant homologs [2, 50].

CombFunc: (http://www.sbg.bio.ic.ac.uk/~mwass/combfunc/) [51]. The CombFunc webserver employs multiple approaches to determine function including BLAST-based sequence similarity, protein fold prediction, gene ontology, protein–protein interaction, and gene co-expression data. Support for function is determined by combining these results using a support vector machine learning approach.

FFPRED: (http://bioinf.cs.ucl.ac.uk/) [52]. The FFPRED server is a powerful tool that aims to assign gene ontology [53] biological process and molecular function terms to difficult to annotate sequences based on the characteristics of the searched amino acid sequence. The FFPRED server performed very well in this task during the recent critical assessment of protein function annotation (CAFA) experiment [52].

Blocks: (http://blocks.fhcrc.org/) [54, 55]. Blocks utilizes a database of ungapped multiple alignments that correspond to the highly conserved regions of proteins. Query sequences can be compared to the Blocks database via the block searcher tool, IMPALA (comparison of query to database of PSSMs) [56] and LAMA (comparison of multiple alignment to Blocks using profile: profile method) [57].

COMPASS: http://prodata.swmed.edu/compass/compass.php [58–61]. COMPASS generates statistical comparisons of multiple protein alignments via profile generation.

HH-pred: (http://toolkit.tuebingen.mpg.de/hhpred) [62, 63]. HH-pred uses the HHsearch algorithm to search protein and domain databases by pairwise alignment of profile-HMMs. Alignment incorporates predicted structural information and can generate an HMM from a single submitted query sequence by automated PSI-BLAST search.

Hmmer: (http://hmmer.janelia.org/) [27, 64]. Hmmer uses a Profile-HMM method of sequence comparison. Tools include: hmmbuild (HMM construction based on a multiple sequence alignment), hmmalign (align sequence(s) to an existing HMM), and hmmsearch (search a database of sequences for a significant match to an HMM).

### 3.3 Analysis Pipeline Step 3: Genomic Sequence Comparison

Recent advances in sequencing and genome annotation have led to the generation of datasets that can provide users with vast amounts of precomputed and cataloged information. Linking of a query sequence to a gene in these databases allows rapid access to functional annotation including predicted orthologs and paralogs, gene structure, gene expression, splice variation, association with disease, chromosomal location, and gene polymorphism data. Thus, inferring homology (either using keyword or similarity searches as

described previously) to a gene from a multigenome resource as those described in this section should be a final step in the analysis pipeline. The annotation of a gene in this way may corroborate findings determined during previous steps and may offer additional data to reinforce a working hypothesis. These databases also have an advantage in that they are regularly updated with improved gene predictions and annotations. The resource used will depend on the organism from which the query sequence was obtained (*see* **Note 7**). Although some data overlap is inevitable, users are encouraged to try each tool to survey the breadth of information available (*see* **Note 8**).

For many eukaryotic organisms the UCSC genome browser [15] and Ensembl genome server [13, 14] are ideal sources of information. Both include a wealth of annotation data for the genomes of multiple organisms, and direct links between the two tools are provided. The contents of these databases are regularly updated and reflect the current trend for whole-genome sequencing of biologically relevant model organisms and increasingly organisms of interest for comparative evolutionary analysis [31]. Searching of these databases can be via a gene identifier or by a similarity query (BLAST at Ensembl and BLAT at UCSC genome browser) [65].

In addition, the COGs database housed at the NCBI contains clusters of orthologous genes (COGs) that are typically associated with a specific and defined function [7, 66]. Although primarily a tool for comparison of prokaryotes and unicellular eukaryotes, the COG database also includes many eukaryotic genomes [67]. Of particular use is the interlinking of the COG database with the other databases at the NCBI [38], allowing direct links from a BLAST hit to a predicted COG via the Blast Link (BLink) tool.

### 3.4 Conclusion

The prediction of function from sequence data alone is a complex procedure. Appropriate prior information regarding data such as the tissue or developmental stage from which sequences were collected should be added to working hypotheses as analysis is conducted. It should also be remembered that predictive tools, although based on robust algorithms, can sometimes produce inconsistent or incorrect results. Therefore, the experimenter should look for the convergence between multiple methods to improve confidence in prediction and seek experimental verification where possible.

## 4 Notes

1. In describing any predicted relationship it must be remembered that the terms similarity and homology are not interchangeable. Often sequences are described as containing n percent similarity. It is not, however, correct to use the term percent homologous;

genes either are homologous (implying an ancestral relationship) or they are not [3].

2. When conducting domain analysis, note the positions of domains detected and conduct some background research of the essential residues and predicted function of these domains. Record the probability associated with any domain prediction and the database version searched.
3. To specifically identify recent homologs or in-paralogs, search an appropriate database with a shallow scoring matrix (BLOSUM80, PAM20) that will have a shorter look-back time, thus biasing toward more recent homologs.
4. Only use PSI-BLAST or DELTA-BLAST if you are attempting to identify distant homologs of unusual sequences. If an abundant domain known to be present in many different protein types (e.g., zf-C2H2 Zinc fingers, of which there are thousands known within any given species), consider masking this region before running a BLAST search to avoid detection of an excess of hits that provide little additional predictive information. If a representative structural sequence is available, comparison of a query sequence to the protein data bank PDB (http://www.rcsb.org/pdb/) can help in identifying structural and functional conserved residues. Increasing the gap penalty may decrease the significance of unrelated sequences, improving the signal-to-noise ratio for a true hit but at a cost of missing true homologs.
5. E-value scores are correlated to database size. Therefore, choosing which database to search will affect the significance or interpretation of results obtained. For example, to identify an ortholog in bacterial genomes, searching a database of only bacterial sequences will reduce the search space and improve the significance of an E-value for a given alignment. In relation to search database size, searching the large numbers of near identical sequences held in the nr database could potentially result in missing a true homolog with threshold significance. Alternatively, a significant hit close to the threshold when searching a small database should be checked carefully and corroborated by other methods to avoid false-positives.
6. The relevant E-value for a hit sequence is the value when it is first identified, not at convergence or at completion of a set number of iterations. This is because inclusion of a sequence refines the PSSM for subsequent searches and will lead to greater significance of that sequence in subsequent iterations.
7. If the organism from which the query sequence was obtained is not currently available, compare to taxonomically related organisms to build a working hypothesis of a similar function for the query gene.

8. Why is there no significant hit when I BLAST the genome of the same or closely related organism? Many methods of whole genome sequencing utilize a process of fragmentation, sequencing, and computational reassembly of genomic DNA (Whole Genome Shotgun sequencing). Depending on the depth of coverage and the heterozygosity of the genomic DNA, this approach will result in varying degrees of incomplete noncontiguous sequences. Genes apparently missing from the genome may be located in these gaps or in repetitive hard-to-sequence regions of the genome. An alternative possibility is that the gene prediction tools used to annotate the genome and predict genes may have not predicted the query gene correctly.

## References


1. Doolittle RF (1981) Similar amino acid sequences: chance or common ancestry? Science 214(4517):149–159
2. Pearson WR, Sierk ML (2005) The limits of protein sequence comparison? Curr Opin Struct Biol 15(3):254–260
3. Fitch WM (2000) Homology a personal view on some of the problems. Trends Genet 16 (5):227–231
4. Henikoff S, Greene EA, Pietrokovski S, Bork P, Attwood TK, Hood L (1997) Gene families: the taxonomy of protein paralogs and chimeras. Science 278(5338):609–614
5. Sonnhammer EL, Koonin EV (2002) Orthology, paralogy and proposed classification for paralog subtypes. Trends Genet 18 (12):619–620
6. Weber MJ (2005) New human and mouse microRNA genes found by homology search. FEBS J 272(1):59–73
7. Tatusov RL, Galperin MY, Natale DA, Koonin EV (2000) The COG database: a tool for genome-scale analysis of protein functions and evolution. Nucleic Acids Res 28(1):33–36
8. Hurles M (2004) Gene duplication: the genomic trade in spare parts. PLoS Biol 2(7):E206
9. Bateman A (1997) The structure of a domain common to archaebacteria and the homocystinuria disease protein. Trends Biochem Sci 22 (1):12–13
10. Ponting CP, Russell RR (2002) The natural history of protein domains. Annu Rev Biophys Biomol Struct 31:45–71
11. Ponting CP (2001) Issues in predicting protein function from sequence. Brief Bioinform 2 (1):19–29
12. Ponting CP, Dickens NJ (2001) Genome cartography through domain annotation. Genome Biol 2(7), Comment 2006
13. Flicek P, Ahmed I, Amode MR, Barrell D, Beal K, Brent S, Carvalho-Silva D, Clapham P, Coates G, Fairley S et al (2013) Ensembl 2013. Nucleic Acids Res 41(Database issue): D48–D55
14. Hubbard T, Barker D, Birney E, Cameron G, Chen Y, Clark L, Cox T, Cuff J, Curwen V, Down T et al (2002) The Ensembl genome database project. Nucleic Acids Res 30 (1):38–41
15. Meyer LR, Zweig AS, Hinrichs AS, Karolchik D, Kuhn RM, Wong M, Sloan CA, Rosenbloom KR, Roe G, Rhead B et al (2013) The UCSC Genome Browser database: extensions and updates 2013. Nucleic Acids Res 41(Database issue):D64–D69
16. Marchler-Bauer A, Panchenko AR, Shoemaker BA, Thiessen PA, Geer LY, Bryant SH (2002) CDD: a database of conserved domain alignments with links to domain three-dimensional structure. Nucleic Acids Res 30(1):281–283
17. Marchler-Bauer A, Zheng C, Chitsaz F, Derbyshire MK, Geer LY, Geer RC, Gonzales NR, Gwadz M, Hurwitz DI, Lanczycki CJ, Lu F, Lu S, Marchler GH, Song JS, Thanki N, Yamashita RA, Zhang D, Bryant SH (2013) CDD: conserved domains and protein three-dimensional structure. Nucleic Acids Res 41 (Database issue):D348–D352
18. Apweiler R, Attwood TK, Bairoch A, Bateman A, Birney E, Biswas M, Bucher P, Cerutti L, Corpet F, Croning MD et al (2001) The InterPro database, an integrated documentation resource for protein families, domains and functional sites. Nucleic Acids Res 29 (1):37–40
19. Jones P, Binns D, Chang HY, Fraser M, Li W, McAnulla C, McWilliam H, Maslen J, Mitchell A, Nuka G, Pesseat S, Quinn AF, Sangrador-Vegas A, Scheremetjew M, Yong SY, Lopez R,

Hunter S (2014) InterProScan 5: genome-scale protein function classification. Bioinformatics 30(9):1236–1240

20. Bateman A, Birney E, Durbin R, Eddy SR, Finn RD, Sonnhammer EL (1999) Pfam 3.1: 1313 multiple alignments and profile HMMs match the majority of proteins. Nucleic Acids Res 27(1):260–262
21. Finn RD, Bateman A, Clements J, Coggill P, Eberhardt RY, Eddy SR, Heger A, Hetherington K, Holm L, Mistry J, Sonnhammer EL, Tate J, Punta M (2014) Pfam: the protein families database. Nucleic Acids Res 42(Database issue):D222–D230
22. Letunic I, Doerks T, Bork P (2012) SMART 7: recent updates to the protein domain annotation resource. Nucleic Acids Res 40(Database issue):D302–D305
23. Schultz J, Milpetz F, Bork P, Ponting CP (1998) SMART, a simple modular architecture research tool: identification of signaling domains. Proc Natl Acad Sci U S A 95 (11):5857–5864
24. Pearson WR, Lipman DJ (1988) Improved tools for biological sequence comparison. Proc Natl Acad Sci U S A 85(8):2444–2448
25. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ (1990) Basic local alignment search tool. J Mol Biol 215(3):403–410
26. Altschul SF, Madden TL, Schaffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res 25(17):3389–3402
27. Eddy SR (1998) Profile hidden Markov models. Bioinformatics 14(9):755–763
28. Gibbs RA, Weinstock GM, Metzker ML, Muzny DM, Sodergren EJ, Scherer S, Scott G, Steffen D, Worley KC, Burch PE et al (2004) Genome sequence of the Brown Norway rat yields insights into mammalian evolution. Nature 428(6982):493–521
29. Lander ES, Linton LM, Birren B, Nusbaum C, Zody MC, Baldwin J, Devon K, Dewar K, Doyle M, FitzHugh W et al (2001) Initial sequencing and analysis of the human genome. Nature 409(6822):860–921
30. Ellsworth RE, Jamison DC, Touchman JW, Chissoe SL, Braden Maduro VV, Bouffard GG, Dietrich NL, Beckstrom-Sternberg SM, Iyer LM, Weintraub LA et al (2000) Comparative genomic sequence analysis of the human and mouse cystic fibrosis transmembrane conductance regulator genes. Proc Natl Acad Sci U S A 97(3):1172–1177
31. Emes RD, Goodstadt L, Winter EE, Ponting CP (2003) Comparison of the genomes of human and mouse lays the foundation of genome zoology. Hum Mol Genet 12 (7):701–709
32. Schultz J, Copley RR, Doerks T, Ponting CP, Bork P (2000) SMART: a web-based tool for the study of genetically mobile domains. Nucleic Acids Res 28(1):231–234
33. Sonnhammer EL, Eddy SR, Birney E, Bateman A, Durbin R (1998) Pfam: multiple sequence alignments and HMM-profiles of protein domains. Nucleic Acids Res 26(1):320–322
34. Finn RD, Mistry J, Schuster-Bockler B, Griffiths-Jones S, Hollich V, Lassmann T, Moxon S, Marshall M, Khanna A, Durbin R, Eddy SR, Sonnhammer EL, Bateman A (2006) Pfam: clans, web tools and services. Nucleic Acids Res 34(Database issue):D247–D251
35. Henikoff S, Henikoff JG (1993) Performance evaluation of amino acid substitution matrices. Proteins 17(1):49–61
36. Henikoff S, Henikoff JG (1992) Amino acid substitution matrices from protein blocks. Proc Natl Acad Sci U S A 89(22):10915–10919
37. Smith TF, Waterman MS (1981) Identification of common molecular subsequences. J Mol Biol 147(1):195–197
38. Wheeler DL, Barrett T, Benson DA, Bryant SH, Canese K, Chetvernin V, Church DM, Dicuccio M, Edgar R, Federhen S et al (2008) Database resources of the National Center for Biotechnology Information. Nucleic Acids Res 36(Database issue): D13–D21
39. Pearson WR (2014) BLAST and FASTA similarity searching for multiple sequence alignment. Methods Mol Biol 1079:75–101
40. Altschul SF, Gertz EM, Agarwala R, Schaffer AA, Yu YK (2009) PSI-BLAST pseudocounts and the minimum description length principle. Nucleic Acids Res 37(3):815–824
41. Altschul SF, Koonin EV (1998) Iterated profile searches with PSI-BLAST—a tool for discovery in protein databases. Trends Biochem Sci 23 (11):444–447
42. Boratyn GM, Schaffer AA, Agarwala R, Altschul SF, Lipman DJ, Madden TL (2012) Domain enhanced lookup time accelerated BLAST. Biol Direct 7:12
43. Jones DT, Swindells MB (2002) Getting the most from PSI-BLAST. Trends Biochem Sci 27 (3):161–164
44. Korf I (2003) Serial BLAST searching. Bioinformatics 19(12):1492–1496
45. Altschul SF, Bundschuh R, Olsen R, Hwa T (2001) The estimation of statistical parameters for local alignment score distributions. Nucleic Acids Res 29(2):351–361

46. Wootton JC, Federhen S (1996) Analysis of compositionally biased regions in sequence databases. Methods Enzymol 266:554–571
47. Altschul SF, Gish W (1996) Local alignment statistics. Methods Enzymol 266:460–480
48. Henikoff S (1996) Scores for sequence searches and alignments. Curr Opin Struct Biol 6(3):353–360
49. Schaffer AA, Aravind L, Madden TL, Shavirin S, Spouge JL, Wolf YI, Koonin EV, Altschul SF (2001) Improving the accuracy of PSI-BLAST protein database searches with composition-based statistics and other refinements. Nucleic Acids Res 29(14):2994–3005
50. Sierk ML, Pearson WR (2004) Sensitivity and selectivity in protein structure comparison. Protein Sci 13(3):773–785
51. Wass MN, Barton G, Sternberg MJ (2012) CombFunc: predicting protein function using heterogeneous data sources. Nucleic Acids Res 40(Web Server issue):W466–W470
52. Minneci F, Piovesan D, Cozzetto D, Jones DT (2013) FFPred 2.0: improved homology-independent prediction of gene ontology terms for eukaryotic protein sequences. PLoS One 8(5):e63754
53. Ashburner M, Ball CA, Blake JA, Botstein D, Butler H, Cherry JM, Davis AP, Dolinski K, Dwight SS, Eppig JT et al (2000) Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. Nat Genet 25 (1):25–29
54. Henikoff S, Pietrokovski S, Henikoff JG (1998) Superior performance in protein homology detection with the Blocks Database servers. Nucleic Acids Res 26(1):309–312
55. Henikoff JG, Pietrokovski S, McCallum CM, Henikoff S (2000) Blocks-based methods for detecting protein homology. Electrophoresis 21(9):1700–1706
56. Schaffer AA, Wolf YI, Ponting CP, Koonin EV, Aravind L, Altschul SF (1999) IMPALA: matching a protein sequence against a collection of PSI-BLAST-constructed position-specific score matrices. Bioinformatics 15 (12):1000–1011
57. Pietrokovski S (1996) Searching databases of conserved sequence regions by aligning protein multiple-alignments. Nucleic Acids Res 24 (19):3836–3845
58. Sadreyev R, Grishin N (2003) COMPASS: a tool for comparison of multiple protein alignments with assessment of statistical significance. J Mol Biol 326(1):317–336
59. Sadreyev RI, Grishin NV (2004) Quality of alignment comparison by COMPASS improves with inclusion of diverse confident homologs. Bioinformatics 20(6):818–828
60. Sadreyev RI, Tang M, Kim BH, Grishin NV (2007) COMPASS server for remote homology inference. Nucleic Acids Res 35(Web Server issue):W653–W658
61. Sadreyev RI, Tang M, Kim BH, Grishin NV (2009) COMPASS server for homology detection: improved statistical accuracy, speed and functionality. Nucleic Acids Res 37(Web Server issue):W90–W94
62. Soding J, Biegert A, Lupas AN (2005) The HHpred interactive server for protein homology detection and structure prediction. Nucleic Acids Res 33(Web Server issue):W244–W248
63. Hildebrand A, Remmert M, Biegert A, Soding J (2009) Fast and accurate automatic structure prediction with HHpred. Proteins 77(Suppl 9):128–132
64. Eddy SR (2011) Accelerated profile HMM searches. PLoS Comput Biol 7(10):e1002195
65. Kent WJ (2002) BLAT—the BLAST-like alignment tool. Genome Res 12(4):656–664
66. Marchler-Bauer A, Anderson JB, Derbyshire MK, DeWeese-Scott C, Gonzales NR, Gwadz M, Hao L, He S, Hurwitz DI, Jackson JD et al (2007) CDD: a conserved domain database for interactive domain family analysis. Nucleic Acids Res 35(Database issue):D237–D240
67. Wheeler DL, Church DM, Federhen S, Lash AE, Madden TL, Pontius JU, Schuler GD, Schriml LM, Sequeira E, Tatusova TA, Wagner L (2003) Database resources of the National Center for Biotechnology. Nucleic Acids Res 31(1):28–33

# Chapter 3

# Inferring Functional Relationships from Conservation of Gene Order

Gabriel Moreno-Hagelsieb

## Abstract

Predicting functional associations using the Gene Neighbor Method depends on the simple idea that if genes are conserved next to each other in evolutionarily distant prokaryotes they might belong to a polycistronic transcription unit. The procedure presented in this chapter starts with the organization of the genes within genomes into pairs of adjacent genes. Then, the pairs of adjacent genes in a genome of interest are mapped to their corresponding orthologs in other, informative, genomes. The final step is to verify if the mapped orthologs are also pairs of adjacent genes in the informative genomes.

**Key words** Conservation of gene order, Operon, Genomic context, Functional inference, Gene neighbor method

## 1 Introduction

Two independent works first presented data supporting the idea that genes conserved together in evolutionarily distant genomes might have functional associations [1, 2]. However, the first thoroughly described method to infer functional relationships using conservation of gene order might be the work published by Overbeek et al. [3]. Nowadays, the method is part of the RAST/MG-RAST system of functional annotations [4, 5]. The method finds support in three main ideas (Fig. 1): (1) the knowledge that genes in operons, stretches of adjacent genes in the same DNA strand transcribed into a single mRNA [6, 7], are functionally related; (2) by the expectation that operons should be conserved throughout evolution; and (3) by the finding that gene order in general is not conserved [8], and is lost much faster than protein sequence identity [9]. Thus, conservation of gene order at long evolutionary distances might indicate a functional relationship. Some divergently transcribed genes (Fig. 1) might also be functionally related (*see*, for instance [10–12]). However, Overbeek et al. [3] found that the conservation of divergently transcribed genes in evolutionarily

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_3, © Springer Science+Business Media New York 2017

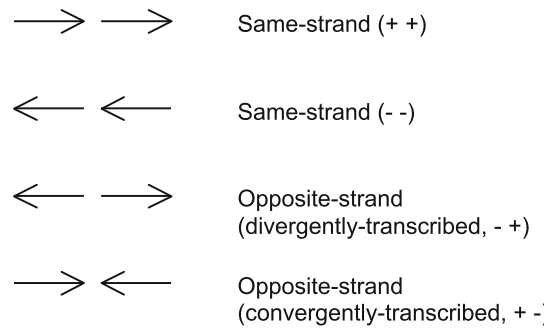



**Fig. 1** Pairs of genes used in the study of conservation of gene order. Genes are represented by *arrows* indicating the direction of transcription. Same-strand genes would be the ones that might be in operons, and thus functionally related. Genes in opposite strands can be either divergently transcribed or convergently transcribed. Comparing the conservation of gene order of genes in the same strand against that of genes in opposite strands helps calculate a confidence value for predictions of functional interactions

distant genomes was minimal compared to the conservation of genes in the same strand. Thus, they limited their analyses to the detection of conservation of adjacency of genes in the same DNA strand. Given that the significance of conservation of adjacency increases with the phylogenetic distance of the genomes compared, Overbeek et al. [3] directly used phylogenetic distances as a score. However, selecting an appropriate threshold was a problem. Ermolaeva et al. [13] proposed the conservation of adjacency of genes in opposite strands as a representative of background conservation useful to calculate a confidence value for the conservation of adjacent genes in the same strand. An approach using a simplified method similar to that presented by Ermolaeva et al. [13] was used to show that conservation of adjacency of paralogous genes is also useful for predicting operons and functional relationships of gene products [14].

Another approach to conservation of gene order is to count the number of genomes in which a given pair of genes are conserved next to each other (*see*, for instance [15, 16]). The main problem of such an approach is that conservation of adjacency in very closely related genomes is not as informative as that among evolutionarily distant genomes. A later approach uses phylogenetic relationships to correct for this problem [17]. I present here the simplified method that uses adjacent genes in opposite strands for calibration mentioned earlier [14]. The confidence values obtained in this method provide a direct measure of significance that is very easy to understand. Moreover, there are no results yet showing that accounting for the number of genomes in which the genes are conserved produces any better results than conservation in evolutionarily distant genomes.

## 2 Systems, Software, and Databases

### 2.1 A UNIX-Based Operative System

I am assuming that the user is working with a UNIX-based operating system. The prevalent UNIX-based systems today include Mac OSX, and many other systems based on Linux, like Ubuntu and Debian. Specialized computer system servers and workstations tend to also run under a UNIX-based operative system.

### 2.2 The RefSeq Bacterial Genome Database

The RefSeq database [18, 19] contains files with diverse information about each genome. The database can be downloaded using programs such as "wget" or "rsync" (*see* **Note 1**). For instance, periodically running the command:

```
rsync -av rsync://rsync.ncbi.nlm.nih.gov/genomes/
   refseq/Bacteria/\
LOCAL_GENOMES --delete
```

would keep an updated directory "LOCAL_GENOMES" with all the information in the directory "/genomes/refseq/Bacteria" of the NCBI rsync server (*see* **Note 2**). Here I will be using three files under each genome directory, those ending with ".gbk," with ".ptt" and with ".rnt" (from now on called GBK, PTT, and RNT files). Though the GBK file generally contains all of the necessary information, the PTT and RNT files are more programmer friendly.

### 2.3 NCBI's BLAST+

To map the corresponding orthologous [homologous] genes it is necessary to compare proteins encoded by the genes within all genomes. Identifying orthologs is important for any genome context method for inferring functional associations [9]. Appropriate binaries of NCBI's BLAST+ [20] program suite can be downloaded from NCBI's servers using rsync at: rsync://rsync.ncbi.nlm.nih.gov/blast/executables/LATEST/.

## 3 Methods

The method starts with the construction of files or databases of gene neighbors. For each gene in a given pair of neighbors in the genome of interest, the method verifies the existence of orthologs in an informative genome. If the orthologs exist, their adjacency is investigated. The following pseudocode summarizes this method:

```
GENE_NEIGHBOR_METHOD
1  for each informative_genome
2     count_conserved <- 0
3     conserved_list <- ""
```

```
4       for each NEIGHBORS(a,b) in genome_of_interest
5       if (ORTH(a) AND ORTH(b)) in informative_genome
6             if   (NEIGHBORS(ORTH(a),ORTH(b)))   in
   informative_genome
7           ADD(a,b) to conserved_list
8            count_conserved <- count_conserved + 1
9     return (informative_genome, count_conserved,
      conserved_list)
```

Notice that the results are returned for each informative genome. This is important in order to calculate confidence scores. Throughout the detailed method later, I will be using PERL code to exemplify each step. The programs and example files can also be downloaded from http://microbiome.wlu.ca/GeneNeighbor/.

### 3.1 Learn Orthologs

Orthologs are defined as genes that diverge after a speciation event [21]. Such genes can also be colloquially referred to as the "same genes" in different species. Accordingly, orthologs are the appropriate genes to compare in the Gene Neighbor method. In comparative genomics, the most commonly used working definition of orthology is reciprocal best hits [22–24]. Two genes in two different genomes are reciprocal best hits if, when each is used as a query, each finds the other as its top scoring hit (*see* **Note 3**).

The possibility of adjacently conserved paralogs, genes that diverge after duplication events [21], was also discussed by Overbeek et al. [3]. Moreover, other work has shown that operons have a tendency toward producing paralog operons [14, 25], and that strict detection of orthologs is not necessary for prediction of functional association [14]. Thus, here I use conservation of unidirectional best hits for predicting interactions by conservation of gene order. In order to detect the top best hits for genes in a target genome, the protein sequences encoded by the genes in the target genome are compared against those encoded by the informative genome using BLASTP (*see* **Note 4**):

```
blastp -query genome_of_interest -db informative_genome
   -evalue 1e-4 \
-seg yes -soft_masking true -use_sw_tback -outfmt 6 -
   out \
genome_of_interest.informative_genome.blastp
```

This will produce a table of BLAST hits between the genome of interest and a given informative genome in the file "genome_of_interest.informative_genome.blastp." The '-outfmt 6' option instructs BLASTP to format the results into a simple, tab-separated, table. The other options are '-evalue 1e-6,' which sets the maximum E-value to 1e-6; '-seg yes -soft_masking true,' which sets filtering

of low information sequences during the blast search, but not during the alignment; '-use_sw_tback,' which indicates a Smith–Waterman alignment to calculate the scores [26] (*see* **Note 5**).

BLAST presents results sorted from best to worst match. Thus, a subroutine in PERL that can get the best hits would look like this (*see* **Note 6**):

```
sub get_best_hits {
   my ($genome_of_interest,$informative_genome) = @_;
   my %best_hits = ();
   my %E_value = ();
   my %bit_score = ();
   my $blast_file
    =    "BLAST_RUNS/$genome_of_interest.$informa-
    tive_genome.blastp";
   open(BLTBL,$blast_file);
   while(<BLTBL>) {
    my ($query_id,$target_id,@stats) = split;

    # both the query and target have a complex name,
    # we only need the gi number to match the neighbor
    # table identifiers
    my ($query) = $query_id =~ /gi\|(\d+)/;
    my ($target) = $target_id =~ /gi\|(\d+)/;

    # the penultimate value is the E value
    my $E_value = $stats[$#stats - 1];
    # the last value is the bit score
    my $bit_score = $stats[$#stats];
    # now we actually learn the best hits
    if($bit_score{$query} > 0) {
       if(
        ($E_value{$query} == $E_value)
        && ($bit_score{$query} == $bit_score)
    ) {
        $best_hits{$query} .= ",".$target;
    }
   }
    else {
        $E_value{$query} = $E_value;
```

```
Escherichia coli K12, complete genome - 0..4639675
4237 proteins
Location      Strand  Length  PID       Gene    Synonym  Code  COG      Product
190..255      +       21      16127995  thrL    b0001    -     -        thr operon leader peptide
337..2799     +       820     16127996  thrA    b0002    E     COG0460  bifunctional
                                                                        aspartokinase
                                                                        I/homeserine
                                                                        dehydrogenase I
2801..3733    +       310     16127997  thrB    b0003    E     COG0083  homoserine kinase
3734..5020    +       428     16127998  thrC    b0004    E     COG0498  threonine synthase
5234..5530    +       98      16127999  yaaX    b0005    -     -        hypothetical protein
5683..6459    -       258     16128000  yaaA    b0006    S     COG3022  hypothetical protein
6529..7959    -       476     16128001  yaaJ    b0007    E     COG1115  inner membrane transport
                                                                        protein
8238..9191    +       317     16128002  talB    b0008    G     COG0176  transaldolase
9306..9893    +       195     16128003  mogA    b0009    H     COG0521  molybdenum cofactor
                                                                        biosynthesis protein
9928..10494   -       188     16128004  yaaH    b0010    S     COG1584  putative regulator,
                                                                        integral membrane protein
10643..11356  -       237     16128005  yaaW    b0011    S     COG4735  hypothetical protein
10725..11315  +       196     16128006  htgA    b0012    -     -        positive regulator for
                                                                        sigma 32 heat shock
                                                                        promoters
11382..11786  -       134     16128007  yaaI    b0013    -     -        hypothetical protein
12163..14079  +       638     16128008  dnaK    b0014    O     COG0443  molecular chaperone DnaK
14168..15298  +       376     16128009  dnaJ    b0015    O     COG0484  chaperone with DnaK; heat
                                                                        shock protein
15445..16557  +       370     16128010  yi81_1  b0016    L     COG3385  IS186 hypothetical
                                                                        protein
15869..16177  -       102     16128011  yi82_1  b0017    -     -        IS186 and IS421
                                                                        hypothetical protein
16751..16960  -       69      16128012  mokC    b0018    -     -        regulatory peptide whose
                                                                        translation enables hokC
                                                                        (gef) expression
16751..16903  -       50      49175991  hokC    b4412    -     -        small toxic membrane
                                                                        polypeptide
17489..18655  +       388     16128013  nhaA    b0019    P     COG3004  Na+/H antiporter, pH
                                                                        dependent
18715..19620  +       301     16128014  nhaR    b0020    K     COG0583  transcriptional activator
                                                                        of cation transport (LysR
                                                                        family)
19811..20314  -       167     16128015  insB_1  b0021    L     COG1662  IS1 protein InsB
20233..20508  -       91      16128016  insA_1  b0022    L     COG3677  IS1 protein InsA
20815..21078  -       87      16128017  rpsT    b0023    J     COG0268  30S ribosomal protein S20
21181..21399  +       72      16128018  yaaY    b0024    -     -        unknown CDS
21407..22348  +       313     16128019  ribF    b0025    H     COG0196  hypothetical protein
22391..25207  +       938     16128020  ileS    b0026    J     COG0060  isoleucyl-tRNA synthetase
25207..25701  +       164     16128021  lspA    b0027    M     COG0597  signal peptidase II
25826..26275  +       149     16128022  fkpB    b0028    O     COG1047  FKBP-type peptidyl-prolyl
                                                                        cis-trans isomerase
                                                                        (rotamase)
26277..27227  +       316     16128023  ispH    b0029    I     COG0761  4-hydroxy-3-methylbut-2-
                                                                        enyl diphosphate
                                                                        reductase
27293..28207  +       304     16128024  rihC    b0030    F     COG1957  nucleoside hydrolase
28374..29195  +       273     16128025  dapB    b0031    E     COG0289  dihydrodipicolinate
                                                                        reductase
29651..30799  +       382     16128026  carA    b0032    E     COG0505  carbamoyl-phosphate
                                                                        synthase small subunit
30817..34038  +       1073    16128027  carB    b0033    E     COG0458  carbamoyl-phosphate
                                                                        synthase large subunit
34195..34695  +       166     49175992  caiF    b0034    -     -        transcriptional regulator
                                                                        of cai operon
34781..35392  -       203     16128029  caiE    b0035    R     COG0663  possible synthesis of
                                                                        cofactor for carnitine
                                                                        racemase and dehydratase
35377..36270  -       297     16128030  caiD    b0036    I     COG1024  carnitinyl-CoA
                                                                        dehydratase
36271..37839  -       522     49175993  caiC    b0037    -     -        crotonobetaine/carnitine-
                                                                        CoA ligase
```

**Fig. 2** A few lines of the PTT table of the genome of *Escherichia coli* K12. The *first column* of the PTT (protein-coding genes) and of the RNT (noncoding genes, those producing rRNAs and tRNAs) tables contains the gene coordinates. The *second column* contains the strand where the gene is found, which is useful for organizing the genes into stretches of adjacent genes in the same strand, called *directons*. The *fourth column* is the GI

```
            $bit_score{$query} = $bit_score;
            $best_hits{$query} = $target;
          }
    }
      close(BLTBL);
      return(%best_hits);
    }
```

### 3.2 Neighbors Database

The natural next step is to build a database of gene neighbors. The minimum information that this database should contain is a list of adjacent gene pairs and information on the strand on which each gene is found. To build this database, a convenient starting point is the RefSeq genomes database, available from the NCBI ftp server.

Several Refseq files could be used to obtain coordinates for each gene within the genome. Here I exemplify with the PTT (protein table) and RNT (robonucleotide table) files. The PTT file contains a table of protein-coding genes, while the RNT file contains a table of rRNA and tRNA genes (*see* **Note 4**). The first column within these tables consists of the gene coordinates. As an example I show a few lines of the PTT file for the genome of *Escherichia coli* K12 [27], accession "NC_000913," version "NC_000913.2 GI:49175990" (Fig. 2).

The first column in these tables corresponds to gene coordinates. Thus, the problem of forming pairs of adjacent genes becomes trivial. All that is needed is to sort the genes and associate each of them with the next gene in the list, formatting them into a table, or a database, of Gene Neighbors. The header of the resulting table might look like this:

| Gene_a | Gene_b | Strands |
|---|---|---|

Genes in the same strand will have either "++" or "−−" in the "Strand" column, while genes in different strands will have either "+−" (convergently transcribed) or "−+" (divergently transcribed) in this field (*see* **Note 7**).

**Fig. 2** (Continued) number (labeled here as a PID or protein identifier). This number is the best identifier for the protein-coding genes in a genome because it is unique. However, in the RNT tables this column is not the best identifier; the best identifiers seem to be the gene name (*fifth column*), and the synonym (*sixth column*). The table in the figure is formatted for display purposes, but the original PTT and RNT tables contain tab-separated plain text

If the genome is circular a final pair should be formed with the last and the first genes in the table. The first line in the GBK file indicates whether the replicons are circular or linear (*see* **Note 8**).

An example program in PERL that will output this table is:

```
#!/usr/bin/perl
$die_msg = "\tI need a genome to work with\n\n";
$genome_of_interest = $ARGV[0] or die $die_msg;
 $die_msg = "\tNo $genome_of_interest directory\n\n";
$genome_dir = "LOCAL_GENOMES/$genome_of_interest";
opendir(GNMDIR,"$genome_dir") or die $die_msg;
@ptt_files = grep {/\.ptt/} readdir(GNMDIR);
$results_dir = "NEIGHBORS";
mkdir($results_dir) unless (-d $results_dir);
open(NGHTBL,"> $results_dir/$genome_of_interest.nghtbl");
 for my $ptt_file (@ptt_files) {
   # get proper name of the RNT and GBK files
   my $rnt_file = $ptt_file;
   my $gbk_file = $ptt_file;
   $rnt_file =~ s/\.ptt/\.rnt/;
   $gbk_file =~ s/\.ptt/\.gbk/;
   # Is the genome circular?
   # The information is in the first line of the GBK
   # file, which starts with the word "LOCUS"
   my $circular = "yes"; # make circular the default
   open (GBK,"$genome_dir/$gbk_file");
   while(<GBK>) {
      if(/^LOCUS/) {
      $circular = "no" if(/linear/i);
      last; # we do not need to read any further
      }
   }
   # now we read the table of protein coding genes
   # and their "leftmost" coordinate so we can
   # order them and find the neighbors
```

```
31   my %strand = ();
32   my %coords = ();
33   my @ids = ();
34   open(PTT,"$genome_dir/$ptt_file");
35   while(<PTT>) {
36      my @data = split;
37      next unless($data[1] =~ /^\+|\-$/);
38      $gi = $data[3];
39      $strand{$gi} = $data[1];
40      my ($coord) = $data[0] =~ /^(\d+)/;
41      $coord{$gi} = $coord;
42  }
43   close(PTT);
44  # we verify that there is a table of rRNA and tRNA
    genes
45  # if so, we get the genes
46  if(-f "$genome_dir/$rnt_file") {
47   open(RNT,"$genome_dir/$rnt_file");
48   while (<RNT>) {
49     my @data = split;
50     next unless($data[1] =~ /^\+|\-$/);
51
52     # The identifier is not a GI
53     # but I rather keep the variable names consistent
54     # the best identifier for an 'RNA' gene is
55     # the gene name (5th column)
56     my $gi = $data[4];
57     $strand{$gi} = $data[1];
58     my ($coord) = $data[0] =~ /^(\d+)/;
59     $coord{$gi} = $coord;
60     }
61  }
62  # now we build the table of direct neighbors
63  my @ids = sort {$coord{$a} <=> $coord{$b}} keys
    %coord;
64  for my $i(0 .. $#ids) {
65   if (exists($strand{$ids[$i+1]})) {
```

```
66      my $str = $strand{$ids[$i]}.$strand{$ids
      [$i + 1]};
67      print NGHTBL $ids[$i],"\t",$ids[$i + 1],"\t",
        $str,"\n";
68    }
69    else {
70     if ($circular eq "yes") {
71      my $str = $strand{$ids[$i]}.$strand{$ids[0]};
72      print NGHTBL $ids[$i],"\t",$ids[0],"\t",$str,
        "\n";
73      }
74     }
75    }
76 }
77
78 close(NGHTBL);
```

and a subroutine that will read this table, learn the neighbors, and classify them as same-strand and opposite-strand neighbors is:

```
sub get_strands_of_neighbors {
   my $genome = $_[0];
   # we will learn the neighbors as hashes where the keys
   # are the neighbor pairs of genes and the values are
   # the strand situations (same strand or opposite
    strand
   my %strands_of = ();
   open(NGH,"NEIGHBORS/$genome.nghtbl");
   while(<NGH>) {
   my($gi,$gj,$strand) = split;
   my $neighbors = join(",",sort($gi,$gj));
   if (($strand eq "--") || ($strand eq "++")) {
      $strands_of{"$neighbors"} = "same";
   }
    elsif (($strand eq "-+") || ($strand eq "+-")) {
      $strands_of{"$neighbors"} = "opp";
   }
  }
 return(%strands_of);
}
```

### 3.3 Putting Everything Together

As originally defined, the Gene Neighbor Method aims to find genes with conserved adjacency in evolutionarily distant genomes. However, Ermolaeva et al. [13] have obviated the need for a phylogenetic distance by using the genomes themselves to determine the significance of the conservation in the form of a confidence value. The idea behind the confidence value is that the proportion of conserved adjacencies in opposite strands represents conservation due to chance alone, or more properly, conservation due to short evolutionary distance and chance rearrangement (*see* **Note 9**). A simplified version of the confidence value calculated under the same assumption is:

$$C = 1 - 0.5 * \frac{P_{Opp}}{P_{Same}}$$

The confidence value ($C$) can be thought of as a positive predictive value (true positives divided by the total number of predictions) for two genes to be conserved due to a functional interaction (they would be in the same operon) (*see* **Note 10**). The value 0.5 in this expression is a prior probability for the genes to be in different transcription units. $P_{Opp}$ is the count of pairs of orthologs conserved next to each other in opposite strands ("+−" and "−+" pairs of neighbor genes) divided by the total number of neighbors in opposite strands in the informative genome. $P_{Same}$ is the count of orthologs conserved next to each other in the same strand ("++" and "−−") divided by the total number of neighbors in the same strand in the informative genome.

Now, with all the necessary data, neighbors, and best hits, and with a way of calculating a confidence value, the previous pseudocode is modified as:

```
GENE_NEIGHBOR_METHOD
1    for each informative_genome
2      count_conserved < - 0
3      conserved_list < - ""
4      for each NEIGHBORS(a,b) in genome_of_interest
5        if (ORTH(a) AND ORTH(b)) in informative_genome
6          if (same-strand(ORTH(a),ORTH(b))) in informative_
           genome
7          ADD(a,b) to conserved_same-strand
8          count_same < - count_same + 1
9        else if (opposite-strand(ORTH(a),ORTH(b))
10         ADD(a,b) to conserved_opposite-strand
11       count_opposite < - count_opposite + 1
```

```
12   confidence <- 1-0.5*proportion(same)/propor-
     tion(opposite)
13  return (informative_genome, confidence, conser-
    ved_same-strand)
```

And a particular example program in PERL would be:

```
1  #!/usr/bin/perl
2  $genome_of_interest = "Escherichia_coli_K12";
3  @genomes = qw(
4         Salmonella_typhi_Ty2
5         Yersinia_pestis_KIM
6         Rhizobium_etli_CFN_42
7         Bacillus_subtilis
8    );
9  $results_dir = "Confidence";
10 mkdir($results_dir) unless(-d $results_dir);
11 my %strands_of = get_strands_of_neighbors
   ($genome_of_interest);
12 open(CONF," >$results_dir/$genome_of_
   interest.confidence");
13 for my $informative_genome (@genomes) {
14   print $informative_genome,"\n";
15   my %best_hits
16     = get_best_hits($genome_of_interest,$
       informative_genome);
17  my %inf_strands_of
18     = get_strands_of_neighbors($informative_
       genome);
19  my $count_same = 0;
20  my $count_opp = 0;
21  my @predictions;
22  for my $neighbors (keys %strands_of) {
23     my ($gi,$gj) = split(/,/,$neighbors);
24     # first see if there are any orthologs
25     if (exists($best_hits{$gi})
26           && exists($best_hits{$gj})) {
27           # since there might be more than one ortho-
             log, and
28           # there might be more than one conserved
             pair,
29           # we use a "flag" (count_conserv = "none") to
```

```
30          # avoid "overcounting"
31          my $count_conserv = "none";
32
33      # now the actual verification of conservation
34      for my $orth_i (split(/,/,$best_hits{$gi})) {
35       for my $orth_j (split(/,/,$best_hits{$gj})) {
36          my $test_neigh = join ",", sort($orth_i,
            $orth_j);
37          if ($inf_strands_of{$test_neigh}
38            eq $strands_of{$neighbors}) {
39            $count_conserv = $strands_of{$neigh
              bors};
40          }
41       }
42      }
43
44      # now we verify the flag and count any conservation
45      if ($count_conserv eq "same") {
46         $count_same++;
47         push(@predictions,$neighbors);
48       }
49      elsif ($count_conserv eq "opp") {
50        $count_opp++;
51        }
52      }
53   }
54   # now we also need to count the number of genes in the
     same
55   # strand and those in opposite strands in the infor-
     mative genome
56   my $total_same = 0;
57   my $total_opp = 0;
58   for my $inf_ngh (keys %inf_strands_of) {
59    if ($inf_strands_of{$inf_ngh} eq "same") {
60     $total_same++;
61     }
62    elsif($inf_strands_of{$inf_ngh} eq "opp") {
63    $total_opp++;
64    }
65   }
```

```
66
67  # now we can calculate the confidence value
68  my $P_same = $count_same/$total_same;
69  my $P_opp = $count_opp/$total_opp;
70  my $conf = 1 - 0.5 * ($P_opp/$P_same);
71  $conf = sprintf("%.2f",$conf);
72  print "CONFIDENCE = ",$conf,"\n";
73   # now print predictions with their confidence
    values
74  for my $prediction (@predictions) {
75    $prediction =~ s/,/\t/;
76    print CONF $prediction,"\t",$conf,"\t",$info
    rmative_genome,"\n";
77    }
78 }
```

When run, this program creates a single file with conserved neighbors, their confidence values, and the genome from which the value was obtained. At the same time, the program prints the following output to the display:

```
% ./neighbor-method.pl
Salmonella_typhi_Ty2
CONFIDENCE = 0.55
Yersinia_pestis_KIM
CONFIDENCE = 0.73
Rhizobium_etli_CFN_42
CONFIDENCE = 0.99
Bacillus_subtilis
CONFIDENCE = 1.00
```

The informative genomes are ordered evolutionarily from closest to farthest. As expected, the evolutionarily closest organism to *E. coli* K12 in this example, *Salmonella typhi* Ty2, gives the lowest confidence value, while the farthest gives the maximum confidence value. The threshold I use to accept predictions is a confidence value $\geq 0.95$.

## 4 Notes

1. Traditionally, UNIX users might have used the ftp program to transfer files. The newer programs, wget and rsync, offer options that might help transferring more than one file with a

single command. Many servers lack rsync capabilities, and then wget can be used. It is still possible to use the ftp command for this task. Conveniently, ftp sites can be displayed in a web browser, the user can then find the files that might be of interest, and then download them.

2. As of this writing, NCBI has reorganized its RefSeq Genome data server. The Bacteria subdirectory has been deleted. The new directory structure is very complicated, which makes me think that people wanting to work with all the complete genomes will have to access them using a program. Some of the changes at NCBI's server allow access to several assemblies for each genome, thus complicating the automatic decision as to which assembly to download. I can advise little more now than consulting the assembly file in order to decide what to download:

```
rsync -avzL \
rsync://rsync.ncbi.nlm.nih.gov/refseq/assembly_summary_refseq.txt
```

3. For finding orthologous genes, what we compare is the proteins encoded by the annotated genes in one genome, against the proteins encoded by the annotated genes in the other. Genes producing directly active RNA, such as tRNA and rRNA genes, are mostly ignored in these kinds of analyses, perhaps because they are fewer than the coding genes, and because comparing DNA sequences and thus determining orthology, especially among evolutionarily distant organisms, can be very difficult.

4. In order for BLASTP to run, the protein sequences found in the files ending with ".faa" (FAA file) have to be formatted into BLAST databases. I prefer to keep each genome separated so it is simpler to update results when a new genome is published. The main caveat to this approach is that some prokaryotic genomes contain more than one replicon. This means that there will be more than one FAA file for these genomes. It is better to have all the protein sequences in a single file. Thus, I concatenate all the FAA files within the directory of each genome into a single file. A simple UNIX command that can do this job is:

```
cat genome_of_interest/*.faa > FAADB/genome_of_interest.faa
```

A file compressed with gzip, like the one used under **Note 5**, would be obtained as follows:

```
cat genome_of_interest/*.faa | gzip -9 > FAADB/genome_of_interest.faa.gz
```

The "−9" gzip option calls for maximum compression. To build BLAST databases the command is:

```
makeblastdb -dbtype prot -in FAADB/genome_
of_interest.faa -parse_seqids \
-hash_index -out BLASTDB/genome_of_interest
–title ““genome_of_interest”
```

5. Given blast's UNIX heritage, the command can be "piped." Because of this important feature, blast results can be compressed as they are produced, if needed. This can be accomplished by taking advantage of blast's default output being the standard output (the screen), which can be piped into the gzip command (or bzip2):

   ```
   blastp -query genome_of_interest.faa -db informa-
   tive_genome -evalue 1e-4 \
   -seg yes -soft_masking true -use_sw_tback -outfmt
   7 | gzip -9 > \
   genome_of_interest.informative_genome.blastp.gz
   ```

   Piping can also be advantageous to run blast when the query fasta file is also compressed:

   ```
   gzip –qdc genome_of_interest.faa.gz | blastp -
   query - \
   -db informative_genome -evalue 1e-4 \
   -seg yes -soft_masking true -use_sw_tback -outfmt
   7 | gzip -9 > \
   genome_of_interest.informative_genome.blastp.gz
   ```

   The "-query -" option is not really necessary (though I prefer using explicit options, to easily understand what is going on when checking commands later on), because the default query is "-" (standard input):

   ```
   gzip –qdc genome_of_interest.faa.gz | blastp \
   -db informative_genome -evalue 1e-4 \
   -seg yes -soft_masking true -use_sw_tback -outfmt
   7 | gzip -9 > \
   genome_of_interest.informative_genome.blastp.gz
   ```

6. It might be tempting to use blastp's option for displaying only one matching sequence per query sequence (-max_target_seqs 1). However, there can be more than just one best hit. Yet the option would only display one. Cases where more than one best hit exists are not very frequent, but they happen. It is up to the user to decide whether to use this option and save downstream computation.

7. It is also possible to allow gaps (i.e., intervening genes) between gene pairs. However, in my experience, allowing gaps neither improves, nor worsens the results. This assessment is based on knowledge of the operons in *Escherichia coli* K12. However, allowing gaps might facilitate calculation of confidence values in very small genomes, where the number of same- and opposite-strand genes might be too small. If gaps

are used, it is important that the pairs of genes are part of the same stretch of genes in the same strand with no intervening genes in the opposite strand (such stretches are called *directons*). For opposite-strand genes it will be enough to confirm that they are in different strands. The extreme example is the same-directon versus different-directon approach. The conservation to be evaluated would be that of two genes in the same directon, regardless of the number of genes in between. The control, or negative set, would consist of genes in different, yet adjacent, directons. This is very similar to a method that is now used at The Institute for Genomics Research (Maria Ermolaeva, personal communication), which is a simplified version of a method published by Ermolaeva et al. [13]. A program that will output a database of genes in the same directon, and genes in different directons, would be:

```
1  #!/usr/bin/perl
2  $genome_of_interest = $ARGV[0] or die "I need a
   genome to work with\n\n";
3                  $genome_dir = "LOCAL_GENOMES/
   $genome_of_interest";
4  opendir(GNMDIR,"$genome_dir") or die $die_msg;
5  @ptt_files = grep {/\.ptt/} readdir(GNMDIR);
6  $results_dir = "NEIGHBORS_DIRECTON";
7  mkdir($results_dir) unless (-d $results_dir);
8    open(NGHTBL,"> $results_dir/$genome_of_interest.
     nghtbl");
9  PTT:
10 for my $ptt_file (@ptt_files) {
11    # get proper name of the RNT and GBK files
12    my $rnt_file = $ptt_file;
13    my $gbk_file = $ptt_file;
14    $rnt_file =~ s/\.ptt/\.rnt/;
15    $gbk_file =~ s/\.ptt/\.gbk/;
16    # Is the genome circular?
17    # The information is in the first line of the
      "gbk"
18    # file, which starts with the word "LOCUS"
19     my $circular = "yes"; # make circular the
       default
20     open (GBK,"$genome_dir/$gbk_file");
21     while(<GBK>) {
```

```
22      if(/^LOCUS/) {
23         $circular = "no" if(/linear/i);
24         last; # we do not need to read any further
25      }
26   }
27   # now we read the table os protein coding genes
28   # and their "leftmost" coordinate so we can
29   # order them and find the neighbors
30   my %strand = ();
31   my %coord = ();
32   open(PTT,"$genome_dir/$ptt_file");
33   while(<PTT>) {
34      my @data = split;
35      next unless($data[1] =~ /^\+|\-$/);
36      my $gi = $data[3];
37      $strand{$gi} = $data[1];
38      my ($coord) = $data[0] =~ /^(\d+)/;
39      $coord{$gi} = $coord;
40   }
41   close(PTT);
42   if(-f "$genome_dir/$rnt_file") {
43      open(RNT,"$genome_dir/$rnt_file");
44      while (<RNT>) {
45        my @data = split;
46        next unless($data[1] =~ /^\+|\-$/);
47
48        # The identifier is not a GI
49        # but I rather keep the variable names
            consistent
50        # the best identifier for an 'RNA' gene is
51        # the gene name (5th column)
52        my $gi = $data[4];
53        $strand{$gi} = $data[1];
54        my ($coord) = $data[0] =~ /^(\d+)/;
55        $coord{$gi} = $coord;
56      }
57   }
```

```
58  # we build directons: stretches of genes in the
    same
59  # strand with no intervening gene in the opposite
60  # strand
61  my  @ids = sort  {$coord{$a} <=> $coord{$b}}
    keys %coord;
62  my @directon = ();
63  my $directon;
64  $prev_str = "none";
65  for my $gi(@ids) {
66  if ($strand{$gi} eq $prev_str) {
67   $directon . = ",".$gi;
68   $prev_str = $strand{$gi};
69   }
70   else {
71     push(@directon,$directon)    if    (defined
       $directon);
72     $directon = $gi;
73     $prev_str = $strand{$gi};
74     }
75  }
76
77  # with circular genomes we make sure that
78  # we close the circle, meaning if first and last
79 # directon are in the same strand, they form a single
80  # directon
81  if ($strand{$ids[0]} eq $strand{$ids[$#ids]}) {
82  if ($circular eq "yes") {
83  $directon[0] = $directon.",".$directon[0];
84  }
85   else {
86     push(@directon,$directon);
87  }
88  }
89  else {
90   push(@directon,$directon);
91  }
92
```

```
93  # now we do form pairs in same directon, and
94  # pairs in different directons
95  for my $i (0 .. $#directon) {
96     my @gi = split(/,/,$directon[$i]);
97     # same directon
98     my @expendable = @gi;
99     while (my $gi = shift @expendable) {
100       for my $gj (@expendable) {
101         print NGHTBL $gi,"\t",$gj
102         ,"\t",$strand{$gi}.$strand{$gj},"\n";
103       }
104  }
105  ## different directon
106  ## assuming circular replicons
107  my $next_directon = "none";
108  if ($i < $#directon) {
109   $next_directon = $directon[$i + 1];
110   }
111   else {
112    if ($circular eq "yes") {
113       $next_directon = $directon[0];
114    }
115      else {
116       next PTT;
117      }
118    }
119  my @gj = split(/,/,$next_directon);
120  for my $gi (@gi) {
121    for my $gj (@gj) {
122         print NGHTBL $gi,"\t",$gj
123        ,"\t",$strand{$gi}.$strand{$gj},"\n";
124     }
125      }
126    }
127  }
128  close(NGHTBL);
```

8. It is important to know that some of the Prokaryotic genomes reported so far have more than one replicon, meaning more than one DNA molecule. Multireplicon genomes can contain two or more chromosomes, mega-plasmids, and plasmids. I consider all the published replicons part of the genome, and thus the programs presented are designed to read all of the replicons under a given genome directory.
9. As stated, Overbeek et al. [3] noted that some divergently transcribed genes could be functionally related, but found that the proportion of conserved, divergently transcribed genes across evolutionarily distant species was very small. The main effect of this possibility is that the confidence value would be an *underestimate*. This is clear in the analyses presented by Ermolaeva et al. [13], and in the particular examination of false positives presented by Janga et al. [14], who found independent evidence that almost all of their false positives had a functional relationship (*see also* **Note 6**). In these analyses, the confidence value of 0.95 seems to correspond to a positive predictive value (true positives divided by the total number of predictions) of 0.98.
10. The relationship between the positive predictive value and the confidence value has been established [13, 14] using data on experimentally determined operons of *Escherichia coli* K12 from RegulonDB [28]. Another useful statistic is coverage (also called sensitivity: true positives divided by the total number of truly related pairs). For protein-coding genes, the current estimate for most genomes is that 0.5 of all same-strand direct neighbors might be in the same operon. In *E. coli* K12, the total number of same-strand protein-coding genes is 2930. Thus, the total number of functionally related neighbors is approximately $2930/2 = 1465$. The maximum number of predictions for *E. coli* K12 compared against all the genomes in the current database is 640 at a confidence value $\geq 0.95$. Thus, the estimated coverage is: 640 * 0.95 / 1465 = 0.41. This coverage might be thought low, but the predictions are of excellent quality.

## Acknowledgments

This work was supported by a Discovery Grant from the Natural Sciences and Engineering Research Council of Canada (NSERC).

## References


1. Dandekar T, Snel B, Huynen M, Bork P (1998) Conservation of gene order: a fingerprint of proteins that physically interact. Trends Biochem Sci 23:324–328
2. Overbeek R, Fonstein M, D'Souza M, Pusch GD, Maltsev N (1999) Use of contiguity on the chromosome to predict functional coupling. In Silico Biol 1:93–108
3. Overbeek R, Fonstein M, D'Souza M, Pusch GD, Maltsev N (1999) The use of gene clusters to infer functional coupling. Proc Natl Acad Sci U S A 96:2896–2901
4. Overbeek R, Olson R, Pusch GD, Olsen GJ, Davis JJ, Disz T, Edwards RA, Gerdes S, Parrello B, Shukla M, Vonstein V, Wattam AR, Xia F, Stevens R (2014) The SEED and the Rapid Annotation of microbial genomes using Subsystems Technology (RAST). Nucleic Acids Res 42:D206–D214
5. Glass EM, Wilkening J, Wilke A, Antonopoulos D, Meyer F (2010) Using the metagenomics RAST server (MG-RAST) for analyzing shotgun metagenomes. Cold Spring Harb Protoc 2010, pdb prot5368
6. Jacob F, Perrin D, Sanchez C, Monod J (1960) Operon: a group of genes with the expression coordinated by an operator. C R Hebd Seances Acad Sci 250:1727–1729
7. Jacob F, Perrin D, Sanchez C, Monod J, Edelstein S (2005) [The operon: a group of genes with expression coordinated by an operator. C R Acad Sci Paris 250 (1960) 1727–1729]. C R Biol 328, 514–520
8. Mushegian AR, Koonin EV (1996) Gene order is not conserved in bacterial evolution. Trends Genet 12:289–290
9. Bork P, Dandekar T, Diaz-Lazcoz Y, Eisenhaber F, Huynen M, Yuan Y (1998) Predicting function: from genes to genomes and back. J Mol Biol 283:707–725
10. Date SV, Marcotte EM (2003) Discovery of uncharacterized cellular systems by genome-wide analysis of functional linkages. Nat Biotechnol 21:1055–1062
11. Korbel JO, Jensen LJ, von Mering C, Bork P (2004) Analysis of genomic context: prediction of functional associations from conserved bidirectionally transcribed gene pairs. Nat Biotechnol 22:911–917
12. Moreno-Hagelsieb G, Jokic P (2012) The evolutionary dynamics of functional modules and the extraordinary plasticity of regulons: the Escherichia coli perspective. Nucleic Acids Res 40:7104–7112
13. Ermolaeva MD, White O, Salzberg SL (2001) Prediction of operons in microbial genomes. Nucleic Acids Res 29:1216–1221
14. Janga SC, Moreno-Hagelsieb G (2004) Conservation of adjacency as evidence of paralogous operons. Nucleic Acids Res 32:5392–5397
15. Snel B, Lehmann G, Bork P, Huynen MA (2000) STRING: a web-server to retrieve and display the repeatedly occurring neighbourhood of a gene. Nucleic Acids Res 28:3442–3444
16. von Mering C, Huynen M, Jaeggi D, Schmidt S, Bork P, Snel B (2003) STRING: a database of predicted functional associations between proteins. Nucleic Acids Res 31:258–261
17. Zheng Y, Anton BP, Roberts RJ, Kasif S (2005) Phylogenetic detection of conserved gene clusters in microbial genomes. BMC Bioinformatics 6:243
18. Tatusova T, Ciufo S, Fedorov B, O'Neill K, Tolstoy I (2014) RefSeq microbial genomes database: new representation and annotation strategy. Nucleic Acids Res 42:D553–D559
19. Maglott DR, Katz KS, Sicotte H, Pruitt KD (2000) NCBI's LocusLink and RefSeq. Nucleic Acids Res 28:126–128
20. Camacho C, Coulouris G, Avagyan V, Ma N, Papadopoulos J, Bealer K, Madden TL (2009) BLAST+: architecture and applications. BMC Bioinformatics 10:421
21. Fitch WM (2000) Homology a personal view on some of the problems. Trends Genet 16:227–231
22. Tatusov RL, Koonin EV, Lipman DJ (1997) A genomic perspective on protein families. Science 278:631–637
23. Ward N, Moreno-Hagelsieb G (2014) Quickly finding orthologs as reciprocal best hits with BLAT, LAST, and UBLAST: how much do we miss? PLoS One 9:e101850
24. Moreno-Hagelsieb G, Latimer K (2008) Choosing BLAST options for better detection of orthologs as reciprocal best hits. Bioinformatics 24:319–324
25. Gevers D, Vandepoele K, Simillion C, Van de Peer Y (2004) Gene duplication and biased functional retention of paralogs in bacterial genomes. Trends Microbiol 12:148–154
26. Schaffer AA, Aravind L, Madden TL, Shavirin S, Spouge JL, Wolf YI, Koonin EV, Altschul SF (2001) Improving the accuracy of PSI-BLAST protein database searches with composition-based statistics and other refinements. Nucleic Acids Res 29:2994–3005

27. Blattner FR, Plunkett G 3rd, Bloch CA, Perna NT, Burland V, Riley M, Collado-Vides J, Glasner JD, Rode CK, Mayhew GF, Gregor J, Davis NW, Kirkpatrick HA, Goeden MA, Rose DJ, Mau B, Shao Y (1997) The complete genome sequence of Escherichia coli K-12. Science 277:1453–1474
28. Salgado H, Gama-Castro S, Peralta-Gil M, Diaz-Peredo E, Sanchez-Solano F, Santos-Zavaleta A, Martinez-Flores I, Jimenez-Jacinto V, Bonavides-Martinez C, Segura-Salazar J, Martinez-Antonio A, Collado-Vides J (2006) RegulonDB (version 5.0): Escherichia coli K-12 transcriptional regulatory network, operon organization, and growth conditions. Nucleic Acids Res 34:D394–D397

# Chapter 4

# Structural and Functional Annotation of Long Noncoding RNAs

**Martin A. Smith and John S. Mattick**

## Abstract

Protein-coding RNAs represent only a small fraction of the transcriptional output in higher eukaryotes. The remaining RNA species encompass a broad range of molecular functions and regulatory roles, a consequence of the structural polyvalence of RNA polymers. Albeit several classes of small noncoding RNAs are relatively well characterized, the accessibility of affordable high-throughput sequencing is generating a wealth of novel, unannotated transcripts, especially long noncoding RNAs (lncRNAs) that are derived from genomic regions that are antisense, intronic, intergenic, and overlapping protein-coding loci. Parsing and characterizing the functions of noncoding RNAs—lncRNAs in particular—is one of the great challenges of modern genome biology. Here we discuss concepts and computational methods for the identification of structural domains in lncRNAs from genomic and transcriptomic data. In the first part, we briefly review how to identify RNA structural motifs in individual lncRNAs. In the second part, we describe how to leverage the evolutionary dynamics of structured RNAs in a computationally efficient screen to detect putative functional lncRNA motifs using comparative genomics.

**Key words** lncRNA, Comparative genomics, RNA secondary structure, Homology search, Functional genome annotation

## 1 Introduction

Functional genome annotation involves the identification of both known and hypothetical genes in uncharacterized genomic DNA sequence. This largely includes protein-coding genes and noncoding RNAs, as well as other genomic features such as telomeric/subtelomeric regions and centromeres. The identification of protein-coding genes can unravel the molecular repertoire of the majority of the genomes of microorganisms, especially prokaryotes, whose genomes are largely composed of protein-coding sequences. However, protein-coding sequences encompass only a small fraction of the genome in higher eukaryotes, which decreases with increasing developmental and cognitive complexity [1, 2] and comprise less than 1.5 % of the human genome.

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_4, 

Most of the human genome is dynamically transcribed into RNA [3, 4], which implies that untranslated RNAs compose the most abundant class of genomic output. In particular, noncoding transcripts greater than 200 nt in length—long noncoding RNAs (lncRNAs)—are emerging as master regulators of development and differentiation in higher eukaryotes [5–10]. There are currently 15,767 lncRNA genes (excluding alternative isoforms and pseudogenes) listed in version 25 of the GENCODE human genome annotation database, compared to 19,950 protein-coding genes. Contrary to protein-coding genes, whose set is relatively well characterized and has remained relatively stable in number and repertoire throughout metazoan evolution [1, 2, 11], although there are novel genes mainly encoding small proteins being discovered [12], the number of identified lncRNAs is steadily increasing as more and more biological conditions are investigated with high-throughput RNA sequencing technologies.

Many lncRNAs appear to regulate gene expression through their association with epigenetic proteins, such as histone modification enzymes and DNA methyltransferases, with which they synergistically organize the nuclear environment [13–15]. Other lncRNA functions include acting as molecular decoys and macromolecular scaffolds, as well as the regulation of splicing and translation, mRNA stability, and the formation of subcellular organelles [5, 16]. A small but growing number of lncRNAs have been functionally validated through knockout and ectopic expression in vivo and in cell culture, and other biochemical studies [17–20], but the precise molecular mechanisms and structures guiding their function remain largely unresolved.

At present, lncRNAs are largely categorized by their position relative to neighboring protein-coding genes, i.e., intergenic, antisense, intronic, or bidirectional. However, the particular functions of lncRNAs do not necessarily correlate with their genomic context. For example, the lncRNA *HOTAIR* functions by recruiting a chromatin modification complex (PRC2) to repress gene expression in *trans* [21], whereas the lncRNA *HOTTIP* recruits another epigenetic complex (WDR5-MLL1) in *cis* to activate gene expression via chromosomal looping [22]. Both are situated in the intergenic regions surrounding HOX genes. The functional annotation of lncRNAs at a genome- or transcriptome-wide scale therefore requires the consideration of additional molecular features that may be unique to each transcript.

A unifying feature of ncRNAs is their propensity to form discrete secondary and tertiary structures through canonical and noncanonical nucleotide base pairings that often dictate their function. Many lncRNAs appear to be very plastic, evolve quickly, and/or have arisen relatively recently in evolution, as evidenced by high turnover rates and reduced primary sequence conservation [23, 24], although there are exceptions that have extraordinarily high

levels of sequence conservation [25–27]. Their evolutionary dynamics are different from protein-coding genes, displaying relaxed structure-function constraints that are synonymous with being under positive selection for adaptive radiation. They are in general (although there are likely to be exceptions) unlikely to have catalytic activities, such as ribosomal RNAs, yet may nonetheless form evolutionarily stable, functional secondary and tertiary structures with different functions, as well as shorter primary sequences that may interact with other RNAs and DNA. For instance, the widespread presence of repetitive sequences derived from mobile elements in the human genome is believed to contribute to modular lncRNA biogenesis by forming a reservoir of functional motifs—or structured templates for RNA-binding proteins—that can be co-opted into RNA regulatory networks via positive selection [28–30].

Computational identification of functional RNA structural motifs encoded in genomic sequences is a challenging task, mainly because almost any RNA sequence can form internal base pairs via classical Watson–Crick, Hoogstein, or ribose 2′OH hydrogen bond formation, and fold into discrete structures [31, 32], but also because RNA structures themselves are dynamic, flexible, and are contingent on the cellular environment (i.e., temperature, ion concentrations, ligand binding, transcriptional kinetics). Functional RNA structures can nonetheless be identified through comparative genomics by observing nucleotide substitutions that are consistent and compatible with a common structural topology. Indeed, a much larger fraction of the human genome seems to function through the formation of RNA structure motifs than through sequence-constrained elements, as evidenced by considering nucleotide covariation events in evolutionary information [33].

In this chapter, we describe how to annotate ncRNAs in genomic or transcriptomic data, where known or putative functions are assigned to uncharacterized sequences to gain insight into their biology. First, we summarize how to identify functional RNA elements in single sequences via homology search as well as prediction of local structures in long transcripts. Finally, we describe how to identify putative functional motifs in lncRNAs that are supported by evolutionarily conserved RNA secondary structures. We provide user friendly, step-by-step instructions on how to perform a multiple genome-wide screen for functional RNA motifs similar to that published in [33].

## 2 Materials

A UNIX-based computing environment should be employed for most of the described methods, preferably with access to a high-performance computing infrastructure. Alternatively, a computer or server with multiple processors and over 4 GB of RAM may be employed.

### 2.1 Genomic Data

Genomic or transcriptomic sequence data should be downloaded and converted (if required) to *fasta* file format, unless it is already available. Genomic data for reference organisms can be obtained from the following sources:

1. UCSC genome browser—select the organism and the desired genome version, then full data set, then the file with suffix "*.fa.gz*" at http://hgdownload.cse.ucsc.edu/downloads.html.
2. NCBI—select the species of interest and then sequence data can be downloaded for each chromosome individually at (ftp://ftp.ncbi.nih.gov/genomes/). A FTP batch download tool or interface should be considered to automate the process.
3. ENSEMBL genome browser—select the appropriate release version, then 'fasta' at ftp://ftp.ensembl.org/pub/.

### 2.2 Transcriptomic Data

LncRNAs are often spliced (including alternatively spliced), generating sequences and structures that would otherwise be missed during computational screens of unprocessed genomic sequences. Depending on the task at hand and the availability of suitable data, the sequences corresponding to processed transcripts should also be considered to improve the robustness of functional lncRNA annotation. For RNA sequencing data, algorithms for de novo assembly should be considered provided the depth of coverage is sufficient. These programs usually produce output files containing genomic coordinates in *.bed* (browser extendible data file, preferably in 12-field format with exon boundary information), *.gtf* (gene transfer format), *.gff* (general feature format), or similar formats. The popular *Cufflinks* program from the *Tuxedo* suite of RNAseq tools [34] produces a *.gtf* file and includes the appropriate software—a program called *gffread* located in the Cufflinks binary folder—to extract and process sequence information from a reference genome into a *.fasta* file. Alternatively, the *Trinity* program for de novo transcriptome assembly without aligning to a reference genome [35] directly outputs a *.fasta* file of assembled transcripts from the *.fastq* files containing deep sequencing data.

### 2.3 Multiple Genome Alignments

Comparative genomics approaches for functional annotation of noncoding RNAs require pairwise or multiple genome alignments for the species of interest. Prealigned genomic sequence alignments for most well-studied vertebrates can be downloaded in *.maf* (multiple alignment format) from the ENSEMBL comparative genomics database [36] or from the UCSC genome browser [37]—which also hosts alignments for nonvertebrate species—as follows:

1. ENSEMBL Compara—Information about downloading multiple genome alignments is available at http://ensembl.org/info/data/ftp/index.html. Multiple alignments in *.maf* from the latest release at the time this was written can be downloaded

via FTP protocol at ftp://ftp.ensembl.org/pub/release-85/maf/ensembl-compara/multiple_alignments/.

2. UCSC Genome Browser—Navigate to the table browser tab at http://genome.ucsc.edu (select 'tools,' then 'table browser' from the drop-down menu bar on the top of the page). Select the reference species of interest, then 'Comparative Genomics' from the group menu, 'Conservation' from the track menu, and 'Multiz Align,' form the table menu. Optionally, regions can be limited to an existing UCSC or custom track (which needs to be uploaded independently prior to this step). This can significantly reduce the size of the download when only interested in a set of transcripts, for example. Next, ensure that 'MAF—multiple alignment format' appears in the output format menu, otherwise the appropriate track or table must be selected. Finally, name the output file and get output (ideally, compressed) or send the output to the Galaxy [38] platform for post-processing (*see later*).

Multiple alignments form the UCSC Genome Browser employ a different synteny and alignment algorithm than those from ENSEMBL. The latter usually present contiguous alignments for large syntenic blocks via the *Enredo* (or *Mercator*) and *Pecan* algorithms [39, 40], whereas the former is optimized for total genomic coverage and presents smaller, fragmented alignment blocks as produced with the *TBA* and *MULTIZ* algorithms [41]. Because of their highly fragmented nature and variable presence of each species in each block, TBA/MULTIZ alignments may require additional processing, such as being 'stitched' together. A good summary of approaches for processing *.maf* files is described by Blankenberg et al. [42]. The ENSEMBL alignments require less processing, as the syntenic blocks are much longer. These alignments can also contain segmental duplications, which should be removed at the user's discretion (ensuring that the coordinates of the segmental duplications for the reference species are saved for future reference).

## 3 Methods

The first step in any analysis of a putative noncoding RNA is to estimate its protein-coding potential. This typically involves excluding known protein-coding genes from a reference genome annotation, from mass spectrometry data (when available), as well as computational estimation of coding potential via the analysis of open reading frames and evolutionary information, such as synonymous codon usage. The *Pinstripe* software suite is one example of a recently developed bioinformatics resource that enables the discrimination of coding versus noncoding transcripts, which is accompanied by a well-described usage manual [12]. Such methods

and additional considerations—i.e., bifunctional RNA transcripts that are both mRNAs and ncRNAs—are reviewed in [43, 44].

There are two general approaches for the functional annotation of noncoding RNAs: (1) homology search against known RNAs; and (2) de novo identification of putative functional domains. The former is more suitable for the annotation of small RNAs (e.g., tRNAs, snoRNAs, 5S rRNAs, snRNAs, miRNAs, etc.); however, an increasing number of lncRNAs have been sufficiently characterized and are amenable to this approach (*see* [45] and the most recent release of RFAM). De novo computational annotation of noncoding RNAs can be applied to both size categories of transcripts and involves the elucidation of both sequence and structural characteristics that are indicative of function. Comparison of sequence similarity to orthologous genes, for instance, with BLAST [46], is a commonly employed method for the identification of protein-coding genes and ribosomal RNAs given their strong dependence on sequence composition as well as crucial cellular functions. However, when comparing genes with similar functions across larger evolutionary distances, sequence homology is outclassed by structural homology, where classical sequence alignment methods are inefficient. Hidden Markov models [47, 48] and codon substitution matrices (e.g., PAM [49] or BLOSUM [50]) are employed to overcome the sequence alignment barrier when faced with greater sequence divergence than for protein-coding genes.

For noncoding RNAs, alternative computational strategies must be employed to overcome the increased diversity of sequences that are compatible with a given secondary or tertiary structure. The evolutionary dynamics of noncoding RNAs are governed by three factors: (1) They do not require the preservation of sequence composition to convey a genetic code, i.e., codons, with the notable exception of the anticodon loop in tRNAs. (2) RNA structures are more tolerant to nucleotide substitutions than proteins for mutated codons. Indeed, 6 out of 16 possible canonical ribonucleotide combinations will form canonical base pairings, which include Watson–Crick and G-U/U-G 'wobble' base pairs. Because RNA structures can accommodate a higher frequency of base substitutions than mRNAs—as long as they are consistent or compatible with their paired nucleotide—bioinformatics tools investigating noncoding RNAs must focus on secondary and tertiary structural characteristics as well as primary sequence, where short patches of high conservation may indicate important biochemical interactions. (3) Since their biological function is often of regulatory nature, they are more likely to be under positive selection for adaptive radiation. This is most notable for lncRNAs.

### 3.1 Detecting Homology to Known Functional RNAs

The RFAM database encompasses several well-characterized noncoding RNA families that are presented in multiple alignments based on both their sequence and higher order structure topologies [51]. Until recently, the RFAM repository was mostly limited

to entire RNA sequences, mainly small noncoding RNAs. Recent updates to RFAM have expanded the repository to include some lncRNAs as well as *bona fide* RNA structural motifs [52]. The latter are defined as "a non-trivial, recurring RNA sequence and/or secondary structure that can be predominantly described by local sequence and secondary structure elements" and can be part of a larger structure or noncoding RNA [53]. RFAM includes Covariance Models (CMs) for each entry, or family, in the database. CMs are a probabilistic representation of RNA structure profiles that can be used to scan a genome (or transcriptome) for sequences compatible with a given consensus structure. They can be used by the *Infernal* program to scan large metazoan genomes in minutes and report homologous hits with high accuracy [54]. The *Infernal* software package can also generate a CM from a given multiple sequence and structure alignment and thus permits using custom CMs to perform a search. Detailed instructions on how to use *Infernal* can be found at http://infernal.janelia.org/ as well as in [55].

There are also alternative bioinformatics resources for RNA structural homology search. The *RNAmotif* program enables users to construct descriptors of a target RNA structure, then scans a sequence database, and reports all compatible sequences [56]. Although the software is somewhat out of date, *RNAmotif*'s capacity to construct detailed and customized RNA structure descriptors manually and with relative ease justify its pertinence. It also enables the inclusion of tertiary structural elements such as pseudoknots, triplexes, and quadruplexes. Unfortunately, it does not consider thermodynamic stability or base-pairing probabilities and, consequently, can produce a large amount of biologically irrelevant hits unless the results are filtered appropriately (for a practical example of how this may be performed, please refer to the last paragraph of Subheading 3). Alternatively, the recently developed *LocaRNAscan* algorithm [57] can consider the local structural environment in the target sequence when performing a scan using a base pair probability matrix (*see later*) as a query, which can be generated from a single sequence or an alignment of several sequences.

### 3.2 Predicting the Structural Landscape of Individual lncRNA Sequences

The computational prediction of RNA secondary structures from sequence alone was one of the first challenges in bioinformatics. Consequently, modern software packages such as *RNAfold* [58], *UNAfold* [59], and *RNAstructure* [60, 61] are quite efficient at predicting the most thermodynamically stable RNA secondary structure—Minimum Free Energy (MFE)—for a given input sequence. Unfortunately, MFE structural predictions do not always represent the biological reality and, on their own, are not usually considered as a robust qualification of function. This is particularly true for lncRNAs, which can be tens of thousands of nucleotides

long. Locally stable RNA secondary structures, which might compose functional units (or modules) of a lncRNA, can be overlooked in favor of long-range base pairings that contribute more toward lowering the overall free energy score. Furthermore, the dynamic structural nature of RNA macromolecules also confounds RNA structure prediction, as noncoding RNAs can form more than a single functional structural topology (riboswitches are a good example). It is therefore beneficial to consider an ensemble of suboptimal structures when characterizing the function of noncoding RNAs, as exemplified in Fig. 1.

A more biologically relevant alternative to the MFE structure is the centroid, which consists of the structure with minimal distance to all other structures in a set of suboptimal structures. The centroid is usually generated through the partition function, which estimates the statistical distribution of all possible RNA structures within a given thermodynamic range (Boltzmann ensemble). Although centroid estimators have been shown to outperform MFE predictions on known RNA structures [62], they do not necessarily inform about the stability or diversity of the structural landscape for a given query sequence. The latter can be evaluated in two ways: (1) through direct visual inspection of a base-pairing probability matrix, such as that produced by the "*RNAfold –p*" program in the Vienna RNA package (Fig. 1a)—a greater quantity of smaller dots is indicative of a larger diversity of compatible base pairings for a particular nucleotide, which is consistent with a reduced likelihood of forming a stable structure; and (2) through the command-line output of *RNAfold*, or the *RNAfold* webserver [63], which produce a numerical estimate of the ensemble diversity, as well as the frequency of the MFE within the ensemble (i.e., how credible the MFE structure prediction is). A larger ensemble diversity value suggests that the queried RNA sequence may form a broader repertoire of structures or dynamically fluctuate between intermediary structures.

As mentioned earlier, secondary structure prediction of individual lncRNA sequences is not a trivial task. Fortunately, the computational prediction of locally stable structural elements has been shown to be more accurate than global RNA structural predictions for long RNA polymers [64]. This finding is consistent with the general hypothesis that lncRNAs function via local structural (or unstructured) domains, such as protein-binding motifs or RNA–DNA interactions (*see* Subheading 1). *RNAplfold* from the Vienna RNA package [58] and its enhancement in *LocalFold* [64] both offer a useful solution for the manual inspection of local structural topologies in long noncoding RNAs. The tools produce a base-pairing probability matrix that spans the entire RNA sequence but limits the range of base-pairing interactions to a user-definable threshold (Fig. 1d). This facilitates the identification of locally stable (or unstable) structures, which can reveal putative

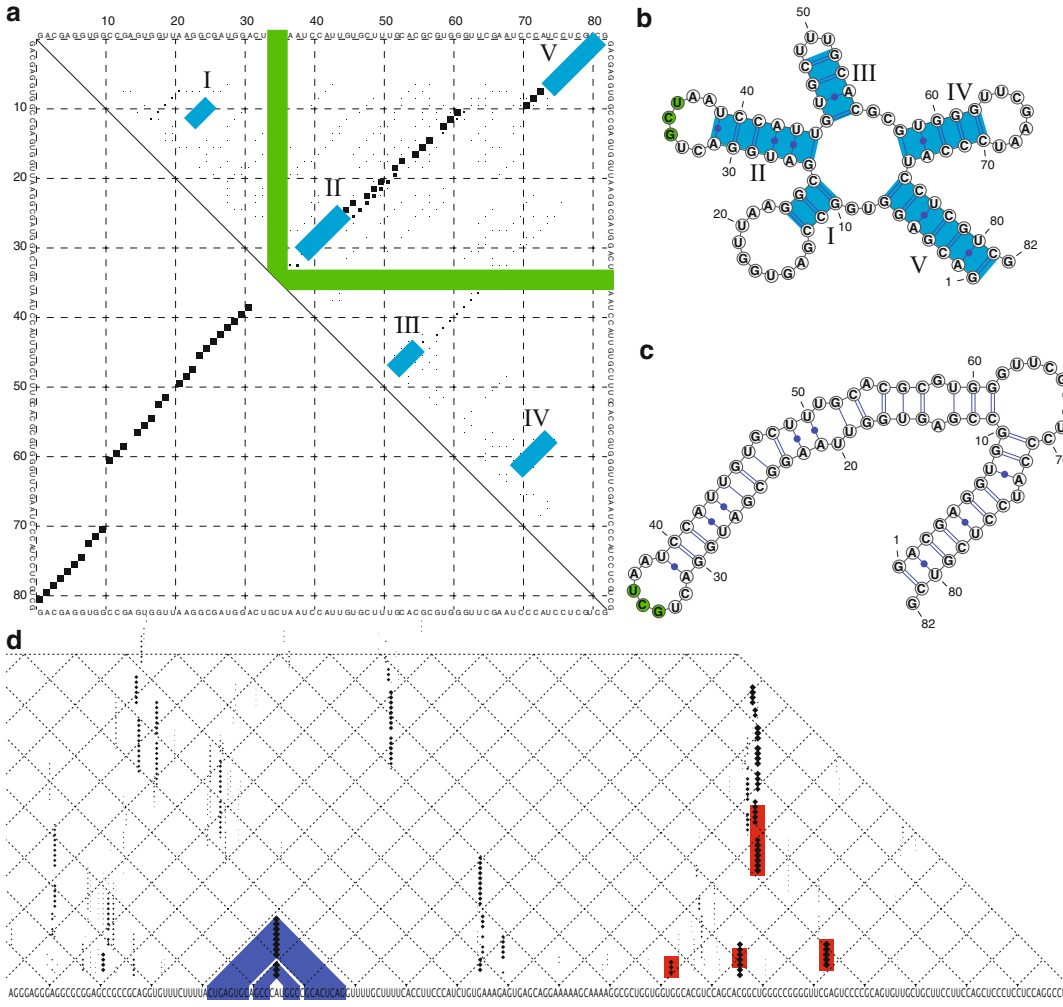



**Fig. 1** Representation of RNA secondary structure predictions for single sequences. (**a**) An RNA base-pairing probability matrix representing both the minimum free energy structure prediction (below the diagonal) and suboptimal base-pairing probabilities (above the diagonal) of a serine tRNA that forms five helices. The RNA sequence of interest is displayed on the *X* and *Y* axes, where each dot represents possible base pairings between bases (*x*,*y*). The size of the dots is indicative of the frequency (or probability) of the base pairings in a Boltzmann ensemble of suboptimal structures, as calculated by McCaskill's partition function algorithm in the Vienna RNA package [58]. The base pairs forming the validated biological structure (**b**) are highlighted in *blue* and numbered accordingly, whereas the unpaired bases forming the anticodon are highlighted in *green*. (**c**) The MFE prediction forms a structure that is quite divergent to the actual tRNA, although the biological structure is perceptible in the suboptimal base pairings. (**d**) A base-pairing probability matrix generated by the *RNAplfold* algorithm on a ~400 nt section of the 3′ end of the NEAT1 lncRNA. Locally stable base pairings are displayed as described for (**a**), however the sequence is represented on the diagonal (i.e., the upper quadrant of (**b**) is rotated 45°). In the lower left, the bases associated to the base pairs (*dots*) are highlighted in *blue*. In the lower right, the tRNA-like structure at the 3′ end of NEAT1 (as illustrated in Fig. 2c) is highlighted in *red*

functional regions as well as guide the design of small interfering RNAs for knockdown experiments. Alternatively, there are software tools, such as *Rnall* [65], *RNAsurface* [66], *RNAlfoldz* (part of the Vienna RNA package [58]), that can facilitate the identification of RNA subsequences presenting strong local structural stability, although a user-defined maximal base-pairing span is required.

### 3.3 Inferring Function from an Individual RNA Sequence

If noncoding RNAs function through the formation of stable secondary structures, can structure predictions alone be used for de novo functional annotation of ncRNAs? This question was first examined over 30 years ago by comparing the RNA structure (or 'folding') score of a native RNA sequence to that of shuffled sequences, under the premise that functional RNAs should form more stable structures than random sequences [67–69]. This strategy produced promising results, but it was consequently shown that the relatively higher stability of native noncoding RNA sequences reflected local biases in sequence composition rather than structural features alone [70]. In particular, the energetic contributions of base-stacking interactions were ignored (the order of consecutively arranged base pairs can significantly alter the free energy score). Some reports have since successfully applied this approach to certain classes of noncoding RNAs by using adequate background models that control for dinucleotide content [71, 72]. Known and novel RNA elements have also been predicted in the yeast genome using a similar strategy, several of which were subsequently experimentally validated [73].

### 3.4 Detecting Functional 2D Motifs via Comparative Genomics

The biological significance of lncRNAs has often been questioned since they (generally) display lower conservation of primary sequence than proteins in evolutionary comparisons [24, 74]. Conservation of RNA secondary or tertiary structure has rarely been considered in such analyses, partially due to the more complex bioinformatic analyses required to investigate such phenomena. However, probing evolutionary data for evidence of RNA structural conservation is not substantially more difficult in practice than evaluating primary sequence conservation. In this section, we describe how to leverage the hallmark signature of RNA structural conservation, i.e., base pair covariation, to identify putative functional RNA motifs in multiple sequence alignments, using existing software.

We recently showed that measuring RNA structure conservation from genomic sequence alignments of 32 mammals could identify evidence of purifying selection on RNA structure motifs that span over 13 % of the human genome, while presenting little overlap with known sequence-constrained regions [33]. Evolutionarily Conserved Structure (ECS) predictions with the human genome as reference can be visualized in the UCSC genome browser (Fig. 2) as follows:

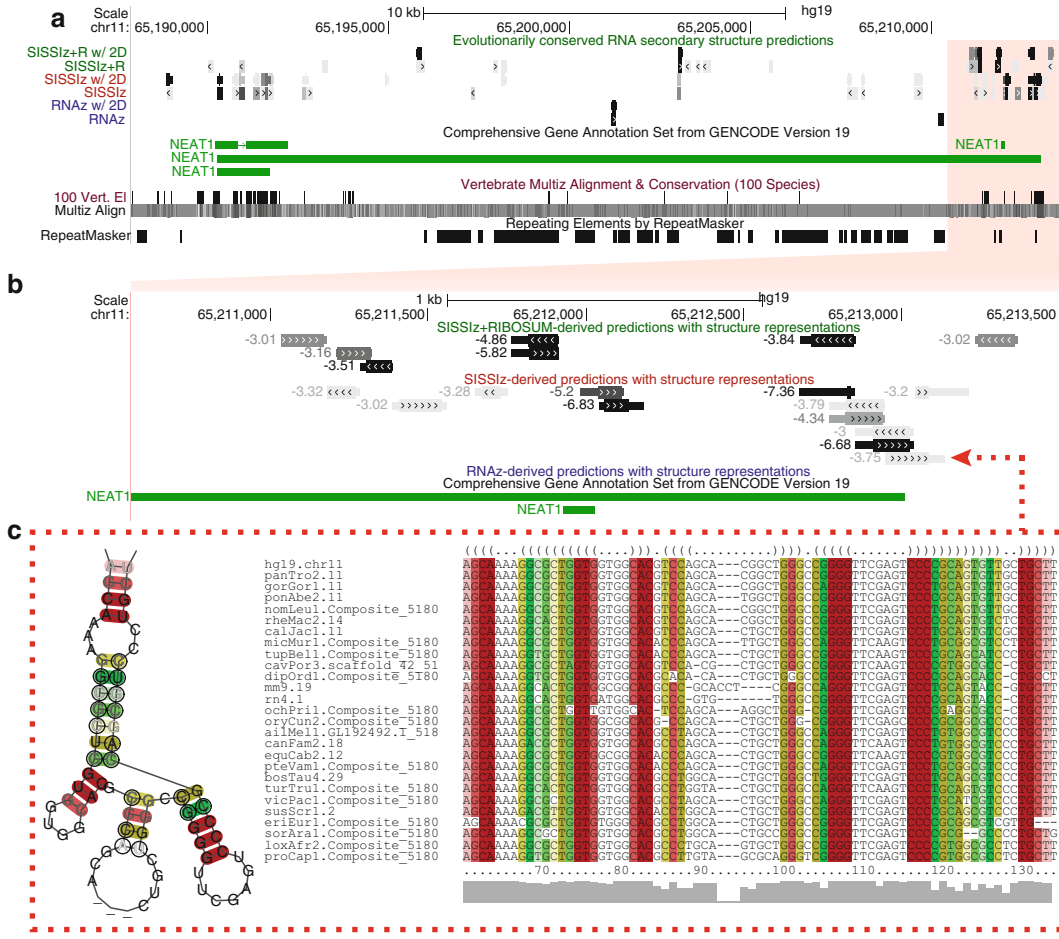



**Fig. 2** Visualization of ECS predictions in the UCSC Genome Browser. (**a**) The NEAT1 lncRNA locus presenting several ECS predictions from [33]. Six subtracks are displayed: *SISSIz*, *SISSIz* with *RIBOSUM* scoring, and *RNAz*-derived results for all significant predictions and those with structure topologies and alignments available to view on a web server (*see* Subheading 3). (**b**) Expanded, zoomed in view of the tracks with structure representations. The RNA secondary structure consensus, flanked by the outermost base pair, is represented by a thicker rectangle. The color of the bars corresponds to a relative measure of their scores (darker = stronger score). (**c**) Detailed illustration of a segment of the predicted structure and alignment obtained by clicking on an ECS prediction from (**b**), which also provides general predictions statistics, a dot-bracket representation of the consensus structure and the consensus sequence generated on the spot via the Vienna RNA package [58]

1. Browse to http://genome.ucsc.edu (or any UCSC Genome Browser mirror), navigate to the 'Genomes' tab, then select the hg19 human genome assembly.
2. Click on the 'track hubs' button, then select the 'My hubs' tab.
3. Paste in the URL for the ECS track hub (http://www.martinalexandersmith.com/hubs/ecs/hub.txt), then 'Add Hub.' The URL can also be obtained via the supplementary information from [33].

4. Browse to any region of interest, zooming out if the ECS track hub titles appear and nothing is displayed under them in the browser (usually, >1 KB of genomic span should be sufficient). ECS predictions are split according to the algorithms that were used to make the predictions (*RNAz*, *SISSIz*, and *SISSIz* + *RIBOSUM*). Although all the ECS predictions are statistically significant (with a $\leq 1$ % false-positive rate), they are color coded based on their relative scores (darker = less likely to arise by chance). After fully expanding the tracks, either by clicking on the title of the track or in the individual track configuration below the browser, the scores associated to the predictions are displayed as the name of each ECS prediction. *SISSIz*-derived predictions will display Z-scores, which represent the degree of observed structural conservation (in number of standard deviations) from the mean of a background distribution produced from *SISSIz*'s null model. There are two subtracks for each employed algorithm: one supporting structure representations, one without. Those with structure representation also have larger segments annotated within individual ECSs; these correspond to the positions within the sampled genomic alignments that contain the outermost base pairs forming the conserved structure prediction (Fig. 1).
5. Expand the ECS track display settings to 'pack' or 'full' view by clicking on the title bar or by selecting the appropriate view in the drop-down menu below the browser interface window.
6. Directly click on a bar corresponding to an ECS prediction of interest. Depending on the nature of the subtrack, this will either: (1) link to a page with a rundown of the statistics for the ECS of interest as well as a description of the methodology; or (2) link to an external page with detailed statistics for the selected ECS, a colored and annotated figure of the consensus secondary structure corresponding, the multiple sequence alignment (colored and annotated) that was used to make the prediction, as well as the consensus structure and sequence in dot-bracket format (Fig. 1c). The ECS tracks with structure representations that link to an external page (as described earlier) will display bars with thin and thick segments; the thinner extremities correspond to regions in the sampled alignment that are not contained within the predicted secondary structure, whereas the thicker internal portion of the bars represents regions contained within the ECS prediction (*see* **Note 1**).
7. Any combination of subtracks (i.e., all ECS predictions, predictions with structure representations, or the results for individual algorithms) can be hidden (or redisplayed) by clicking on the link in the title of the ECS predictions track, located in the drop-down controls section of the UCSC browser below the main window.

There are several caveats pertaining to the data currently contained within the ECS track hub for the UCSC browser. These data are derived from genome-wide screens that are resource intensive and, consequently, were applied to heuristic and not necessarily accurate genome-scale multiple sequence alignments (alignment errors can often be observed via close inspection of alignments from **step 6**). The quality and amount of significant ECS predictions will undoubtedly improve by realigning the queried sequences with more robust algorithms, such as *Clustal Omega* [75], *MAFFT* [76] or, ideally, RNA structure alignment algorithms (reviewed in [77]).

Another caveat is that the above-mentioned ECS predictions are generated from sliding windows of ≤200 nucleotides (nt), which includes multiple genome alignment columns that can primarily be composed of indels. This means RNA base pairs that are more than 200 nt apart are ignored. Furthermore, the sampled alignment windows are offset by 100 nt, therefore conserved RNA structures smaller than 200 nt may also be missed given an incomplete sampling of the structure's boundaries.

An additional issue with the functional annotation of lncRNAs is that many are spliced, often comprising relatively small exons. Although the biological motives for lncRNA splicing remain enigmatic, one possibility is that constitutively spliced exons are joined to maintain the formation of higher order structures, whereas alternatively spliced exons contain self-contained modular units. Probing multiple alignments for evidence of RNA structural conservation in spliced transcripts would thus require pasting the alignment blocks together first (reviewed in [42]), as well as additional considerations like splice site conservation and syntenic continuity in other species.

Performing a de novo scan for ECSs in multiple sequence alignments, either from another reference species or from a set of spliced alignments, can be quite computationally intensive. The approach used for the genomic screen published in [33] can nonetheless be performed by anyone with basic command-line experience. For large alignments (whole genomes or chromosomes)

1. Download and install the following software packages (requires compilation and linking the binaries to the environmental $PATH variable):
   (a) *SISSIz* 2.0 and *RNAz* 2.0 [78] available at http://martinalexandersmith.com/ecs or via links provided in their original manuscripts (N.B. *SISSIz* 2.0 was released in [33]).
   (b) The Vienna RNA package at http://www.tbi.univie.ac.at/RNA [58], preferably version 1.8.5 (newer versions may not be compatible with the software in **step 2**).

2. Download the JAVA archive containing the binary code required to scan .maf files from the following URL (in the software section): http://martinalexandersmith.com/ecs.
3. Ensure that the multiple (genome) sequence alignments have the reference species in the first row with genomic coordinates in the appropriate field of the *.maf* file. This will be used to output the genomic coordinates of the predictions during the scan. Also, ensure that the alignments present sufficiently long blocks (*see* Subheading 2 and **Note 2**).
4. Launching the following command (in the appropriate directory) from a UNIX terminal will provide more verbose information on the basic usage and available parameters: '*java –jar MafScanCcr.jar*.' Some options include window size, step or 'sliding' distance, realignment of the input with the multiple sequence alignment program *MAFFT*, number of processors to use, etc.
5. Execute the program with the selected parameters. The program will load one alignment block of the *.maf* input file at a time, with an optional realignment step to increase accuracy at the expense of computation time. Next, *N* windows are sampled concurrently, where *N* is the number of specified processors (the alignments can also be run in parallel on a computer cluster).
6. The program will save all sampled subalignments that score above the respective thresholds for each employed algorithm. Genomic coordinates associated to significant ECS predictions for the alignment's reference species are also emitted to the standard output in browser extendable (*.bed*) format. Simply redirect the standard output to a file, e.g., '> *output.bed*' from the UNIX terminal. Alternatively, genomic coordinates can be recovered from the file names of the saved alignments, which encode a 6-field underscore delimited bed-compatible entry. Furthermore, the name field of the *.bed* entries also encodes colon-delineated statistical information about the alignment used to make the ECS prediction. This includes (in order):
   (a) Number of retained sequences.
   (b) Raw mean pairwise identity (including indels).
   (c) Mean pairwise identity (normalized to the shortest gapless sequence length).
   (d) Relative gap (indel) content.
   (e) Standard deviation of the (normalized) mean pairwise identity.
   (f) Normalized Shannon entropy.
   (g) Relative GC content.

(h) Scoring algorithm employed: s = *SISSIz* 2.0; r = *SISSIz* 2.0 with *RIBOSUM* scoring; z = *RNAz*-2.0.

The fifth field of the *.bed* entries represents the score associated with the predictions. The scores have been modified to accommodate representation in the UCSC genome browser, which only supports integer values. Z-scores from *SISSIz* predictions are multiplied by −100 (−2.54 = 254), whereas *RNAz*-derived scores are simply multiplied by 100 (0.85 = 85).

7. The topology of a given ECS prediction can be visualized by running the *RNAalifold* program from the Vienna RNA package on the multiple alignment associated to the predicted ECS. The default *RNAalifold* options are suitable for ECS predictions from *SISSIz* and *RNAz*, but the *RIBOSUM* scoring option '*-r*' should be used otherwise.

8. Because the ECS predictions are based on a consensus, it is possible that the reference species forms a structure that is not compatible with the consensus. To evaluate the likelihood of this structural congruence, an auxiliary program is available to process the alignments output from **step 6** (see the supplementary information of [33]). The *ParseAlifold.jar* program performs two main tasks: (1) trimming the genomic coordinates of the reference species to the outermost base pairs of the consensus structure; (2) measuring the relative difference between the native secondary structure for the sampled reference sequence and that produced from constraining the structure to the consensus, as produced from the '*RNAfold −C*' command from the Vienna RNA package [58]. This is done for both the minimum free energy and the base-pairing probabilities generated from the partition function implemented in *RNAfold*, where the probabilities of base pairs from the consensus are extracted from the base-pairing probability matrix. The *.bed* 6 plus formatted output prints to the terminal's standard output and contains the following additional fields:

   (a) Average base-pairing probability of the minimum free energy structure for the reference species. If the base is unpaired, this value is calculated as 1—the sum of all probabilities for the given base.

   (b) Average base-pairing probability of consensus-constrained reference structure.

   (c) Base-pairing probability ratio (constrained/native).

   (d) Free energy (kcal/mol) of the consensus-constrained reference sequence.

(e) Minimum free energy (kcal/mol) of the native reference sequence.

(f) Free energy ratio (constrained/native).

(g) Length of prediction (nt).

(h) Dot-bracket secondary structure mask of *RNAalifold* consensus. Ex: (((((...))))).

### 3.5 The Next Frontier: Functional Parsing of lncRNAs

In higher eukaryotes, recurring RNA structural motifs that display evidence of evolutionary conservation provide a tangible basis for the functional annotation of noncoding sequences, as they may indicate protein-interaction domains that potentially nucleate regulatory networks. For example, Parker et al. [79] performed a similar analysis using evolutionarily conserved RNA secondary structures predicted with *EvoFold* [80] to generate profile Stochastic Context-Free Grammars (SCFGs), which were then used to scan the human genome for paralogs to the RNA structural predictions. The results were grouped into RNA families based on their structural similarities and revealed 220 families of RNA structures, including 172 novel RNA structure families.

However, as effective as bioinformatic methods may be, they seldom indicate what biological functions or processes are involved (unless, of course, there is a high level of homology to well-characterized RNAs). Assigning biological functions to novel RNA structural motifs can be achieved via modern experimental techniques predicated on high-throughput sequencing, such as RNA immunoprecipitation (RIP-Seq), crosslinking immunoprecipitation (CLIP-Seq), and chromatin isolation by RNA purification (ChiRP-Seq). These methods can identify the RNAs interacting with specific proteins, providing sets of RNA sequences that share the same protein-binding characteristics. The increasing availability of next-generation sequencing technologies will likely increase contributions to public specialized databases such as *starBase* [81], which contains numerous RNAseq data sets relating to RNA–protein interactions. Mining these data with advanced bioinformatics tools will bridge the gap between functional annotation of lncRNAs and RNA structure prediction.

Computational identification of RNA structures common to a set of sequences can currently be performed via clustering algorithms based on pairwise comparison scores, obtained through either RNA structure alignment algorithms (e.g., *CARNA* [82], *LocaRNA* [83], *FOLDALING* [84, 85]) or other secondary structure comparison strategies (e.g., *GraphClust* [86], RNACluster [87], and *NoFold* [88]). These approaches have been applied to small RNA sequences and have successfully identified both known and yet to be characterized noncoding RNA families based on their shared secondary structures [79, 83, 85, 87, 88]. Unfortunately, lncRNA sequences are not directly amenable to such structure-motif enrichment approaches because they may harbor extraneous

sequence elements, thus requiring additional processing such as the extraction of subsequences presenting stable RNA structure domains. Refining the aforementioned methods and applying them to sequencing data that target RNA–protein interactions will help identify new functional RNA structure motifs, which can, in turn, serve to index genomic sequences. This strategy will lay the foundations required to unravel the structure–function relationships of lncRNAs, categorize their repertoires, and annotate the expanses of noncoding sequences in vertebrate genomes.

## 4 Notes

1. *Sense or antisense?* Given the complementary nature of canonical RNA base pairs (G–C/C–G), it is not uncommon to find that both strands of DNA produce high scoring, consensus secondary structure predictions. When these bidirectional structure predictions arise in regions with little or no associated transcription, determining the most likely orientation of the putative transcript can be quite difficult. Sequences with high GC content are more susceptible to this phenomenon because there are fewer G–U base pairs, which can effectively be used to discriminate the host transcript's orientation (the antisense A–C base pair does not contribute to canonical Watson–Crick base pairing). Occasionally, visual inspection of the alignments and consensus RNA secondary structures can be sufficient to identify the most likely orientation, i.e., the strand that produces more base pairs (G–U in particular). Otherwise, the most likely orientation can sometimes be determined by using the *RNAstrand* program [89], a machine learning algorithm which was specifically developed for this purpose (not covered here). *RNAstrand* generates a score which estimates the orientation of a consensus RNA secondary structure from a given multiple sequence alignment used as input.
2. *Genomic alignments and block sizes.* As a strict minimum, the blocks should be at least the length of the window size for sampling structure conservation (by default, 200 nt). The longer the blocks are, the more consecutive overlapping windows will be sampled, which will provide greater genomic coverage of the computational screen. Usually, alignments with more species will present shorter blocks given the greater diversity of synteny. In this case, 'stitching' the alignment blocks together can also abrogate synteny in nonreference sequences (i.e., all but the first row in the alignment), which may introduce uncertainty in the consensus structure evaluation as noncontiguous sequences are treated as contiguous. For example, a 500 nt segment from human chromosome 12 might align to a

250 nt segment from mouse chromosome 3 and 250 nt from mouse chromosome 6, therefore any windows sampled between the segment joining both mouse chromosomes will not reflect the biological reality (unless these regions are prone to fusion or trans-splicing events, an unlikely predicament). From a practical viewpoint, the multiple genome alignments produced by the *Enredo-Pecan-Ortheus* pipeline [39, 90] (available via the ENSEMBL comparative genomics portal: http://ensembl.org/info/genome/compara/index.html) present much longer syntenic blocks than those from *TBA/Multiz* [41] (accessible via the UCSC Genome Browser comparative genomics tracks), thus avoiding the need to 'stitch' several small alignments together.

## References


1. Liu G, Mattick JS, Taft RJ (2013) A meta-analysis of the genomic and transcriptomic composition of complex life. Cell Cycle 12 (13):2061–2072
2. Taft RJ, Pheasant M, Mattick JS (2007) The relationship between non-protein-coding DNA and eukaryotic complexity. Bioessays 29 (3):288–299
3. Djebali S, Davis CA, Merkel A et al (2012) Landscape of transcription in human cells. Nature 489(7414):101–108
4. Mercer TR, Gerhardt DJ, Dinger ME et al (2012) Targeted RNA sequencing reveals the deep complexity of the human transcriptome. Nat Biotechnol 30(1):99–104
5. Morris KV, Mattick JS (2014) The rise of regulatory RNA. Nat Rev Genet 15(6):423–437
6. Fatica A, Bozzoni I (2014) Long non-coding RNAs: new players in cell differentiation and development. Nat Rev Genet 15(1):7–21
7. Mattick JS (1994) Introns: evolution and function. Curr Opin Genet Dev 4(6):823–831
8. Mattick JS (2001) Non-coding RNAs: the architects of eukaryotic complexity. EMBO Rep 2(11):986–991
9. Mattick JS (2011) The central role of RNA in human development and cognition. FEBS Lett 585(11):1600–1616
10. Mattick JS (2010) RNA as the substrate for epigenome-environment interactions: RNA guidance of epigenetic processes and the expansion of RNA editing in animals underpins development, phenotypic plasticity, learning, and cognition. Bioessays 32(7):548–552
11. Ezkurdia I, Juan D, Rodriguez JM et al (2014) Multiple evidence strands suggest that there may be as few as 19,000 human protein-coding genes. Hum Mol Genet 23(22):5866–5878
12. Gascoigne DK, Cheetham SW, Cattenoz PB et al (2012) Pinstripe: a suite of programs for integrating transcriptomic and proteomic datasets identifies novel proteins and improves differentiation of protein-coding and non-coding genes. Bioinformatics 28(23):3042–3050
13. Mercer TR, Mattick JS (2013) Structure and function of long noncoding RNAs in epigenetic regulation. Nat Struct Mol Biol 20 (3):300–307
14. Koziol MJ, Rinn JL (2010) RNA traffic control of chromatin complexes. Curr Opin Genet Dev 20(2):142–148
15. Mattick JS, Amaral PP, Dinger ME et al (2009) RNA regulation of epigenetic processes. Bioessays 31(1):51–59
16. Wang KC, Chang HY (2011) Molecular mechanisms of long noncoding RNAs. Mol Cell 43(6):904–914
17. Li L, Chang HY (2014) Physiological roles of long noncoding RNAs: insight from knockout mice. Trends Cell Biol 24(10):594–602
18. Mattick JS (2009) The genetic signatures of noncoding RNAs. PLoS Genet 5(4):e1000459
19. Quek XC, Thomson DW, Maag JL et al (2014) lncRNAdb v2.0: expanding the reference database for functional long noncoding RNAs. Nucleic Acids Res 43:D168–D173. doi:10.1093/nar/gku988
20. Sauvageau M, Goff LA, Lodato S et al (2013) Multiple knockout mouse models reveal lincRNAs are required for life and brain development. Elife 2:e01749
21. Rinn JL, Kertesz M, Wang JK et al (2007) Functional demarcation of active and silent chromatin domains in human HOX loci by noncoding RNAs. Cell 129 (7):1311–1323

22. Wang KC, Yang YW, Liu B et al (2011) A long noncoding RNA maintains active chromatin to coordinate homeotic gene expression. Nature 472(7341):120–124
23. Ulitsky I, Shkumatava A, Jan CH et al (2011) Conserved function of lincRNAs in vertebrate embryonic development despite rapid sequence evolution. Cell 147(7):1537–1550
24. Johnsson P, Lipovich L, Grander D et al (2014) Evolutionary conservation of long non-coding RNAs; sequence, structure, function. Biochim Biophys Acta 1840 (3):1063–1071
25. Bejerano G, Haussler D, Blanchette M (2004) Into the heart of darkness: large-scale clustering of human non-coding DNA. Bioinformatics 20(Suppl 1):i40–i48
26. Calin GA, Liu CG, Ferracin M et al (2007) Ultraconserved regions encoding ncRNAs are altered in human leukemias and carcinomas. Cancer Cell 12(3):215–229
27. Stephen S, Pheasant M, Makunin IV et al (2008) Large-scale appearance of ultraconserved elements in tetrapod genomes and slowdown of the molecular clock. Mol Biol Evol 25 (2):402–408
28. Kapusta A, Kronenberg Z, Lynch VJ et al (2013) Transposable elements are major contributors to the origin, diversification, and regulation of vertebrate long noncoding RNAs. PLoS Genet 9(4):e1003470
29. Matylla-Kulinska K, Tafer H, Weiss A et al (2014) Functional repeat-derived RNAs often originate from retrotransposon-propagated ncRNAs. Wiley Interdiscip Rev RNA 5 (5):591–600
30. Smith M, Bringaud F, Papadopoulou B (2009) Organization and evolution of two SIDER retroposon subfamilies and their impact on the Leishmania genome. BMC Genomics 10:240
31. Stombaugh J, Zirbel CL, Westhof E et al (2009) Frequency and isostericity of RNA base pairs. Nucleic Acids Res 37 (7):2294–2312
32. Cruz JA, Westhof E (2009) The dynamic landscapes of RNA architecture. Cell 136 (4):604–609
33. Smith MA, Gesell T, Stadler PF et al (2013) Widespread purifying selection on RNA structure in mammals. Nucleic Acids Res 41 (17):8220–8236
34. Trapnell C, Hendrickson DG, Sauvageau M et al (2013) Differential analysis of gene regulation at transcript resolution with RNA-seq. Nat Biotechnol 31(1):46–53
35. Haas BJ, Papanicolaou A, Yassour M et al (2013) De novo transcript sequence reconstruction from RNA-seq using the Trinity platform for reference generation and analysis. Nat Protoc 8(8):1494–1512
36. Flicek P, Amode MR, Barrell D et al (2014) Ensembl 2014. Nucleic Acids Res 42(Database issue):D749–D755
37. Karolchik D, Barber GP, Casper J et al (2014) The UCSC Genome Browser database: 2014 update. Nucleic Acids Res 42(Database issue): D764–D770
38. Goecks J, Nekrutenko A, Taylor J et al (2010) Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. Genome Biol 11(8):R86
39. Paten B, Herrero J, Beal K et al (2008) Enredo and Pecan: genome-wide mammalian consistency-based multiple alignment with paralogs. Genome Res 18(11):1814–1828
40. Dewey CN (2007) Aligning multiple whole genomes with Mercator and MAVID. Methods Mol Biol 395:221–236
41. Blanchette M, Kent WJ, Riemer C et al (2004) Aligning multiple genomic sequences with the threaded blockset aligner. Genome Res 14 (4):708–715
42. Blankenberg D, Taylor J, Nekrutenko A et al (2011) Making whole genome multiple alignments usable for biologists. Bioinformatics 27 (17):2426–2428
43. Ilott NE, Ponting CP (2013) Predicting long non-coding RNAs using RNA sequencing. Methods 63(1):50–59
44. Dinger ME, Pang KC, Mercer TR et al (2008) Differentiating protein-coding and noncoding RNA: challenges and ambiguities. PLoS Comput Biol 4(11):e1000176
45. Burge SW, Daub J, Eberhardt R et al (2013) Rfam 11.0: 10 years of RNA families. Nucleic Acids Res 41(Database issue):D226–D232
46. Altschul SF, Madden TL, Schaffer AA et al (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res 25(17):3389–3402
47. Eddy SR (1996) Hidden Markov models. Curr Opin Struct Biol 6(3):361–365
48. Krogh A, Brown M, Mian IS et al (1994) Hidden Markov models in computational biology. Applications to protein modeling. J Mol Biol 235(5):1501–1531
49. Dayhoff MO, Schwartz RM, Orcutt BC (1978) A model of evolutionary change in proteins. National Biomedical Research Foundation, Washington, DC
50. Henikoff S, Henikoff JG (1992) Amino acid substitution matrices from protein blocks.

Proc Natl Acad Sci U S A 89 (22):10915–10919
51. Griffiths-Jones S, Bateman A, Marshall M et al (2003) Rfam: an RNA family database. Nucleic Acids Res 31(1):439–441
52. Nawrocki EP, Burge SW, Bateman A et al (2014) Rfam 12.0: updates to the RNA families database. Nucleic Acids Res 43: D130–D137. doi:10.1093/nar/gku1063
53. Gardner PP, Eldai H (2014) Annotating RNA motifs in sequences and alignments. Nucleic Acids Res 43:691–698. doi:10.1093/nar/gku1327
54. Nawrocki EP, Eddy SR (2013) Infernal 1.1: 100-fold faster RNA homology searches. Bioinformatics 29(22):2933–2935
55. Griffiths-Jones S (2005) Annotating noncoding RNAs with Rfam. Curr Protoc Bioinformatics Chapter 12, Unit 12.15
56. Macke TJ, Ecker DJ, Gutell RR et al (2001) RNAMotif, an RNA secondary structure definition and search algorithm. Nucleic Acids Res 29(22):4724–4735
57. Will S, Siebauer MF, Heyne S et al (2013) LocARNAscan: incorporating thermodynamic stability in sequence and structure-based RNA homology search. Algorithms Mol Biol 8:14
58. Lorenz R, Bernhart SH, Honer Zu Siederdissen C et al (2011) ViennaRNA Package 2.0. Algorithms Mol Biol 6:26
59. Markham NR, Zuker M (2008) UNAFold: software for nucleic acid folding and hybridization. Methods Mol Biol 453:3–31
60. Mathews DH (2004) Using an RNA secondary structure partition function to determine confidence in base pairs predicted by free energy minimization. RNA 10(8):1178–1190
61. Mathews DH, Disney MD, Childs JL et al (2004) Incorporating chemical modification constraints into a dynamic programming algorithm for prediction of RNA secondary structure. Proc Natl Acad Sci U S A 101 (19):7287–7292
62. Hamada M, Kiryu H, Sato K et al (2009) Prediction of RNA secondary structure using generalized centroid estimators. Bioinformatics 25(4):465–473
63. Gruber AR, Lorenz R, Bernhart SH et al (2008) The Vienna RNA websuite. Nucleic Acids Res 36(Web Server issue):W70–W74
64. Lange SJ, Maticzka D, Mohl M et al (2012) Global or local? Predicting secondary structure and accessibility in mRNAs. Nucleic Acids Res 40(12):5215–5226
65. Wan XF, Lin G, Xu D (2006) Rnall: an efficient algorithm for predicting RNA local secondary structural landscape in genomes. J Bioinform Comput Biol 4(5):1015–1031
66. Soldatov RA, Vinogradova SV, Mironov AA (2014) RNASurface: fast and accurate detection of locally optimal potentially structured RNA segments. Bioinformatics 30 (4):457–463
67. Seffens W, Digby D (1999) mRNAs have greater negative folding free energies than shuffled or codon choice randomized sequences. Nucleic Acids Res 27 (7):1578–1584
68. Chen JH, Le SY, Shapiro B et al (1990) A computational procedure for assessing the significance of RNA secondary structure. Comput Appl Biosci 6(1):7–18
69. Le SY, Maizel JV Jr (1989) A method for assessing the statistical significance of RNA folding. J Theor Biol 138(4):495–510
70. Rivas E, Eddy SR (2000) Secondary structure alone is generally not statistically significant for the detection of noncoding RNAs. Bioinformatics 16(7):583–605
71. Bonnet E, Wuyts J, Rouze P et al (2004) Evidence that microRNA precursors, unlike other non-coding RNAs, have lower folding free energies than random sequences. Bioinformatics 20(17):2911–2917
72. Clote P, Ferre F, Kranakis E et al (2005) Structural RNA has lower folding energy than random RNA of the same dinucleotide frequency. RNA 11(5):578–591
73. Kavanaugh LA, Dietrich FS (2009) Noncoding RNA prediction and verification in Saccharomyces cerevisiae. PLoS Genet 5(1): e1000321
74. Kutter C, Watt S, Stefflova K et al (2012) Rapid turnover of long noncoding RNAs and the evolution of gene expression. PLoS Genet 8 (7):e1002841
75. Sievers F, Higgins DG (2014) Clustal Omega, accurate alignment of very large numbers of sequences. Methods Mol Biol 1079:105–116
76. Katoh K, Standley DM (2014) MAFFT: iterative refinement and additional methods. Methods Mol Biol 1079:131–146
77. Gorodkin J, Hofacker IL (2011) From structure prediction to genomic screens for novel non-coding RNAs. PLoS Comput Biol 7(8): e1002100
78. Gruber AR, Findeiss S, Washietl S et al (2010) RNAz 2.0: improved noncoding RNA detection. Pac Symp Biocomput, 69–79
79. Parker BJ, Moltke I, Roth A et al (2011) New families of human regulatory RNA structures identified by comparative analysis of vertebrate genomes. Genome Res 21(11):1929–1943

80. Pedersen JS, Bejerano G, Siepel A et al (2006) Identification and classification of conserved RNA secondary structures in the human genome. PLoS Comput Biol 2(4):e33
81. Li JH, Liu S, Zhou H et al (2014) starBase v2.0: decoding miRNA-ceRNA, miRNA-ncRNA and protein-RNA interaction networks from large-scale CLIP-Seq data. Nucleic Acids Res 42(Database issue):D92–D97
82. Sorescu DA, Mohl M, Mann M et al (2012) CARNA—alignment of RNA structure ensembles. Nucleic Acids Res 40(Web Server issue): W49–W53
83. Will S, Reiche K, Hofacker IL et al (2007) Inferring noncoding RNA families and classes by means of genome-scale structure-based clustering. PLoS Comput Biol 3(4):e65
84. Havgaard J, Kaur S, Gorodkin J (2012) Comparative ncRNA gene and structure prediction using Foldalign and FoldalignM. Curr Protoc Bioinformatics Chapter 12, Unit12.11
85. Torarinsson E, Havgaard JH, Gorodkin J (2007) Multiple structural alignment and clustering of RNA sequences. Bioinformatics 23(8):926–932
86. Heyne S, Costa F, Rose D et al (2012) GraphClust: alignment-free structural clustering of local RNA secondary structures. Bioinformatics 28(12):i224–i232
87. Liu Q, Olman V, Liu H et al (2008) RNACluster: an integrated tool for RNA secondary structure comparison and clustering. J Comput Chem 29(9):1517–1526
88. Middleton SA, Kim J (2014) NoFold: RNA structure clustering without folding or alignment. RNA 20(11):1671–1683
89. Reiche K, Stadler PF (2007) RNAstrand: reading direction of structured RNAs in multiple sequence alignments. Algorithms Mol Biol 2:6
90. Paten B, Herrero J, Fitzgerald S et al (2008) Genome-wide nucleotide-level mammalian ancestor reconstruction. Genome Res 18 (11):1829–1843

# Chapter 5

# Construction of Functional Gene Networks Using Phylogenetic Profiles

**Junha Shin and Insuk Lee**

## Abstract

Functional constraints between genes display similar patterns of gain or loss during speciation. Similar phylogenetic profiles, therefore, can be an indication of a functional association between genes. The phylogenetic profiling method has been applied successfully to the reconstruction of gene pathways and the inference of unknown gene functions. This method requires only sequence data to generate phylogenetic profiles. This method therefore has the potential to take advantage of the recent explosion in available sequence data to reveal a significant number of functional associations between genes. Since the initial development of phylogenetic profiling, many modifications to improve this method have been proposed, including improvements in the measurement of profile similarity and the selection of reference species. Here, we describe the existing methods of phylogenetic profiling for the inference of functional associations and discuss their technical limitations and caveats.

**Key words** Phylogenetic profiling, Functional association, Gene network

## 1 Introduction

The discovery of all the functional components of cells and the elucidation of all their interactions are the grand challenges in systems biology. Phylogenetic profiling [1–3] is a method in which interactions between genes are inferred using their similarity in inheritance across species. During speciation, genetic information about functional components, such as proteins, is passed from ancestral species to new species. Given that most, if not all, cellular processes are operated by a set of genes, which are often represented as a complex or a pathway, the functional interdependence among genes is often coinherited. Functional constraints during speciation therefore would display a similar phylogenetic pattern across species, which provides the opportunity to infer functional association between genes.

Large-scale gene networks, which can be constructed from the functional associations inferred from various types of biological

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_5, 

data, including phylogenetic profiles, have proven useful to the study of gene and pathway functions [3, 4]. The propagation of function and phenotype information through the network facilitates the identification of novel gene functions and/or loss-of-function phenotypes [5]. Given that phylogenetic profiling methods require only sequence data, these methods are likely to benefit significantly from the recent advances in genome sequencing technology such as next-generation sequencing. The rapid growth of the number of genome-sequenced species potentiates the power of phylogenetic profiling methods, because additional species can fill the current gaps in knowledge about the evolutionary trajectories of cellular functions. The expansion of genome projects therefore can make a direct contribution to the understanding of gene and pathway functions.

Since the initial development of phylogenetic profiling methods [2, 3], various approaches have been explored to improve the performance of these methods. These approaches differ mainly with respect to the measurement of profile similarity and the selection of reference species. In this chapter, we will describe conceptual and technical differences among these methods as well as their strengths and weaknesses. In addition, we will discuss the limitations and caveats in the construction of gene networks using phylogenetic profiling methods.

## 2 Materials

Proteins are the major biomolecule in cellular processes. Amino acid sequences therefore may be more relevant to biological function than nucleic acid sequences. Hence, we generally use protein sequences to profile functional conservation across species. To look for protein conservation between species, we generally use the standard sequence homology search software, BLAST (Basic Local Alignment Search Tool).

### 2.1 Protein Sequence Data

Protein sequence data for completely sequenced species are available from major archive databases such as the National Center for Biotechnology Information (NCBI, ftp://ftp.ncbi.nlm.nih.gov/genomes), the European Bioinformatics Institute—European Nucleotide Archive (EBI-ENA, ftp://ftp.ebi.ac.uk/pub), and the Ensembl Genome Browser (ftp://ftp.ensembl.org/pub). Although protein sequence data are generally provided in FASTA format (*see* **Note 1**), the file extensions of FASTA protein sequence files differ across data providers (e.g., *.faa* from NCBI and *.pep.all.fa* from EBI). These sequence data originate from public data repositories for genome projects that are maintained by either genome sequencing centers or genome project consortiums (Table 1). These repositories are replenishing sources for genome

**Table 1**
**Public data repositories for genome sequences**

| Epository | URL |
|---|---|
| *Maintained by genome sequencing centers* | |
| The Broad Institute (BI) | http://www.broadinstitute.org/scientific-community/data |
| Department of Energy Joint Genome Institute (DOE-JGI) | ftp://ftp.jgi-psf.org/pub/JGI_data |
| The J. Craig Venter Institute (JCVI) | ftp://ftp.jcvi.org/pub/data |
| Génolevures | http://www.genolevures.org/download.html |
| Genoscope | http://www.genoscope.cns.fr/spip/Genoscope-s-Resources.html |
| The Beijing Genomics Institute | ftp://ftp.genomics.org.cn/pub |
| The Genome Database for Rosaceae (GDR) | http://www.rosaceae.org/ |
| VectorBase | https://www.vectorbase.org/downloads |
| *Maintained by project consortiums* | |
| Consensus CDS Project (CCDS) | ftp://ftp.ncbi.nlm.nih.gov/pub/CCDS/ |
| Saccharomyces Genome Database (SGD) | http://www.yeastgenome.org/download-data |
| Wormbase | ftp://ftp.wormbase.org/pub/wormbase/species/ |
| FlyBase | ftp://ftp.flybase.org/genomes/ |
| The Arabidopsis Information Resource (TAIR) | ftp://ftp.arabidopsis.org/home/tair/Sequences/ |

project data that recently have been completed. Note that when we use sequence data obtained directly from the original repository, we need to confirm the data publication policy. Some genomes that recently have been sequenced may be under data usage restrictions. These repositories also may contain data from incomplete genome projects. In these cases, genome sequence data are presented as contigs or scaffolds, which may not provide a comprehensive list of proteins for a given species. An incomplete list of proteins may generate inaccurate phylogenetic profiles, which in turn may affect the quality of inferred functional links.

### 2.2 Homology Search Software

Functional conservation between species can be represented by the occurrence of homologous proteins. NCBI BLAST is the most popular software by which to search for homologous proteins across species. Installation files and source codes for the latest version of BLAST are available from the ftp site (ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/).

## 3 Methods

### 3.1 Protein Homology Search Using the Stand-alone BLAST Software

FASTA input files should be carefully prepared to run an efficient homologous search in BLAST. Two input files are required to execute the stand-alone BLAST program, a query-sequence file and a reference-sequence file. The following steps should be taken to run the homology search:

1. Create a query-sequence file that contains query protein sequences, which are usually from a target species, for gene network inference.
2. Create a reference-sequence file of concatenated protein sequences from the genomes of reference species. No specific order for concatenated protein sequences is required, but unique identifiers for each protein sequence are warranted. The assignment of a specific "genome sequence code (GC)" is recommended by users as well as the homology search program. For example, "GC120-077-SEQ0009" stands for "the 9th protein sequence encoded in the 77th reference species out of a total of 120 reference species." Once a GC is assigned to a protein sequence in the reference-sequence file, the FASTA metadata lines of the reference-sequence file need to be modified by adding the GC after the ">" symbol, as shown as follows:

   ```
   >gi|158249333|ref|YP_0015144.1| response regulator
   >GC120-077-SEQ0009  gi|158249333|ref|YP_0015144.1|
   response regulator
   ```

3. Install the stand-alone BLAST program. Installation methods are well described on the BLAST help webpage (http://www.ncbi.nlm.nih.gov/books/NBK52638/) for each operating system.
4. Format both the query-sequence file and the reference-sequence file using the BLAST "formatdb" program by typing the following command line:

   ```
   /[path to blast]/bin/formatdb –i [input FASTA file]
   –p T –o F
   ```

   Factors that can cause errors during the formatdb procedure are described in **Note 2**.
5. Run "blastp," a BLAST program for protein sequences. Commands and switches for executing the stand-alone BLAST programs are well described on the BLAST manual webpage (http://www.ncbi.nlm.nih.gov/books/NBK1763/). Here is an example that uses the default settings:

   ```
   /[path to blast]/bin/blastp –db [reference-
   sequence file] –query [query-sequence file] –out
   [output filename]
   ```

The "blastp" program searches for sequence homology for only one query sequence at a time against all the reference sequences. If parallel searches for many query sequences are intended, then an additional "loop" script for iterative executions is recommended.

6. Obtain the results of the "blastp" analysis, which includes the sequence IDs, homology scores, lengths of the matches, and the actual matched positions for both the query and matched reference sequences.

### 3.2 Construction of the Phylogenetic Profiles

A phylogenetic profile for a query protein is represented as a vector of homology scores across reference species. Two alternative types of homology scores may be used: binary scores for simple occurrences of homology or quantitative measures of homology between a query protein and a protein of the reference species. If the binary score is used, then all the scores of the profile vectors are represented as either a "1" or a "0," which indicate the presence or absence of a homologous protein in the reference species, respectively. The assignment of binary scores based on the BLAST results requires a threshold for the BLAST hit score (e.g., assign "1" if the E-value $< 1e{-}03$). If the quantitative measure is used, then BLAST hit scores are used as the profile score; this score is often transformed for the purpose of mathematical procedures [6]. If multiple homologous proteins exist in a reference species, only a single BLAST hit score for the best homolog remains in the profile. Both binary and quantitative score types have strengths and weaknesses (*see* **Note 3**).

A phylogenetic profile matrix is comprised of the profiles from multiple query proteins (Fig. 1). Note that the order of the reference species (i.e., the order of the columns in the phylogenetic profile matrix) does not affect the measurement of profile similarity, because each reference species is assumed to be orthogonal. It has been reported that the composition of reference species datasets significantly impact the performance of phylogenetic profiling methods in retrieving functional associations between genes [7–10] (*see* **Note 4**).

### 3.3 Measuring the Similarity Between Phylogenetic Profiles

The identification of functional associations between query protein-coding genes can be accomplished by measuring the similarity between phylogenetic profiles. Various similarity measures, such as the Hamming distance [11, 12], the Jaccard coefficient [13, 14], Pearson's correlation coefficient [13], and Mutual Information (MI) [6, 15, 16], have been used in phylogenetic profiling analyses. Different measures focus on different aspect of the profiles. Consequently, similarity scores may differ significantly across measures. Testing multiple measures and then choosing the measure with the best performance for the given profiles will yield the optimal analysis.

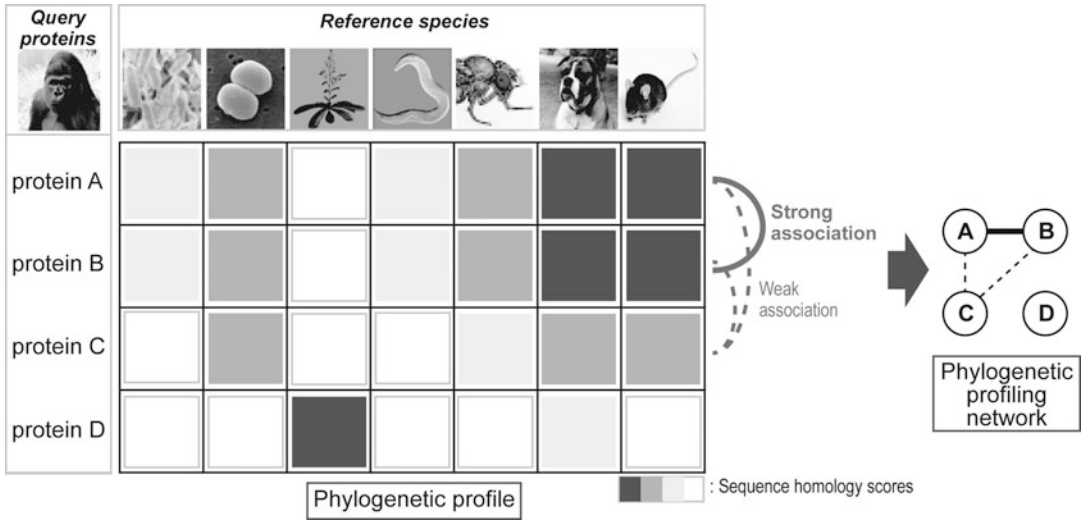



**Fig. 1** A schematic summary of the phylogenetic profile matrix. The phylogenetic profile of a query protein (a row) is a vector that consists of listed scores (gray-scaled rectangle), which indicate the occurrence of a homolog within a reference species (columns). Two proteins with similar profiles (e.g., proteins A and B) are expected to be functionally associated because of their coinheritance pattern

To illustrate how to measure profile similarity based on homology scores in practice, we present a specific analysis that uses MI [6]. MI was developed originally for categorical values. The BLAST scores of profiles, however, are continuous values. The MI calculation therefore requires discretization of the BLAST scores (*see* **Note 5**). The MI can be calculated for the discretized BLAST scores using the following procedures:

1. Calculate the "marginal entropy" and the "joint entropy."

   The marginal entropy of gene A, $H(A)$, is calculated by

$$H(A) = -\sum_{i=1}^{N} p(A_i)\ln p(A_i),$$

   where $N$ is the number of the assigned bins and

$$p(A_i) = \frac{\#\text{ of profile scores that belong to bin } i \text{ for protein A}}{\text{total }\#\text{ of profile scores for protein A}}$$

   The joint entropy between gene A and gene B, $H(A,B)$, is calculated by

$$H(A,B) = -\sum_{i=1}^{N}\sum_{j=1}^{N} p(A_i, B_j)\ln p(A_i, B_j),$$

   where

$$p(A_i, B_j) = \frac{\#\text{ of profile scores that belong to bin } i \text{ for protein A and bin } j \text{ for protein B for the same reference species}}{\text{total \# of profile scores for proteins A and B}}$$

2. Calculate the MI of genes A and B, which is calculated by

$$\mathrm{MI}(A, B) = H(A) + H(B) - H(A, B)$$

3. A gene pair with a higher MI value is more likely to have a functional association.

### 3.4 Benchmarking Functional Associations Inferred by Phylogenetic Profile Similarity

Functional gene networks are highly applicable to the study of cellular systems (*see* **Note 6**). We can construct a network of functional associations by phylogenetic profile similarity. An evaluation of the inferred functional associations is critical to network construction. To benchmark the inferred functional links, we use the "gold standard" (GS) functional associations derived from functional annotation databases, such as the Gene Ontology biological process (GOBP) [17] and the Kyoto Encyclopedia of Genes and Genomes (KEGG) [18], by pairing two genes that share any annotation term. This process generates "GS positives." We can also generate "GS negatives" by pairing two genes that are annotated but do not share any annotation terms.

One simple way to benchmark inferred links using the gold standard set is by using the frequency of the gold standard link among all the inferred links with functional annotations, which is represented by the following equation:

$$\text{Benchmark score} = \frac{\#\text{ GS positives}}{(\#\text{GS positives}) + (\#\text{ GS negatives})}$$

In practice, inferred gene pairs are ordered by decreasing similarity scores. Benchmark scores then are calculated for each bin of 1000 gene pairs from the top scores.

Benchmarking is also useful for finding the optimal analysis condition that achieves the maximal inference power. Many variables and parameters can be selected during the analysis process. These variables include: (1) the composition of the reference species dataset, (2) the types of profile scores (e.g., binary or quantitative), (3) the similarity measures between phylogenetic profiles, and (4) other free parameters. The optimal analysis parameters are those in which the best performance is observed.

### 3.5 Caveats and Limitations

Homologous proteins can be classified into two major classes: orthologs and paralogs. Orthologs are homologous proteins passed from ancestral species to their descendants; these proteins tend to retain their function. In contrast, paralogs are homologous proteins that have appeared as a consequence of gene duplications within a species. A gene duplication event followed by a beneficial modification is a major evolutionary mechanism that creates either a new function (neofunctionalization) or diversifies an existing function

(subfunctionalization). Paralogs therefore tend to have different functions. These functionally divergent paralogs have the same phylogenetic profiles, however, which results in a high similarity score. Paralogous pairs therefore need to be excluded from inferred functional associations. A simple way to detect paralogs is to "self-BLAST" using two identical query sequence files. Alternatively, predefined paralogous relationships can be obtained from databases such as the KEGG Sequence Similarity DataBase (SSDB, http://www.kegg.jp/kegg/ssdb/).

The phylogenetic profiling method relies on homology information; therefore, this method may not be applicable for proteins that lack homology across reference species. This limitation can be overcome by adding more sequenced species to the phylogenetic analysis. For example, the phylogenetic profiling method has not been successful for human proteins. This lack of success may be due to the fact that most of the reference species used in previous analyses have been unicellular microbes. It is expected that recently launched large-scale genome project consortiums such as the 'Genome 10K Project' (https://genome10k.soe.ucsc.edu/) [19], which proposes to sequence 10,000 vertebrates, will improve the application of the phylogenetic profiling method to human proteins in the future.

### 3.6 Case Study: Construction of a Yeast Gene Network Using Phylogenetic Profiles

Here we present an example of a yeast (*Saccharomyces cerevisiae*) gene network to illustrate how to construct a gene network using the phylogenetic profiling method (Fig. 2).

1. Download the yeast protein sequences (i.e., query sequences) from the Saccharomyces Genome Database repository (SGD) [20] and download protein sequences for multiple reference species from the major archive databases, such as NCBI, EBI, and Ensembl. Add systematic genome codes (GC) to the FASTA metadata lines and concatenate all protein sequences of the reference species to create a single 'reference-sequence' file.
2. Format the input sequence files and execute the BLAST program. Exclude insignificant hits (e.g., an E-value $\geq 1$) from the BLAST results.
3. Construct a phylogenetic profile matrix of the BLAST E-values (Fig. 2a). Include only the best hit score for each reference species per query protein in the matrix. If there is no BLAST hit of a query gene for a reference species, assign a score of "1" for that reference species.
4. Calculate the MI scores between all the query protein pairs and sort them in descending order (Fig. 2b).
5. Benchmark the inferred protein pairs with the gold standard functional pairs. The optimal parameters (e.g., the number of bins for the BLAST E-value discretization in the MI calculation) can be determined from the best benchmark curve (Fig. 2c).

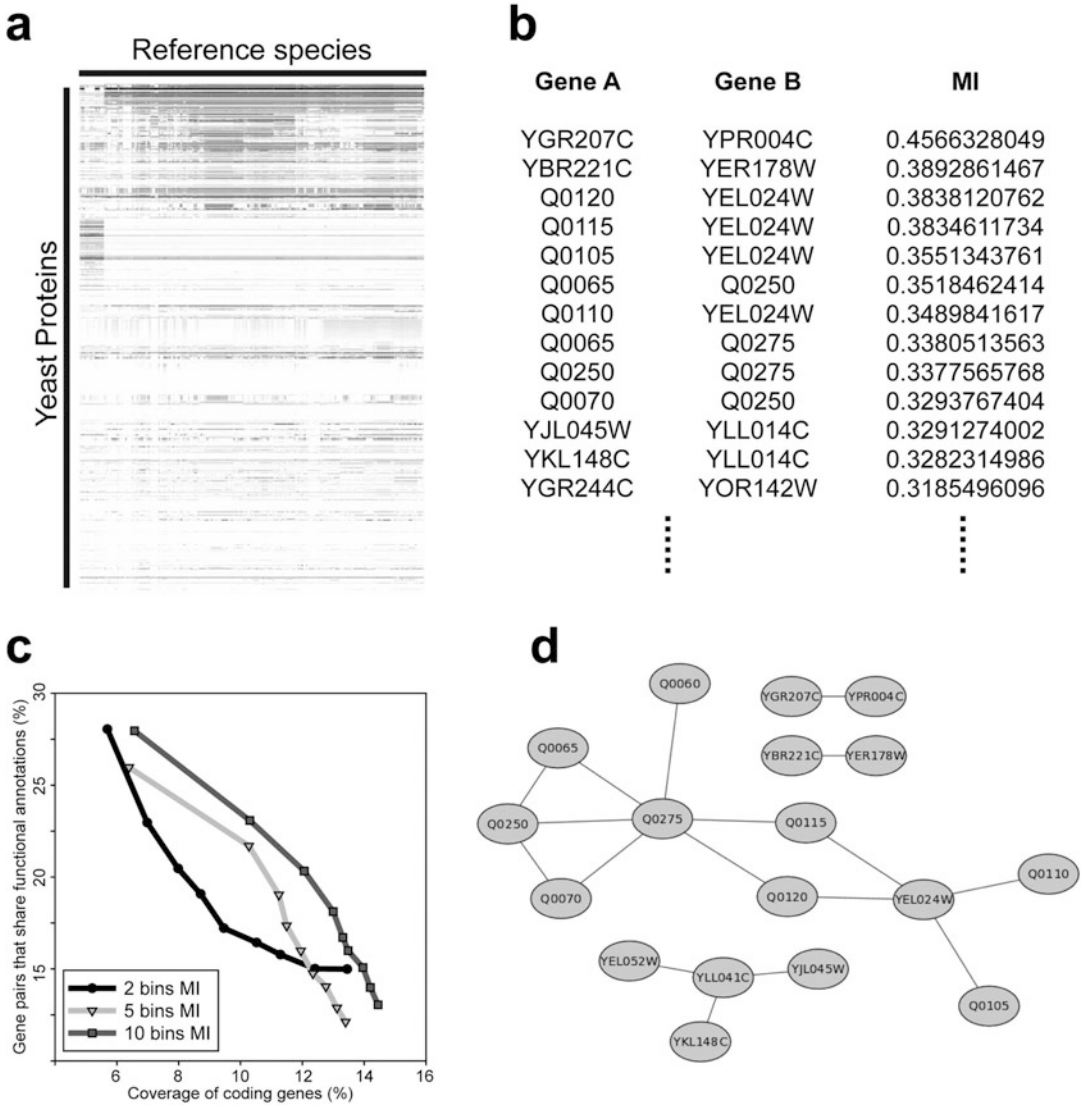



**Fig. 2** Construction of the yeast gene network using phylogenetic profiles. (**a**) A heat-map view of a phylogenetic profile matrix of yeast proteins. The rows represent yeast proteins and the columns represent reference species. The homology score is indicated by the grayscale such that a stronger protein homology (i.e., a lower BLAST E-value) is represented as a *darker color*. (**b**) Yeast gene pairs are listed in descending order of the Mutual Information (MI) scores. (**c**) Benchmark curves for the inferred functional associations between yeast genes from the MI calculation are constructed using different numbers of bins for the score discretization. The percentage of gene pairs that share functional annotations (GS positives) among all annotated gene pairs by the Gene Ontology biological process terms (*y*-axis) is measured for different coverage of all the yeast coding genes (*x*-axis). The best network was inferred using 10 score bins for the MI calculation. (**d**) A part of the inferred gene network was visualized by the Cytoscape program

6. Define a gene network by applying a threshold to the benchmark score. The resulting network is analyzed by various network algorithms and visualized in Cytoscape [21] (Fig. 2d).

## 4 Notes

1. The first line of the FASTA format file starts with a ">" symbol followed by metadata, such as the name of the protein, the name of the origin organism, and a short description of the molecule. The sequence information, which is written in single letters, starts from the next line. The sequences are provided either as a single line for the whole sequence or as multiple text lines of ~60–80 letters. Both styles can be used for BLAST input files.
2. One factor that causes frequent errors during the BLAST "formatdb" process is the existence of the "*" symbol within or at the end of a sequence. For some sequence data repositories, the "*" is used either to mark an unidentified or suspicious amino acid position or to designate the last position in the sequence. The "formatdb" program cannot handle this symbol. If the symbol is located at the last position of the sequence, then it must be removed. If the symbol is located within a sequence, then the entire sequence must be excluded from the analysis. In addition, if the BLAST program runs in a UNIX-affiliated operating system such as LINUX or MAC OS, then "^M" (i.e., "control-M"), which is placed at the end of every text line, also causes errors in the "formatdb" procedure. "^M" represents a carriage return in the DOS system, and UNIX systems do not recognize it. This conflict occurs when the FASTA file is generated in a DOS system. Various ways exist to remove the "^M" from the file, including the use of a shell command "dos2unix."
3. The simplicity of the binary score is advantageous for calculating the profile similarity measure due to the low computational burden. In contrast, quantitative scores provide high-resolution information, which potentially leads to a more accurate measure of similarity between profiles.
4. There exists controversy about the proper composition of reference species datasets. Several studies have reported that phylogenetic profiles consisting of only prokaryotic genomes perform well, and that the addition of eukaryotic genomes reduces performance [7, 9]. In contrast, another study has reported that eukaryotic genomes are improving the performance of phylogenetic profiling methods as the number of completely sequenced eukaryotic genomes grows [8]. Furthermore, the effect of an increased number of reference species and the selection of representative genomes from the reference species on the performance of phylogenetic profiling methods have been investigated thoroughly [10].
5. The BLAST E-value scores of the profiles range from "0" (i.e., perfect hit) to "1" (i.e., no valid hit). The discretization of

continuous scores requires the assignment of score bins. The number of bins can be settled arbitrarily. For example, we may define score bins by dividing the entire score range into equal intervals. The distribution of E-values, however, is skewed toward 1. Equal intervals therefore will result in a heavily biased binning of the scores. To resolve this problem, we fix a bin for all scores of 1 and divide the remaining scores with an equal bin distribution. For example, suppose that the profile matrix consists of ten BLAST E-values: $\{x \mid \text{BLAST E-values of the profile matrix}\} = \{0, 0, 0.2, 0.4, 0.6, 0.8, 1, 1, 1, 1\}$. If we create four score bins (i.e., categories) for discretization, the score sets would be {1, 1, 1, 1}, {0, 0}, {0.2, 0.4}, and {0.6, 0.8}.

6. Network approaches have proven useful for biological studies [22]. For example, the novel functions of a gene can be inferred from network neighbor genes that are functionally annotated using the guilt-by-association principle [23]. A subnetwork structure, which is often called a module, is another useful network feature to investigate functional associations. If a group of genes is highly interconnected, representing a pathway, then this group of genes is likely to operate within the same cellular process. Other genes that are connected to this group may be new members of the cellular process.

## Acknowledgements

This work was supported by grants from the National Research Foundation of Korea (2015R1A2A1A15055859, 2012M3A9B4028641, 2012M3A9C7050151).

### References

1. Kensche PR, van Noort V, Dutilh BE, Huynen MA (2008) Practical and theoretical advances in predicting the function of a protein by its phylogenetic distribution. J R Soc Interface 5 (19):151–170
2. Huynen MA, Bork P (1998) Measuring genome evolution. Proc Natl Acad Sci U S A 95(11):5849–5856
3. Marcotte EM, Pellegrini M, Thompson MJ, Yeates TO, Eisenberg D (1999) A combined algorithm for genome-wide prediction of protein function. Nature 402(6757):83–86
4. Lee I (2011) Probabilistic functional gene societies. Prog Biophys Mol Biol 106 (2):435–442
5. Lee I (2013) Network approaches to the genetic dissection of phenotypes in animals and humans. Anim Cells Syst 17(2):75–79
6. Date SV, Marcotte EM (2003) Discovery of uncharacterized cellular systems by genome-wide analysis of functional linkages. Nat Biotechnol 21(9):1055–1062
7. Jothi R, Przytycka TM, Aravind L (2007) Discovering functional linkages and uncharacterized cellular pathways using phylogenetic profile comparisons: a comprehensive assessment. BMC Bioinformatics 8:173
8. Singh S, Wall DP (2008) Testing the accuracy of eukaryotic phylogenetic profiles for prediction of biological function. Evol Bioinform Online 4:217–223
9. Snitkin ES, Gustafson AM, Mellor J, Wu J, DeLisi C (2006) Comparative assessment of performance and genome dependence among phylogenetic profiling methods. BMC Bioinformatics 7:420

10. Sun J, Xu J, Liu Z, Liu Q, Zhao A, Shi T, Li Y (2005) Refined phylogenetic profiles method for predicting protein-protein interactions. Bioinformatics 21(16):3409–3415
11. Pagel P, Wong P, Frishman D (2004) A domain interaction map based on phylogenetic profiling. J Mol Biol 344(5):1331–1346
12. Pellegrini M, Marcotte EM, Thompson MJ, Eisenberg D, Yeates TO (1999) Assigning protein functions by comparative genome analysis: protein phylogenetic profiles. Proc Natl Acad Sci U S A 96(8):4285–4288
13. Glazko GV, Mushegian AR (2004) Detection of evolutionarily stable fragments of cellular pathways by hierarchical clustering of phyletic patterns. Genome Biol 5(5):R32
14. Yamada T, Goto S, Kanehisa M (2004) Extraction of phylogenetic network modules from prokaryote metabolic pathways. Genome Inform 15(1):249–258
15. Bowers PM, Cokus SJ, Eisenberg D, Yeates TO (2004) Use of logic relationships to decipher protein network organization. Science 306 (5705):2246–2249
16. Huynen M, Snel B, Lathe W 3rd, Bork P (2000) Predicting protein function by genomic context: quantitative evaluation and qualitative inferences. Genome Res 10(8):1204–1210
17. Ashburner M, Ball CA, Blake JA, Botstein D, Butler H, Cherry JM, Davis AP, Dolinski K, Dwight SS, Eppig JT, Harris MA, Hill DP, Issel-Tarver L, Kasarskis A, Lewis S, Matese JC, Richardson JE, Ringwald M, Rubin GM, Sherlock G (2000) Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. Nat Genet 25(1):25–29
18. Kanehisa M, Goto S (2000) KEGG: kyoto encyclopedia of genes and genomes. Nucleic Acids Res 28(1):27–30
19. Genome KCS (2009) Genome 10K: a proposal to obtain whole-genome sequence for 10,000 vertebrate species. J Hered 100(6):659–674
20. Costanzo MC, Engel SR, Wong ED, Lloyd P, Karra K, Chan ET, Weng S, Paskov KM, Roe GR, Binkley G, Hitz BC, Cherry JM (2014) Saccharomyces genome database provides new regulation data. Nucleic Acids Res 42(Database issue):D717–D725
21. Cline MS, Smoot M, Cerami E, Kuchinsky A, Landys N, Workman C, Christmas R, Avila-Campilo I, Creech M, Gross B, Hanspers K, Isserlin R, Kelley R, Killcoyne S, Lotia S, Maere S, Morris J, Ono K, Pavlovic V, Pico AR, Vailaya A, Wang PL, Adler A, Conklin BR, Hood L, Kuiper M, Sander C, Schmulevich I, Schwikowski B, Warner GJ, Ideker T, Bader GD (2007) Integration of biological networks and gene expression data using Cytoscape. Nat Protoc 2(10):2366–2382
22. Barabasi AL, Oltvai ZN (2004) Network biology: understanding the cell's functional organization. Nat Rev Genet 5(2):101–113
23. Aravind L (2000) Guilt by association: contextual information in genome analysis. Genome Res 10(8):1074–1077

# Chapter 6

# Inferring Genome-Wide Interaction Networks

Gökmen Altay and Onur Mendi

## Abstract

The inference of gene regulatory networks is an important process that contributes to a better understanding of biological and biomedical problems. These networks aim to capture the causal molecular interactions of biological processes and provide valuable information about normal cell physiology. In this book chapter, we introduce GNI methods, namely C3NET, RN, ARACNE, CLR, and MRNET and describe their components and working mechanisms. We present a comparison of the performance of these algorithms using the results of our previously published studies. According to the study results, which were obtained from simulated as well as expression data sets, the inference algorithm C3NET provides consistently better results than the other widely used methods.

**Key words** Gene network inference, Gene network inference (GNI) algorithms, Bioinformatics

## 1 Introduction

The inference of gene regulatory networks (GRN), which can be seen as a reverse engineering problem, is a process of estimating direct physical associations among genes from gene expression data [1]. This process can provide valuable information about normal cell physiology, development, and pathogenesis and contribute to a better understanding of biological and biomedical problems [2–4]. Gene network inference (GNI) algorithms are widely used in bioinformatics to detect the activator genes of genetic diseases, to determine the functions of the regulating and regulated genes, and to obtain drug targets [5]. GRNs aim to capture the interactions between molecular entities and are represented as graphs in which nodes represent genes, proteins or metabolites and edges represent molecular interactions [6]. In vivo or in vitro, molecular interactions can be detected accurately by classical molecular biology approaches. Unfortunately these methods are laborious and the number of interactions that can be studied by these approaches is limited [7]. Gene networks such as transcriptional regulatory networks, protein networks, or metabolic networks represent

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_6, 

blueprints of dynamical processes within cells. Different types of gene networks have different effects on the dynamical processes of cellular systems [8, 9]. Hence, gene network inference has been identified as a focal point in systems biology. However, GNI is a challenging problem because of the current very large-scale biological datasets and the noise caused by experimental and computational processes.

The steps of the gene network inference process are shown in Fig. 1. The dataset obtained from microarray data analysis consists of gene expression levels. Firstly, by using these preprocessed expression values, a gene expression matrix is created. In this matrix, each row corresponds to a gene whereas each column corresponds to a sample. The second step is estimating interaction scores of gene pairs. In this step, association score estimators such as correlation-based, entropy-based and direct mutual information (MI) estimators are used to obtain interaction scores. A dataset discretization operation is required in order to use MI estimators. At the end of the second step, a square gene association matrix is

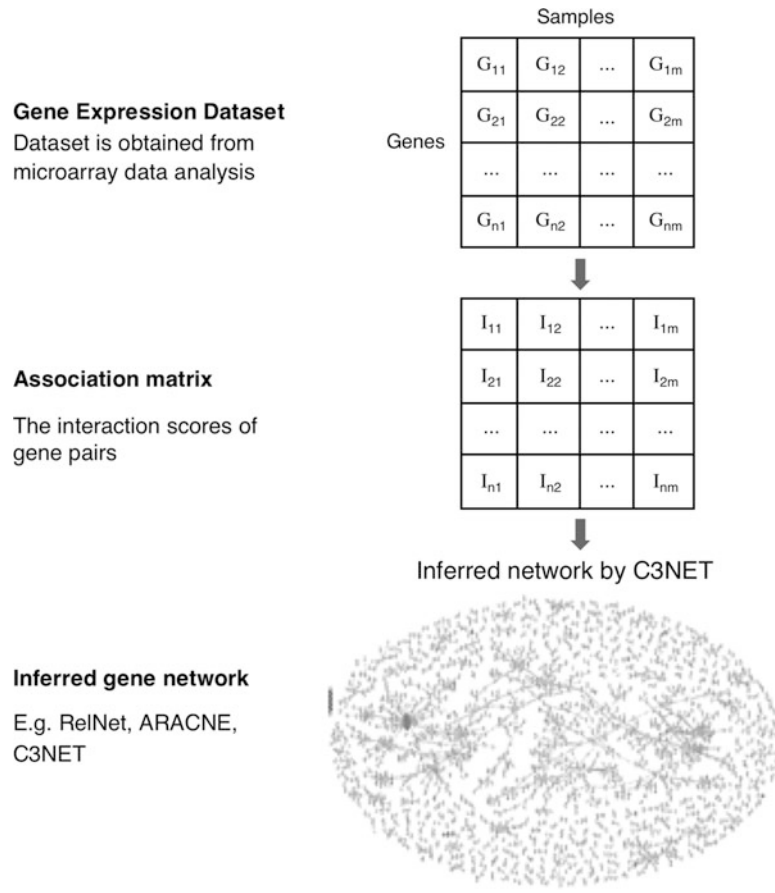



**Fig. 1** The work flow of gene network inference

obtained. Finally, GNI algorithms are applied to this association matrix and the inference of gene regulatory network process is completed.

The most crucial process of GNI algorithms is to obtain the interaction scores among cell molecules. The interaction scores among gene pairs are determined from the gene expression datasets by the association score estimators. However, there is no commonly accepted estimator that is known to provide the best performance for GNI methods. In the study [5], 27 different interaction estimators were reviewed and 14 most promising estimators were evaluated. According to the study results; BS with spline order 2 (BS2), BS with spline order 3 (BS3), Kernel Density Estimator (KDE), Pearson-based Gaussian (PBG), and Spearman-based Gaussian (SPG) were found to be the best association score estimators regarding the performance and runtime (*see* **Note 1**). Therefore, we preferred Pearson-based Gaussian estimator in our study [5, 10].

Several popular techniques have been developed to infer GRNs from microarray gene expression data. The best of these methods are based on information theory [11, 12]. The main principle of information-based methods is estimating mutual information (MI) values among gene pairs [13, 14]. MI based methods are able to detect linear and nonlinear effects among gene pairs [15, 16]. Furthermore, they enable us to work with large sample sizes such as 25,000 genes [17].

One of the first algorithms introduced was RN (relevance network) [18]. This algorithm computes all mutual information values for all pairs of genes and eliminates the edges among genes that have MI values that are not statistically significant. The second well-known method is ARACNE [19]. ARACNE uses data processing inequality and, in addition to RN, ARACNE performs a second step to eliminate the least significant edge of a triplet of genes. This results in a more conservative estimation of the inferred network.

CLR (Context Likelihood of Relatedness) [20] is another method that employs a background sensitive estimator between the gene pairs by converting MI estimates to values similar to z-scores. In contrast to RN and ARACNE, CLR estimates *individual* thresholds by considering an *individual* background for each pair of genes. In addition to these methods, MRNET (maximum relevance/minimum redundancy network) [21] infers a network using the maximum relevance/minimum redundancy feature selection method. Finally, C3NET (conservative causal core network inference) [22, 23] has been introduced. The basic idea of C3NET is selecting the edge for each gene with maximum mutual information (MI) value (*see* **Notes 2** and **3**).

The book chapter is organized as follows. In the next section we introduce GNI methods by describing their components and

working mechanisms. We start with a detailed review of C3NET and then give brief descriptions of the other most widely known inference algorithms, namely RN, ARACNE, CLR, and MRNET. Then we present a comparison of the performance of inference algorithms using the results of the study [22]. In the study, performance was evaluated using simulated as well as expression data from *E. coli*. Finally, we discuss the comparison results.

## 2 Methods

The inference of gene networks from high-throughput data is an important and very complex process. Recent advances in biotechnology enable us to obtain large-scale expression data. The availability of this type of data ushered in the development of gene inference methods [1, 22]. In this section, firstly we demonstrate the inference algorithm C3NET by describing its components and working mechanism. We describe the implementation of the algorithm and usage of its R package. Then, we briefly introduce the other most widely known inference algorithms, namely RN, ARACNE, CLR, and MRNET.

### 2.1 C3NET (Conservative Causal Core Network Inference)

The inference algorithm C3NET consists of two main steps. The first step is the elimination of nonsignificant connections among gene pairs, whereas the second step selects for each gene the edge with maximum mutual information (MI) value [22]. The first step is similar to previous methods, e.g., RN, ARACNE, or CLR. In this step, C3NET tests the statistical significance of pairwise mutual information values using resampling methods and eliminates nonsignificant edges according to a chosen significance level $\alpha$. Mathematical formulation of the mutual information [24] of two random variables $X$ and $\Upsilon$ is defined as

$$I(X,\ \Upsilon) = \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\,\log\frac{p(x,y)}{p(x)p(y)}.$$

In order to calculate a statistical threshold, C3NET uses resampling methods that estimate the distribution under the null hypothesis corresponding to a vanishing mutual information. For this purpose, it randomizes the expression data set by permuting the gene expression measurements $n$ times and recalculating the distribution of the new pairwise mutual information for each permutation. Then C3NET creates a vector combining these permuted mutual information matrices and determines the threshold value according to a chosen significance level $\alpha$. Visualization of this vector is shown in Fig. 2. The vertical ($\Upsilon$) axis represents the frequency of mutual information values, whereas the horizontal ($X$) axis represents mutual information values. The threshold, denoted by $I_C$, is

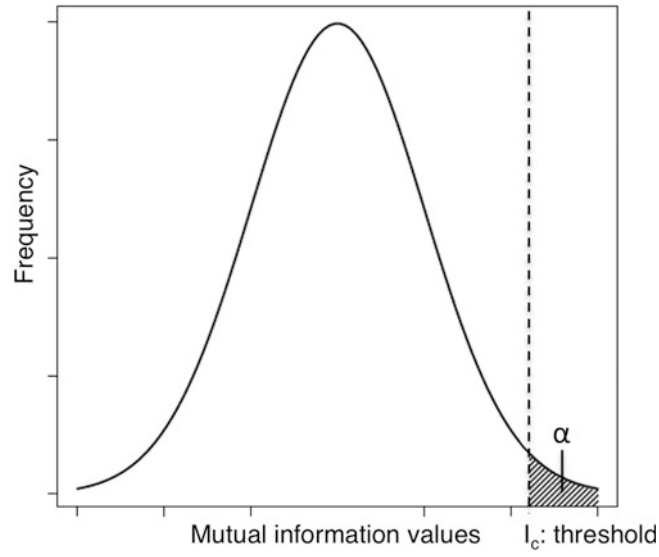



**Fig. 2** Determining MI threshold, $I_C$

1: $B$: initiate adjacency matrix, $B_{ij} = 0$ for all $i, j \in V$
2: $C$: initiate connectivity matrix, $C_{ij} = 0$ for all $i, j \in V$
3: estimate mutual information $I_{ij}$ for all $i, j \in V$
4: **repeat**
5: Set $C_{ij} = 1$ if $I_{ij}$ != 0 is statistically significant (hypothesis test)
6: **until** all pairs $i \neq j$ are tested
7: **for all** $i \in V$ **do**
8: $N_s(i) = \{j: C_{ij} = 1 \text{ and } j \neq i\}$
9: **if** $N_s(i) \neq \varnothing$
10: $j_c(i) = \arg\max_{j \in N_s(i)} \{I_{ij}\}$
11: **else**
12: $j_c(i) = \varnothing$
13: **endif**
14: **end for**
15: **for all** $i \in V$ **do**
16: **if** $j_c(i) = \varnothing$
17: $B_{ij_c(i)} = B_{j_c(i)i} = 1$
18: **endif**
19: **end for**
20: **return** adjacency matrix $B$

**Fig. 3** The principal steps of C3NET. C3NET consists of two main steps. The first step is for the elimination of nonsignificant connections among gene pairs, whereas the second step selects for each gene the edge among the remaining ones with maximum mutual information (MI) value [22]

determined as the maximum mutual information value for the significant region of the null distribution, as illustrated in Fig. 2 by the dashed line.

Figure 3 shows the principle steps of the C3NET algorithm. Primarily, C3NET creates a mutual information matrix (MIM) by

estimating the mutual information values from the data by using an appropriate estimator allowing a close approximation of the theoretical value of the population. Starting from zero matrices $C$ and $B$ (with $C_{ij} = 0$ and $B_{ij} = 0$ for all $i, j \in V$) C3NET thoroughly tests all pairwise mutual information values $I_{ij}$, $i, j \in V$, and sets $C_{ij} = C_{ji} = 1$ if the null hypothesis $H_0$: $I_{ij} = 0$ can be rejected, for a given significance level $\alpha$ [22].

In the second step, the most significant connection for each gene is selected. The algorithm first determines the neighborhood $N_s$ for all genes $i \in V$. The neighborhood of gene $i$ is defined by $N_s(i) = \{j: C_{ji} = 1 \text{ and } j \neq i\}$. For this purpose, it uses the connectivity matrix $C$. The link corresponding to the highest mutual information value in the neighborhood for each gene is determined by using $N_s$ and $I$. This link is identified by

$$j_c(i) = \underset{j \in N_s(i)}{\operatorname{argmax}} \{I_{ij}\}.$$

It is possible that all mutual information values $I_{ij}$ for $j \in V$ are nonsignificant ($N_s(i) \neq \emptyset$). In this case, no index is assigned to $j_c(i)$. The algorithm constructs the adjacency matrix $B$ of the estimated undirected network by setting $B_{ij_c(i)} = B_{j_c(i)i} = 1$ if $j_c(i)$ has been set to a valid index. The rest of the entries of $B$ remain zero [22].

A visualization of the principal working mechanism of C3NET is shown in Fig. 4. Suppose that we have the mutual information values given by $I$. The mutual information values which are statistically significant appear as "1" entries, whereas the remaining ones appear as "0" entries in the corresponding connectivity matrix $C$.

$$I = \begin{pmatrix} 1.0 & 0.9 & 0.7 & 0.1 \\ 0.9 & 1.0 & 0.8 & 0.7 \\ 0.7 & 0.8 & 1.0 & 0.4 \\ 0.1 & 0.7 & 0.4 & 1.0 \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}.$$

Then the algorithm determines statistically significant connections with neighboring genes with maximum mutual information. This is the critical step in C3NET, resulting in $j_c = (1, 2, 2, 2)$. The next

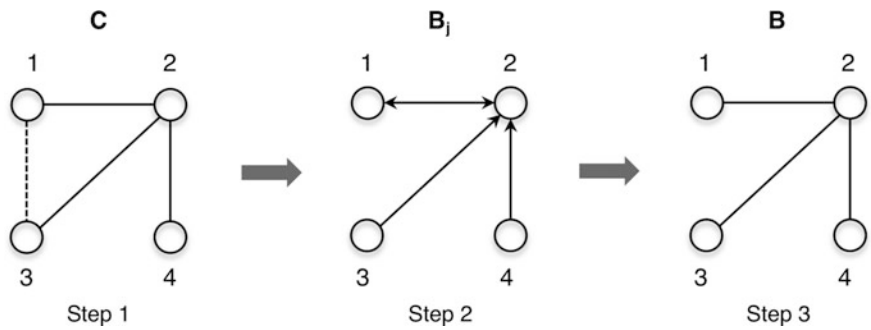



**Fig. 4** Visualization of the principal working mechanism of C3NET. The edges shown in *solid* and *dashed lines* correspond to significant edges. In the third step, the edges in solid lines correspond to the edges with maximum mutual information value

step is determining auxiliary matrix $B_{j,}$ directly from $j_c$. $B_j$ contains exactly the edges added by each node. Due to its symmetry in its arguments, MI does not provide directional information, so the resulting adjacency matrix, $B$, is symmetric.

$$B_j = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

The resulting network represented by adjacency matrix $B$ is a star-like network where gene 2 is connected to three other genes. It is obtained from the conversion of the asymmetric matrix $B_j$ to a symmetric matrix $B$ as shown in the example of Fig. 4. It is important to realize that each gene can add at most one connection, but different genes $i$ can select the same gene $j_c(i)$. For this reason, the final undirected network can consist of genes having more than one connection to other genes (*see* **Note 4**).

### 2.1.1 Implementation of C3NET: Usage of the R package

An R package called *c3net* is available from the website https://r-forge.rproject.org/ projects/c3net and also downloadable through the CRAN package repository. To illustrate the principal working mechanism of C3NET, an example data set is provided in the R package. This package includes both experiment and true network data which can be loaded in R by executing the *data(expdata)* and *data(trunet)* commands of C3NET. There is a core function available in *c3net* package that takes the data set as input and outputs the inferred network. This function hides individual steps of *c3net* and provides an inferred network in an all-in-one single command. An example usage of this function and its default parameters is as follows [23]:

```
c3net(dataset, alpha = 0.01, methodstep1 = "MTC", MTCmethod
= "BH", itnum = 5, network = TRUE)
```

The first parameter *dataset* is the data set where rows are variables (e.g., genes) and columns are samples. The second parameter *alpha* is a user defined statistical significance threshold. The *methodstep1* parameter is set to define the procedure that will be used to eliminate nonsignificant edges in **Step 1** of C3NET. {"cutoff", "MTC", and "justp"} are the options that can be used for the parameter *methodstep1*. If *cutoff* and *MTC* options are selected, then "*cutoffMI*" or "*MTCmethod*" additional parameters must be set, respectively. If *methodstep1* = "*cutoff*", then it is mandatory *cutoffMI* needs to be set to a numerical value that is used as the cutoff value to eliminate nonsignificant MI value of edges. The *cutoffMI* value can be set to 0 to use default mean MI as *cutoffMI*.

In case MTC option is used as *methodstep1,* then *MTCmethod* needs to be set to employ the specific multiple testing correction method. The six available MTC options are; (1) Benjamini and Hochberg (*"BH"*), (2) Bonferroni (*"bonferroni"*), (3) Benjamini and Yekutieli (*"BY"*), (4) Hochberg (*"hochberg"*), (5) Holm (*"holm"*), and (6) Hommel (*"hommel"*). Additionally, *itnum* parameter needs to be set to specify the number of iterations to obtain a null distribution and *alpha* the statistical significance level. If *methodstep1* = *"justp"*, then only *alpha* and *itnum* need to be set and the elimination step of C3NET is done only with the *p*-values and the significance level of $\alpha$ [23].

Besides providing the inference procedure of C3NET [22], the *c3net* package can also visualize the inferred network by using the *igraph* package [25]. The visualization can be enabled by setting the parameter *network* to *TRUE*.

```
net = c3net(expdata, network = TRUE)
```

Further, *c3net* can validate the performance of the inference by its *checknet* function. The *checknet* function outputs the following six values: precision, F-score, recall, TP, FP, and FN. C3NET package provides additional functions that allow individual steps to be performed only instead of performing the whole inference step. This flexibility allows users to combine internal functions of *c3net* with components outside the package [23].

In order to demonstrate the *checknet* function, the example command above was performed on the example data set located in the software package. *"BH"* multiple testing correction method was used for the elimination of nonsignificant edges. Statistical significance threshold and the number of iteration parameters were set to 0.01 and 5, respectively. Also, the network parameter was enabled for the visualization of the inferred network. The *checknet* results for the example data set are shown in Table 1.

Figure 5 is the topological representation of the inferred network obtained by C3NET using the Fruchterman–Reingold algorithm [26].

For ease of usage, *c3net* package provides a file with the name EXAMPLE.TXT containing examples. One can easily learn the functionality of *c3net* by executing the examples line-by-line. For

**Table 1**
**Results of C3NET obtained by *checknet* function**

| Precision | F-score | Recall | TP | FP | FN |
|---|---|---|---|---|---|
| 0.68 | 0.42 | 0.30 | 263 | 123 | 601 |

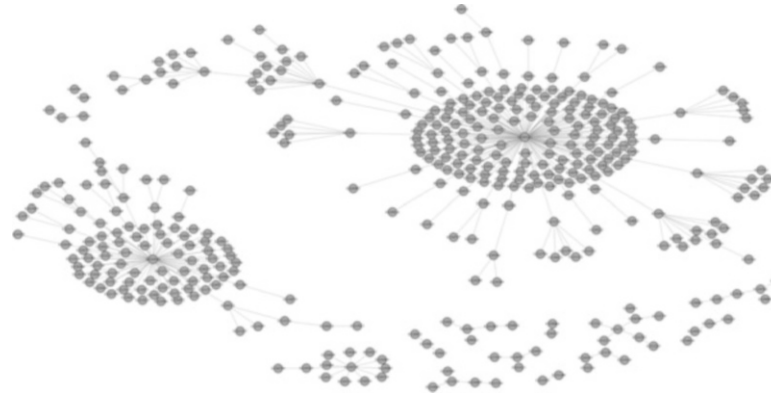


**Fig. 5** Inferred network of example data set by C3NET. Fruchterman–Reingold algorithm is used in the topological representation of the inferred network

additional help, *c3net* also provides an internal help function for each command which can be called by using the command line. Further, there is a manual file accessible from *inst/doc* folder of *c3net* which contains detailed explanations and examples of the functions of *c3net* [23].

### 2.2 RN (Relevance Networks)

The approach of relevance networks [13] consists in inferring a genetic network by computing all mutual information values for all pairs of genes, and linking a pair of genes $(ij)$ by an edge if their corresponding mutual information value $I_{ij}$ is larger than a given threshold $I_0$. In the resulting network, two genes connect to each other only if $I_{ij} > I_0$, otherwise no edge is included between $i$ and $j$. The threshold value $I_0$ was found by randomization of the gene expression dataset.

The complexity of the algorithm is $O(n^2)$ since all pairwise interactions are computed. Note that RN does not eliminate all the indirect interactions between genes since it can set an edge between two genes which do not interact directly, but both are regulated by a third gene. For example, suppose that gene $i$ and $j$ are regulated by gene $k$. This will result in high mutual information between gene pairs $(ij)$, $(ik)$ and $(jk)$. Therefore, the algorithm will set an edge between $i$ and $j$ although these two genes interact only through gene $k$ [27].

### 2.3 ARACNE (Algorithm for the Reconstruction of Accurate Cellular Networks)

The algorithm for the reconstruction of accurate cellular networks (ARACNE) [19] is an extension of the RN approach. The algorithm starts with estimating the pairwise mutual information values for all genes. Then it eliminates nonsignificant values according to the obtained threshold $I_0$. This step is basically equivalent to relevance networks since it computes mutual information and declares mutual information values significant if $I_{ij} > I_0$. If $I_{ij}$ is found to be significant, then an edge is included in the corresponding adjacency

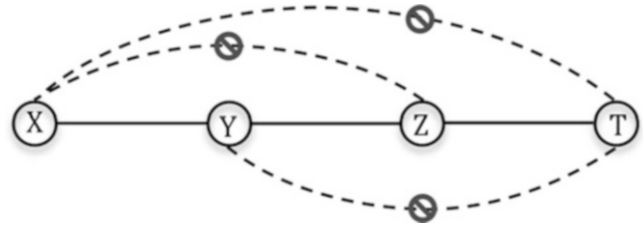



**Fig. 6** Working mechanism of DPI. Although all six gene pairs have significant mutual information values, the DPI will infer the most likely path of information flow. For example, $X \leftrightarrow Z$ will be eliminated because $I(X,Y) > I(X,Z)$ and $I(Y,Z) > I(X,Z)$. $Y \leftrightarrow T$ will be eliminated because $I(Y,Z) > I(Y,T)$ and $I(Z,T) > I(Y,T)$. $X \leftrightarrow T$ will be eliminated in two ways: (1) because $I(X,Y) > I(X,T)$ and $I(Y,T) > I(X,T)$, and (2) because $I(X,Z) > I(X,T)$ and $I(Z,T) > I(X,T)$ [19]

matrix between gene $i$ and $j$, $A_{ij} = A_{ji} = 1$. In addition to the first step, ARACNE performs a second step called *data processing inequality* (DPI). The DPI is a relation between mutual information values which means loosely that a post-processing of data cannot increase its information content [24]. DPI serves as a filtering step. DPI states that, if gene $X$ interacts with gene $Z$ through gene $\Upsilon (X \rightarrow \Upsilon \rightarrow Z)$, then

$$I(X,Z) \leq \text{ argmin } \{I(X,\ \Upsilon),\ I(\Upsilon,\ Z)\}.$$

Here, the weakest edge of the gene triplet $I(X, Z)$, corresponds to the indirect interaction and hence is eliminated by the DPI approach. The working mechanism of DPI is shown in Fig. 6.

In this step, ARACNE tests all gene-triplets (three genes with mutual information values larger than $I_0$) and then, for each $(ijk)$, it eliminates the edge corresponding to the lowest mutual information value $I_1 = I_{i'j'}$, with $(i'j') = \text{argmin}\{I_{ij}, I_{jk}, I_{ik}\}$ from the adjacency matrix, if it is smaller than the second smallest MI value $I_2$ multiplied by a factor [19].

$$A_{i'j'} = A_{j'i'} = \begin{cases} 0 & I_1 \leq I_2(1 - \epsilon) \\ 1 & \text{otherwise.} \end{cases}$$

Here $0 \leq \epsilon \leq 1$. $\epsilon$ is the tolerance parameter. Simulation studies that allow a comparison with the underlying true network are used to obtain optimal values for $\epsilon$. For this reason, it can be said that $I_0$ is found in an unsupervised and $\epsilon$ in a supervised manner of learning.

In ARACNE, each gene triplet is analyzed independently from the other triplets. Hence, it is possible that an edge can be included in the resulting network although it has been marked for removal by prior DPI applications to different triplets. Consequently, the order of examination of gene triplets does not affect the resulting network. ARACNE has a complexity in $O(n^3)$ since the algorithm considers all triplets of genes [19].

### 2.3.1 Implementation of ARACNE: Usage of the R Package

The ARACNE algorithm is implemented in an R/Bioconductor package called *minet*. MINET (Mutual Information NETworks) is an open-source Bioconductor package that includes network inference methods RELNET, ARACNE, CLR, MRNET, and MRNETB. It can be downloaded from the CRAN package repository at http://cran.r-porject.org as well as from the Bioconductor website http://bioconductor.org [27].

Once the R platform is launched, *minet* package can be activated by using "*library(minet)*" command. The example usage of ARACNE algorithm with the example data set in C3NET package is as follows:

```
data(expdata)                          # Load data
mim < - build.mim(expdata,             # Build mutual information matrix
estimator = "pearson")                 using pearson estimator
net < - aracne(mim)                    # Inferring network by using aracne
netplot(net)                           algorithm
                                       # Visualize inferred network
```

The topological representation of the inferred network obtained by ARACNE in using Fruchterman–Reingold algorithm is shown in Fig. 7.

### 2.4 CLR (Context Likelihood of Relatedness)

The CLR algorithm is also an extension of the RN approach which starts by computing the pairwise mutual information values for all genes. Then it estimates the statistical likelihood of each mutual information value $I_{ij}$ by comparing this MI value to a "background" distribution of the MI values. In particular, two z-scores are obtained for each gene pair $(ij)$ by comparing the MI value $I_{ij}$ with gene specific distributions, $p_i$ and $p_j$. Here, $p_i$ and $p_j$ distributions are equivalent to the distributions of MI values related to genes $i$ and $j$, respectively [20]. CLR takes into account the score

$$\overline{z_{ij}} = \sqrt{z_i^2 + z_j^2}$$

by making a normality assumption about these distributions. Here, $z_i$ and $z_j$ are the z-scores of $I_{ij}$, whereas $\overline{z_{ij}}$ corresponds to the joint

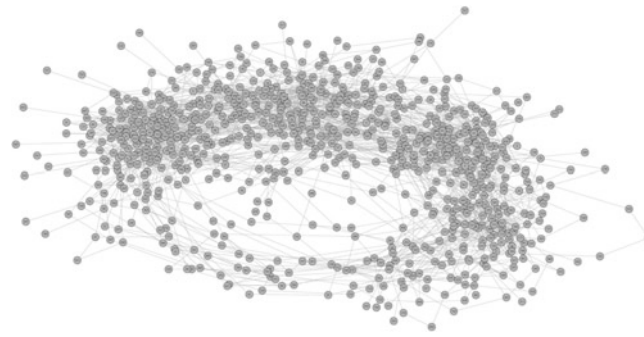


**Fig. 7** Inferred network of example data set by ARACNE. Fruchterman–Reingold algorithm is used in the topological representation of the inferred network

likelihood measure. In contrast to RN and ARACNE, which employ a global threshold $I_0$ for each MI value related to pair of genes, CLR estimates *individual* thresholds by considering an *individual* background for each pair of genes. The complexity of CLR is $O(n^2)$ since mutual information matrix is computed once for each gene pair [20].

### 2.5 MRNET (Maximum Relevance, Minimum Redundancy)

MRNET is an iterative algorithm that infers a network using the maximum relevance/minimum redundancy feature selection method. The algorithm identifies potential interaction partners of a target gene $\Upsilon$ that maximize a scoring function. The algorithm starts with ranking the set of input variables $V$ according to a score that is the difference between the MI with the output variable $\Upsilon$ and the average MI value with the previously ranked variables. The basic idea is ranking the direct interactions higher than indirect interactions [21]. The working mechanism is shown below.

$$X_j^s = \underset{X_j \epsilon V \setminus S}{\operatorname{argmax}}\left(s_j\right)$$

$$s_j = I\left(X_j; \Upsilon\right) - \frac{1}{|S|}\sum_{X_k \in S} I\left(X_j; X_k\right).$$

Here, the score $s_j$ is the difference between the mutual information of $X_j$ with the target variable $\Upsilon$ (relevance term) and the average redundancy of $X_j$ to each already selected variable $X_k \in S$ (redundancy term). A gene is added to the set $S$ only if the $s_j$ is above the threshold value, $s_0$ and the score of gene $X_j$ maximizes the value $X_j^s$. The algorithm repeats the iteration procedure until no further gene can be found that passes the threshold test. The MRNET approach consists in finding interaction partners for $\Upsilon$ that are of maximal relevance for $\Upsilon$, but have a minimum redundancy for the already found interaction partners in the set $S$. The algorithm starts with a fully connected, undirected network among all genes and then it eliminates the edges between $\Upsilon$ and $V \setminus S$, which have not maximized the value of $X_j^s$ [21].

MRNET has a complexity in $O(f \times n^2)$ since the feature selection step is repeated for each of the $n$ genes. Therefore, it can be said that the complexity of the algorithm ranges between $O(n^2)$ and $O(n^3)$ according to the value of $f$ [21].

## 3 Comparison of Inference Methods

The inference performance of the GNI methods may vary according to the data sets used in the assessment. Usually, a synthetic or a few real biological datasets are used in the analysis, but this may result in variations in the performance of the methods over different datasets. In order to assess the performance of a GNI algorithm on

a de novo dataset, a framework called *GANET* has been developed. GANET assesses the performance of GNI algorithms employing the available literature of interaction databases. Any new real dataset of any size can be assessed by using GANET [28].

In the study [22], the performance of C3NET is compared with four of the most widely known inference algorithms, RN, ARACNE, CLR, and MRNET. Simulated as well as expression data from microarray experiments were used in the analysis. The simulations were performed by considering the ensemble approach mentioned in Refs. [29, 30].

The performance of inference algorithms is assessed by using the error measure F-score. F-score is obtained by using the formula $F = 2pr/(p + r)$ where $p$ and $r$ values correspond to the precision and recall. Here precision, $p = \mathrm{TP}/(\mathrm{TP} + \mathrm{FP})$, and recall, $r = \mathrm{TP}/(\mathrm{TP} + \mathrm{FN})$, is a function of the number of true positive (TP), false positive (FP), and false negative (FN) edges in an inferred network. In the simulation study, two biological networks were used which represent sub-networks of the transcriptional regulatory network (TRN) of *E. coli* [31, 32] and *Yeast* [33].

SynTReN was used to randomly sample sub-networks from these TRNs. SynTReN is a network generator that produces synthetic gene expression data for approximating the experimental data [34]. Both networks consist of $n = 100$ genes. Synthetic expression data, which mimicks the mRNA concentration, was generated by using the *neighbor addition* method of SynTReN. In this process, nonlinear transfer functions based on Michaelis–Menten and Hill enzyme kinetic equations were used [35–37].

In this section, we present the results obtained by using the simulated ensemble data. Following that we give the results of expression data from *E. coli*.

### 3.1 Simulated (Synthetic) Data

The boxplots of the resulting F-scores for two different sample sizes ($p = \{50, 200\}$) are shown in Fig. 8. The results were obtained by using a sub-network of Yeast GRN [33]. According to the results, C3NET provides better results than all four other inference methods considering the median value of the F-score as well as the other statistical measures, e.g., minimum, maximum, or mean F-scores.

Table 2 provides a summary of the results obtained for the sub-networks of Yeast and *E. coli*. These numerical results reveal that C3NET gives the best result in all cases except one: minimum F-score value for $\mathrm{Yeast}_{50}$. For this case, the score of C3NET (0.2844) is quite close to the score of the best performing algorithm, MRNET (0.2879).

The analysis was repeated using a sub-network of *E.* coli [31, 32]. The ensemble size was 300, which results in 300 different data sets, each consisting of 1000 samples. In data generation, the same procedure was followed as for yeast. The boxplots of the resulting F-scores for the three best performing algorithms C3NET,

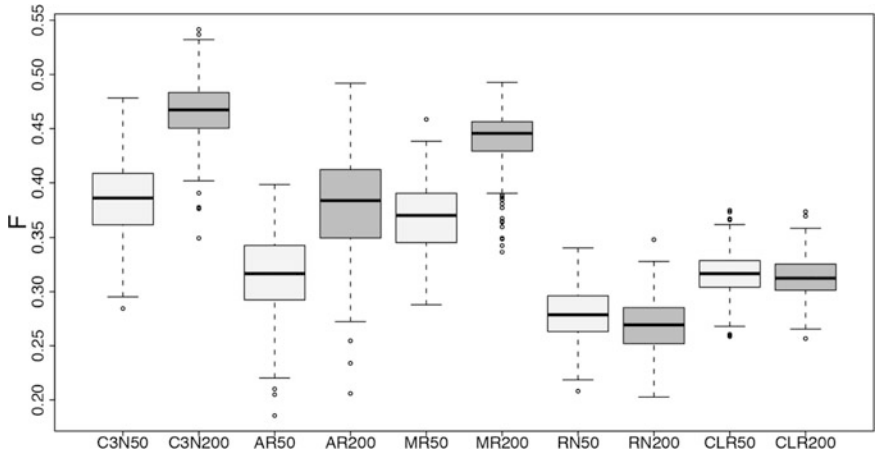



**Fig. 8** Boxplots of F-scores for C3NET (C3N50, C3N200), ARACNE (AR50, AR200), MRNET (MR50, MR200), RN (RN50, RN200), and CLR (CLR50, CLR200). *Light gray color* corresponds to sample size 50, whereas *dark gray color* corresponds to sample size 200 for each method. A sub-network of Yeast GRN with ensemble size $N = 300$ is used for the simulations [22]

**Table 2**
**Summary of F-scores (max, min, mean and median) for C3NET, ARACNE and MRNET obtained from our simulations**

| Dataset | Sample size | Statistical measure | C3NET | ARACNE | MRNET |
|---|---|---|---|---|---|
| Yeast | 200 | Max | 0.5478 | 0.4919 | 0.4927 |
| | | Min | 0.336 | 0.2058 | 0.336 |
| | | Median | 0.4628 | 0.3836 | 0.4455 |
| | | Mean | 0.4628 | 0.3795 | 0.4410 |
| Yeast | 50 | Max | 0.4782 | 0.3983 | 0.4585 |
| | | Min | 0.2844 | 0.1854 | 0.2879 |
| | | Median | 0.3859 | 0.3166 | 0.3698 |
| | | Mean | 0.3848 | 0.3161 | 0.3683 |
| *E. coli* | 1000 | Max | 0.6046 | 0.4973 | 0.5608 |
| | | Min | 0.4131 | 0.1866 | 0.3512 |
| | | Median | 0.5308 | 0.3803 | 0.500 |
| | | Mean | 0.5269 | 0.3758 | 0.4948 |

The sample size for Yeast is 200 and 50, whereas the sample size of *E. coli* is 1000 [22]

ARACNE, and MRNET are shown in Fig. 9. These results also indicate that C3NET provides the best results [22].

Figure 10 shows the true sub-network of Yeast obtained by using SynTReN. In the figure, gene names are shown on the labels

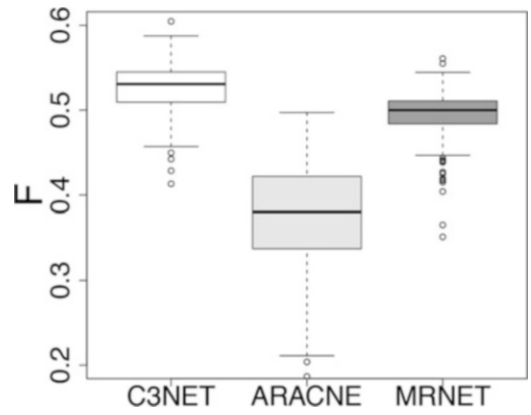



**Fig. 9** Boxplots of F-scores for C3NET (*white*), ARACNE (*gray*) and MRNET (*dark gray*). A sub-network of *E. coli* TRN with is used for the simulations. Sample size is 1000 and ensemble size is $N = 300$ [22]

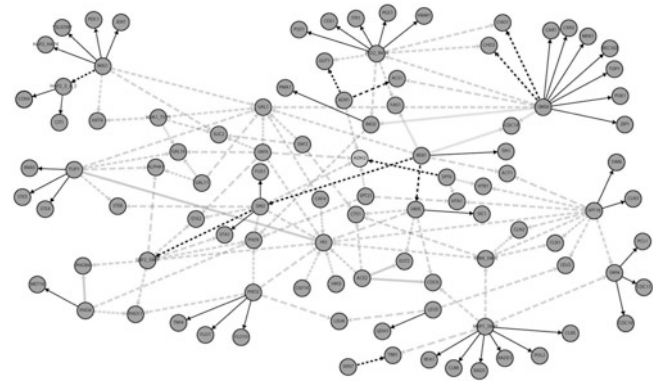


**Fig. 10** Sub-network of yeast consisting of 100 genes, sample size is 200. Edge colors are obtained from simulations of 300 data sets. The color of each edge reflects its mean TPR. Specifically, for *solid line black edges*, $1 \geq \overline{\text{TPR}} > 0.75$, for *dashed line black edges*, $0.75 \geq \overline{\text{TPR}} > 0.5$, for *solid line gray edges*, $0.5 \geq \overline{\text{TPR}} > 0.25$, and for *dashed line gray edges*, $0.25 \geq \overline{\text{TPR}} \geq 0.0$ [22]

of the nodes. The type of each edge corresponds to the mean true positive rate ($\overline{\text{TPR}}$). The edge types for true positive rates are as follows: for solid line black edges, $1 \geq \overline{\text{TPR}} > 0.75$, for dashed line black edges, $0.75 \geq \overline{\text{TPR}} > 0.5$, for solid line gray edges, $0.5 \geq \overline{\text{TPR}} > 0.25$, and for dashed line gray edges, $0.25 \geq \overline{\text{TPR}} \geq 0.0$. It is obvious that all leaf edges inferred by C3NET are correct because the edges connecting to leaf nodes are solid line black in both networks. Leaf node corresponds to a node that has only one incoming edge and no outgoing edges. Here, the incoming edge is called leaf edge. This observation indicates the efficiency of C3NET in inferring leaf edges. Additionally, dashed line gray edges help to observe that colliders cause difficulties for

the respective edges. Here, collider means a node that has two incoming edges [22].

The observations in the study [22] show that leaf nodes can be inferred easily whereas the nodes which are not leaf nodes are more difficult to infer. Hence, the detection of hubs is not easy. However, due to the fact that they are connected to many other nodes, it is possible that one or more of these nodes may be a leaf node. Therefore, they are more likely to appear in the inferred network.

### 3.2 Expression Data from E. coli

The expression data, which consists of 524 microarrays, was obtained from Ref. [20]. For this data set, it has been shown that the results of CLR obtained by a manually assembled reference network, $G_{2007}^{EC}$ is better than ARACNE and RN. Hence, in this section, the authors compared the inference algorithm C3NET only with CLR. Table 3 provides a summary of the results obtained for the comparison of C3NET and CLR [22].

The inferred network of *E. coli* is shown in the Fig. 11. In the figure, edges with solid line correspond to TP edges whereas the edges with dashed lines correspond to FP edges. The genes with gray color correspond to regulated genes whereas the genes with black color correspond to regulating (transcription factors) genes.

**Table 3**
**Summary of results for C3NET and CLR obtained from our simulations**

| Algorithm | Interactions | TP | FP | FN | Precision |
|---|---|---|---|---|---|
| C3NET[a] | 99 | 74 | 25 | 3017 | 0.75 |
| CLR[b] | 274 | 169 | 105 | 2922 | 0.62 |

[a]A threshold value of 6.974 obtained for the z-scores used by CLR
[b]A threshold value of 0.414 obtained as a result of significance test of the MI values for C3NET [22]

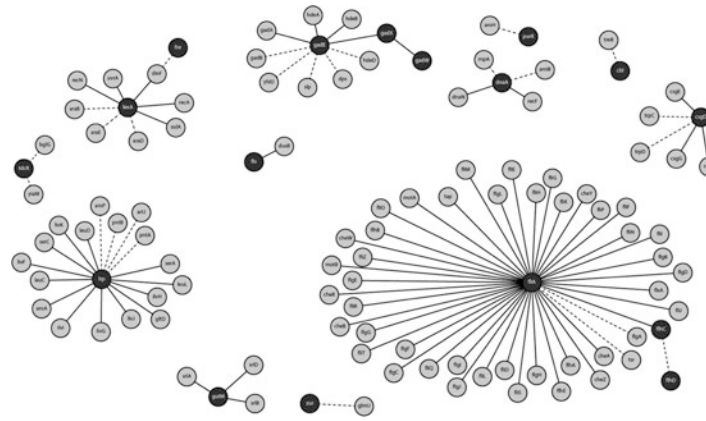


**Fig. 11** Inferred *E. coli* network by C3NET. *Black genes* correspond to transcription factors and *gray genes* to regulated genes. Edges with *solid line* indicate true positive results whereas edges with *dashed lines* correspond to false positives [22]

### 3.3 Conclusions

In this book chapter, we examine the five most widely known network inference methods—C3NET, RN, ARACNE, CLR, and MRNET—and discussed their performance in various biological and synthetic datasets and simulation conditions. The performance of GNI algorithms is assessed using a global performance metric. We also provide the implementation and usage of R packages of the algorithms C3NET and ARACNE.

Sample size is an important factor affecting the performance of GNI algorithms. However, there is no commonly recommended sample size that provides the best or optimal performance of the GNI methods. The general opinion in this subject is that a larger sample size results in better inference performance. According to the study by Altay [38], the inference performance of the information-theory-based GNI algorithms tends to converge after a particular sample size region around $\simeq 64$ (*see* **Note 5**). The results of the study show that increasing the sample size over this region does not improve the performance substantially [38].

The study by Altay and Emmert-Streib [22] reveals that the conservative approach of C3NET, which allows each gene to add at most one edge to the inferred network, provides consistently better results in comparison with the other methods widely used [18–21]. According to the study results, C3NET gives a precision of 0.81 for the expression data from *E. coli*. This result is 31 % better than the nearest precision obtained by CLR algorithm which performs better than ARACNE and RN (*see* **Note 6**). This robust result indicates that the performance of a well-structured algorithm can be better than other methods that are more complex [22]. In another study, it was found that the other inference algorithms show a more sensitive behavior in dependence on the network type used [23].

## 4 Notes

1. Obtaining the interaction scores among cell molecules is the main process in almost all GNI algorithms. A failure in this step often leads to an erroneous result in the ultimate inference process. According to the study results [5], BS2, BS3, KDE, PBG, and SBG are observed as the best performing estimators. However, the runtime of the KDE is large. Therefore, we advise using BS, PBG, and SBG estimators for applications in which the runtime is more important than the precision result.
2. The inference algorithms RN, ARACNE, CLR, and MRNET are becoming more and more complex. This may cause serious difficulties in obtaining a balanced statistical analysis [22].
3. All other methods than C3NET aim to infer the entire regulatory network for a given data set. However, achieving this goal is not easy for a large sample size. Observational data may not be

able to detect all dynamical interrelations that would allow a reliable estimation. Hence, C3NET aims to infer only the strongest interactions among covariates; this is called as *conservative causal core* or C3 [22].

4. An important characteristic of C3NET that is different to all previous methods is that it can infer at most as many edges as genes. The reason for this is that the second step of C3NET allows each gene to add at most one edge to another gene [22].
5. The sample size region of $\simeq$64 is sufficient to achieve good inference precision even for very large networks. However, F-score should be used to observe how close is the inferred network to the whole of the true network [38].
6. The performance of the inference method depends crucially on the characteristics of the data [22].

## References


1. Emmert-Streib F, Glazko GV, Altay G, de Matos Simoes R (2012) Statistical inference and reverse engineering of gene regulatory networks from observational expression data. Front Genet 3:8
2. Emmert-Streib F, Dehmer M (2010) Medical biostatistics for complex diseases. Wiley-Blackwell, Weinheim
3. Rual JF, Venkatesan K, Hao T, Hirozane-Kishikawa T (2005) Towards a proteome-scale map of the human protein-protein interaction network. Nature 437:1173–1178
4. Schadt E (2009) Molecular networks as sensors and drivers of common human diseases. Nature 461:218–223
5. Kurt Z, Aydin N, Altay G (2014) A comprehensive comparison of association estimators for gene network inference algorithms. Bioinformatics btu182v2–btu182
6. Hecker M, Lambeck S, Toepfer S, van Someren E, Guthke R (2009) Gene regulatory network inference: data integration in dynamic models—a review. Biosystems 96(1):86–103
7. Klipp E, Herwig R, Kowald H, Wierling C, Lehrach H (2005) Systems biology in practice: concepts, implementation, and application. Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim
8. Emmert-Streib F, Dehmer M (2009) Information processing in the transcriptional regulatory network of yeast: functional robustness. BMC Syst Biol 3:35
9. Emmert-Streib F, Dehmer M (2009) Predicting cell cycle regulated genes by causal interactions. PLoS One 4(8):e6633
10. Kurt Z, Aydin N, Altay G (2014) Comprehensive review of association estimators for the inference of gene networks. Turk J Electr Eng Comput Sci. doi:10.3906/elk-1312-90
11. Gallager R (1968) Information theory and reliable communication. Wiley, New York
12. Shannon C, Weaver W (1949) The mathematical theory of communication. University of Illinois Press, Champaign
13. Butte A, Tamayo P, Slonim D, Golub T, Kohane I (2000) Discovering functional relationships between RNA expression and chemotherapeutic susceptibility using relevance networks. Proc Natl Acad Sci U S A 97 (22):12182–12186
14. Meyer P, Kontos K, Bontempi G (2007) Information-theoretic inference of large transcriptional regulatory networks. EUROSIP J Bioinform Syst Biol 79879
15. Li W (1990) Mutual information functions versus correlation functions. J Stat Phys 60 (5–6):823–837
16. Steuer R, Kurths J, Daub CO, Weise J, Selbig J (2002) The mutual information: detecting and evaluating dependencies between variables. Bioinformatics 18(2):231–240
17. Altay G, Asim M, Markowetz F, Neal DE (2011) Differential C3NET reveals disease networks of direct physical interactions. BMC Bioinformatics 12:296
18. Butte A, Kohane I (2000) Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. Pac Symp Biocomput 5:415–426

19. Margolin A, Nemenman I, Basso K, Wiggins C, Stolovitzky G, Dalla Favera R, Califano A (2006) ARACNE: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. BMC Bioinformatics 7:7
20. Faith JJ, Hayete B, Thaden JT, Mogno I, Wierzbowski J, Cottarel G, Kasif S, Collins JJ, Gardner TS (2007) Large-scale mapping and validation of Escherichia coli transcriptional regulation from a compendium of expression profiles. PLoS Biol 5:8
21. Meyer PE, Kontos K, Latiffe F, Bontempi G (2007). Information-theoretic inference of large transcriptional regulatory networks. EUROSIPJ Bioinform Syst Biol, Article ID 79879
22. Altay G, Emmert-Streib F (2010) Inferring the conservative causal core of gene regulatory networks. BMC Syst Biol 5:132
23. Altay G, Emmert-Streib F (2010) Structural influence of gene networks on their inference: analysis of C3NET. Biol Direct 6:31
24. Cover T, Thomas J (1991) Information theory. John Wiley & Sons, New York
25. Csardi G, Nepusz T (2008) igraph-package. http://cneurocvs.rmki.kfki.hu/igraph/doc/R/aaa-igraph-package.html
26. Fruchterman TMJ, Reingold EM (1991) Graph Drawing by Force-Directed Placement. Softw Pract Exp 21(11):1129–1164
27. Meyer PE, Lafitte F, Bontempi G (2008) minet: a R/Bioconductor package for inferring large transcriptional networks using mutual information. BMC Bioinformatics 9:461
28. Altay G, Altay N, Neal D (2013) Global assessment of network inference algorithms based on available literature of gene/protein interactions. Turk J Biol 37:547–555
29. Emmert-Streib F, Altay G (2010) Local network-based measures to assess the inferability of different regulatory networks. IET Syst Biol 4(4):277–288
30. Altay G, Emmert-Streib F (2010) Revealing differences in gene network inference algorithms on the network-level by ensemble methods. Bioinformatics 26(14):1738–1744
31. Shen-Orr S, Milo R, Mangan S, Alon U (2002) Network motifs in the transcriptional regulatory network of Escherichia coli. Nat Genet 31:64–68
32. Ma HW, Kumar B, Ditges U, Gunzer F, Buer J, Zeng AP (2004) An extended transcriptional regulatory network of Escherichia coli and analysis of its hierarchical structure and network motifs. Nucleic Acids Res 32:6643–6649
33. Guelzim N, Bottani S, Bourgine P, Kepes F (2002) Topological and causal structure of the yeast transcriptional regulatory network. Nat Genet 31:60–63
34. Van den Bulcke T, Van Leemput K, Naudts B, van Remortel P, Ma H, Verschoren A, De Moor B, Marchal K (2006) SynTReN: a generator of synthetic gene expression data for design and analysis of structure learning algorithms. BMC Bioinformatics 7:43
35. Fersht A (1985) Enzyme structure and mechanism. W.H. Freeman and Company, New York
36. Hofmeyr J, Cornish-Bowden A (1997) The reversible Hill equation: how to incorporate cooperative enzymes into metabolic models. Comput Appl Biosci 13:377–385
37. Mendes P, Sha W, Ye K (2003) Artificial gene networks for objective comparison of analysis algorithms. Bioinformatics 19:122–129
38. Altay G (2012) Empirically determining the sample size for large-scale gene network inference algorithms. IET Syst Biol 6(2):35–43

# Chapter 7

# Integrating Heterogeneous Datasets for Cancer Module Identification

**A.K.M. Azad**

## Abstract

The availability of multiple heterogeneous high-throughput datasets provides an enabling resource for cancer systems biology. Types of data include: Gene expression (GE), copy number aberration (CNA), miRNA expression, methylation, and protein–protein Interactions (PPI). One important problem that can potentially be solved using such data is to determine which of the possible pair-wise interactions among genes contributes to a range of cancer-related events, from tumorigenesis to metastasis. It has been shown by various studies that applying integrated knowledge from multi-omics datasets elucidates such complex phenomena with higher statistical significance than using a single type of dataset individually. However, computational methods for processing multiple data types simultaneously are needed. This chapter reviews some of the computational methods that use integrated approaches to find cancer-related modules.

**Key words** Cancer modules, Cancer systems biology, Data integration, Gene-gene network, Multi-omics dataset

## 1 Introduction

Cancer is a common genetic disease involving a range of factors. Genomic, epigenomic, and differential gene expression aberrations all play vital roles in a cancer's initiation, development, and malignance [1]. It has been reported by various studies that cancer-related activities including cell proliferation, angiogenesis, and metastasis are associated with abrupt changes in regulatory and signaling pathways [2–6]. Mutations involving somatic and copy number aberrations of some genes can either directly affect some key pathways or have a cumulative effect when they occur across network modules representing common functional activities in cancer [7, 8]. Consequently, identifying cancer modules is of primary importance to the effective diagnosis and treatment of cancer patients.

One of the core steps of cancer module identification involves modeling gene–gene relationships in a network. Many algorithms

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_7, 

have been developed for this purpose, but most apply only to homogeneous datasets, that is, data of only one type, usually GE data or PPI information [9–15]. Most of the methods relying only on GE data apply differential expression analysis but it is often hard to determine whether such variations in expression are causative or merely an effect of complex diseases [16]. Differential expression analysis can produce false negatives and false positives: some important genes in cancer-related pathways may not be identified as differentially expressed, whereas some differentially expressed genes may not be relevant to cancer [17]. Typically CNA regions identified by some approaches [18–20] using only CNA datasets are spatially extensive, which makes it difficult to identify a specific gene causing genomic aberration [21]. PPI can provide important information in characterizing topological properties of the network involving cancer genes [7]. However, PPI information for multiple cell types and developmental stages is still incomplete, which limits its usefulness in developing methods for cancer module identification.

Recent studies have demonstrated the "genomic footprint" of driver mutations on gene expression [21–23]. This happens when somatic mutations and copy number aberrations affect a gene's transcriptional changes directly or indirectly [24] and thus perturb some core pathways relevant to cancer growth and malignance [1]. Research carried out for The Cancer Genome Atlas on both glioblastoma [25] and ovarian carcinoma [26] demonstrated the simultaneous occurrences of mutations, copy number aberrations, and gene expression changes in a significant number of patients in the core components of some key pathways (*see* **Note 1**). In this chapter we discuss some methods that find cancer-related modules by integrating multiple heterogeneous datasets.

This chapter is organized as follows. We first briefly introduce some of the main sources of data that can be used and the required preprocessing steps essential for subsequent integrated analysis. Then, we describe methods that integrate information from heterogeneous data sources to find cancer-related modules/subnetworks (*see* **Note 1**). Finally, we address some approaches for validating identified modules.

## 2 Data Sources

**Gene Expression** data from cancer samples can be primarily found in the database GEO (Gene Expression Omnibus) [27]. It is a database of gene expression values measured using high-throughput hybridization arrays (also known as chips or microarrays). Sample values are reposited both in raw and normalized versions. Another comprehensive collection of gene expression data from various cancer samples is The Cancer Genome Atlas

(TCGA) [28]. There are three different levels of datasets available in TCGA: Level 1 consists of low-level (not normalized) data for a single sample probe, Level 2 consists of normalized single sample probe data, and Level 3 consists of aggregated gene-level data (grouped by mapped probes with gene symbols). **Mutation, Copy number aberration, DNA methylation,** and **miRNA expression** datasets can also be found in TCGA data portal.

Preprocessing is an important step in data integration, especially when paired samples are used (*see* **Note 2**). Preprocessing of GE values includes scale transformation, imputing missing values, handling redundancies, pattern standardization (i.e., normalizing to a zero mean and unit standard deviation), and other transformations [29]. Preprocessing of CNA data in microarray chips is typically more complex than that of GE data, and can include quantile normalization, imputing missing values, summarizing multiple probes at a single locus (with mean or median), segmentation of genomic regions, and mapping segmented CNA values in genomic regions into corresponding gene symbols [17, 30]. Probe level methylation data from CpG sites can be normalized between 0 and 1 by finding the following ratio [31]:

$$\beta_i = \frac{\max(M_i, 0)}{(\max(M_i, 0) + \max(U_i, 0) + \alpha)} \quad (1)$$

where $\beta_i$ is the Beta-value for an $i$th interrogated CpG site, and $M_i$ and $U_i$ are the intensities measured by the $i$th methylated and unmethylated probes. After background adjustment, intensities ($M_i$ and $U_i$) may become negative, but in the above definition those negative values are reset to 0. Again, when both $M_i$ and $U_i$ intensities are very low, a constant offset $\alpha$ (default value = 100) is added to the denominator to regularize Beta-value, as suggested by Illumina [31].

## 3 Methods for Integrating Heterogeneous Datasets

Figure 1 generalizes a possible approach that integrates multiple heterogenous datasets in order to find cancer-related modules in a gene–gene network. The gene–gene network can be modeled either by exploiting combined knowledge from multiple datasets or by merging individual networks built upon corresponding datasets. In these networks, nodes represent genes and the edges can be modeled as the relationships (i.e., directed and/or undirected) among them. PPI information can be useful at various stages of network modeling. After modeling the integrated network various module detection techniques such as optimization models, hierarchical clustering, etc. can be applied to find cancer-related modules. The following sections describe some of the methods that use integrated approaches for cancer module identification.

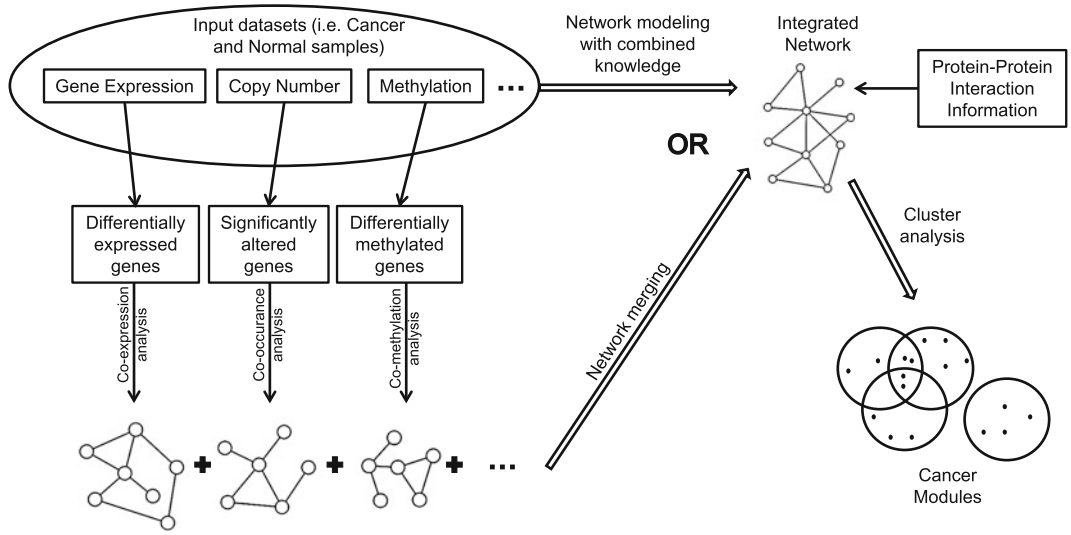



**Fig. 1** Schematic diagram of a possible integrated approach for cancer module identification. Each input dataset contains both cancer and normal samples. In network modeling, genes are identified based on differential information in the two-conditional studies (cancer vs normal), and edges can be defined according to pair-wise correlation

### *3.1 iMCMC*

A method known as iMCMC (**i**dentify **M**utated **C**ore **M**odule in **C**ancer) [32] was developed for the simultaneous analysis of three heterogeneous datasets: Gene expression (GE), copy number Aberration (CNA), and sequence mutations. These are combined to infer a network in which core cancer modules are identified (*see* **Note 3**). The method involves an optimization model followed by statistical significance tests. This method initially starts with building two different networks, one generated from GE data and the other by combining somatic mutations with CNAs over common samples. These two networks are then combined to construct an integrated network.

First, a binary matrix $A_0$ is constructed in which the columns represent the paired samples containing somatic mutations and CNAs, and the rows represent genes that the samples have in common. Each entry in $A_0$ is set to 1 if a mutation occurs in the corresponding gene and sample, or if there is a statistically significant copy number variation detected; otherwise the entry is set to 0. Genes that are mutated in the same samples in $A_0$ are combined into larger *metagenes*, and thus a new matrix $A$ called the mutation matrix is obtained. Another data matrix, $B$ is built from the expression values. Its entries are real values representing the relative expression of a given gene in a particular sample. The following two paragraphs explain the methodologies for constructing the *Expression Network* (**EN**) and *Mutation Network* (**MN**) from the data matrices B and A, respectively.

### 3.1.1 Constructing the Expression Network

The *Expression Network* is based on the gene expression dataset. In this network, both nodes and edges are weighted. Nodes represent genes and their corresponding weights reflect the extent to which a mutation in that gene affects the expression levels of other genes. Each edge weight is defined as the absolute correlation between the expression levels of the two corresponding genes.

The definition of nodes in the **EN** depends on both data matrices A and B. New sets of genes and samples are defined as: $G' = G_A \cap G_B$ and $S' = S_A \cap S_B$, where $(G_A, S_A)$ and $(G_B, S_B)$ are the sets of genes and samples in the two data matrices A and B, respectively. For each gene $g_i \in G'$, the corresponding samples in $S'$ are classified into two groups, based on that gene's mutation status in A. The numbers of samples in each group are denoted $n_i^{(1)}$ and $n_i^{(2)}$. Then, for each $g_i$, a mutation-correlated expression vector $e_i = \left(e_i^{(1)}, e_i^{(2)}\right)$ is constructed, where $e_i^{(1)}$ and $e_i^{(2)}$ are defined as follows:

$$\begin{aligned} e_i^{(1)} &= \left\{ b_{ki} : a_{ki} = 1, k \in S' \right\}, \\ e_i^{(2)} &= \left\{ b_{ki} : a_{ki} = 0, k \in S' \right\}. \end{aligned} \tag{2}$$

Here $a_{ki}$ and $b_{ki}$ denote the entries for the $i$-th gene and $k$-th sample in the data matrices A and B, respectively. To determine whether there are significant differences between the expression levels in $e_i^{(1)}$ and $e_i^{(2)}$, $p$-values are calculated using *mattest* in MATLAB. A small $p$-value indicates that mutations in the gene in question affect the expression levels of other genes. Since there should be a minimum of two samples in each group for conducting this test, the set of nodes $G$ in the **EN** is defined as follows:

$$G = \left\{ g_i \in G' : n_i^{(1)} \geq 2, n_i^{(2)} \geq 2 \right\}. \tag{3}$$

And the weight of each node in **EN** is defined as follows:

$$f_i = 1 - \frac{1}{d} \sum_{r=1}^{d} p_r, \quad \forall g_i \in G \tag{4}$$

where $d$ is the total number of genes in $G_B$ and $p_r$ is the $p$-value calculated for gene $g_r$ as described above. The weight $u_{ij}$ of any edge in $G$ is defined as the absolute Pearson correlation between two mutation-correlated expression vectors $e_i$ and $e_j$, among the samples in $S'$. In the case of *metagenes*, node and edge weights are defined as the averages of the corresponding values of their constituent genes.

### 3.1.2 Constructing the Mutation Network

To build the *Mutation Network* (**MN** ) from the mutation matrix A, the same gene set $G$ is used as for the network **EN**. The weight of any node (or gene), $g_i \in G$ is defined as follows:

$$h_i = \frac{m_i}{m}, \tag{5}$$

where $m$ is the total number of samples in A and $m_i$ is the total number of mutations occurring in the samples of A for a particular gene $g_i$. The weight $v_{ij}$ of any edge between genes $(g_i, g_j)$ in **MN** is defined as the ratio of the number of samples in which *exactly* one of the gene pairs is mutated to the number of samples in which *at least* one of the gene pairs is mutated in A.

### 3.1.3 The Integrative Network

An integrative network $\mathcal{M}$ is constructed by combining the expression network **EN** with the mutation network **MN**. It is necessary to first adjust the weights of nodes and edges in **EN** and **MN** so that they become comparable. Two balancing terms, $\xi$ and $\eta$, are defined for the networks **EN** and **MN**, respectively, as follows:

$$\xi = \frac{u}{f}, \eta = \frac{v}{h}, \tag{6}$$

where $f = \max(f_i)$ and $u = \max(u_{ij})$ in **EN**, and $h = \max(h_i)$ and $v = \max(v_{ij})$ in **MN**. Now, if $F = \{f_i\}$ and $U = \{u_{ij}\}$, then the edge weights $U$ and node weights $\xi F$ are said to have *balanced values* in **EN**. Similarly, if $H = \{h_i\}$ and $V = \{ v_{ij}\}$, then the edge weights $V$ and node weights $\eta H$ have balanced values in **MN**. A relative importance term can also be introduced to modify the relative impact of the two networks **EN** and **MN** on the integrated network. Let $k$ denote the relative importance of **MN** relative to **EN** and set $\delta \cdot \left(\frac{u}{v}\right) = k$, so $\delta = k \cdot \frac{v}{u}$. In the remainder of this description, we set $k = 1$. Thus, node weights $c_i$ and edge weights $w_{ij}$ can be defined as follows:

$$\begin{aligned} w_{ij} &= \delta \cdot u_{ij} + v_{ij}, \\ c_i &= \delta\xi \cdot f_i + h_i, \end{aligned} \tag{7}$$

where $i, j = 1, \ldots, n$. Here, $n$ is the total number of genes in $G$.

### 3.1.4 An Optimization Model for Identifying Core Cancer Pathways

The final step of this approach is to identify some core modules in the integrative network $\mathcal{M}$, where each such module contains genes with both high node-weights and high edge-weights. For this purpose, an optimization model (previously reported by Wang et al. [33]) is employed. The optimization problem is stated as follows:

$$\begin{aligned} \max \quad & \sum_i \sum_j w_{ij} x_i x_j + \lambda c_i x_i, \\ s.t. \quad & x_1^\beta + x_2^\beta + \cdots + x_n^\beta = 1, \\ & x_i \geq 0, i = 1, \ldots, n, \end{aligned} \tag{8}$$

where the non-negative vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ contains the degrees of each node in a particular module (sub-network). The first term in the objective function states the inter-connectivity within the module, whereas the second term specifies the degree of association between the nodes and the module. The role of the positive parameter $\lambda$ here is to balance these two terms (*see* **Note 4**). In this model, the regularization constraint over the variable $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ controls the number of nodes to be selected in the module and the parameter $\beta$ adjusts the strength of this regularization. Here we set $\beta$=1 to find small-sized core modules.

The following iterative algorithm [33] provides an easy solution of the above optimization model by finding a local maximum in the vicinity of a predetermined initial approximate solution:

$$x_i^{t+1} = \left( x_i^t \frac{2(WX)_i + \lambda c_i}{2X^T WX + \lambda \sum_i c_i x_i^t} \right)^{\frac{1}{\beta}}, \tag{9}$$

where $W = \{w_{ij}\}$ is the $n \times n$ edge weight matrix, and $X = \left(x_1^t, x_2^t, \ldots, x_n^t\right)^T$ is the solution vector at the $t$-th iteration. The non-zero entries in solution vector $\mathbf{x}$ define a particular module (sub-network) where in practice the entries are defined as zero if they are less than 0.1. Once a locally optimal solution is obtained, corresponding nodes are removed from the network and the whole process is repeated again to find additional modules.

### 3.2 Wen et al.

The method of Wen et al. integrates DNA methylation, gene expression, and protein–protein interaction datasets to identify causal network modules in colorectal cancer [34]. The method starts with collecting a set of candidate causal genes. This collection is the union of a set of differentially methylated genes and a common subset of known cancer genes from DNA methylation chips, the Cancer Gene Census (CGC) [35], and tumor associated genes in the TAG database. Employing a minimum multi-set cover strategy due to Kim et al. [36], a gene is determined to be differentially methylated if its comparative $\beta$ value (a measurement of DNA methylation level) between tumor and paired non-tumor samples is $\geq 0.2$ [37, 38].

Next, a comprehensive protein–protein interaction (PPI) network is developed integrating five curated human PPI databases: HPRD [39], BioGrid [40], IntAct [41], MINT [42], and Reactome [43]. Only those interactions that are found in at least three of these databases are considered. The resulting network contained 7001 nodes and 19,188 edges, where each edge $e$ is assigned a weight calculated as follows:

$$\begin{aligned} w(e) &= 1 - |cor(x,y)| \\ &= 1 - \left| \frac{\sum_{i=1}^{m}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{m}(x_i - \overline{x})^2}\sqrt{\sum_{i=1}^{m}(y_i - \overline{y})^2}} \right|. \end{aligned} \tag{10}$$

Here $x = (x_1, \ldots, x_m)$ and $y = (y_1, \ldots, y_m)$ are expression profiles of the two nodes in an edge $e$, and $\overline{x}$ and $\overline{y}$ are mean values of $x$ and $y$, respectively. This PPI network is further decomposed into network modules by applying the Markov Clustering algorithm [44], but only those modules are selected which contain at least one candidate causal gene. The activities of each network module $M_i$ in sample $S_j$ are calculated as follows:

$$M_{ij} = \frac{\sum_{g_m \in \{CGC \cap M_i\}} \sum_{(g_m, g_n) \in E(g_m)} \frac{g_{mj} + g_{nk}}{2}}{\sqrt{\sum_{g_m \in \{CGC \cap M_i\}} \#(E(g_m))}}, \tag{11}$$

where $E(g_m)$ is the set of edges belonging to the candidate causal gene $g_m$ in module $M_i$, $\#(E(g_m))$ represents the total number of edges in $E(g_m)$, and $g_{mj}$ is the normalized gene expression value of the gene $g_m$ in sample $S_j$. Next, a classifier is built for selecting the causal modules as follows:

$$\begin{aligned} \left\|S - \overline{S}_{control}\right\|_2^2 - \left\|S - \overline{S}_{case}\right\|_2^2 < 0, \quad for \quad S \in S_{control}, \\ \left\|S - \overline{S}_{case}\right\|_2^2 - \left\|S - \overline{S}_{control}\right\|_2^2 < 0, \quad for \quad S \in S_{case}, \end{aligned} \tag{12}$$

where $S$, $S_{\text{control}}$, $S_{\text{case}}$, $\overline{S}_{control}$, and $\overline{S}_{case}$ are the sample, the set of non-tumor samples, tumor samples, the center of non-tumor samples, and the center of tumor samples set, respectively. These classifier conditions can be further simplified as follows (for details, see supplementary texts of original article):

$$\mathbf{C} \cdot (x_1, x_2, \ldots, x_k)^T \leq 0, \tag{13}$$

where $x_i$ is an indicator variable having value 1 if module $M_i$ is selected, and 0 otherwise; and $\mathbf{C}$ is a matrix that is defined as a function of $M_{ij}$ as follows:

$$\mathbf{C} := \left\langle \left( M_{1i} - \frac{M_{11} + \cdots + M_{1n}}{n} \right)^2, \ldots, \left( M_{ki} - \frac{M_{k1} + \cdots + M_{kn}}{n} \right)^2 \right\rangle \tag{14}$$

Here, any element $C_{ij}$ of the above matrix $\mathbf{C}$ represents the contribution of the module $M_j$ to the $i$th sample condition. The objectives of this classifier are twofold: (1) classifying tumor and non-tumor samples, (2) identifying a small number of modules. This module identification problem is modeled as a binary integer linear programming problem as follows:

$$\begin{aligned} &\min_{x_1, x_2, \ldots, x_k} \sum_{j=1}^{k} x_j + \lambda \sum_{i=1}^{s} \sum_{j=1}^{k} C_{ij} \cdot x_j \\ &s.t. \quad \mathbf{C} \cdot (x_1, x_2, \ldots, x_k)^T \leq 0 \\ &\sum_{i=1}^{k} x_i \geq 1, \quad x_i = 0, 1, \quad i \in \{1, 2, \ldots, k\}, \end{aligned} \tag{15}$$

where $s$ is the number of samples. In this objective function, the first term encourages a small number of modules to be found whereas the second term implies the maximization of the classification abilities of modules by minimizing $\mathbf{C} \cdot (x_1, x_2, \ldots, x_k)^T$. $\lambda$ is the controlling parameter which balances the trade-off between those two terms. However, this binary integer linear programming model for module identification is computationally extensive. Therefore, this problem is further resolved by reformulating the model to a simple linear programming model where the binary variables $x_i \in \{0, 1\}$ are relaxed to a continuous variables $x_i \in [0, 1]$. For further detail, *see* **Note 5**.

### 3.3 Cerami et al.

The method of Cerami et al. [45] is an integrated approach for identifying core pathways altered in glioblastoma. It combines sequence mutation, copy number aberration (CNA), and protein–-protein interaction (PPI) datasets. The first step of this method is to construct a global Human Interaction Network (HIN) from literature curated data sources only. To cover more interaction information, the HIN is constructed based on the union of (a) interactions obtained from the HPRD website (http://www.hhprd.org/) and (b) various signaling pathway databases, specifically Reactome, NCI/Nature Pathway Interaction DB, and MSKCC Cancer Cell Map from Pathway Common (http://www.pathwaycommons.org). Information from the last of these pathway sources was in BioPAX format, which is represented as subgraphs of biochemical

networks. A set of rules was defined to map these subgraphs into binary interaction data. After removing all redundancies and self-directed interactions, the HIN contained 9264 genes and 68,111 interactions.

Sequence mutation and copy number datasets of Glioblastoma Multiforme (GBM) for paired samples were collected from TCGA data portal (https://tcga-data.nci.nih.gov/tcga/). Copy number aberration data was analyzed using the RAE algorithm [19] which discretizes all isoforms of autosomal genes into multiple putative aberration states, and finds statistically aberrant regions with $q$-values. Next, the statistical significance of each gene's aberration is defined as the minimum of the $q$-values of all the spanning regions over the corresponding gene's coding locus. A set of altered genes is identified, where a gene is defined as altered if it has a validated non-synonymous somatic nucleotide substitution, or a homozygous deletion, or a multi-copy amplification only.

Next, a GBM-specific network was constructed in which the node set is the union of the set of altered genes and a set of linker genes. For each gene in the altered gene set, the corresponding neighbor genes are identified in the HIN. Neighbor genes having degree one are trivially ignored, as they are connected to exactly one altered gene. The remaining neighbor genes with degree $\geq 2$ have the potential to connect two or more altered genes, and are thus considered to be candidate linker genes. Only linker genes that are found to be statistically significant by a hypergeometric test among all other candidate linker genes are further assessed. The null hypothesis is: *the linker genes connect the observed number of altered genes in HIN only by chance*. $p$-values from the statistical assessment of this hypothesis are further corrected using the Benjamini–Hochberg procedure [46] giving corresponding $q$-values, and the genes having $q$-values $\leq 0.05$ are selected as a final list of linker genes. The final network contained six linker genes connecting 66 GBM altered genes, and their corresponding PPI interactions in the HIN.

To find network modules in the resulting GBM-specific network, the *edge-betweenness algorithm* was applied. Originally proposed by Girvan and Newman [47], this algorithm applies a divisive approach where at each iteration an edge with the highest edge-betweenness score among all other edges is identified and removed from the network in order to reveal modular structure. The *edge-betweenness score* of a particular edge is defined as the number of shortest paths between pairs of nodes that traverse that edge [47]. More specifically, the shortest paths between all pairs of vertices are identified, and then for each edge the number of shortest paths that include that edge is counted and considered as the *edge-betweenness score* for that particular edge. After each edge removal, the *edge-betweenness scores* of the edges of the updated network are recalculated. (Only those edges which are affected by

the particular edge removal require recalculation of this score.) To obtain a partition yielding the best modular structure, *network modularity* [48] is also calculated after each edge removal. This process continues until there are no remaining edges. The maximum network modularity score obtained during this process indicates the optimal number of edges to be removed. The *network modularity score* is defined as follows:

$$M = \sum_{s=1}^{N_M} \left[ \frac{l_s}{L} - \left( \frac{d_s}{2L} \right)^2 \right], \tag{16}$$

where $N_M$ is the number of modules, $l_s$ is the number of edges within module $s$, $L$ is the total number of edges in the network, and $d_s$ is the summation of the degrees of all the edges within module $s$. *Modularity* quantifies the fraction of network edges connecting the nodes within modules minus the expected number of network edges obtained by forming random connections among the nodes within the module, subject to the same modular divisions. A value of $M$ close to 0 indicates that the number of within-module edges is consistent with random formation, whereas a value close to 1 indicates stronger modular structure. This procedure results in a set of modules extracted from the GBM-specific network.

### 3.4 VToD

VToD [17] integrates gene expression (GE), copy number aberration (CNA), and PPI (protein–protein interaction) datasets in order to find cancer-related modules in glioblastoma and ovarian cancer patients. The GE and CNA data matrices are obtained from TCGA data portal [28]; both are Level 3 datasets. The PPI dataset is obtained from Cerami et al. [45]. This method provides an integrated framework that infers pair-wise relationships between genes based on both *data-driven* and *topological properties* (*see* **Note 3**). A data-driven property of a pair of genes is a correlation observed between the data obtained for those genes. These correlations may be of three types: GE-GE, GE-CNA, or CNA-CNA correlations. Data-driven properties also include the indirect relationships discussed below. Topological properties are connections observed in PPI networks.

#### 3.4.1 Constructing a Gene–Gene Relationship Network

The method starts with a set of seed genes $S$, thought to be related to cancer progression and malignance. This set is a union of a set of differentially expressed and a set of significantly altered genes. Differential expression is identified using a two-tailed pooled $t$-test, and the corresponding $p$-values are corrected using the Bonferonni correction. A set of significantly altered genes is found by mapping gene symbols to the collected focal aberrant regions [25, 26] identified by GISTIC [18] and RAE [19] algorithms. Next, the Gene–Gene Relationship Network (*GGR*), a weighted undirected

network, is defined. Nodes of this network represent the seed genes and edges represent direct or indirect pair-wise relationships among genes. The absolute value of the Pearson correlational coefficient (PCC) is used to identify pair-wise relationships between genes, and as a weight on each edge.

For any gene-pair ($gene_i$,$gene_j$), all three types of absolute PCC value (GE-GE, GE-CNA, and CNA-CNA) are calculated, depending on data availability. The maximum of these is defined as the data-driven property of that particular gene-pair and termed its *r_value*. For the gene-pairs ($gene_i$, $gene_i$) this *r_value* is considered to be 0. If an *r_value* is greater than some threshold, then a direct relationship is defined for that particular gene-pair. The gene-pairs for which a direct relationship is not found may still be connected if an indirect relationship is identified. An indirect relationship between two particular genes is a statistically significant simple path joining those two genes in the PPI network (*see* **Note 6**). To identify such statistically significant paths, the observed paths between particular gene-pairs are compared with the path in a random PPI network, which is generated in such a way that gene interactions are randomly assigned while the network topology and gene expression values are the same as those in the observed PPI network. In other words, the random PPI network has the same number of interactions (edges) as the observed one, but the genes (nodes) of the observed PPI network are shuffled in the random PPI network. The null hypothesis for this statistical significance test is: *the geometric mean of r_values of the simple path found in random PPI network is greater or equal to that of the observed path.* In order to reduce the time complexity, a heuristic search is applied only for those gene-pairs for which there is a connection in the PPI network (*see* **Note 6**). All the simple paths between two genes with a fixed path length are identified using a breadth first search (BFS) algorithm. Furthermore, only those simple paths are selected in which all the constituent genes have either GE, or CNA, or both datasets available. Since there can be multiple such paths found, a path $P^*$ with maximum average PPI connectivity is selected:

$$P^* = \max_P \left\{ \frac{1}{n} \sum_{l=1}^{n} norm\_deg(gene_i) \right\} \tag{17}$$

where $norm\_deg(gene_i)$ is the degree of connectivity for $gene_i$ normalised by the global maximum connectivity in the PPI network, and $n$ is the number of genes along the path. The statistical significance of the path $P^*$ is measured as above, and is selected if its corresponding $p$-value is below 0.05. For the gene-pairs for which a statistically significant path is found, an edge is added to the *GGR* network, where the edge weight is the average of all the pair-wise *r_values* of gene-pairs along the path $P^*$.

### 3.4.2 Module Detection

Next, a **V**oting based module detection algorithm identifies overlapping modules in the GGR network by combining **To** pological and **D**ata-driven properties. The name of the method—VToD—is an acronym for this procedure. First, a pairwise score (vote) is calculated for every pair $\{g, m\} \in S$ using the following equation:

$$vote(g, m) = \frac{norm\_deg(m)}{SPL(g, m)} + r\_value(g, m) \qquad (18)$$

where above $norm\_deg(m)$ is the degree of connectivity of $m$ normalized by the global maximum PPI connectivity, $SPL(g, m)$ is the shortest path length between the two genes in the PPI network, and $r\_value(g, m)$ is the relationship value calculated for the constructed network *GGR*. This definition states how much vote-score a gene $m$ can get from another gene $g$, for any pair $\{g, m\} \in S$. Note, the $vote(g, m)$ score in the above equation is not a symmetrical measure because of the definition of the topological property ($norm\_deg(m)$ in above equation). A high score indicates either (1) a gene-pair $\{g, m\}$ has high data-driven relationship *r_values* or (2) any gene $g$ is interacting with a gene $m$ with a high topological value in the PPI network. Note, the shortest path length *SPL* is constrained by a user-defined threshold to control the compactness of the module. If any of the shortest paths has length above that threshold, that path is ignored.

Next, for any gene $g \in S$, corresponding vote-scores with all the genes $m \in S$ (including $g$) are stored in a table. Here, $vote(g, g)$ is defined with the $norm\_deg(g)$ only, since $r\_value(g, g) = 0$, and $SPL(g, g)$ is not defined for the PPI network as it doesn't contain any self-loop. Next, the table for the gene $g$ containing vote-scores of all the genes $m \in S$ is sorted in descending order of vote-score. In that sorted table, the ranking of each gene $m$ is defined as its local rank. Then, in that sorted table, the cumulated vote-score from the top-ranked vote-scores of the $m$ ($\in S$) genes is calculated. If the cumulated vote-score of the top-ranked $m$ gene(s) is(are) within the top $k\%$ (a user-defined threshold) of total cumulative vote-score in that particular table (for gene $g$), then that(those) top-ranked $m$ gene(s) are considered as candidate representative gene(s) of that particular gene $g$. Next, if the $vote\ (g, m)$ score(s) of this(these) top-ranked $m$ gene(s) are within top $vote\_th\%$ (a user-defined threshold) of the distribution of all pair-wise vote-scores (considered as the global rank of the gene $m$), then this(these) $m$ gene(s) are finally selected as a representative gene(s) of the particular gene $g$. Thus, this technique makes it possible to find overlapping modules in the network by allowing multiple representative $m$ genes to be selected for a particular gene $g$. More importantly, this method can select a gene $m \in S$ (i.e. a hub-gene in PPI network) as a representative gene for multiple $g \in S$ genes, thus revealing a modular structure.

Next, these modular structures, called "pre-modules," are formed, each with a representative gene $m$ in the center and aggregating all the genes $g$ that chose $m$. A pre-module is defined as the initial state of a module before merging it with other pre-modules to get the final module. After removing redundancies and small pre-modules (typically with $\leq 3$ genes), a module merging algorithm is conducted. Two pre-modules merge if their pair-wise members are closely connected in the PPI network (topological property) or highly related in *GGR* (data-driven property). For this purpose, a pair-wise merging value $\mathrm{MV}(C_i, C_j)$ between any two pre-modules $C_i$ and $C_j$ is calculated as follows:

$$MV(C_i, C_j) = \frac{IC(C_i, C_j)}{n_i} + \frac{1}{n_i \times n_j} \sum_{g_k \in C_i} \sum_{g_l \in C_j} r\_value(g_k, g_l) \quad (19)$$

where $n_i$ and $n_j$ are the sizes of two pre-modules $C_i$ and $C_j$, respectively, and $n_i \leq n_j$ (Note, here it is assumed that $C_i$ is bigger than $C_j$). Inter-connectivity $\mathrm{IC}(C_i, C_j)$ is a kind of topological property relating $C_i$ to $C_j$: it is the proportion of genes in $C_i$ having at least one PPI partner in $C_j$. The second term in the above equation denotes the data-driven property for the pair $C_i$ and $C_j$: it is the average of the gene–gene relationship values over all pairs of a gene in $C_i$ with a gene in $C_j$. At each iteration of the module merging procedure, two pre-modules with the highest pair-wise merging value (calculated using the above equation) are merged together and replaced by the newly merged module. This merging process continues until the highest pair-wise merging value at some iteration becomes less than some threshold *merging_th* (for the details of this threshold selection, see supplementary method of original article).

## 4 Validating Cancer Sub-networks

There are several ways to validate cancer modules identified by the above procedures. Most of them involve statistical hypothesis testing and are specific to the methodology used to identify modules. However, there are a few general techniques that can be used to validate modules, as follows:

### 4.1 Topological Validation

Ideally, a modular network is expected to have dense intra-module connections but sparse inter-module connections. Therefore, proposed networks can be assessed for both high density of connections within modules and high separability of component modules. Equation 16 states the modularity measurement [48] which

compares the connection-density of a particular module with that of a module formed by making random connections among its constituent genes. Similarly, the following equation quantifies the *separability* of modules [48].

$$seperationScore = \sum_{s=1}^{N_M} \left[ 1 - \left( \frac{2l_s}{d_s} \right)^2 \right] \tag{20}$$

where $N_M$ is the number of modules, $l_s$ is the number of edges within module $s$, and $d_s$ is the summation of the degrees of all the edges within module $s$. Both "Modularity" and "Separability" scores can be calculated using above equations (Eqs. 16 and 20, respectively) where higher "Modularity" value indicates stronger modular structure, and higher "Separability" score indicates a particular module is more easily separable from the original network topology by deleting some edges, respectively.

### 4.2 Enrichment Analysis

**f-Measure** Modules can also be validated using a quantity known as an *f*-measure [48]. This quantity evaluates the accuracy of identified modules by comparing them with known reference modules such as: GO functional categories, known biological pathways, and others. *f*-measure can be calculated using the following equation:

$$f-measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{21}$$

where $Precision = \frac{|M \cap F_i|}{|M|}$ and $Recall = \frac{|M \cap F_i|}{|F_i|}$ are the *true positive rate* and *positive predictive value*, respectively. Here, $M$ is a particular module and $F_i$ is a known functional module. For example, a Module $M$ (typically, a set of genes) is mapped to a known functional category $F_i$: "Cell Cycle," then the *Precision* and the *Recall* are the fractions of genes common to both $M$ and $F_i$ to the size of $M$, and to the size of $F_i$, respectively. Bigger modules will have higher *Recall* values, whereas smaller modules will have higher *Precision* values. Therefore, the accuracy of any identified module $M$ can be measured by calculating the harmonic mean of these two values as *f-measure*.

**Hypergeometric Analysis** A hypergeometric test can also be used to assess modules statistically [48]. *P*-values can be calculated using the hypergeometric distribution to indicate the significance of correspondence between a module and a known functional category.

$$p-\text{value} = 1 - \sum_{i=0}^{k-1} \frac{\left( \binom{|X|}{i} \binom{|V| - |X|}{n-i} \right)}{\binom{|V|}{n}} \tag{22}$$

where $|V|$ is the total number of genes (i.e., all the genes in human genome), $|X|$ is the number of genes in a known functional category (such as a GO term or known pathway), $n$ is the number of genes in an identified module, and $k$ is the number of genes in the intersection of that particular module with the known functional category. Here, a low $p$-value indicates that the identified module is significantly enriched in known functions or pathways. For example, a "dhyper" function in a built-in R-package called "stats" can be used to calculate $p$-values of the hypergeometric test [49].

## 5 Notes

1. In general, most of the integrative approaches that aim to find cancer-related modules are based on a common hypothesis: *tumors are characterized by aberrations in specific biological modules that are critical in terms of cancer initiation and malignance*. There are two major steps in such methods, (1) building the network model, and (2) identifying modules (sub-networks). In defining gene dependencies in network models, some methods rely on PPI information only [15, 45], some on data-driven information only [32, 50–52], and some on both of those properties [13, 17].
2. In any integrated approach, higher statistical significance is achieved by using paired sample data rather than unpaired data. Moreover, pair-wise relationships between genes obtained by integrative approaches applied to unpaired sample data may produce false positive results [24]. Here, paired data indicates using various heterogeneous data types (e.g., GE, CNA, methylation, miRNA) measured on the same samples. However, appropriate data normalization and standardization techniques are crucial to obtain correct inferences using paired data.
3. Integrating as many heterogeneous datasets as possible can improve characterizations of driver genes and cancer modules. Zhang et al. found that the integration of three heterogeneous datasets (GE + CNA + mutation) provides additional useful information and can produce statistically significant core modules in both glioblastoma and ovarian cancer compared to the integration of two heterogeneous datasets (GE + mutation, or CNA + mutation) [32]. Similarly, Azad et al. showed that modules found by combining topological and data-driven properties (PPI + GE + CNA) of gene-pairs result in better functional enrichment than those found by using only topological (PPI), or only data-driven (GE + CNA) properties [17]. Akavia et al. reported that combining CNA and GE provides greater sensitivity for identifying RAB27A as a novel driver gene in a

melanoma dataset. They also showed that this gene would not be selected based on CNA alone [21].

4. The parameters of the iMCMC method for integrating somatic mutation, CNA, and GE datasets are set in such a way that the method can balance the influence of different data sources on the network, and on the vertex and edge weights [32].
5. The problem of module identification in Wen et al. is formulated as a binary Integer Linear Programming (ILP) problem, which is NP-hard. To resolve this issue, the binary variables $x_i \in \{0, 1\}$ are relaxed to continuous variables $x_i \in [0, 1]$. The problem is then solved using a simple linear programming algorithm. To choose the penalty parameter $\lambda$, the classification ability of the identified modules is defined as follows:

$$CP = \max\left(C \cdot (x_1, x_2, \ldots, x_k)^T\right) \tag{23}$$

where the term on the right-hand side is the maximum element of the vector. The ILP is then solved for each value of $\lambda$ between 0 and 1, in increments of 0.01, and the value of $\lambda$ that produces the smallest value of CP is selected. The justification for this is the observation that smaller values of the elements of $\mathbf{C} \cdot (x_1, x_2, \ldots, x_k)^T$ indicate a greater ability to distinguish between cancer and normal samples.

6. VToD combines GE, CNA, and PPI information among gene pairs to find cancer-related modules. In searching for indirect relationships among gene-pairs, VToD considers the sub-network (with the genes for which pair-wise direct relationships are not defined) as fully connected. Therefore, to find a statistically significant indirect relationship considering a set of intermediate genes is an NP-hard problem. This problem is solved heuristically by restricting pair-wise adjacency among gene-pairs employing PPI information only, and converting that problem into finding a statistically significant simple path between gene-pairs. However, a threshold for the length of a simple path is a crucial parameter for handling time-complexity in this regard.

## References


1. Zhang S, Liu CC, Li W, Shen H, Laird PW, et al (2012) Discovery of multi-dimensional modules by integrative analysis of cancer genomic data. Nucleic Acids Res 40:9379–9391
2. Davies H, Bignell GR, Cox C, Stephens P, Edkins S, et al (2002) Mutations of the BRAF gene in human cancer. Nature 417:949–954
3. Wan PT, Garnett MJ, Roe SM, Lee S, Niculescu-Duvaz D, et al (2004) Mechanism of activation of the RAF-ERK signaling pathway by oncogenic mutations of B-RAF. Cell 116:855–867
4. Santarosa M, Ashworth A (2004) Haploinsufficiency for tumour suppressor genes: when you don't need to go all the way. Biochim Biophys Acta 1654:105–122
5. Vogelstein B, Kinzler KW (2004) Cancer genes and the pathways they control. Nat Med 10:789–799

6. Hanahan D, Weinberg R (2011) Hallmarks of cancer: the next generation. Cell 144:646–674
7. Jonsson PF, Bates PA (2006) Global topological features of cancer proteins in the human interactome. Bioinformatics 22:2291–2297
8. Qiu YQ, Zhang S, Zhang XS, Chen L (2010) Detecting disease associated modules and prioritizing active genes based on high throughput data. BMC Bioinf 11:26
9. de Lichtenberg U, Jensen LJ, Brunak S, Bork P (2005) Dynamic complex formation during the yeast cell cycle. Science 307:724–727
10. Segal E, Shapira M, Regev A, Pe'er D, Botstein D, et al (2003) Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. Nat Genet 34:166–176
11. Subramanian A, Tamayo P, Mootha VK, et al (2005) Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. Proc Natl Acad Sci USA 102:15545–15550
12. Liu X, Liu ZP, Zhao XM, Chen L (2012) Identifying disease genes and module biomarkers by differential interactions. J Am Med Inform Assoc 19:241–248
13. Wen Z, Liu ZP, Yan Y, Piao G, Liu Z, et al (2012) Identifying responsive modules by mathematical programming: an application to budding yeast cell cycle. PLoS One 7:e41854
14. He D, Liu ZP, Honda M, Kaneko S, Chen L (2012) Coexpression network analysis in chronic hepatitis B and C hepatic lesions reveals distinct patterns of disease progression to hepatocellular carcinoma. J Mol Cell Biol 4:140–152
15. Nepusz T, Yu H, Paccanaro A (2012) Detecting overlapping protein complexes in protein-protein interaction networks. Nat Methods 9:471–472
16. Iorns E, Lord CJ, Turner N, Ashworth A (2007) Utilizing RNA interference to enhance cancer drug discovery. Nat Rev Drug Discov 6:556–568
17. Azad AKM, Lee H (2013) Voting-based cancer module identification by combining topological and data-driven properties. PLoS One 8: e70498
18. Beroukhim R, Getz G, Nghiemphu L, Barretina J, Hsueh T, et al (2007) Assessing the significance of chromosomal aberrations in cancer: methodology and application to glioma. Proc Natl Acad Sci 104:20007–20012
19. Taylor BS, Barretina J, Socci ND, DeCarolis P, Ladanyi M, et al (2008) Functional copy-number alterations in cancer. PLoS One 3: e3179
20. Hur Y, Lee H (2011) Wavelet-based identification of DNA focal genomic aberrations from single nucleotide polymorphism arrays. BMC Bioinf 12:146
21. Akavia UD, Litvin O, Kim J, Sanchez-Garcia F, Kotliar D, et al (2010) An integrated approach to uncover drivers of cancer. Cell 143:1005–1017
22. Jornsten R, Abenius T, Kling T, Schmidt L, Johansson E, et al (2011) Network modeling of the transcriptional effects of copy number aberrations in glioblastoma. Mol Syst Biol 7:486
23. Schadt EE, Lamb J, Yang X (2005) An integrative genomics approach to infer causal associations between gene expression and disease. Nat Genet 37:710–717
24. Lee H, Kong SW, Park PJ (2008) Integrative analysis reveals the direct and indirect interactions between DNA copy number aberrations and gene expression changes. Bioinformatics 24:889–896
25. TCGA (2008) Comprehensive genomic characterization defines human glioblastoma genes and core pathways. Nature 455:1061–1068
26. TCGA (2011) Integrated genomic analyses of ovarian carcinoma. Nature 474:609–615
27. Edgar R, Domrachev M, Lash AE (2002) Gene expression omnibus: Ncbi gene expression and hybridization array data repository. Nucleic Acids Res 30:207–210
28. The cancer genome atlas - data portal (2005) https://tcga-data.nci.nih.gov/tcga
29. Herrero J, Diaz-Uriarte R, Dopazo J (2003) Gene expression data preprocessing. Bioinformatics 19:655–656
30. van de Wiel MA, Picard F, van Wieringen WN, Ylstra B (2011) Preprocessing and downstream analysis of microarray DNA copy number profiles. Brief Bioinform 12:10–21
31. Du P, Zhang X, Huang CC, Jafari N, Kibbe WA, et al (2010) Comparison of beta-value and m-value methods for quantifying methylation levels by microarray analysis. BMC Bioinf 11:1–9
32. Zhang J, Zhang S, Wang Y, Zhang XS (2013) Identification of mutated core cancer modules by integrating somatic mutation, copy number variation, and gene expression data. BMC Syst Biol 7:S4
33. Wang Y, Xia Y (2008) Condition specific subnetwork identification using an optimization model. Proc Sec Int Symp Opt Syst Biol. http://www.aporc.org/LNOR/9/OSB2008F42.pdf
34. Wen Z, Liu ZP, Liu Z, Zhang Y, Chen L (2013) An integrated approach to identify

causal network modules of complex diseases with application to colorectal cancer. J Am Med Inform Assoc 20:659–667

35. Futreal PA, Coin L, Marshall M, Down T, Hubbard T, et al (2004) A census of human cancer genes. Nat Rev Cancer 4:177–183
36. Kim YA, Wuchty S, Przytycka TM (2011) Identifying causal genes and dysregulated pathways in complex diseases. PLoS Comput Biol 7:e1001095
37. Hinoue T, Weisenberger DJ, Lange CP, Shen H, Byun HM, et al (2012) Genome-scale analysis of aberrant DNA methylation in colorectal cancer. Genome Res 22:271–282
38. Bibikova M, Le J, Barnes B, Saedinia-Melnyk S, Zhou L, et al (2009) Genome-wide DNA methylation profiling using Infinium assay. Epigenomics 1:177–200
39. Peri S, Navarro JD, Kristiansen TZ, Amanchy R, et al (2004) Human protein reference database as a discovery resource for proteomics. Nucleic Acids Res 32:497–501
40. Stark C, Breitkreutz BJ, Reguly T, Boucher L, Breitkreutz A, et al (2006) BioGRID: a general repository for interaction datasets. Nucleic Acids Res 34:D535–D539
41. Hermjakob H, Montecchi-Palazzi L, Lewington C, Mudali S, Kerrien S, et al (2004) IntAct: an open source molecular interaction database. Nucleic Acids Res 32:D452–D455
42. Ceol A, Chatr Aryamontri A, Licata L, Peluso D, Briganti L, et al (2010) MINT, the molecular interaction database: 2009 update. Nucleic Acids Res 38:D532–D539
43. Matthews L, Gopinath G, Gillespie M, Caudy M, Croft D, et al (2009) Reactome knowledgebase of human biological pathways and processes. Nucleic Acids Res 37:D619–D622
44. Enright AJ, Van Dongen S, Ouzounis CA (2002) An efficient algorithm for large-scale detection of protein families. Nucleic Acids Res 30:1575–1584
45. Cerami E, Demir E, Schultz N, Taylor BS, Sander C (2010) Automated network analysis identifies core pathways in glioblastoma. PLoS One 5:e8918
46. Benjamini Y, Hochberg Y (1995) Controlling the false discovery rate: a practical and powerful approach to multiple testing. J R Stat Soc Ser B Methodol 57:289–300
47. Newman MEJ, Girvan M (2004) Finding and evaluating community structure in networks. Phys Rev E 69:026113
48. Zhang A (2009) Modularity analysis of protein interaction networks. In: Zhang A (ed) Protein interaction networks: computational analysis, 1st edn. Cambridge University Press, Cambridge
49. R Core Team (2015) R: a language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. http://www.R-project.org/. ISBN 3-900051-07-0
50. Vandin F, Upfal E, Raphael BJ (2012) De novo discovery of mutated driver pathways in cancer. Genome Res 22:375–385
51. Zhao J, Zhang S, Wu LY, Zhang XS (2012) Efficient methods for identifying mutated driver pathways in cancer. Bioinformatics 28:2940–2947
52. Miller CA, Settle SH, Sulman EP, Aldape KD, Milosavljevic A (2011) Discovering functional modules by identifying recurrent and mutually exclusive mutational patterns in tumors. BMC Med Genomics 4:34

# Chapter 8

# Metabolic Pathway Mining

Jan M. Czarnecki and Adrian J. Shepherd

## Abstract

Understanding metabolic pathways is one of the most important fields in bioscience in the post-genomic era, but curating metabolic pathways requires considerable man-power. As such there is a lack of reliable, experimentally verified metabolic pathways in databases and databases are forced to predict all but the most immediately useful pathways.

Text-mining has the potential to solve this problem, but while sophisticated text-mining methods have been developed to assist the curation of many types of biomedical networks, such as protein–protein interaction networks, the mining of metabolic pathways from the literature has been largely neglected by the text-mining community. In this chapter we describe a pipeline for the extraction of metabolic pathways built on freely available open-source components and a heuristic metabolic reaction extraction algorithm.

**Key words** Metabolic pathway, Metabolic interaction extraction, Text-mining, Natural language processing, Named entity recognition, Information extraction

## Abbreviations

**NER** Named entity recognition
**NLP** Natural language processing
**PPI** Protein–protein interaction

## 1 Introduction

PubMed currently (as at February 2016) contains over 25 million article records, and this number is increasing at a faster rate than ever [1]. In some fields, researchers are encouraged, or even required, to submit results to databases in a standard format. For instance, upon solving the structure for a protein, an X-ray crystallographer will submit the structure to the Protein Databank (PDB) as well as submitting the results in a paper for peer review [2]. This allows anybody with an Internet connection to find curated

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_8, 

structures of proteins of interest quickly and efficiently. The data, being in a standard format, is also easily consumed by computer programs allowing large scale studies involving many structures to be carried out. For instance, the CATH project has developed a semi-automated system for classifying protein domain structures through the comparison of structures in the PDB [3].

Unfortunately, this method of submitting results in a standard, computer-readable language is found in few areas of bioscience. Currently, the vast majority of data in many biomedical subdomains is only available as unstructured text spread across many publishers' websites.

The study of metabolic pathways is one such field that suffers from a lack of manually curated data in databases. BRENDA is a large database of curated metabolic reactions, but individual reactions are not linked together to form pathways (meaning that there is little motivation to curate complete pathways from single organisms) [4]. KEGG [5] and BioCyc [6] are two databases that were developed to curate metabolic pathways. Ultimately, however, the databases are populated by human curators, which means it is practically impossible to keep up with all new articles being published. Training of a FlyBase Genetic Literature Curator, for instance, can take 6 months in addition to the time taken to actually perform a curation [7].

Computer processors are designed to follow very strict commands. This is reflected in the languages that are used to command computers, which, while incredibly varied, ultimately come down to providing a list of instructions for the various pieces of hardware within the computer. Therefore, the data which a computer program is designed to read and process must be stored in a strict format which the program can follow strict instructions to parse. Computer hardware, programs, and data storage formats are all designed from the bottom up with this philosophy in mind. Natural language, however, isn't designed, but is constantly evolving and rules can often be hard to define. The English language is particularly notorious for having significant exceptions to the majority of spelling and grammatical rules.

Text-mining development in bioscience initially focused on systems for named entity recognition (NER), the process of classifying elements in text into predefined categories. Current state-of-the-art NER tools (focusing on entities such as proteins, small molecules, drugs, and organisms) are able to achieve very high levels of accuracy—typically with *F*-scores greater than 90 % (*see* **Note 1** ). Focus has, therefore, shifted to interaction extraction, the process of determining the nature of relationships between different named entities. Interaction extraction can be used to determine abstract relationships, such as gene–disease relationships, or more direct, physical relationships, such as protein–protein interactions (PPIs). As metabolic reactions fall into the latter category and the

extraction of PPIs is the topic upon which most research has focused, a review of PPI extraction methods provides a useful backdrop for the development of an extraction method for metabolic reactions.

## 2 Existing Protein–Protein Interaction Extraction Methods

There are a range of experimental methods that have been developed to characterize PPIs ranging from narrow focused methods such as X-ray crystallography, which offers the most convincing evidence that two proteins form a stable complex, to broad scoped methods such as yeast two-hybrid screens, which can find potential binding partners from a large pool of proteins. The IntAct database [8] (which contains curated PPIs from the 14 members of the IMEx Consortium [9]) contains interactions extracted from more than 14,000 publications (as of April 2016). While this is a monumental manual effort, it is still only a small fraction of the available material.

PPI extraction was the subject of one task at BioCreative II [10] in 2006, where teams were tasked with extracting PPIs from documents curated by IntAct and MINT (which were separate databases at the time before merging in the IMEx Consortium). Extracted interactions could then be compared to the gold standard, manually curated interactions. The best performing tool achieved an *F*-score of 29 %, far lower than the high performance achieved by NER tools. Two general approaches to the problem were identified in the subsequent analysis of the submitted tools—which have been termed as local association analysis and global association analysis. Local association analysis identifies co-occurring proteins at either the sentence or passage level and may use other approaches such as interaction word lists and/or machine learning techniques to determine if an interaction between the co-occurring pair is described. Global association analysis focuses less on the characteristics of individual sentences, but rather looks at the co-occurrence of protein names multiple times in a document or over the whole collection. Global association analysis is more suitable for extracting well-known interactions that are described frequently in the literature, but only local association analysis is able to determine novel interactions that have only been described once. The method described here incorporates both local and global association analysis.

Kabiljo et al. [11] carried out a comparison on a range of PPI extraction tools including AkanePPI [12], OpenDMAP [13], and Whatizit [14]. AkanePPI is a state-of-the-art tool that utilizes many natural language processing (NLP) methods. OpenDMAP is a general purpose information extraction platform which uses a heuristic approach. The patterns for PPI recognition were created

manually to adapt the tool to the task. Whatizit is a suite of tools that can perform many bioscientific NLP tasks. The PPI extraction tool in Whatizit, Protein Corral, uses three methods which utilize co-occurrence and heuristic techniques.

A simple baseline method was also developed for the comparison. The method was co-occurrence based, looking for two protein or gene names within the same sentence as well as an "interaction" verb, such as *binds* or *phosphorylates* (a manually curated list of "interaction" verbs was used), in between the two entities—a similar methodology to the Co3 method of Protein Corral. The tools were evaluated on five PPI corpora. While performance across the five corpora by each tool was variable, the simple baseline method showed an overall performance that was comparable to the more sophisticated methods, while being far simpler. We followed this simple methodology in the development of a metabolic pathway extraction method.

BioCreative III [15] proposed a slightly different PPI extraction task to that in BioCreative II. The task required the development of a tool capable of classifying and ranking abstracts according to their suitability for manual curation of PPIs in the full text. This behavior is required by PPI databases, such as IntAct, to effectively manage their curator man-hours and to prevent the needless curation of irrelevant articles. Semi-automated selection of articles for manual inspection is common across the majority of biological annotation databases, but is typically carried out using simple PubMed searches. While effective at selecting articles relevant to a particular entity, this method is inadequate when dealing with complex events and interactions involving multiple entities [16].

Jamieson et al. used text-mining to recreate the HIV-1, Human Protein Interaction Database [17]. Protein NER was carried out by BANNER [18] while interactions in text were identified using 2 tools, the Turku event extraction system [19] and EventMiner [20]. The NER and event extraction were applied to 3090 titles and abstracts and 49 full-text articles achieving a precision, recall, and *F*-score of 87.5 %, 90.0 %, and 88.6 %, respectively. The pipeline was able to completely replicate over 50 % of the database. The team observed that the greatest obstacles to the automated extraction were grammatically complex sentences and sentences containing poor grammar.

While new methods for extracting PPIs are regularly released [21, 22], attention is increasingly shifting towards more complex relationships, with a particular focus on biomolecular networks and pathways [23] such as PPI networks [24, 25], signal-transduction pathways [26–28], protein metabolism (synthesis, modification, and degradation) [23], and regulatory networks [29, 30]. This protein/gene-centric focus has been enshrined in most of the competitive text-mining events (such as BioCreative [31–33] and BioNLP [23]). Indeed, in spite of this new focus on networks and

pathways, one of the most important sub-topics—the construction and curation of metabolic pathways—has largely been ignored.

## 3 Metabolic Interaction Extraction

Humphreys et al. [34] developed the template-based EMPathIE system which aimed to extract metabolic reactions with contextual information such as the source organism and pathway name. The system achieved 23 % recall and 43 % precision on a small corpus of seven journal articles. EMPathIE is no longer under active development (R. Gaizauskas, personal communication) and no tool has since been released to attempt to solve the problem. There are certain generic systems, such as the GeneWays system for "extracting, analyzing, visualizing and integrating molecular pathway data" [26] and the MedScan sentence parsing system [35], that could potentially be applied to metabolic pathway extraction, but neither are freely available.

There appears to be a perception that metabolic pathway extraction is a significantly more difficult problem compared to gene/protein interaction extraction. In evaluating the performance of GeneWays, the developers chose the extraction of signal-transduction pathways instead of metabolic pathways, suggesting that the former problem was an "easier target." Metabolic pathway extraction has a number of specific challenges compared to PPI extraction:

- Multiple entity types: Metabolic reactions consist of both proteins and small molecules, while PPIs consist of a single entity type (for the purposes of NER, genes and proteins are indistinguishable).
- Entity mismatch: There is a mismatch between the type of entities involved in a metabolic reaction, enzymes, and metabolites, and those extracted by popular NER tools, genes/proteins, and small molecules.
- Ternary and *n*-ary interactions: While PPIs are typically considered binary (e.g., "protein A binds to protein B"), metabolic reactions can consist of many entities including an enzyme, multiple substrates, and multiple products. Furthermore the enzyme is optional and may or may not be present in a description of a reaction, and the number of metabolites can vary significantly. With a greater number of entities the likelihood of information being split over multiple sentences increases.

In developing the method we describe here, we focused on the goal of providing assistance to database curators and model builders as opposed to the fully automated curation of the literature. Such systems are commonplace in large scale curation efforts, such as

FlyBase [36] and the Comparative Toxicogenomics Database [37]. In this context, high recall is typically deemed to be of paramount importance, although excessive numbers of false positives detract from the usability of such systems [38].

## 4 Components for Developers

Although the systems described so far cover a wide variety of general purpose and specialist requirements, the unique characteristics of many bioinformatics projects necessitate the development of bespoke solutions. Also, it is frequently the case that algorithms tuned for one specific domain will outperform their general purpose rivals. Fortunately, there are various standalone programs and libraries available that encapsulate functional building blocks of text-mining systems and are available for free, some of which have been developed from the ground up for bioinformatics applications or retrained on biological data. Since there are hundreds of NLP components and libraries available from the broader computational linguistics community, this section will give preference to those that are of particular utility or interest to bioinformatics developers. All the tools we discuss are available as Java libraries, the most popular language in the biomedical text-mining community.

### 4.1 General Text-Mining Tools

Libraries written in most programming languages exist for carrying out basic NLP tasks such as sentence parsing and part-of-speech tagging. One toolkit was found to fit our criteria: Apache OpenNLP [39].

OpenNLP is a machine learning-based Java library and, as such, requires extensive training to create a suitable probabilistic model. While a large number of models are provided alongside the library, they are all the result of training the library on general-use language (typically from newspaper articles). The language used in biomedical articles, however, is highly specialized [40]. Buyko et al. [41] showed that transferring OpenNLP components to the biomedical domain was as simple as retraining the tool using a biomedical corpus, however, and that a specially designed tool was not necessary—for the low level text-mining tasks that OpenNLP deals with, at least. OpenNLP was retrained separately on two corpora, GENIA [42] and PennBioIE [43], and the subsequent performance of five OpenNLP components was assessed. Each component performed well when trained with either corpus, with the sentence splitter, tokenizer, and parts-of-speech tagger achieving accuracies of approximately 99 % and the chunker and parser achieving average *F*-scores of 92 % and 86 %, respectively. The group have released the trained models [44], allowing OpenNLP to be implemented with no need of further training.

OpenNLP was incorporated into the previously mentioned Mayo clinical Text Analysis and Knowledge Extraction System where a number of components were built on OpenNLP components trained on clinical data [45].

### 4.2 Named Entity Recognition

NER, typically the first step taken in a text-mining operation, aims to find entities within text and assign each to a predefined category. Solutions have been developed for the recognition of a wide range of biological entities, but here we focus on those relevant to metabolic networks: genes/proteins, small molecules, and organism names.

#### 4.2.1 Gene/Protein NER

The recognition of gene and protein names is one of the best studied fields in biomedical text-mining with the task being the focus of many early competitions, such as BioNLP and BioCreative. Until 2008 the best performing, freely available tool was ABNER which was able to achieve an *F*-score of 83.7 % on the BioCreative I test corpus.

Leaman and Gonzalez, recognizing a lack of freely available tools, developed BANNER, an open-source gene NER tool based on conditional random fields [18]. BANNER achieved an *F*-score of 82.0 %—coming between the 9th and 10th ranked entries of BioCreative II—while ABNER achieved an *F*-score of 78.3 % on the same test corpus. This performance was repeated by Kabiljo et al. [11] who found that BANNER outperformed ABNER on four different corpora.

#### 4.2.2 Small Molecule NER

The recognition of small molecules has been the focus of significantly less research. In 2006, Corbett and Murray-Rust released the first freely available chemical NER tool, OSCAR3 [46], followed by OSCAR4 in 2011 [47]. OSCAR, with its ability to recognize both vernacular and systematic names, is widely used. The first tool to compete against OSCAR4, ChemSpot, was released in 2012 [48]. The only comprehensive comparison of these tools is found in the ChemSpot paper where both tools were tested against the SCAI chemical corpus [49]. OSCAR4 achieved an *F*-score of 57.3 % while ChemSpot achieved 68.1 %. The method described here utilizes OSCAR4, but ChemSpot would certainly be worth investigating in its place.

#### 4.2.3 Organism Name NER

Recognizing mentions of organism names is necessary to determine the context of any entities or interactions found in an article. LINNAEUS is principally a dictionary-based method which implements some heuristic rules [50]. A dictionary of organism name synonyms was created using the NCBI Taxonomy database and abbreviated names were generated for each entry. Additional synonyms were identified that occur frequently in the literature—such as *patient* referring to *Homo sapiens*.

The tool performed well on a manually annotated corpus of 100 full-text articles from the PMC Open-Access Subset with 94.3 % recall and 97.1 % precision. The BioCreative III gene normalization task required the entries to determine the source organism of genes in order to link them to database entries [33]. LINNAEUS was the only publicly available organism NER tool at the time and was used by the vast majority of teams. While the teams did note some ambiguity in species names and taxonomy IDs, the performance of LINNAEUS was well regarded.

## 5 Case Study: A Metabolic Pathway Extraction Tool

Here we present a heuristic metabolic pathway extraction method. The method can be split into four principal subtasks:

1. The retrieval of relevant documents to mine.
2. The extraction of individual metabolic pathways using a heuristic text-mining algorithm.
3. The merging and linking together of individual reaction extractions.
4. The determination of reactions relevant to the user.

### 5.1 Document Retrieval

The challenge of retrieving full-text articles has long held back biomedical text-mining. All article titles and abstracts can be obtained using the mature and stable E-Utils API provided by the NCBI [1]. As the API allows article records, containing the article abstract, to be retrieved in bulk and in a common format, early text-mining work in the biomedical community concentrated on the mining of these easily obtainable abstracts. While mining abstracts can return important data (as the significant findings of a paper will be repeated in the abstract), a great deal of potential useful data is only found in the full article. There has been a clear move towards developing tools using full-text articles with the BioCreative III tasks using corpora of full-text articles for the competition [33].

Unfortunately, publishers have been reluctant to allow their publications to be mined. While it is possible to retrieve content programmatically from the publishers' websites [51] (known as "screen scraping"), typically the Robots Exclusion Standard of most publishers' websites disallows access to screen scraping tools (with the exception of search engine spiders, such as the Googlebot). While the rules set by the `robots.txt` file are purely advisory and rely on the cooperation of the spider, web administrators can block access if they wish (*see* **Note 2** ).

Here we will discuss the use of the NCBI E-Utils API to retrieve relevant abstracts.

1. The user specifies a pathway (or multiple pathways) of interest, by MetaCyc ID(s), and an organism of interest, by NCBI Taxonomy ID.
2. A series of PubMed queries (one for each metabolite found in the "seed" pathway(s)) are constructed from this user-supplied information. For instance, consider the user specifies MetaCyc pathway "glycolysis I (from glucose-6P)" and the organism *Mycobacterium tuberculosis*. The following query would be constructed for the metabolite *pyruvate* (truncated lists of synonyms are used for presentation purposes):

```
(("M.tuberculosis"[All Fields]) OR ("Mycobacterium
    tuberculosis"[All Fields]) OR
  ("Bacterium tuberculosis"[All Fields])) AND
    (("pyruvate"[All Fields]) OR
  ("pyruvic acid"[All Fields]) OR ("alpha-ketopro-
    pionic acid"[All Fields]))
```

   Organism synonyms are retrieved using the dictionary provided by the LINNAEUS organism NER library (which is based on the NCBI Taxonomy database) [50]. Pathway data is retrieved using the BioCyc API, while metabolite synonyms are retrieved from a local ChEBI database [52]. It is not advisable to include all metabolites in the query—currency molecules, such as *ATP* and *ADP*, should be excluded as they are not meaningful in the identification of pathways and may lead to the retrieval of many irrelevant articles.
3. Use the NCBI E-Utils API to find articles matching each query. The API uses a REST interface and PubMed can be queried using the following general URL:

   http://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi

   The following general parameter string can be supplied as either GET or POST parameters (*see* **Note 3**), where `{term}` is the search query and `{retmax}` is the maximum number of articles to retrieve:

```
db=pubmed&term={term} &retmax={retmax}
```

   The request will return an XML document containing a list of PubMed IDs corresponding to articles matching the query (*see* **Note 4**).
4. The metadata for each article can be retrieved in bulk using the following URL:

   http://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi

   The following general parameter string can be supplied as either GET or POST parameters, where `{id}` is a PubMed ID retrieved in the previous step:

`db=pubmed&retmode=xml&id={id} ,{id} ,{id} ...`
The request will return an XML document containing metadata (such as the title, authors, and abstract) for each PubMed ID supplied (*see* **Note 5**).

### 5.2 A Heuristic Metabolic Reaction Extraction Method

Czarnecki et al. describe a pattern-based method for extracting individual metabolic reactions from text [53], which was implemented in this pipeline. Proteins and small molecules are recognized using the NER tools BANNER and OSCAR4, respectively. Patterns (such as substrate–product or enzyme–product–substrate) are assigned to individual sentences and each pattern assignment is scored regarding the match of entities extracted by BANNER and OSCAR4 and also words occurring between the entities. For instance, if two small molecules are assigned as substrate and product, the substrate would be expected to be preceded by a reaction word, while a production word would be expected to occur between the two entities (*see* **Note 6**). Other words such as variants of the verb *catalyze*, prepositions (e.g., *to*, *from*, and *by*), and the coordinating conjunction *and* are also accounted for. A training set was used to calculate appropriate scores for these individual factors, which are added together to create a final assignment score. The highest scoring assignment, if greater than a given threshold, is returned to the user.

### 5.3 Forming Networks from Individual Metabolic Reactions

The previous stage results in a list of putative individual metabolic reactions. Before a network can be formed, reactions must first be assigned to the correct host organism (*see* **Note 7**). We developed a simple heuristic method—similar to entries in the gene normalization task of BioCreative III [33]. Based on a small development corpus of 30 documents, we propose the following simple rules in order of priority:

1. If an organism is mentioned within a reaction sentence, the reaction is associated with this organism. If multiple organisms are mentioned, the reaction is assigned to all.
2. If the previous point does not apply, the reaction described will belong to the first organism mentioned in the paper.

Multiple individual extractions may describe the same metabolic reactions, but extracted from separate sources (or from separate sentences within the same source). Working out that multiple extractions refer to the same reaction is not trivial, however, as different sources may use different names for the metabolites and not all metabolites may be included (particularly side-metabolites, such as ATP and ADP).

Solving both of these problems requires linking metabolite names to common identifiers (such as SMILES and InChI). Fortunately there are a number of databases that catalogue small

molecule synonyms—the largest of which, PubChem [54], has catalogued approximately 50 million compounds. As a great many reactions may be extracted by the algorithm using web services to retrieve the InChI for a given small molecule would be unsuitable. Rather, a local database should be utilized. Here we describe the use of ChEBI [55]—a smaller database than PubChem, more focused on metabolic pathways.

1. The entire ChEBI database can be downloaded as a series of tab-delimited files. The three files containing data necessary for this task are `compounds.tsv`, `names.tsv`, and `inchis.tsv`. These files can be simply read into an SQL database (such as MySQL) and the synonyms indexed. More complex searching is required, however, as names used by authors may not correspond exactly the those held in the database (for instance, punctuation is often variable in small molecule names).
2. While complex querying can be provided through the implementation of a full search index (see Lucene for a standard Java solution), a simpler solution is to pregenerate variants that disregard elements that are likely to be variable in small molecule names. Consider the small molecule *aldehydo-D-glucose 6-phosphate(2−)*. The following variants can be generated:
   (a) Remove round and square brackets with their content.
   *aldehydo-D-glucose 6-phosphate*
   (b) Remove stereochemistry identifiers.
   *aldehydo-glucose 6-phosphate*
   (c) Remove all whitespace.
   *aldehydo-glucose6-phosphate*
   (d) Remove non-word characters (the set of word characters contain the 26 letters, 10 numbers, and underscore).
   *aldehydoglucose6phosphate*
   (e) Remove any non-letters.
   *aldehydoglucosephosphate*
3. Putative small molecules extracted by OSCAR4 undergo the same variant generation and the variants (a) to (e) are queried until a match is found. If a match is found the putative metabolite is assigned to the corresponding InChI.
4. Determining whether two extracted reactions are referring to the same reaction is not trivial even once InChIs have been assigned to all the metabolites. While two extractions containing the exact same metabolites can safely be merged, consider the following two reactions:

$$\begin{aligned} A &\rightarrow C \\ A + B &\rightarrow C \end{aligned}$$

There are two different ways of merging the reactions:

- Taking the union (creating a reaction including all metabolites from all the merged reactions)—this is the correct approach when B is a correct extraction, but is a side metabolite (such as ATP) which is not always included.
- Taking the intersection (creating a reaction including only those metabolites found in all the merged reactions)—this is the correct approach if B is an incorrect extraction.

5. We have evaluated both methods, but found weaknesses with both. Ultimately we decided to only merge reactions that were exactly the same, but other strategies may be viable.
6. While joining reactions together to form pathways is usually trivial (i.e., if the product of one reaction has the same InChI as a substrate of a different reaction, the reactions can be joined), currency molecules can be problematic. Currency molecules, such as ATP, tend to form a small number of highly connected nodes which the rest of the network clusters around (due to their involvement in many unrelated reactions). We use a manually curated list of currency molecules and recognized each mention of a particular currency molecule as unique entities (*see* **Note 8**). Therefore, completely separate reactions that both happen to convert *ATP* to *ADP* will not be linked together. There are problems with using a static list to identify currency molecules, however, as a metabolite's status as currency or non-currency can depend on context. While *acetyl-coenzyme A* is often confined to side reactions, it is an integral metabolite in the TCA cycle—pathways downloaded from BioCyc using the API make no distinction between "currency" and "integral" metabolites.

### 5.4 Assessing Reaction Relevance

A network created by following the previous steps will typically be very large, containing hundreds (and often thousands) of individual metabolic reactions. The extracted reactions can be classified into three categories:

- Extractions that do not match the content of the source sentence—either a described reaction is extracted incorrectly or the sentence does not even describe a reaction.
- Extractions that accurately represent a described reaction, but are irrelevant to the user.
- Correct extractions that are relevant to the user.

Unfortunately the third category is invariably in the minority. While there are a number of possible methods for determining whether an extraction accurately represents a real reaction (for example, counting the number of times the reaction is extracted,

or finding the reaction in a database of known reactions such as BRENDA—*see* **Note 9**), here we will focus on possible methods for determining whether a given reaction is relevant to the user.

Initially if the reaction matches a reaction in the seed pathway submitted by the user to construct the PubMed queries, the reaction will typically be relevant to the user. We have found, however, that a reaction simply containing just a single metabolite found in the seed pathway will typically not be relevant—particularly with metabolites that are found in many pathways.

MetaCyc groups pathways together that have the same purpose (such as the biosynthesis or degradation of a specific metabolite), but in different contexts. Such pathways are distinguished by their titles ending with Roman numerals (for instance, "glycolysis I" and "glycolysis II" describe glycolysis from two different starting metabolites). Pathways containing an extracted reaction can be found and their name compared to those used to construct the seed pathway. If a match is found, the extracted reaction is likely to be relevant.

These methods, however, rely on the extracted reaction already being known, though not necessarily in the same organism, and being present in MetaCyc—the reactions in a novel alternative pathway will not be found as relevant. We have identified two general properties that correlate with relevance, however, that do not require prior knowledge:

- The number of times a reaction is extracted.
- The similarity of the set of metabolites in the seed pathway and the set of metabolites mentioned in a source document as measured using Jaccard Index (*see* **Note 10**).

Depending on the task at hand, however, more properties may be identified. For instance, if links between pathways are of interest, reactions in an unbroken route connecting a metabolite in the seed pathway to that of a pathway of interest could be marked as relevant. Here we consider the identification of alternative pathways where one pathway takes a different route between two metabolites. The following steps describe how to correctly weight these properties using a training set:

1. Identify pairs of pathways in MetaCyc from two organisms which show different routes between the same two metabolites. Separate a small number out to use as a training set while the others will form a test set.
2. Run the reaction extraction algorithm and network building method using one pathway in the pair as a seed pathway, but specifying the organism of interest as the host of the other pathway.

3. Using the other pathway in the pair as gold standard relevant reactions, calculate the probabilities of a reaction being relevant with regard to the number of times it has been extracted. For instance, if from 20 reactions that have been extracted 3 times, 5 are found in the gold standard relevance pathway, reactions extracted 3 times have a 0.25 probability of being relevant.
4. Calculate the similarity between the sets of metabolites in the seed pathway and in each source document using Jaccard Index. Sort all extracted reactions by this score (if a reaction is extracted from multiple sources, simply take the greatest Jaccard Index of any of the sources). Calculate the precision of relevant reactions (i.e., reactions found in the gold standard relevance pathway) in a moving window at each reaction in the sorted list. Plot the Jaccard Index assigned to each reaction against the precision of relevant items and calculate a curve of best fit. The equation of the curve will be used to calculate a probability of relevance using a given Jaccard Index.
5. Find branches connecting metabolites from the seed pathway. Starting with such a metabolite, identify each reaction containing the metabolite as a substrate. Then do the same with the products of these identified reactions and so on until another metabolite from the seed pathway is discovered, the route loops back on itself, or the route simply ends (*see* **Note 11**).
6. As the pairs of metabolic pathways in the training set only have one alternative branch each, it is not possible to calculate a meaningful probability for whether a given branch in the extracted pathway is relevant. Branch relevance is instead calculated from the length of the branch (a route containing only two reactions connecting 2 metabolites found in the seed pathway is more likely to be relevant than a route containing 6 reactions) and the correctness of each individual reaction (a single incorrectly extracted reaction should lower the relevance of the entire branch). The product of the individual correctness probabilities can be calculated to take into account both of these factors.

### 5.5 Outputting the Extracted Pathway

There are a number of different use cases that demand the extracted pathway in different formats. For a developer implementing the tool as a library the extracted pathway should be returned in a computer readable format, such as the XML-based format SBML (for which there are mature Java libraries). An end-user, however, would typically prefer the pathway in a human readable format. This could entail outputting the extracted reactions (ordered by relevance score, for instance) or as a network diagram. While there are many network drawing libraries, outputting files that can be drawn by a separate package, such as Cytoscape (*see* **Note 12**), would be trivial to develop and could fit well into the user's workflow.

## 6 Concluding Remarks

While machine-learning methods typically produce the best performance of methods applied to text-mining tasks, heuristic methods, such as the method described here, still have an important role in the field. In a field such as metabolic pathway extraction, where little work has been carried out, heuristic methods can be prototyped relatively quickly and can act as a baseline for more sophisticated methods that follow. In addition, machine-learning methods rely on the availability of large quantities of marked-up text for training. For tasks such as NER and PPI extraction, large corpora have been developed which have greatly aided the development of methods. With no metabolic reaction corpus currently available, however, machine-learning methods are not a serious option.

Despite this, we have found that the methods described here perform strongly—roughly in line with methods used for PPI extraction. Nevertheless, there are many opportunities to improve the method:

- We have described the retrieval of abstracts and full-text open-access articles from PMC. Unfortunately full-text non-open-access articles have traditionally been impossible to obtain (without contravening web scraping rules), limiting the practical use of text-mining. There are signs that this is changing, however, with publishers such as Elsevier releasing APIs for accessing their libraries (although this has not been without controversy). Such APIs, or the general CrossRef API, could be implemented in the strategy we have described.
- The assignment of reactions to a specific organism is a key step in the pipeline. Incorrect assignments can lead to reactions from the wrong organism being included in the results or to the hiding of correct information. The organism assignment method described here uses a very simple method and was tested on a small corpus. While the method generally performs well, the significant effect of any errors may warrant the development of a more sophisticated method.
- The assignment of InChIs to metabolites is another key step in the pipeline. A failure to assign an InChI to a metabolite would prevent the reaction from being merged with other occurrences of the same reaction and from being linked to other reactions through the particular metabolite. While the variant generation method we have shown for the fuzzy searching of metabolite names performs well with anticipated variants (such as missing stereochemistry identifiers), other variants, such as incorrect or rare spellings, are not possible to foresee (*see* **Note 13**). A full search index, provided by a Java library such as Lucene, would allow more sophisticated fuzzy matching.

- When discussing the relevance of extracted reactions, we have focused on the identification of alternative pathways (specifically different routes between the same two metabolites). Other reactions may be relevant, however, depending on the user's needs. The user may simply want to identify evidence for an already established pathway or they might be interested in links to other pathways or metabolites. Consequently, it would be possible to develop multiple relevance algorithms and allow the user choose the most appropriate. However, a more detailed discussion of these options lies outside the scope of this chapter.

## 7 Notes

1. The quantitative assessment of text-mining systems tends to involve the use of corpora—collections of documents with entities and relationships manually annotated. The system being assessed is run on the text within a corpus and the results compared to the marked-up elements using the following measures:

   **Precision** The proportion of extracted instances that are correct extractions.

   $$Precision = \frac{true\ positives}{true\ positives + false\ positives}$$

   **Recall** The proportion of relevant instances that are correctly extracted.

   $$Recall = \frac{true\ positives}{true\ positives + false\ negatives}$$

   **$F_1$-score** (Often abbreviated to **F-score** ) An overall measure of accuracy—the harmonic mean of precision and recall.

   $$F_1\text{-}score = 2 \times \frac{precision \times recall}{precision + recall}$$

2. We have investigated the use of web scraping to retrieve full articles and while most publishers did not block access when using a very conservative method with long delays, some publishers blocked us regardless.
3. While submitting the query as a GET parameter is typically easier to code, there is a character limit which is easy to reach with a small molecule with many synonyms. POST, however, has no such limit.
4. If the full-text of the article is held in PubMed Central, this XML document will contain the PMC ID. This does not guarantee that the article is open-access and can be retrieved using the E-Utils API, however.

5. A similar query can be used to obtain articles from PMC if you wish by changing `db=pubmed` to `db=pmc`. If the article is open-access the returned XML document will contain the full-text article.
6. A reaction word typically refers to the substrate and describes the reaction, either specifically (e.g., *hydrolysis*) or generally (e.g., *converts*). A list of specific reaction words were inferred from the naming of enzymes in the Enzyme Classification (for instance, *alcohol dehydrogenase* leads to the reaction word *dehydrogenates*) while the general reaction words were simply complied from example text. Production words (e.g., *forms*, *produces*) refer to the product. The word lists, in fact, hold the word stems so that all variants need not be included. Stemming was performed using a Java implementation [56] of the standard Porter stemming algorithm [57]. *See* [53] for the full list of words.
7. While the literature search strategy should retrieve articles relevant to a specific organism, it remains necessary to assign individual reactions to an organism as articles rarely mention just a single organism. The organism of interest may be mentioned in passing in an article abstract while the article deals principally with a different organism. Reactions in such an article should not be assigned to the organism of interest. An article dealing with the organism of interest may compare against reactions in other organisms. Such reactions should be recognized as not belonging to the organism of interest.
8. The following molecules are recognized (by their InChI) as currency molecules: NAD+, NADH, NADP+, NADPH, ATP, ADP, AMP, C, O, N, H+, $CO_2$, and $H_2O$.
9. To develop a correctness measure extract a number of pathways using the algorithm and calculate the probability of a reaction being correct given one or more features. For instance, consider that the number of times a reaction is extracted is identified as the sole feature relevant to reaction correctness. If 60 % of the reactions extracted once in the training set were correct extractions, reactions extracted once would achieve a correctness score of 0.6.
10. The Jaccard Index measuring the similarity of two sets is calculated by dividing the number of features common to both sets by the total number of features. For instance, if a seed pathway contains 10 metabolites and an article mentions 5 of these metabolites in addition to a further 20 small molecules not found in the seed pathway, the Jaccard Index would be calculated as follows:

$$
\begin{aligned}
J(A,B) &= \frac{|A \cap B|}{|A \cup B|} \\
&= \frac{5}{10 + 25 - 5} \\
&= \frac{5}{30} \\
&= 0.167
\end{aligned}
$$

11. While this process can continue until these conditions are met, we have found it useful to include a branch length threshold as longer branches are more computationally expensive to calculate and are typically not significant. We used a maximum branch length of 6.
12. While Cytoscape can draw SBML files directly, it cannot read custom annotations where relevance scores would be stored. Reactions should instead by outputted to CSV files (one containing entity attributes and another containing relationships between entities) which can be used to read in arbitrary attributes.
13. For instance, consider the small molecule D-*glucose 1,6-bisphosphate*. The search strategy would identify *glucose-bisphosphate* as the same molecule, but not D-*glucose 1,6-bisphosfate*, despite the latter's closer spelling overall.

## References


1. PubMed Help [Internet] (2005) National Center for Biotechnology Information (US), Bethesda, MD. Available from https://www.ncbi.nlm.nih.gov/books/NBK3830/
2. Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN et al (2000) The protein data bank. Nucleic Acids Res 28:235–242
3. Orengo CA, Michie AD, Jones S, Jones DT, Swindells MB et al (1997) Cath–a hierarchic classification of protein domain structures. Structure 5:1093–1108
4. Schomburg I, Chang A, Placzek S, Söhngen C, Rother M et al (2013) Brenda in 2013: integrated reactions, kinetic data, enzyme function data, improved disease classification: new options and contents in BRENDA. Nucleic Acids Res 41:D764–D772
5. Ogata H, Goto S, Sato K, Fujibuchi W, Bono H et al (1999) Kegg: Kyoto encyclopedia of genes and genomes. Nucleic Acids Res 27:29–34
6. Caspi R, Altman T, Dale JM, Dreher K, Fulcher CA et al (2010) The MetaCyc database of metabolic pathways and enzymes and the BioCyc collection of pathway/genome databases. Nucleic Acids Res 38:D473–D479
7. McQuilton P, FlyBase Consortium (2012) Opportunities for text mining in the flybase genetic literature curation workflow. Database (Oxford) 2012:bas039
8. Orchard S, Ammari M, Aranda B, Breuza L, Briganti L et al (2013) The MIntAct project–IntAct as a common curation platform for 11 molecular interaction databases. Nucleic Acids Res 42:D358–D363
9. Orchard S, Kerrien S, Abbani S, Aranda B, Bhate J et al (2012) Protein interaction data curation: the international molecular exchange (imex) consortium. Nat Methods 9:345–350
10. Krallinger M, Leitner F, Rodriguez-Penagos C, Valencia A (2008) Overview of the protein-protein interaction annotation extraction task of biocreative ii. Genome Biol 9(Suppl 2):S4
11. Kabiljo R, Clegg AB, Shepherd AJ (2009) A realistic assessment of methods for extracting gene/protein interactions from free text. BMC Bioinf 10:233

12. Miyao Y, Sagae K, Saetre R, Matsuzaki T, Tsujii J (2009) Evaluating contributions of natural language parsers to protein-protein interaction extraction. Bioinformatics 25:394–400
13. Hunter L, Lu Z, Firby J, Baumgartner WA, Johnson HL et al (2008) Opendmap: an open source, ontology-driven concept analysis engine, with applications to capturing knowledge regarding protein transport, protein interactions and cell-type-specific gene expression. BMC Bioinf 9:78
14. Rebholz-Schuhmann D, Arregui M, Gaudan S, Kirsch H, Jimeno A (2008) Text processing through web services: calling Whatizit. Bioinformatics 24:296–298
15. Krallinger M, Vazquez M, Leitner F, Salgado D, Chatr-Aryamontri A et al (2011) The protein-protein interaction tasks of biocreative iii: classification/ranking of articles and linking bio-ontology concepts to full text. BMC Bioinf 12(Suppl 8):S3
16. Kwon D, Kim S, Shin SY, Chatr-aryamontri A, Wilbur WJ (2014) Assisting manual literature curation for protein-protein interactions using BioQRator. Database 2014:bau067
17. Jamieson DG, Gerner M, Sarafraz F, Nenadic G, Robertson DL (2012) Towards semi-automated curation: using text mining to recreate the hiv-1, human protein interaction database. Database (Oxford) 2012:bas023
18. Leaman R, Gonzalez G (2008) Banner: an executable survey of advances in biomedical named entity recognition. Pac Symp Biocomput 13:652–663
19. Björne J, Ginter F, Pyysalo S, Tsujii J, Salakoski T (2010) Complex event extraction at pubmed scale. Bioinformatics 26:i382–i390
20. Miwa M, Saetre R, Kim JD, Tsujii J (2010) Event extraction with complex event classification using rich features. J Bioinform Comput Biol 8:131–146
21. Li L, Zhang P, Zheng T, Zhang H, Jiang Z et al (2014) Integrating semantic information into multiple kernels for protein-protein interaction extraction from biomedical literatures. PLoS One 9:e91898
22. Quan C, Wang M, Ren F (2014) An unsupervised text mining method for relation extraction from biomedical literature. PLoS One 9: e102039
23. Kim J, Ohta T, Pyysalo S, Kano Y, Tsujii J (2009) Overview of bionlp'09 shared task on event extraction. In: Proceedings of the BioNLP 2009 workshop companion volume for shared task. Association for Computational Linguistics, Boulder, CO, pp 1–9. http://www.aclweb.org/anthology-new/W/W09/W09-1401.bib
24. Blaschke C, Valencia A (2002) The frame-based module of the SUISEKI information extraction system. IEEE Intell Syst 17:14–20
25. Iossifov I, Krauthammer M, Friedman C, Hatzivassiloglou V, Bader JS et al (2004) Probabilistic inference of molecular networks from noisy data sources. Bioinformatics 20:1205–1213
26. Rzhetsky A, Iossifov I, Koike T, Krauthammer M, Kra P et al (2004) Geneways: a system for extracting, analyzing, visualizing, and integrating molecular pathway data. J Biomed Inform 37:43–53
27. Santos C, Eggle D, States DJ (2005) Wnt pathway curation using automated natural language processing: combining statistical methods with partial and full parse for knowledge extraction. Bioinformatics 21:1653–1658
28. Yuryev A, Mulyukov Z, Kotelnikova E, Maslov S, Egorov S et al (2006) Automatic pathway building in biological association networks. BMC Bioinf 7:171
29. Marshall B, Su H, McDonald D, Eggers S, Chen H (2006) Aggregating automatically extracted regulatory pathway relations. IEEE Trans Inf Technol Biomed 10:100–108
30. Rodríguez-Penagos C, Salgado H, Martínez-Flores I, Collado-Vides J (2007) Automatic reconstruction of a bacterial regulatory network using natural language processing. BMC Bioinf 8:293
31. Hirschman L, Yeh A, Blaschke C, Valencia A (2005) Overview of biocreative: critical assessment of information extraction for biology. BMC Bioinf 6(Suppl 1):S1
32. Smith L, Tanabe LK, nee Ando RJ, Kuo CJ, Chung IF et al (2008) Overview of biocreative ii gene mention recognition. Genome Biol 9 (Suppl 2):S2
33. Lu Z, Kao HY, Wei CH, Huang M, Liu J et al (2011) The gene normalization task in biocreative iii. BMC Bioinf 12(Suppl 8):S2
34. Humphreys K, Demetriou G, Gaizauskas R (2000) Two applications of information extraction to biological science journal articles: enzyme interactions and protein structures. Pac Symp Biocomput 5:505–516
35. Novichkova S, Egorov S, Daraselia N (2003) MedScan, a natural language processing engine for MEDLINE abstracts. Bioinformatics 19:1699–1706
36. Karamanis N, Lewin I, Seal R, Drysdale R, Briscoe E (2007) Integrating natural language processing with flybase curation. Pac Symp Biocomput 12:245–256
37. Wiegers TC, Davis AP, Cohen KB, Hirschman L, Mattingly CJ (2009) Text mining and manual curation of chemical-gene-disease networks

for the comparative toxicogenomics database (CTD). BMC Bioinf 10:326

38. Winnenburg R, Wächter T, Plake C, Doms A, Schroeder M (2008) Facts from text: can text mining help to scale-up high-quality manual curation of gene products with ontologies? Brief Bioinform 9:466–478
39. Kottmann J, Margulies B, Ingersoll G, Drost I, Kosin J, Baldridge J, Goetz T, Morton T, Silva W, Autayeu A, Galitsky B (2011) Apache opennlp. Online. www.opennlp.apache.org
40. Clegg AB, Shepherd AJ (2007) Benchmarking natural-language parsers for biological applications using dependency graphs. BMC Bioinf 8:24
41. Buyko E, Wermter J, Poprat M, Hahn U (2006) Automatically adapting an NLP core engine to the biology domain. In: Proceedings of the ISMB 2006 joint linking literature, information and knowledge for biology and the 9th bio-ontologies meeting.
42. Kim JD, Ohta T, Tateisi Y, Tsujii J (2003) Genia corpus–semantically annotated corpus for bio-text mining. Bioinformatics 19(Suppl 1):i180–i182
43. Kulick S, Bies A, Liberman M, Mandel M, Mcdonald R et al (2004) Integrated annotation for biomedical information extraction. In: Biolink: linking biological literature, ontologies and databases, proceedings of HLT-NAACL, pp 61–68
44. Hahn U, Matthies F, Faessler E, Hellrich J (2016) UIMA-based JCoRe 2.0 goes GitHub and Maven central—state-of-the-art software resource engineering and distribution of NLP pipelines. In: Calzolari N (Conference Chair), Choukri K, Declerck T, Grobelnik M, Maegaard B, Mariani J, Moreno A, Odijk J, Piperidis S (eds.) Proceedings of the tenth international conference on language resources and evaluation (LREC 2016), Portorož, Slovenia
45. Savova GK, Masanz JJ, Ogren PV, Zheng J, Sohn S et al (2010) Mayo clinical text analysis and knowledge extraction system (cTAKES): architecture, component evaluation and applications. J Am Med Inform Assoc 17:507–513
46. Corbett P, Murray-Rust P (2006) High throughput identification of chemistry in life science texts. In: Proceedings of the 2nd international symposium on computational life science (CompLife '06), pp 107–118
47. Jessop DM, Adams SE, Willighagen EL, Hawizy L, Murray-Rust P (2011) Oscar4: a flexible architecture for chemical text-mining. J Cheminform 3:41
48. Rocktäschel T, Weidlich M, Leser U (2012) Chemspot: a hybrid system for chemical named entity recognition. Bioinformatics 28:1633–1640
49. Kolarik C, Klinger R, Friedrich CM, Hofmann-Apitius M, Fluck J (2008) Chemical names: Terminological resources and corpora annotation. In: Workshop on Building and evaluating resources for biomedical text mining (6th edition of the Language Resources and Evaluation Conference). Marrakech, Morocco
50. Gerner M, Nenadic G, Bergman CM (2010) Linnaeus: a species name identification system for biomedical literature. BMC Bioinf 11:85
51. Yepes AJ, Verspoor K (2014) Literature mining of genetic variants for curation: quantifying the importance of supplementary material. Database (Oxford) 2014:bau003
52. de Matos P, Ennis M, Darsow M, Guedj M, Degtyarenko K et al (2006) Chebi — chemical entities of biological interest. Database Summary Paper 646, EMBL Outstation - The European Bioinformatics Institute
53. Czarnecki J, Nobeli I, Smith AM, Shepherd AJ (2012) A text-mining system for extracting metabolic reactions from full-text articles. BMC Bioinf 13:172
54. Bolton EE, Wang Y, Thiessen PA, Bryant SH (2008) Chapter 12 PubChem: integrated platform of small molecules and biological activities. Annu Rep Comput Chem 4:217–241
55. de Matos P, Alcantara R, Dekker A, Ennis M, Hastings J et al (2010) Chemical entities of biological interest: an update. Nucleic Acids Res 38:D249–D254
56. (2006) Porter stemming algorithm implementations. http://tartarus.org/~martin/PorterStemmer/
57. Porter M (1980) An algorithm for suffix stripping. Program 14:130–137

# Part II

## Applications

# Chapter 9

# Analysis of Genome-Wide Association Data

Allan F. McRae

## Abstract

The last decade has seen substantial advances in the understanding of the genetics of complex traits and disease. This has been largely driven by genome-wide association studies (GWAS), which have identified thousands of genetic loci associated with these traits and disease. This chapter provides a guide on how to perform GWAS on both binary (case–control) and quantitative traits. As poor data quality, through both genotyping failures and unobserved population structure, is a major cause of false-positive genetic associations, there is a particular focus on the crucial steps required to prepare the SNP data prior to analysis. This is followed by the methods used to perform the actual GWAS and visualization of the results.

**Key words** Genome-wide association, SNP cleaning, Population stratification, Imputation, Case–-control, Quantitative trait

## 1 Introduction

Unlike single gene disorders (such as Cystic fibrosis or Huntington's disease), complex traits (e.g., height, body mass index) and diseases (e.g., schizophrenia, Type 2 diabetes) are the result of the combination of many genes and environmental factors, with each gene variant individually affecting an individual's trait or disease risk by a small amount [1]. While we had been successful at the elucidation of the genes and genetic variants underlying single gene disorders, no genes underlying complex traits had been identified prior to the advent of genome-wide association studies (GWAS) [2].

GWAS test the association of hundreds-of-thousands or millions of single-nucleotide polymorphisms (SNPs) across the genome with complex traits and diseases. While the first GWAS occurred earlier [3, 4], the first large GWAS using SNPs that cover a large percentage of the genome came from the Wellcome Trust Case-Control Consortium (WTCCC) in 2007 [5]. In the 5 years following that, over 2000 loci that were significantly and robustly associated with complex traits were identified [6], with the number of significant SNP-trait associations surpassing 14,000 in 2013 [2].

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_9, 

The analysis of GWAS data has a number of statistical challenges and potential pitfalls. In particular, false-positive associations can be generated through population stratification and poor quality genotyping [7]. This chapter will discuss stringent preparation of data for GWAS to avoid these pitfalls and the subsequent analysis and visualization of the results.

## 2 Data Quality Control and Cleaning

The most important and time-consuming task in a GWAS is the preparation of the data. As the final analysis will be testing the association of hundreds of thousands, or millions, of SNPs with a phenotype, even small systematic biases can lead to large numbers of false-positive results, and potentially false-negative, findings [8]. The cleaning process is generally divided into two steps: first removing any individuals with poor quality data and then removing SNP markers that have substandard genotyping performance. Performing the per-individual steps first prevents individuals with poor quality genotypes having an undue influence on the removal of SNP markers in the later step.

### 2.1 Per-Individual Quality Control

Quality control at an individual level aims to remove samples that have issues affecting their genotypes throughout the genome. This will be divided into five steps: (1) removal of individuals with excess missing genotypes, (2) removal of individuals with outlying homozygosity values, (3) remove of samples showing a discordant sex, (4) removal of related or duplicate samples, and (5) removal of ancestry outliers. As the removal of an individual from the analysis is costly—both in terms of the cost of the genotyping and the time spent preparing the DNA sample—it is important to spend time during the initial study design to ensure to the extent possible that all individuals are from a common ancestral background and that extracted DNA is of high quality.

1. Removal of individuals with excess missing genotypes: Modern genotyping arrays call the genotype at a SNP by comparing the intensity of florescence of the two alleles [9, 10]. Clusters are formed for individuals with high intensity for the A allele and low intensity for the B allele, for individuals with high intensity for the B allele and low intensity for the A allele, and for those with intermediate intensities for both alleles. Individuals falling in these clusters are given the genotypes AA, BB, and AB, respectively (Fig. 1). Any individual falling outside these genotype clusters is given a missing genotype. Large numbers of missing SNP calls for an individual indicate that the genotype is failing to fall into any of these clusters, which can be caused by low quality or concentration of the DNA used for genotyping.

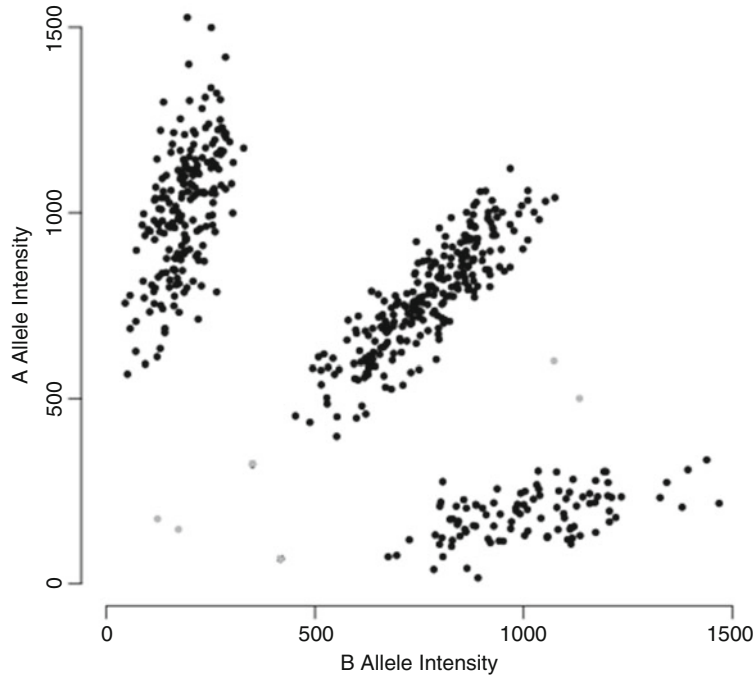



**Fig. 1** Example of SNP genotype clustering. Three distinct classes of SNPs can be seen corresponding to the AA, AB, and BB genotypes. A few individuals (colored *gray*) fall outside of these clusters and are unable to be assigned a genotype

Samples with a high missingness rate also tend to have higher genotyping error in the genotypes that are called, so need to be removed completely from analysis. Typically, a threshold in the order of 5 % missingness is used to determine which samples need to be removed; however, an appropriate threshold should be determined for each experiment by looking at the distribution of missingness across samples and removing the outliers. This step is particularly important when using a case–control design, especially when the DNA extraction was performed separately for cases and controls, as differential genotype quality may correlate with disease status and thus introduce a bias to the analysis [7].

2. Removal of individuals with outlying homozygosity values: The proportion of homozygous (or inversely heterozygous) genotypes across an individual's genome (excluding sex chromosomes) can detect several issues with genotyping. Average heterozygosity correlates with genotype missingness such that samples with high missingness tend to have lower average heterozygosity, although a reduction in heterozygosity can also reflect inbreeding. Sample contamination, where multiple samples are accidentally genotyped on a singe array, results in

high average heterozygosity. The average value of the proportion of heterozygous genotypes will vary across populations and genotyping platforms and as such the high and low thresholds for sample removal need to be determined by examining the distribution in your cohort.

3. Removal of samples showing a discordant sex: Determining whether an individual is male or female is straightforward from genotyping array data. Because males only have a single copy of the X chromosome, they cannot be heterozygous for any markers (outside the small psuedo-autosomal regions at the end of the chromosome). Depending on the genotype-calling algorithm, males may have a few heterozygous genotypes due to the low background genotyping error rate, although some platforms consider these as missing values. Starting with females, individuals with low heterozygosity across the X chromosome are indicative of a sample mix-up with a male. For males, samples with high heterozygosity—or excess missingness—are likely to be females. Provided males and females have been randomly placed on plates for genotyping, patterns of mismatching sex can be used to rectify potential plating errors.
4. Removal of related or duplicate samples: When using population cohorts for GWAS, it is important to exclude related or duplicated individuals from the analysis. While the case for removing duplicated individuals is obvious, even individuals with a relatively distant relationship can bias the analysis. For example, if we have two related cases in a case–control analysis, their genotypes being on average more similar to each other than the rest of the cohort will provide a slight bias to the estimate of the allele frequency in cases and its associated standard error [8]. Even this small bias is important when considering the number of statistical tests being performed. Duplicate and related individuals are detected using the Identity-by-State (IBS) metric, which measures the average proportion of alleles shared by two individuals across the autosomal genome. The IBS measure can be converted to a measure of the degree of relatedness between a pair called Identity-by-Descent (IBD). IBD is interpreted as the proportion of the genome that is shared between two individuals. For duplicate samples, or monozygotic twins, we expect that IBD = 1, IBD = 0.5 for first-degree relatives and 0.25 for second-degree relatives. A maximal threshold of IBD = 0.1875 (which is halfway between that expected of second- and third-degree relatives) is common [8], although selecting a smaller threshold that removes outlying pairs from your cohort is warranted. If any pair of individuals is found to be related, it is best to remove the one with the lowest genotyping rate as determined earlier.

5. Removal of ancestry outliers: Population stratification is a major source of bias in GWAS, as it is common for disease or quantitative traits to have different frequencies or distributions across populations [11]. This is a particularly important consideration for case–control studies although quantitative trait studies are also affected. For example, Campbell et al. [12] performed an association analysis on two groups of individuals of European descent that were discordant for height and identified an association with the *LCT* locus. This locus has historically undergone strong selection in certain European populations, resulting in the frequency of its variants differing significantly across the populations who also differed in average height. While this may be considered an extreme example, even small amounts of population stratification in an apparently homogeneous population can bias association results.

   The most common method used to detect ancestral outliers is principal component analysis (PCA), although other methods such as multidimensional scaling can equally be used. Principal component analysis is a multivariate statistical method that produces a set of uncorrelated variables (principal components) such that the first principal component explains the most variation in the data, followed by the second, and so on. Using a reference population with large ancestry differences ensures (at least) the first two principal components reflect major ancestral differences, with a commonly used reference being the HapMap data [13] that includes populations from Africa (YRI), Asia (CHB + JPT), and Europe (CEU). When calculating principal components it is important to first filter the SNPs into an approximately independent subset of approximately 50,000 SNPs to avoid undue influence of regions with high linkage disequilibrium.

   Once the principal components are generated from the reference populations, the PCA model is applied to the study cohort to calculate their principal component values, allowing them to be clustered alongside the HapMap individuals (Fig. 2). Any outlying individuals should be removed from the analysis, with a typical threshold being any individuals that are further than four standard deviations away from the cluster mean.

### 2.2 Per-Marker Quality Control

The second stage of genotype cleaning involves looking at individual SNPs to determine genotype accuracy. As discussed earlier, genotypes at each SNP are typically called by clustering individuals into three clusters representing AA, AB, and BB genotypes. Ideally each individual cluster plot would be investigated to confirm the separation of clusters and thus accuracy of genotype calling. However, this is impractical with current genotyping arrays that have hundreds of thousands of SNPs. Instead, statistical metrics of SNP

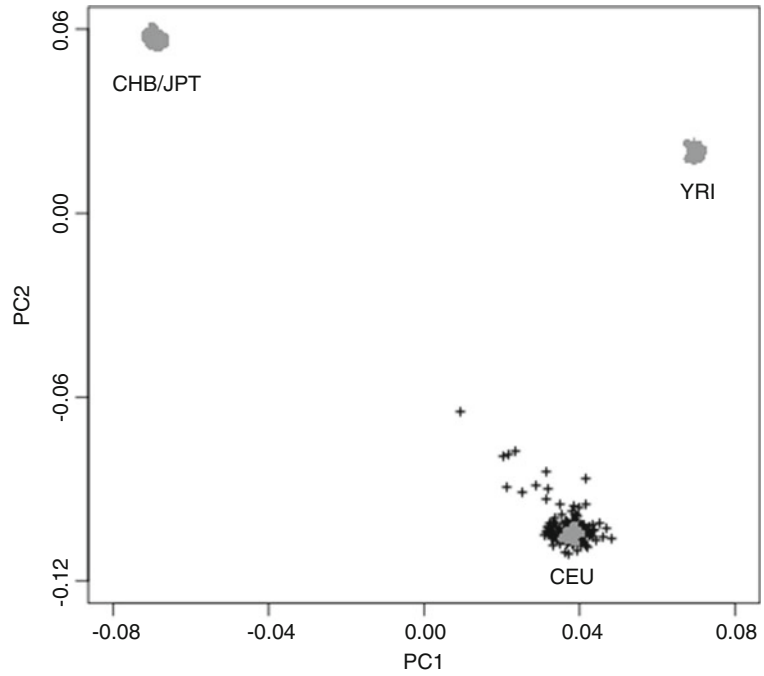



**Fig. 2** Principal component analysis to detect ancestral outliers in a study population. The four HapMap populations representing the three major ancestral groups are used as anchors to analyze the study population (*black crosses*). In this example, the majority of the study population is of primarily European ancestry and is clustered around the CEU population. A few individuals show admixture with a proportion of Asian ancestry and need to be removed before performing the GWAS analysis

quality are used to determine whether to remove them from the analysis. While every SNP removed is potentially a missed association, an attempt will be made to recover them using imputation in the next section. This stage will be divided into four steps: (1) removal of SNPs with excess missing genotypes, (2) removal of SNPs that deviate from Hardy–Weinberg equilibrium, (3) removal of SNPs with low minor allele frequency, and (4) comparing minor allele frequency to known values.

1. Removal of SNPs with excess missing genotypes: Poor separation of clusters during genotype calling will result in an increase in the number of individuals falling into a region that is not clearly in one genotype cluster or another and thus having their genotype called as missing. A SNP should be excluded from further analysis if it has greater than 5 % missing genotypes, although more stringent thresholds have also been applied.

   When analyzing case–control association studies, a secondary check of the missing data rates is required. As missingness can be nonrandom with respect to the underlying genotype, differential missing genotype rates between cases and controls

can lead to false-positive results. Any SNP showing a difference in missingness with $p < 10^{-6}$ should be removed from further analysis. This step is particularly important if the case and control cohorts have been collected at different times as there may be a difference in DNA quality between the cohorts.

2. Removal of SNPs that deviate from Hardy–Weinberg equilibrium: The principle of Hardy–Weinberg equilibrium (HWE) provides us expected genotype frequencies given an allele frequency. Poor genotype calling, including issues due to poor cluster separation, can result in deviations from genotype frequencies expected under HWE [14]. Typically a threshold of $p < 10^{-6}$ is used to exclude SNPs from further consideration. One caveat is that selection can also cause deviations from HWE, and thus exclusion due to this could result in missing important disease associations [15]. Therefore, in case–control studies, only the controls should be used to screen for deviations from Hardy–Weinberg.
3. Removal of SNPs with low minor allele frequency: Typically, any SNPs with a minor allele frequency less than 1 % are removed from the analysis. SNPs at this frequency have very few individuals with heterozygous genotype or the rare homozygous genotype and thus it is difficult to determine accurate genotype clusters when calling genotypes. Also, there is low power to detect associations for SNPs at low frequencies and alternative strategies for association testing combining multiple variants should be used [16].
4. Comparing minor allele frequency to known values: A final check of genotype quality is to compare the minor allele frequency of the called genotypes to highly genotyped cohorts such as the HapMap [13] and 1000 Genomes samples [17]. While there will be difference in allele frequency between the two cohorts due to ancestry differences and sampling variation, an overall similarity in allele frequencies ensures the annotation of the SNPs being analyzed is correct and individual deviations are a sign of genotyping error.

## 3 Imputation

Genotype imputation is the process of predicting, or imputing, genotypes that are not known in a sample of individuals. This uses a reference panel of densely genotyped individuals to impute SNP genotypes for a study set that have only been genotyped at a subset of those SNPs. Predicting these unobserved SNPs increases the amount of SNPs that can be tested for association, which in turn increases the power of the study and the ability to fine-map the

causal variants [18]. It also facilitates a meta-analysis of multiple cohorts by ensuring that all cohorts have a common set of SNPs.

Genotype imputation generally is performed in two steps. First, the samples being imputed are phased—that is their genotypes are separated into their two component haplotypes. While this step is not strictly necessary for some imputation software, performing the haplotyping step separately can improve overall computational performance. The second step is to fill in missing genotypes in the target sample using a reference panel of haplotypes with a dense set of SNP. The basic principle relies on finding sections of a haplotype in the reference population that share SNP alleles with the haplotype being imputed, which indicates that region of the genome is descended from a common ancestor and thus carries the same SNP alleles across the whole haplotype.

Clearly, the larger the reference panel of haplotypes, the greater chance of finding a matching haplotype and the more accurate the imputation will be. Initially the HapMap populations were used as a reference for imputing, but larger reference samples are continuously becoming available, including the 1000 Genomes individuals. Even when imputing a sample from a single ancestral background, the imputation accuracy is improved by using a reference panel covering many ancestral backgrounds [19].

After imputation, each SNP is given a set of probabilities of having the three possible genotypes. A number of measures have been proposed to describe the accuracy of imputation at a given SNP, although there is a strong correlation between each of them. These statistics provide an $R^2$ measure that lies within the range of 0–1, with a value of 1 indicate there is no uncertainty in the imputed genotypes and 0 indicating that imputation provided no information on that SNPs genotypes. A SNP imputation $R^2$ value in a sample of $N$ individuals has the equivalent power to having $N \times R^2$ actually genotyped individuals [18]. Any SNPs with poor imputation accuracy are typically filtered out before any association analysis. The threshold used depends whether the genotypes probabilities are being used in the analysis, or whether a "best-guess" genotype will be used by selecting the genotype with the highest probability. When using genotype probabilities, a low threshold of $R^2 > 0.3$ can be used, although there is no bias introduced by including all SNPs. A much more stringent threshold of at least $R^2 > 0.8$ should be used when considering best-guess genotypes. If using best-guess genotypes, it is important to repeat the filtering for Hardy–Weinberg equilibrium after the imputation.

## 4 Association Testing

After preparing the genetic data, performing the actual association analysis is relatively straightforward. At each SNP in the genome, a simple statistical test is performed to assess the association between

the SNP and trait of interest. The analysis of quantitative traits and disease (or binary) traits will be considered separately.

### 4.1 Quantitative Traits

Genetic association analysis for quantitative traits is performed using a linear regression. Typically a simple model is considered, where each SNP is encoded 0, 1, or 2 representing the number of B alleles in the genotypes AA, AB, and BB, respectively. For imputed SNPs the equivalent value is calculated by taking $2 \times P_{BB} + P_{AB}$ where $P_{BB}$ and $P_{AB}$ are the probabilities of genotypes BB and AB, respectively. This is referred to as the additive model of association as each copy of the B allele is represented as adding to the trait value. While it is possible to test more complex modes of genetic inheritance, in general we do not know the mechanism of inheritance at a SNP a priori and in this case the simple additive model is usually used.

Genetic association at each SNP in the genome is tested by performing a linear regression of the trait value of the SNP value. The use of a regression framework allows the inclusion of any known covariates that may affect the trait, such as age or sex. Due to the large number of statistical tests being performed, it is particularly important that the assumptions underlying the regression model are satisfied. Of particular importance, the normality of residuals needs to be ensured under the null hypothesis of no association. If needed, this can be achieved by rank normalizing the trait data, in which the trait is scaled such that it becomes normally distributed. If covariates with large effects are included in the analysis, it is preferential to correct for these in a linear regression first and then ensure the normality of the residuals from this model.

### 4.2 Disease Traits

Genetic association tests for disease traits test whether the proportion of B alleles at a SNP differs between cases and controls. This tests for a multiplicative model of association, where each copy of the B allele increases the risk of developing the disease by a factor $r$ for each B allele carried, i.e., a baseline risk of $b$ for genotype AA, a risk of $br$ for genotype AB, and a risk of $br^2$ for genotype BB. Again more complex models could be applied to the data, but the true inheritance model is generally unknown and thus multiplicative models are used in the first instance.

Testing for association at a SNP can be done using a simple chi-square contingency table test with a $2 \times 2$ matrix containing the counts of A and B alleles for cases and controls in each row. A Fisher's Exact Test could also be used, although cell counts are generally large enough to use the chi-square approximation given the filtering of rare SNPs that was performed during the data cleaning. For imputed data, the number of B alleles in an individual can be generated as earlier for quantitative traits and the total across all cases and controls taken. This will end up with noninteger

number counts for each cell in the contingency table, meaning the Fisher's Exact Test cannot be used. If covariates are to be included in the analysis, a logistic regression framework is used instead. When no covariates are used, a logistic regression is statistically equivalent to the contingency table model.

### 4.3 Significance

It is important to correct for the large number of tests performed in a GWAS study when assessing the significance of a result. Correcting for the number of SNPs tested using (e.g.) a Bonferroni correction is overly conservative due to the linkage disequilibrium between SNPs, particularly when using imputed data. It has been shown that a significance threshold of $5 \times 10^{-8}$ corrects for the effective number of independent tests genome-wide [20]. A less stringent threshold of $1 \times 10^{-5}$ is widely used to indicate "suggestive" significance, although many results are expected to achieve this level of significance in a typical GWA study.

### 4.4 Visualization of Results

GWAS results are typically represented using a Manhattan plot, with genomic locations along the $X$-axis and the negative logarithm (base 10) of the $p$-value along the $\Upsilon$-axis, with each point signifying an individual SNP (Fig. 3a). The SNPs with the strongest associations will have the greatest negative logarithms and will tower over the background of unassociated SNPs—much like skyscrapers in the Manhattan skyline. This plot provides an additional check on the quality of the association test, as multiple SNPs should be contributing to each peak, especially when using imputed data. A single outlying SNP can indicate poor quality genotyping and the initial clustering of the genotypes for that SNP should be inspected. In addition, a Manhattan plot showing significant points occurring across the genome should be considered suspect and the presence of confounder effects such as genotyping batch effects (particularly for case–control studies), undetected relatedness and sample duplications, or population stratification will need to be reassessed.

A QQ plot is a common way to demonstrate the lack of confounding effects. In this plot, the ordered observed negative logarithm of the p-values is plotted against the expected distribution. Ideally, the points in the plot should align along the $X = \Upsilon$ line, with deviation at the end for the significant associations (Fig. 3b). One way to quantify the lack of global inflation in the QQ plot is the genomic inflation factor ($\lambda_{GC}$). This is calculated by determining the median $p$-value of your GWAS test statistics, and calculating the quantile in a chi-squared distribution with one degree of freedom that would give this $p$-value. This is divided by the median of a chi-squared distribution with one degree of freedom (0.4549), to give $\lambda_{GC}$. Deviations of this value away from 1.0 indicate genome-wide confounding in the data.

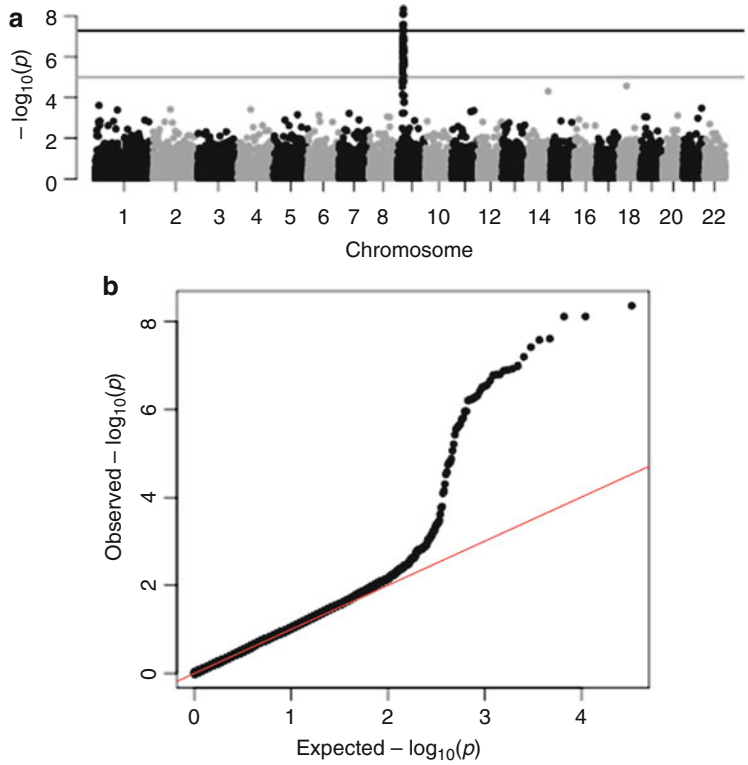



**Fig. 3** Example (**a**) Manhattan and (**b**) QQ plots from a genome-wide association study. The Manhattan plot shows a clear association signal on chromosome 9 that passes the genome-wide significance threshold (*black line*). No other significant or suggestive (*gray line*) signals are present. The QQ-plot shows no deviation from the expected line for low *p*-values, indicating no confounding due to population structure is present

### *4.5 Replication of Significant Results*

Despite the care taken in ruling out confounding factors in any study analysis, the gold standard in GWAS is to replicate all significant results in a second independent population [21]. When replicating, only individual SNPs need be considered. Thus, the burden of multiple testing is much lower and a more modest level of significance can be used.

## 5 Software

The majority of the initial data cleaning and association analysis can be performed using any statistical package. For example, there are several libraries available in the R suite of software. However, it is advantageous to use software specifically designed for the genetic association analysis. One of the most widely used pieces of software in this field is PLINK [22], which has functions for performing both data cleaning and association analyses. PLINK can perform multidimensional scaling to investigate ancestral outliers, otherwise

the principal component analysis can be performed using EIGENSOFT [23]. A range of software packages is available to perform haplotyping and imputation, with several of the leading software implementations being roughly equivalent. One potential combination that provides robust, high-quality imputed genotypes is using SHAPEIT for haplotyping [24, 25] and IMPUTE2 for the imputation [19, 26].

## References


1. Stranger BE, Stahl EA, Raj T (2011) Progress and promise of genome-wide association studies for human complex trait genetics. Genetics 187:367–383
2. Welter D, MacArthur J, Morales J, Burdett T, Hall P, Junkins H et al (2014) The NHGRI GWAS Catalog, a curated resource of SNP-trait associations. Nucleic Acids Res 42: D1001–D1006
3. Dewan A, Liu M, Hartman S, Zhang SS-M, Liu DTL, Zhao C et al (2006) HTRA1 promoter polymorphism in wet age-related macular degeneration. Science 314(5801):989–992
4. Klein RJ, Zeiss C, Chew EY, Tsai J-Y, Sackler RS, Haynes C et al (2005) Complement factor H polymorphism in age-related macular degeneration. Science 308(5720):385–389
5. The Wellcome Trust Case Control Consortium (2007) Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls. Nature 447 (7145):661–678
6. Visscher PM, Brown MA, McCarthy MI, Yang J (2012) Five years of GWAS discovery. Am J Hum Genet 90(1):7–24
7. Clayton DG, Walker NM, Smyth DJ, Pask R, Cooper JD, Maier LM et al (2005) Population structure, differential bias and genomic control in a large-scale, case-control association study. Nat Genet 37(11):1243–1246
8. Anderson CA, Pettersson FH, Clarke GM, Cardon LR, Morris AP, Zondervan KT (2010) Data quality control in genetic case-control association studies. Nat Protoc 5 (9):1564–1573
9. Rabbee N, Speed TP (2006) A genotype calling algorithm for affymetrix SNP arrays. Bioinformatics 22(1):7–12
10. Teo YY, Inouye M, Small KS, Gwilliam R, Deloukas P, Kwiatkowski DP et al (2007) A genotype calling algorithm for the Illumina BeadArray platform. Bioinformatics 23 (20):2741–2746
11. Cardon LR, Palmer LJ (2003) Population stratification and spurious allelic association. Lancet 361:598–604
12. Campbell CD, Ogburn EL, Lunetta KL, Lyon HN, Freedman ML, Groop LC et al (2005) Demonstrating stratification in a European American population. Nat Genet 37 (8):868–872
13. The International HapMap 3 Consortium (2010) Integrating common and rare genetic variation in diverse human populations. Nature 467:52–58
14. Turner S, Armstrong LL, Bradford Y, Carlson CS, Crawford DC, Crenshaw AT et al (2011) Quality control procedures for genome-wide association studies. Curr Protoc Hum Genet Chapter 1, Unit 1.19
15. Wittke-Thompson JK, Pluzhnikov A, Cox NJ (2005) Rational inferences about departures from Hardy-Weinberg equilibrium. Am J Hum Genet 76(6):967–986
16. Spencer CCA, Su Z, Donnelly P, Marchini J (2009) Designing genome-wide association studies: Sample size, power, imputation, and the choice of genotyping chip. PLoS Genet 5 (5):e1000477
17. The 1000 Genomes Project Consortium (2012) An integrated map of genetic variation from 1,092 human genomes. Nature 491 (7422):56–65
18. Marchini J, Howie B (2010) Genotype imputation for genome-wide association studies. Nat Rev Genet 11(7):499–511
19. Howie B, Marchini J, Stephens M (2011) Genotype imputation with thousands of genomes. G3 1(6):457–470
20. Pe'er I, Yelensky R, Altshuler D, Daly MJ (2008) Estimation of the multiple testing burden for genomewide association studies of nearly all common variants. Genet Epidemiol 32(4):381–385
21. Chanock SJ, Manolio T, Boehnke M, Boerwinkle E, Hunter DJ, Thomas G et al (2007) Replicating genotype-phenotype associations. Nature 447(7145):655–660

22. Purcell S, Neale B, Todd-Brown K, Thomas L, Ferreira MAR, Bender D et al (2007) PLINK: a tool set for whole-genome association and population-based linkage analyses. Am J Hum Genet 81(3):559–575
23. Price AL, Patterson NJ, Plenge RM, Weinblatt ME, Shadick NA, Reich D (2006) Principal components analysis corrects for stratification in genome-wide association studies. Nat Genet 38(8):904–909
24. Delaneau O, Marchini J, Zagury J-F (2011) A linear complexity phasing method for thousands of genomes. Nat Methods 9 (2):179–181
25. Delaneau O, Howie B, Cox AJ, Zagury JF, Marchini J (2013) Haplotype estimation using sequencing reads. Am J Hum Genet 93 (4):687–696
26. Howie B, Fuchsberger C, Stephens M, Marchini J, Abecasis GR (2012) Fast and accurate genotype imputation in genome-wide association studies through pre-phasing. Nat Genet 44(8):955–959

# Chapter 10

# Adjusting for Familial Relatedness in the Analysis of GWAS Data

**Russell Thomson and Rebekah McWhirter**

## Abstract

Relatedness within a sample can be of ancient (population stratification) or recent (familial structure) origin, and can either be known (pedigree data) or unknown (cryptic relatedness). All of these forms of familial relatedness have the potential to confound the results of genome-wide association studies. This chapter reviews the major methods available to researchers to adjust for the biases introduced by relatedness and maximize power to detect associations. The advantages and disadvantages of different methods are presented with reference to elements of study design, population characteristics, and computational requirements.

**Key words** Genome-wide association studies, GWAS, Relatedness, Confounding, Population stratification, Cryptic relatedness, Familial structure

## 1 Introduction

Genome-wide association studies (GWAS) represent an effective means of identifying genetic variants associated with disease risk, and increasing sample sizes allow variants of diminishing effect size to be identified. However, GWAS results can be confounded by population stratification, in which ancestry differences result in systematic differences in allele frequencies leading to spurious associations, and familial relatedness, in which the presence of related individuals within the sample have the potential to violate the assumptions of common analytical tools and to artificially inflate test statistics leading to false positives. Familial relatedness can be further categorized into familial structure, in which relationships are known and pedigrees can be constructed, and cryptic relatedness, in which relationships are unknown.

**Electronic supplementary material**: The online version of this chapter (doi:10.1007/978-1-4939-6613-4_10) contains supplementary material, which is available to authorized users.

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_10, 

It can be argued that population stratification and cryptic relatedness are really two aspects of a single confounder: unknown relationships between study participants [1]. The difference between these two ideas is scale; while population structure refers to the effect of ancient relatedness between groups of participants, cryptic relatedness is unknown but relatively recent relatedness between individual participants. However, it has been argued that population structure and cryptic relatedness as such are not the source of confounding, but rather these issues are proxies for the real source of confounding, in which other causative loci confound the estimate of the effect of a given locus, collectively known as "genetic background" [2]. In order to avoid the reporting of false associations, methods for addressing these potential sources of confounding have had to be built into the study design and analytical phases of GWAS.

Early studies addressed these issues by recruiting participants from homogeneous populations, making use of pedigree information, and removing related individuals from analyses. It became increasingly apparent, however, that in order to identify variants of moderate or small effect size, as well as rarer variants, much larger sample sizes were necessary [3]. As sample sizes have increased, ensuring homogeneity has become an increasingly unrealistic goal, rendering these early methods based on study design less viable.

There are also many circumstances in which pedigree information is unobtainable or unreliable, introducing the problem of cryptic relatedness. This is also a problem for studies undertaken in population isolates, which are particularly useful in order to increase power to detect rare variants, as well as for undertaking homozygosity analyses to identify recessive variants [4, 5]. While population isolates tend to exhibit greater phenotypic, environmental, and genetic homogeneity, as well as increased rare allele frequency resulting from bottlenecks, they also highlight the need to mitigate the confounding effects of relatedness [6]. Similarly, family based designs are robust to population stratification, but need to account for relatedness.

For all these reasons, there have been substantial efforts directed at developing statistical methods to correct for the biases introduced by relatedness within a sample. Initially, simple corrective techniques were employed for population stratification, such as genomic control and principal components analysis (PCA). Genomic control has been widely used both to identify and to correct inflation of test statistics resulting from population stratification [7], whereas PCA uses genotype data to identify the most important dimensions of genetic variation, which can then be used to adjust the data to account for the population structure they describe [8]. The problem with these approaches is that they are

aimed primarily at addressing population structure and are less effective at addressing familial relatedness [9].

While these methods remain useful in certain circumstances, they have since been superseded by a range of more computationally sophisticated methods that have the additional benefit of also addressing the problem of cryptic relatedness and family structure. This chapter examines the more widely used tools and compares the utility of these methods in terms of their advantages and limitations for different study designs.

## 2 Methods

### 2.1 Association Using Pedigree Data

In the past, genetic association was carried out using population samples, and scientists with familial data carried out linkage studies. Linkage methods involve searching for segments of chromosome that segregate with a trait of interest in families. This approach has proven useful for Mendelian diseases and rare variants in complex diseases [10]. However, it has proven less effective for identifying common variants in complex disease, leading to efforts to develop approaches for combining association and linkage modeling methods in pedigree data [11]. One such method is contained in the software `LAMP` (Linkage and Association Modeling in Pedigrees) [12], which implements joint linkage and association using a maximum likelihood approach. Through this, it is possible to test whether SNPs within a linkage signal are in linkage disequilibrium with the putative disease allele. This can reduce the chromosomal area of interest, from an often quite broad linkage peak.

The first family-based association method was developed for nuclear families and dubbed the Transmission-Disequilibrium Test (TDT) [13]. The TDT can detect linkage only when genetic association is present. While association can be observed through the confounding, linkage is unaffected and so the TDT is robust to population structure. There are a suite of methods that extend the TDT to larger pedigrees, known collectively as the Family Based Association Tests [14]. While the methods are designed for nuclear family data, they can be used on large pedigree data. This is achieved by treating nuclear families within pedigrees as independent, under the null hypothesis of no association between the marker and trait of interest. These methods, like the TDT, allow for population stratification between families. Family based association testing is implemented in the software `FBAT`.

Variance component (VC) methods have a long history of use in the analysis of pedigree data in human quantitative genetics, animal genetics and animal breeding. They can be used to measure the genetic influence on a continuously varying quantitative trait, and usually assume that the trait is normally distributed. The idea behind these methods is to divide the phenotypic variance into

genetic and environmental components, with the result that heritability, linkage and association can all be determined using pedigree data. VC methods rely on a linear mixed effects regression model, where the non-independence among family members is accounted for by modeling the variance structure of the relationships between individuals as a random effect. As it is based on a linear regression model, covariates (like age and sex), as well as relatedness, can be adjusted for. However, VC methods do not account for population structure.

SOLAR (Sequential Oligogenic Linkage Analysis Routines) is a program designed to carry out linkage and association analyses, based on a variance component method [15]. The SOLAR package is well maintained and includes many extra features, such as estimating identity-by-descent (IBD), allowing for a household effect, multiple trait analyses, and genetic interactions, among many others. An eigen-simplification of the calculation of the likelihood has been incorporated into SOLAR, which has increased the speed of the program, while still calculating exact $p$-values [16].

Another method that uses pedigree data in this way is based on the $M_{\mathrm{QLS}}$ statistic [17]. It is an extension of $W_{\mathrm{QLS}}$ and $CC_{\mathrm{QLS}}$ statistics; in fact, $M_{\mathrm{QLS}}$ stands for "modified" or "more powerful" quasi-likelihood score, giving its name to the program, MQLS. These methods calculate a statistic based on the difference in the allele frequencies of the cases and controls, using a quasi-likelihood method with a $\chi^2$ distribution on 1 degree of freedom under the null hypothesis. The $M_{\mathrm{QLS}}$ statistic provides greater weight to individuals with closely related disease-carrying relatives, thus producing an even more powerful test. This method has been shown to be more powerful than other VC methods for a dichotomous trait (such as case status); however, the disadvantage is that it is not possible to adjust for covariates. GLOGS (Genome-wide LOGistic mixed model/Score test) also uses a quasi-likelihood method for dichotomous traits, while adjusting for relatedness [18]. This approach is based on a logistic regression model, and it can be used to adjust for up to three covariates while testing for association.

MASTOR uses a variance components method to adjust for relatedness from the known pedigrees [19]. This method is akin to the MQLS method, but for detecting association with a continuous trait. Unlike MQLS, it is possible to adjust for covariates. The advantage MASTOR has over an earlier method, known as GTAM, is that, similar to the MQLS method, it can account for missing phenotype, genotype or covariate data, even when missingness is related to the trait of interest, and exploiting the relatedness within the sample to model heritability and increase power. Approaches like these are useful for studies containing participants with known relatedness as well as unrelated participants.

### 2.2 Association with Adjustment for Cryptic Relatedness and Population Structure

There are numerous methods that use variance components to adjust for relatedness by using genetic data to infer relatedness between individuals. These methods can also adjust for population stratification and are sometimes called genetic control methods. All methods described in this section are based on variance components methods, using a linear mixed effect model and assuming a continuous trait. They can also be applied to a dichotomous trait (such as disease status), under the assumption of a threshold-liability model [20]. This assumes that, while the observed disease status is dichotomous, the unobserved disease liability follows a normal distribution and that there is a linear transformation between the two. However, when analyzing a dichotomous trait with a variance component method, it is not possible to calculate the standard effect size of an odds ratio, which is often used for pooling comparable data in genome-wide association meta-analyses [21].

Variance component methods can be loosely divided into approximate and exact methods. Approximate methods are useful because of their improved computer speed and memory usage compared to their exact counterparts, which become computationally impracticable in larger cohorts. While the approximate methods maintain the same type I error rates as exact methods, for some pedigree structures approximate methods suffer a loss of power [22].

Nevertheless, there have been some advances in the speed of exact algorithms. For example, the software `GEMMA` and `FaST-LMM` are able to obtain the same *p*-values as the `EMMA` method, but within a fraction of the time [22–24]. The `GEMMA` method fits a Bayesian sparse linear mixed model (BSLMM) using Markov chain Monte Carlo (MCMC) for estimating the proportion of variance in phenotypes explained. Similarly, the `BOLT-LMM` method uses a Bayesian mixed model to more accurately model genetic architecture, thereby increasing the power to detect associations in larger cohorts with reduced time and memory demands [25].

The simplest method to implement is the GRAMMAR-Gamma algorithm [26]. It fits a linear mixed model using the kinship matrix and the phenotype of interest for the null model of no association between phenotype and genotype. It then uses the residuals of the null model to search for an association with genotype using a standard linear model. This method is the fastest method available and it is easy to implement within the *R* statistical framework, using the library `GenABEL` [27]. However, it has been shown to have a substantial loss of power for more complicated pedigree structures [22].

There are a number of approximate methods available that avoid re-estimating the variance components for each genotype by rewriting the model in terms of a single parameter (the ratio of genetic variance to residual variance). This keeps the heritability

estimated from the null model fixed when testing each genotype, resulting in a substantial reduction in memory and computer run time. Methods such as P3D (Population Parameters Previously Determined), GCTA, FaST_LMM and EMMAX can implement this [28]. GCTA also can be used to estimate the variance explained from all SNPs in a genome-wide study, providing the scope to predict the trait of an individual in a replication set, based on genome wide data [29]. The software that implements the P3D method (TASSEL) also includes a method to further reduce computer time (called compression), by clustering the individuals into fewer groups based on the kinship among the individuals [30].

While FaST-LMM can calculate either exact or approximate $p$-values, it can also capitalize on a few other approximate tricks to speed the process up. One is to use a realized relationship matrix instead of IBD, so that just one spectral decomposition is needed to test all SNPs [31]. Another is to choose a subset of 4000 or 8000 equally spaced markers to estimate cryptic relatedness. A recent simulation study, however, suggests that using a subset of markers will adjust for cryptic relatedness, although population stratification correction may be compromised as a result [32]. FaST-LMM-Select addresses this issue by extending FaST-LMM to include principal components of the genotype matrix as fixed effects [33].

Finally, there can be a loss of power when using the locus of interest to estimate cryptic relatedness. FaST-LMM and GCTA-LOCO can overcome this by implementing a method to leave out the chromosome containing the candidate locus for estimating genetic relatedness.

### 2.3 Known and Unknown Relatedness

Methods that can incorporate both pedigree information and cryptic relatedness are the most successful in accounting for confounding due to population structure and relatedness. Two such packages include ROADTRIPS and Mendel. The ROADTRIPS program uses the quasi-likelihood methods (implemented in the $M_{\mathrm{QLS}}$ and similar statistics), to incorporate both known and unknown relatedness [34]. The inclusion of pedigree data, where it is known, increases the power of this method to detect association in case–control studies, and uses genomic data to estimate a covariance matrix for unknown relationships, both recent and ancient.

Mendel is a software package that was first designed to implement a number of linkage algorithms. It has evolved over time to now include the capability to run linkage, association, gene dropping and many other genetic tools. Recently, it has included a "Pedigree GWAS" option, that implements a rapid variance components method that can adjust for both cryptic and known relationships in a genome wide association study [35].

### 2.4 Rare Variant and Sequence Analysis

Genome-wide association studies are usually undertaken using SNP array data, with newer chips covering millions of variants, including

some rarer SNPs identified in the 1000 Genomes Project [36]. Furthermore, as rare variants are likely to contribute substantially to heritability of complex diseases and the cost of whole exome and whole genome sequencing continues to drop, the issue of maintaining adequate power while adjusting for familial relatedness in association testing of sequencing data becomes increasingly pressing. Early work suggests that similar methods will be effective, such as combining LMM with a kernel score test to aggregate the effects of rare variants within a gene [11, 37]. That is, when investigating the role of rare variants on the trait of interest, the assumption is that multiple variants within a region have an effect, and it is therefore useful to be able to group the rare variants together, rather than assess them individually, as in standard GWAS methods.

Several other methods have been proposed to incorporate information across genes or chromosome regions, to test for association with the trait of interest. While it is beyond the scope of this chapter to go in to detail of each of these methods, we only list them here. Some examples of methods that incorporate relatedness adjustment include RHM (regional heritability mapping) with software `REACTA` [38] and `VEGAS` [39]. The sequence analysis tool, `VAAST` has been extended to allow the use of pedigree data (`pVAAST`) [40]. The extension to the Quasi Likelihood Methods for rare variants is implemented in `MONSTER` [41]. The *R* library that implements the GRAMMAR-Gamma method, `GenABEL` also has a function for rare variant analyses, called `cocohet` [42]. The adjusted SKAT method, `ASKAT`, is useful for investigating rare variants in family-based study designs as it combines the combines SKAT and Fast-LMM methods to control for cryptic and family relatedness [43]. And finally, the software `FBAT`, contains a test FBAT-v that is specifically designed for rare variants [44]. All of the above methods are summarized in Tables 1 and 2.

## 3 Example

To illustrate, we will work through an example of a genome wide association scan using a simulated dataset, where the study participants are related (*see* **Note 1**).

### 3.1 Dataset

The data consist of 165 individuals from the CEPH (Utah residents with ancestry from northern and western Europe) *hapmap* data set [45], including 50 trios, three parent–child relationships and nine unrelated individuals. The genotype data contain 65,173 SNPs across the genome.

This data set is designed to emulate a small GWAS study with the intention to find evidence for genetic regions that are associated with schizophrenia. Rather than a case–control study, this is a population-based study, where the subclinical quantitative trait of

**Table 1**
**A list of available methods, with the authors and certain capabilities given**

| Method | Authors | Trait types | Adjust for covariates? | Cryptic or known pedigree structure |
|---|---|---|---|---|
| FBAT | NM Laird | Continuous/ Categorical/ Time-to-event | No | Known |
| LAMP | M Li and G Abecasis | Categorical | No | Known |
| MQLS | T Thornton and M-S McPeek | Categorical | No | Known |
| GLOGS | S Stanhorpe and M Abney | Categorical | Yes | Known |
| MASTOR | L Jakobsdottir and M-S McPeek | Continuous | Yes | Known |
| SOLAR | J Blangero, K Lange, T Dyer, L Almasy, H Göring, J Williams, M Boehnke, C Peterson | Continuous/ Categorical[a] | Yes | Known |
| P3D | E Buckler, T Casstevens, P Bradbury | Continuous/ Categorical[a] | Yes | Cryptic |
| EMMAX | H Kang | Continuous/ Categorical[a] | Yes | Cryptic |
| FaST-LMM | C Lippert, J Listgarten, and D Heckerman | Continuous/ Categorical[a] | Yes | Cryptic |
| GEMMA | X Zhou and M Stephens | Continuous/ Categorical[a] | Yes | Cryptic |
| BOLT-LMM | P-R Lou | Continuous/ Categorical[a] | Yes | Cryptic |
| GRAMMAR-Gamma | YS Aulchenko | Continuous/ Categorical[a] | Yes | Cryptic |
| GCTA | J Yang and P Visscher | Continuous/ Categorical[a] | Yes | Cryptic |
| ROADTRIPS | T Thornton | Categorical | No | Both |
| Mendel | H Zhou, J Blangero, T Dyer, K Chan, E Sobel, K Lange | Continuous/ Categorical[a] | Yes | Both |

[a]These variance component methods are designed for a continuous trait; however, it is possible to use them to analyze a categorical trait if proper attention is given to rare variants, rare diseases and small sample sizes

the Ekman 60-Faces emotion recognition test [46] was used as an endophenotype for schizophrenia. All individuals in the data set have both genotype and phenotype information (*see* **Note 2**).

The phenotype is correlated with the covariates: age, sex and education. The variance explained by the SNPs in this study was 50 %. There are six causal variants, with equal effect size distributed

**Table 2**
**A list of available methods with software format and links to the software page**

| Method | Software format | Link |
|---|---|---|
| FBAT | Command Line and menu driven. Source Code and Win/Mac/Linux executables | http://www.hsph.harvard.edu/fbat |
| LAMP | Command Line. Source Code and Win/Mac/Linux executables | http://www.sph.umich.edu/csg/abecasis/LAMP |
| MQLS | Command Line. Source Code and Win/Mac/Linux executables | http://www.stat.uchicago.edu/~mcpeek/software/MQLS/index.html or http://www.sph.umich.edu/csg/liang/MQLS |
| GLOGS | Command Line. C Source Code | http://www.bioinformatics.org/~stanhope/GLOGS |
| MASTOR | Command Line. C Source Code | http://www.stat.uchicago.edu/~mcpeek/software/MASTOR |
| SOLAR | Command Line. Win/Mac/Linux Executable | http://solar.txbiomedgenetics.org/ |
| P3D | `TASSEL`: JAVA Command Line, Win/Mac/Linux Executable. R Library `GAPIT`. SAS Code | http://www.maizegenetics.net/statistical-genetics |
| EMMAX | Command Line. Source Code and Win/Linux executables | http://genetics.cs.ucla.edu/emmax |
| FaST-LMM | Command Line. Win/Linux executables | http://mscompbio.codeplex.com/ |
| GEMMA | Command Line. Linux executable | http://stephenslab.uchicago.edu/software.html#gemma |
| BOLT-LMM | Command Line. Linux executable | http://www.hsph.harvard.edu/alkes-price/software/ |
| GRAMMAR-Gamma | R library `GenABEL` | http://www.genabel.org |
| GCTA | Command Line. Linux executable | http://www.complextraitgenomics.com/software/gcta |
| ROADTRIPS | Command Line. Source code | http://www.stat.uchicago.edu/~mcpeek/software/ROADTRIPS |
| Mendel | Command Line. Source code | http://genetics.ucla.edu/software/mendel |

across the genome. They are SNPs: rs2393646, rs10793370, rs6769400, rs6774660, rs210400, rs1481440, and rs9465317 (*see* **Note 3**).

### 3.2 Method

Here we describe, step-by-step the implementation of the `GCTA` software to carry out the association while adjusting for relatedness.

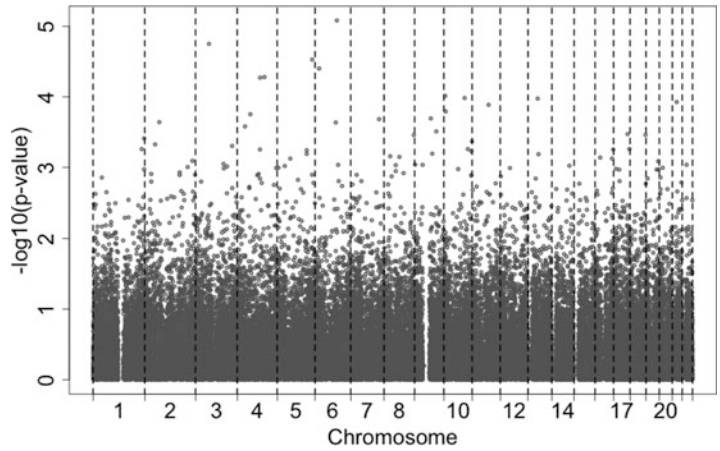



**Fig. 1** A Manhattan plot of the results from the MLMA option in GCTA

1. Download the data (*see* **Note 1**) and the `GCTA` software (*see* Table 2) and run the commands:

   ```
   gcta64 --bfile CEU --make-brm --autosome --out CEU
   gcta64 --mlma-loco --mlma-no-adj-covar --bfile CEU
   --grm-bin CEU --out EK --pheno EK.txt --qcovar
   age_edu.txt --covar gender.txt
   ```

2. Load results in `R` [47] or `Haploview` [48] and create a Manhattan plot (*see* **Note 4**).
3. Create a QQ-plot and genomic inflation factor ($\lambda$) (*see* **Note 5**). Convention suggests that with $\lambda < 1.05$, your $p$-values are not overly inflated due to relatedness, population stratification or some other reason.

### 3.3 Results

As can be seen in Fig. 1, there are no SNPs with a $p$-value less than the conventional threshold for genome wide significance of $5 \times 10^{-8}$. This study consists of multiple causal variants and a small sample size. To reach genome wide significance would require one causal variant with a large effect size and/or a very large sample size. If you examine the $p$-values for the causal SNPs in this simulation you will find that, while they have very small $p$-values, they do not appear in a list of the top ten SNPs with the best evidence for association. This indicates that any study with similar attributes, the top ten hits are likely to be false positives.

Figure 2 shows the QQ-plot for the p-values produced when adjusting for relatedness using the `GCTA` software, while Fig. 3 shows the QQ-plot when relatedness was not adjusted for (*see* **Note 6**). Note that the points do not lie on the straight line in Fig. 3 but they do in Fig. 2. This indicates that, unless relatedness is taken into account, the $p$-values are spuriously inflated. When using the leave-one-chromosome-out command in `GCTA`, the $p$-values

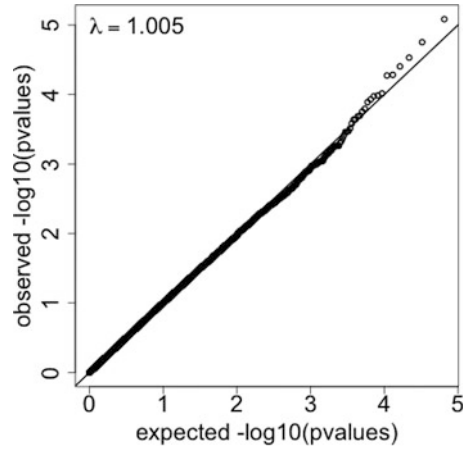



**Fig. 2** The QQ-plot for the GCTA MLMA results (*see* **Note 5**)

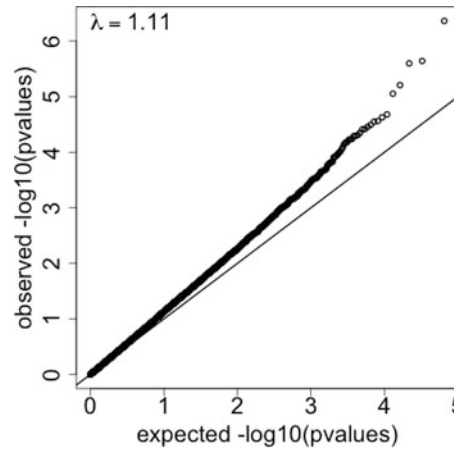



**Fig. 3** The QQ-plot for association results, not adjusted for relatedness (*see* **Note 5**)

show a similarly high level of genomic inflation as in Fig. 3. This suggests that, with as few as 65,000 SNPs, it is important to use all available SNPs to estimate the genetic relationships matrix.

### 3.4 Discussion

There is thus a large and continually expanding array of tools available for undertaking genome-wide association studies in the presence of relatedness, whether known or unknown, ancient or recent. It is apparent that much research has focused on development of linear mixed model methods, with an emphasis on refining methods to reduce computational costs. Precisely which tool will be most appropriate for a given study will depend on the characteristics of the sample and the population from which it was drawn, as well as considerations relating to computation time and memory usage. The tables above provide a starting point for identifying the

salient features of each method, and for narrowing down the choices.

All of the methods described here, except two (LAMP and MQLS), can either adjust for unknown population stratification or include principal components as a covariate. Care should be taken when using principal components as a covariate as standard PCA methods can potentially be confounded by the presence of related individuals in a sample and so PCA methods accounting for this should be used in preference [50].

Overall, the GRAMMAR-Gamma method implemented in GenABEL is the fastest method, and one of the easiest to implement. However, it utilizes an approximate, rather than exact, method that means it is not well-suited to samples with complicated pedigrees or studies that are potentially under-powered [22]. In such instances, an exact method would be preferable, despite the increase in computational demands.

For case–control studies, the methods MQLS and ROAD-TRIPS are best for incorporating participants with incomplete data. This is a relatively common scenario, where a given study may include participants with missing genotype, phenotype or covariate data. In a familial prostate cancer study, for example, the authors found that by using MQLS, and thus incorporating individuals with unknown case status (women and young men) and unknown genotype (ungenotyped affected brothers), an increase in power was obtained that was of similar size to the loss of power that was a result of the cases being related [51]. It is also worth noting that adjusting for relatedness using a mixed model approach can be of value even in an ideal, unrelated population, because it will address the issue of confounding introduced by "genetic background"; that is, other unidentified causal loci elsewhere in the genome [2].

One final consideration is that case–control GWA studies often oversample disease cases to increase study power. This leads to ascertainment bias that can result in a loss of power when adjusting for covariates [52]. Yang et al. [32] show, through simulation, that in studies with large ascertainment bias (that is, when the disease prevalence is small and the sample size is large), the linear mixed effects methods described in this review will suffer a substantial loss of power. For studies with unrelated individuals, it is better to adjust for population structure using standard PCA methods. For studies with ascertainment bias and related individuals that require adjusting for covariates, it is possible that the method based on logistic regression, GLOGS will not suffer the same loss of power. However, further simulation studies are required to confirm this.

To minimize false GWAS positives, it is important to adjust for confounding due to familial relatedness within samples. This review has identified a range of tools that can be used to achieve this goal in different scenarios. This is an area currently receiving a great deal of

attention, and new packages are continually being developed. As whole genome (and whole exome) sequencing becomes more common, and sample sizes continue to expand, we can expect to see more tools produced that address the needs of these studies and address some of the limitations of the current methods.

## 4 Notes

1. This can be accessed at https://cloudstor.aarnet.edu.au/plus/index.php/s/fzn0GhsDovA1A6z.
2. The nature of the data will influence the choice of analysis software. In the current scenario, genotype information is available for all individuals and GCTA is therefore appropriate. When working with a dataset that included ungenotyped individuals, using `ROADTRIPS` for case–control studies or `MASTOR` for quantitative trait studies would allow these individuals to be included and potentially increase the power of the study.
3. The simulation of the phenotype was based on (a)—linear regression in *R* for the correlation with the covariates and (b)—`GCTA` for the association with the causal variants. The `GCTA` simulation used the option `--simu-qt`, which assumes a simple additive genetic model.
4. To generate the Manhattan plot in `Haploview`, the output file from `GCTA` (filename: *EK.mlma*) can be entered in to `Haploview` using the PLINK format option. The `R` code to generate the Manhattan plot is as follows:

```
gcta <- read.table("EK.mlma",header=T)
lengthchr <- tapply(gcta$bp,gcta$Chr,max)
pos <- c(0,sapply(1:22,function(i)
sum(as.numeric(lengthchr[1:i]))))
png("manhattan_plot_gcta.png",width=800,
   height=500)
par(mar=c(4,5,1,1))
plot(pos[gcta$Chr]+gcta$bp,-1*log10(gcta$p),
   cex=0.6,xaxt="n",ylab="-log10(p-value)",
   xlab="Chromosome",lwd=2,cex.lab=2,cex.
   axis=2,lwd=2,col=grey(0.4))
abline(v=pos,lty=2)
midpoint <- sapply(1:22,function(i) (pos[i]+pos[i
   +1])/2)
axis(1,at=pos,labels=F)
axis(1,at=midpoint,tick=FALSE,labels=1:22,cex.
   axis=2)
dev.off()
```

5. The R code used to generate the QQ-plot is as follows:

```
gcta <- read.table("EK.mlma",header=T)
pv = gcta$p
m=length(pv)
expect.stats=-log10(seq(1/(m+1),m/(m+1),length.
   out=m))
lambda=median(-log10(pv))/median(expect.stats)
png("QQplot_gcta.png")
par(mar=c(5,5,1,1))
qqplot(expect.stats,-log10(pv),xlab="expected -
   log10(pvalues)",ylab="observed   -log10(pva-
   lues)",cex.axis=2,cex.lab=2,lwd=2)
abline(a=0,b=1)
text(0,5,bquote(lambda==.(round(lambda,3))),
   adj=0,cex=2)
dev.off()
```

6. This association analysis was conducted without adjusting for relatedness using PLINK [49], via the command:

```
plink --noweb --bfile CEU --pheno EK.txt --linear --
   covar age_edu.txt --sex --out CEU_assoc_plink
```

## References


1. Astle W, Balding DJ (2009) Population structure and cryptic relatedness in genetic association studies. Stat Sci 24:451–471
2. Vilhjálmsson BJ, Nordborg M (2013) The nature of confounding in genome-wide association studies. Nat Rev Genet 14:1–2
3. Manolio TA, Collins FS, Cox NJ, Goldstein DB, Hindorff LA, Hunter DJ, McCarthy MI, Ramos EM, Cardon LR, Chakravarti A et al (2009) Finding the missing heritability of complex diseases. Nature 461:747–753
4. Jakkula E, Leppä V, Sulonen A-M, Varilo T, Kallio S, Kemppinen A, Purcell S, Koivisto K, Tienari P, Sumelahti M-L et al (2010) Genome-wide association study in a high-risk isolate for multiple sclerosis reveals associated variants in *STAT3* gene. Am J Hum Genet 86:285–291
5. McQuillan R, Leutenegger A-L, Abdel-Rahman R, Franklin CS, Pericic M, Barac-Lauc L, Smolej-Narancic N, Janicijevic B, Polasek O, Tenesa A et al (2008) Runs of homozygosity in European populations. Am J Hum Genet 83:359–372
6. Zeggini E (2012) Next-generation association studies for complex traits. Nat Genet 43:287–288
7. Devlin B, Roeder K (1999) Genomic control for association studies. Biometrics 55:997–1004
8. Price AL, Patterson NJ, Plenge RM, Weinblatt ME, Shadick NA, Reich D (2006) Principal components analysis corrects for stratification in genome-wide association studies. Nat Genet 38:904–909
9. Price AL, Zaitlen NA, Reich D, Patterson N (2010) New approaches to population stratification in genome-wide association studies. Nat Rev Genet 11:459–463
10. Wooster R, Neuhausen SL, Mangion J, Quirk Y, Ford D, Collins N, Nguyen K, Seal S, Tran T, Averill D et al (1994) Localization of a breast cancer susceptibility gene, *BRCA2*, to chromosome 13q12-13. Science 265:2088–2090
11. Li Y, Foo JN, Liany H, Low H-Q, Liu J (2014) Combined linkage and family-based association analysis improved candidate gene

detection in Genetic Analysis Workshop 18 simulation data. BMC Proc 8:S29

12. Li M, Boehnke M, Abecasis GR (2005) Joint modeling of linkage and association: identifying SNPs responsible for a linkage signal. Am J Hum Genet 76:934–949
13. Spielman RS, Ewens WJ (1998) A sibship test for linkage in the presence of association: the sib transmission/disequilibrium test. Am J Hum Genet 62:450–458
14. Zhou JJ, Yip W-K, Cho MH, Qiao D, McDonald M-LN, Laird NM (2014) A comparative analysis of family-based and population-based association tests using whole genome sequence data. BMC Proc 8:S33
15. Almasy L, Blangero J (1998) Multipoint quantitative-trait linkage analysis in general pedigrees. Am J Hum Genet 62:1198–1211
16. Blangero J, Diego VP, Dyer TD, Almeida M, Peralta J, Kent JWJ, Williams JT, Almasy L, Göring HH (2013) A kernel of truth: statistical advances in polygenic variance component models for complex human pedigrees. Adv Genet 81:1–31
17. Thornton T, McPeek MS (2007) Case-control association testing with related individuals: a more powerful quasi-likelihood score test. Am J Hum Genet 81:321–337
18. Stanhope SA, Abney M (2012) GLOGS: a fast and powerful method for GWAS of binary traits with risk covariates in related populations. Bioinformatics 28:1553–1554
19. Jakobsdottir J, McPeek MS (2013) MASTOR: mixed-model association mapping of quantitative traits in samples with related individuals. Am J Hum Genet 92:652–666
20. Falconer DS (1965) The inheritance of liability to certain diseases, estimated from the incidence among relatives. Ann Hum Genet 29:51–76
21. Chen MH, Liu X, Larson MG, Fox CS, Vasan RS, Yang Q (2011) A comparison of strategies for analyzing dichotomous outcomes in genome-wide association studies with general pedigrees. Genet Epidemiol 35:650–657
22. Zhou X, Stephens M (2012) Genome-wide efficient mixed model analysis for association studies. Nat Genet 44:821–824
23. Eu-ahsunthornwattana J, Howey RAJ, Cordell HJ (2014) Accounting for relatedness in family-based association studies: application to Genetic Analysis Workshop 18 data. BMC Proc 8:S79
24. Listgarten J, Lippert C, Kadie CM, Davidson RI, Eskin E, Heckerman D (2012) Improved linear mixed models for genome-wide association studies. Nat Methods 9:525–526
25. Loh P-R, Tucker G, Bulik-Sullivan BK, Vilhjalmsson BJ, Finucane HK, Salem RM, Chasman DI, Ridker PM, Neale BM, Berger B, Patterson N, Price AL (2015) Efficient Bayesian mixed model analysis increases association power in large cohorts. Nat Genet 47:284–290
26. Svishcheva GR, Axenovich TI, Belonogova NM, van Duijn CM, Aulchenko YS (2012) Rapid variance components-based method for whole-genome association analysis. Nat Genet 44:1166–1170
27. Aulchenko YS, Ripke S, Isaacs A, van Duijn CM (2007) GenABEL: an R library for genome-wide association analysis. Bioinformatics 23:1294–1296
28. Kang HM, Sul JH, Service SK, Zaitlen NA, Kong SY, Freimer NB, Sabatti C, Eskin E (2010) Variance component model to account for sample structure in genome-wide association studies. Nat Genet 42:348–354
29. Yang J, Lee SH, Goddard ME, Visscher PM (2011) GCTA: a tool for genome-wide complex trait analysis. Am J Hum Genet 88:76–82
30. Bradbury PJ, Zhang Z, Kroon DE, Casstevens TM, Ramdoss Y, Buckler ES (2007) TASSEL: software for association mapping of complex traits in diverse samples. Bioinformatics 23:2633–2635
31. Lynch M, Ritland K (1999) Estimation of pairwise relatedness with molecular markers. Genetics 152:1753–1766
32. Yang J, Zaitlen NA, Goddard ME, Visscher PM, Price AL (2014) Advantages and pitfalls in the application of mixed-model association methods. Nat Genet 46:100–106
33. Tucker G, Price AL, Berger B (2014) Improving the power of GWAS and avoiding confounding from population stratification with PC-Select. Genetics 197:1045–1049. doi:10.1534/genetics.1114.164285
34. Thornton T, McPeek MS (2010) ROADTRIPS: case-control association testing with partially or completely unknown population and pedigree structure. Am J Hum Genet 86:172–184
35. Lange K, Papp JC, Sinsheimer JS, Sripracha R, Zhou H, Sobel EM (2013) Mendel: the Swiss army knife of genetic analysis programs. Bioinformatics 29:1568–1570
36. The 1000 Genomes Project Consortium (2012) An integrated map of genetic variation from 1,092 human genomes. Nature 491:56–65
37. Svishcheva GR, Belonogova NM, Axenovich TI (2014) FFBSKAT: fast family-based sequence kernel association test. PLoS One 9: e99407

38. Uemoto Y, Pong-Wong R, Navarro P, Vitart V, Hayward C, Wilson JF, Rudan I, Campbell H, Hastie ND, Wright AF et al (2013) The power of regional heritability analysis for rare and common variant detection: simulations and application to eye biometrical traits. Front Genet 4, Article 232
39. Liu JZ, Mcrae AF, Nyholt DR, Medland SE, Wray NR, Brown KM, Investigators AMFS, Hayward NK, Montgomery GW, Visscher PM et al (2010) A versatile gene-based test for genome-wide association studies. Am J Hum Genet 87:139–145
40. Hu H, Roach JC, Coon H, Guthery SL, Voelkerding KV, Margraf RL, Durtschi JD, Tavtigian SV, Shankaracharya, Wu W et al (2014) A unified test of linkage analysis and rare-variant association for analysis of pedigree sequence data. Nat Biotechnol 32:663–669
41. Jiang D, McPeek MS (2014) Robust rare variant association testing for quantitative traits in samples with related individuals. Genet Epidemiol 38:10–20
42. Liu F, Struchalin MV, van Duijn K, Hofman A, Uitterlinden AG, Aulchenko YS, Kayser M (2011) Detecting low frequent loss-of-function alleles in genome wide association studies with red hair color as an example. PLoS One 6: e28145
43. Oualkacha K, Dastani Z, Li R, Cingolani PE, Spector TD, Hammond CJ, Richards JB, Ciampi A, Greenwood CMT (2013) Adjusted sequence kernel association test for rare variants controlling for cryptic and family relatedness. Genet Epidemiol 37:366–376
44. De G, Yip W-K, Ionita-Laza I, Laird N (2013) Rare variant analysis for family-based design. PLoS One 8:e48495
45. Thorisson GA, Smith AV, Krishnan L, Stein LD (2005) The international HapMap project web site. Genome Res 15:1592–1593
46. Ekman P, Friesen WV (1976) Pictures of facial affect. Consulting Psychologists Press, Palo Alto, CA
47. R Core Team (2014) R Foundation for Statistical Computing, Vienna, Austria
48. Barrett JC, Fry B, Maller J, Daly MJ (2005) Haploview: analysis and visualization of LD and haplotype maps. Bioinformatics 21:263–265
49. Purcell S, Neale B, Todd-Brown K, Thomas L, Ferreira MAR, Bender D, Maller J, Sklar P, de Bakker PIW, Daly MJ et al (2007) PLINK: a tool set for whole-genome association and population-based linkage analyses. Am J Hum Genet 81:559–575
50. Thornton TAA, Austin MA (2013) Software and data resources for genetic association studies: Mini Review. CAB Rev 8:1–6
51. Fitzgerald LM, Patterson B, Thomson R, Polanowski A, Quinn S, Brohede J, Thornton T, Challis D, Mackey DA, Dwyer T et al (2009) Identification of a prostate cancer susceptibility gene on chromosome 5p13q12 associated with risk of both familial and sporadic disease. Eur J Hum Genet 17:368–377
52. Pirinen M, Donnelly P, Spencer CC (2012) Including known covariates can reduce power to detect genetic effects in case-control studies. Nat Genet 44:848–851

# Chapter 11

# Analysis of Quantitative Trait Loci

David L. Duffy

## Abstract

Although the term quantitative trait locus (QTL) strictly refers merely to a genetic variant that causes changes in a quantitative phenotype such as height, QTL analysis more usually describes techniques used to study oligogenic or polygenic traits where each identified locus contributes a relatively small amount to the genetic determination of the trait, which may be categorical in nature. Originally, too, it would be clear that it covered segregation and genetic linkage analysis, but now genetic association analysis in a genome-wide SNP or sequencing experiment would be the commonest application. The same biometrical genetic statistical apparatus used in this setting—analysis of variance, linear or generalized linear mixed models—can actually be applied to categorical phenotypes, as well as to multiple traits simultaneously, dealing with and taking advantage of genetic pleiotropy. Most recently, they are being used to make inferences about population and evolutionary genetics, with applications ranging from human disease to control of disease-causing organisms. Several computer software packages make it relatively straightforward to fit these statistically complex models to the large amounts of genotype and phenotype data routinely collected today.

**Key words** Biometrical genetics, Mixed model, Kinship, Linkage analysis, Association analysis, Linkage disequilibrium, Population genetics

## 1 Introduction

The basic QTL model is a simple linear regression equation with one trait and one measured locus [1]:

$$y_i = g_i + e_i$$

where $y_i$ is the trait value for the $i$th individual; $g_i$ is the average trait value in that population for the particular genotype the individual carries (genotypic mean); and $e_i$ is the perturbation from the genotypic mean in that individual (residual), due to the buffeting of environmental and developmental factors, which we will model as being a random value drawn from some statistical distribution such as the Gaussian. The simplest extension of this model includes terms for those environmental and developmental factors that can be measured—a multiple regression.

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_11, 

The usual tests for the adequacy of a linear model must be applied here; otherwise statistical tests of significance will not perform correctly. That is, the observed distribution of residuals should match the chosen theoretical distribution, residuals should be uncorrelated with genotype, and residuals should not be correlated between different individuals. In most genome-wide association scans (GWASs), a simple regression of this type is the workhorse, fitting one regression per measured locus. The quantile–quantile (QQ) plot seen in almost every GWAS paper is one graphical test that can highlight failures in the distributional assumptions. For example, when fewer significant loci are detected than would be expected by chance, this suggests model misspecification (or widespread data mismeasurement)—this will not be due to confounding by population genetic structure, though it can reflect batch effects in genotyping error rates if batches also differ by phenotype [2]. A formal test for determining whether a mathematical transformation of the trait values (e.g., log, square root) will improve model adequacy is the Box–Cox maximum likelihood approach [3].

The usual linear model is fitted via ordinary least squares (OLS), which is equivalent to assuming a residual Gaussian distribution. The far more flexible generalized linear model (GLM) [4] incorporates a link function (a phenotype transformation) and a variety of residuals distributions in a maximum likelihood framework fitted by iterative methods. The most familiar GLM is logistic regression, which allows the basic model to be applied to binary traits (most usually disease states in the human literature), but GLMs extend to ordered and unordered (multi-)categorical traits, survival times, and odd-shaped continuous distributions, the commonest being Poisson, negative binomial, and Gamma distributions. These models are all available using standard statistical software.

The cause of a correlation between residuals and genotype that most concerns geneticists is the existence of ethnic stratification, where different subpopulations differ in genotype frequencies and in phenotype distribution [2]. This is particularly important because the effects of individual QTLs studied are generally small. If the presence of this stratification in a study sample is not recognized, combining the subpopulations in an analysis leads to spurious QTL detection. A more extreme example is when the sample contains close relatives. One approach to dealing with these phenomena is to elaborate the linear model to allow correlated residuals, a *random effects* model. Since we will retain the measured genotypes and other covariates in the model, these are usually called *mixed models*, because they contain both fixed and random effects [5]. Random effects in genetic models are relatively unusual in that we can strongly specify the correlations between the individuals using either genetic theory (in the case of relatives we know these

from Mendel's laws) or genomic sharing (an empirical or realized relationship) based on genome-wide genotyping [2]. This model can be written [1]:

$$\mathbf{y} = \mathbf{Qg} + \mathbf{Xb} + \mathbf{Zu} + \mathbf{e}$$

where **y**, **u**, **e** are now vectors of values for the sample (of length $n$); **g** the regression coefficient for each genotype class (e.g., a vector of length 3 for a simple nucleotide variant); **b** the regression coefficients for other measured covariates; **u** that portion of the residuals that are correlated with one another; **e** the uncorrelated portion; and **Q** and **X** are matrices indicating the genotype or covariate value for each individual. The **Z** matrix maps individuals onto the **u** values and will usually be an identity matrix for our purposes, but can vary to handle monozygotic twins and repeated measures. For the model to be fittable by iterative likelihood-based methods, we must specify the correlation matrices for each random effect: for **e**, an $n \times n$ diagonal matrix **E** (zero correlation between $e_i$ and $e_j$, $i \neq j$), while for **u**, it is the known relatedness of individuals i and j, an $n \times n$ matrix **G** estimated from pedigree and/or GWAS data [6–8]. Members of the same subpopulation (or family) will have similar **u** values, which will capture the relationship between ethnicity and trait values that causes confounding in the simpler models. Again a number of computer packages are now available for this task [9–11], but the computational task is intensive (i.e., time consuming) for the large datasets currently studied.

Fitting of a *generalized linear mixed model* to, for example, a binary trait is even more intensive, and so is usually not feasible for an entire modern GWAS. Most current published studies using mixed models for binary traits treat them as if they are continuous (coding them as 0 and 1), a practice that can only ever be approximately correct (and which should be borne in mind assessing significance tests from these studies). Categorical traits can also be analyzed using the same software, by creating dummy binary variables that indicate membership of each category (dropping one category to make the model identified). In practice, if the category and genotypes proportions are high enough, then results are reasonably trustworthy.

In the case that QTL genotypes are not directly measured, these too can be treated as additional random effects. The appropriate correlation matrix for this random effect is the empirical relationship matrix for the genetic region of the QTL—the correlation matrix that measures the probability that alleles at the QTL in a pair of individuals were inherited from the same ancestor (i.e., are *identical by descent*). This is a variance components *genetic linkage analysis*, and a likelihood ratio test can be used to determine linkage of trait phenotypes to QTL genotypes. Statistical power of linkage analysis in the natural pedigrees accessible for humans is much

lower than that of association analysis using measured QTL genotypes. It does have the advantage that linkage of markers to the QTL extends over much greater distances along the chromosome than linkage disequilibrium, so that fewer markers need to be genotyped. In addition, because linkage extends further, a less severe correction for multiple testing is required, so that a linkage test $P < 6 \times 10^{-5}$ is genome-wide significant with an experiment-wise Type 1 error rate of ~0.05 [12]. For historical reasons, the linkage test result is usually displayed as the decimal log likelihood ratio—*lod*, where the above *P*-value corresponds to a lod of 3.5.

More elaborate models that incorporate segregation and linkage disequilibrium information from pedigrees can be fitted using specialized software [13], but again these are most useful when the effect size for the QTL is sizeable—examples would include the ACE insertion–deletion polymorphism and serum ACE levels, major "Mendelian" QTLs like those for familial hypercholesterolemia, and many variants affecting gene expression (*eQTLs*). In families segregating these variants, the trait distribution is often obviously multimodal, each mode representing a genotype. There is much current interest in incorporating linkage information in the identification of rare causative variants in sequence data as although the information contribution from linkage analysis may not be large, when combined with association and functional evidence, it may be decisive in choosing between several likely variants [14].

The same equations can be extended to multiple phenotypes simultaneously and to incorporate genotypes at multiple loci simultaneously. In the case of multitrait analyses, one is able to estimate the contribution of the QTL to the phenotypic correlation between pairs of traits. Furthermore, if traits are not strongly phenotypically correlated, and a QTL is pleiotropic in action, then this increases statistical power for gene detection [15].

When modeling multiple QTLs simultaneously, one often encounters model identification problems due to having too few data points per regression coefficient to be estimated. The first simplification we can carry out is model genotypic effects as a sum of allelic effects (a regression of allele dose on genotype means). This *additive model* is usually quite appropriate (it neglects dominance effects, which are mostly small) and halves the parameter count. In another direction, approaches similar in spirit to stepwise regression attempt to reduce the number of loci in the model to a core set of influential QTLs and can be fitted more easily. These can be Bayesian in nature or various types of penalized regression [16].

A related problem is the use of large-scale genotypic data to predict an individual phenotype value (genomic prediction)—this is of great interest to animal breeders, and in the populations they study can be quite precise. In human populations, such a prediction can be of interest for genetic epidemiology, notably when testing if QTLs for one trait can be used to predict a second trait (a test of

multilocus pleiotropy). A currently theoretical application is to use the predicted value for a disease phenotype to tailor individual disease screening regimens or therapy. In the dosage polygenic risk score model, one fits an additive model to each locus in turn, and uses the resulting coefficients in a full prediction model using the observed genotypes. The assumption under this model is that there is no significant epistasis, that is, gene-by-gene interaction.

## 2 A Standard Workflow for QTL Association Analysis of a GWAS

Almost all the computer programs suggested here are run from the command line, usually in Unix or related operating systems such as OSX, although many will run under Windows (*see* **Note 1**). Most require a lot of memory. GWAS analyses are usually best run on computer clusters, as the analyses can be parallelized easily by dividing up data by chromosome or chromosome chunk.

Determine an appropriate transformation of the phenotype under study. Usually, information will be available in the literature, for example, log transformation of body mass index or serum cholesterol levels. In the case of eQTLs, quite elaborate transformation (*normalization*) is essential. The Box–Cox procedure can be used to confirm or select an optimal transform, ideally including a number of known covariates in the regression. Alternatively, a rankit transformation is not uncommonly used, where each value is replaced with the normal score corresponding to the same rank in the observed distribution. This preliminary is not specific to genetic analysis but is an important first step.

If the dataset includes closely related individuals, carry out a purely family based *phenometric* mixed model analysis, using the kinship matrix based on the known pedigree. This gives an estimate of the trait heritability. This can point to errors, and if enough data is available, gives an upper bound to combined effects of the QTLs. Most mixed model packages will allow one to enter a pedigree, from which the appropriate correlation matrix (often the matrix inverse, as this saves some calculation) will be calculated for this analysis, e.g., *polygenic*() and *polygenic_hglm*() in the R *GenABEL* package [17], the R *MCMCglmm* package [18], *MERLIN* [19], *MENDEL* [13], or *SOLAR* [20].

Carry out appropriate cleaning of the available genotype data (as discussed in Chapter 9 in this volume). Imputation of unobserved genotypes has the advantage of increasing statistical power by ~10 % and improving localization of the true QTL, as opposed to nearby variants in linkage disequilibrium. Principal components for study participant genotypes can be calculated to use as covariates that offer another way of controlling effects of ethnic stratification. Genotype data can be stored as PLINK or compressed VCF files. The *PLINK2* program is one package that can convert backwards and forwards between these formats (https://www.cog-genomics.org/plink2/).

There is a large and increasing number of computer programs that fit QTL models. The *GEMMA* package (http://www.xzlab.org/software.html) [10, 21] is a (relatively) friendly and fast program for estimating empirical kinship matrices and performing linear mixed model association analysis. It also offers a Bayesian sparse linear mixed model approach to reducing the number of QTLs in the model to a manageable number. It reads a PLINK (or a BIMBAM) genotype file and a plain text phenotype file. An initial run calculates the empirical relationship matrix, which can then be used in subsequent mixed model fitting runs. Here is output from the calculation of the relationship matrix (based on PLINK format data files `colour4.fam`, `colour4.bim`, `colour4.bed`, and phenotype file `dummy4.dat`):

```
## GEMMA Version = 0.93
##
## Command Line Input = -b colour4 -p dummy4.dat -gk -miss 0.5 -o GABC
##
## Summary Statistics:
## number of total individuals = 2190
## number of analyzed individuals = 2190
## number of covariates = 1
## number of total SNPs = 911857
## number of analyzed SNPs = 811485
##
## Computation Time:
## total computation time = 54.8707 min
## computation time break down:
## time on calculating relatedness matrix = 54.395 min
```

Since a small correlation must exist between overall relationship and resemblance at a QTL, it is now usual to produce a relationship matrix excluding the chromosome of the genotype currently being tested for association—this slightly increases power to detect QTLs [22]. The results from testing for QTLs for human hair color treated as four unordered categories using GEMMA, where the -n option is used to select 3 of the 4 dummy variables (cols 1, 2, and 4) encoding these in the file `dummy4.dat`:

```
##
## GEMMA Version = 0.94beta
##
```

```
## Command Line Input = -b colour4 -p dummy4.dat -n 1 2 4
-k GABC.cXX.txt -lmm -miss 0.5 -o hc4
##
## Summary Statistics:
## number of total individuals = 2190
## number of analyzed individuals = 1179
## number of covariates = 1
## number of phenotypes = 3
## number of total SNPs = 1109421
## number of analyzed SNPs = 810682
## REMLE log-likelihood in the null model = 183.076
## MLE log-likelihood in the null model = 185.732
## REMLE estimate for Vg in the null model:
0.0267261
0.000351117     0.187691
-0.00381588     -0.0265142      0.12686
## se(Vg):
0.00581771
0.00906147      0.0289283
0.00429616      0.0102545       0.00876679
```

followed by results for known hair color loci, processed using the *postgwas* R package [23], which automatically recognizes GEMMA output files (Table 1).

The R commands which automatically generated plots and the previous table were as follows:

```
library(postgwas)
results = read.delim("hc4.assoc.txt")
postgwas(results)
quit(save=no)
```

**Table 1**
**Significant associations for known hair color loci**

| SNP | CHR | BP | *P* | Gene name |
|---|---|---|---|---|
| rs35395 | 5 | 33948589 | 3.6e−08 | *SLC45A2* |
| rs12203592 | 6 | 396321 | 2.8−11 | *IRF4* |
| rs12913832 | 15 | 28365618 | 2.0e−31 | *HERC2* |
| **rs1805008** | **16** | **89986144** | **6.2e−21** | ***MC1R*** |

## 3 A Standard Workflow for QTL Linkage Analysis

I will not specifically discuss experimental breeding studies, where particular tailored methods can take advantage of the design (e.g., the *R/QTL* package [24]), but the general approaches later can be applied to such data.

QTL linkage analysis of natural pedigrees is now usually performed where SNP genotype data has been collected, though microsatellite or RAPD markers may still be used in nonhuman pedigrees. Specialized linkage SNP arrays with 7000–11,000 markers are still available from some suppliers, but the fall in cost of denser SNP panels and of sequencing has made these less attractive. Since linkage extends over long genetic distances, and linkage disequilibrium actually makes linkage analysis more technically difficult, the data are first thinned to retain ~10,000 informative SNPs in linkage equilibrium with one another. Marker informativeness for linkage is proportional to the number of heterozygous genotypes in the pedigrees to be analyzed (i.e., common SNPs are more useful). SNP genotype data is cleaned as for association analysis (*see* Chapter 27), with the availability of pedigree data meaning that testing for genotype errors can take advantage of *Mendelian error detection*, where genotypes of parents and offspring must also be consistent at each locus, and also in terms of apparent recombination rates between neighboring loci (close double recombinants are very likely to represent a genotyping error). Similarly, the pedigree structure must be checked for nonpaternity and sample mix-up by comparing the empirical kinship matrix to the expected kinship matrix based on the reported pedigree.

Parametric QTL linkage analysis requires one to estimate or specify a model for the relationship between QTL genotype and the trait. If the QTL effect is large, this can be estimated using *segregation analysis* of pedigree data. The "nonparametric" variance components linkage (i.e., mixed model) approach does not require this step, and we concentrate on that later.

The *MERLIN* program (http://csg.sph.umich.edu/abecasis/Merlin/index.html) [19] supports both parametric and nonparametric linkage analysis, association analysis, and Mendelian error detection. It also imputes missing genotypes utilizing the pedigree structure. Three data files must be prepared (Table 2): one listing the traits and markers (".dat" file), a list of marker genetic map positions in sex-averaged centiMorgans (".map" file), and the parents and actual genotype and phenotype data for each individual (".ped" file)—the first five fields of the pedigree file are pedigree ID, individual ID, father ID, mother ID, and sex. Genetic map positions for SNPs can be estimated from files under http://hapmap.ncbi.nlm.nih.gov/downloads/recombination/, which give sequence coordinates (bp) and genetic map positions (cM)

**Table 2**
**Input files required by MERLIN**

| .dat file | .map file | .ped file |
|---|---|---|
| T trait1 | CHR MARKER POS | Ped1 id1 0 0 m 12.1 1.1 1 3/ 1 1/ 1 4/ 4 |
| C covariate1 | 13 rs9579484 0 | Ped1 id2 0 0 f 14.4 2.3 2 0/ 0 0/ 0 0/ 0 |
| C covariate2 | 13 rs9552488 0.021786 | Ped1 id3 id1 id2 f 9.1 5.3 1 1/ 3 1/ 1 4/ 4 |
| M rs9579484 | 13 rs9510743 4.99304 | Ped1 id4 id1 id2 f 16.2 2.7 1 1/ 3 1/ 1 4/ 4 |
| M rs9552488 | | |
| M rs9510743 | | |

along each chromosome. These *MERLIN* file formats differ only slightly from the *PLINK2* ".ped" and ".map" file formats (the latter has fields for both sequence and genetic map positions).

The *MERLIN*-associated *pedstats* program produces summary statistics and checks the pedigree file for errors. For example, all included individuals need to have both parents specified and a record for each included in the .ped file, or to have neither parent specified (where the individual is a *pedigree founder*). It is called from the command line as follows:

```
pedstats -d example.dat -p example.ped
```

*MERLIN's* Mendelian errors checking is invoked, as are all the other analyses, by using a command line flag:

```
merlin -d example.dat -m example.map -p example.ped --
errors
```

This produces a list of putative errors and a *P*-value in a file merlin.err. Another program, *pedwipe*, can then delete these automatically. The resulting "wiped.ped" file can then be analyzed. For example,

```
merlin -d skincol.dat -m skincol.map -p skincol.ped --
assoc
```

gives the following output for an analysis of the region around the *SLC45A2* gene associated with human pigmentation (as evidenced for hair color in the previous section), examining skin color measured as an ordinal trait (the "—assoc" option automatically performs linkage and association analyses, while the "—vc" option gives linkage only):

**Phenotype: skincol (ASSOC) (503 families, h2 = 100.00 %)**

| Linkage test results | | | | Association test results | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Position | H2 | LOD | *P*-value | Marker | Allele | Effect | H2 | LOD | *P*-value |
| 33.008 | 20.7 % | 1.75 | 0.0023 | rs7730923 | 1 | −1.280 | 2.93 % | 8.69 | 2.5e−10 |
| 33.008 | 20.9 % | 1.77 | 0.0022 | rs7715554 | 1 | −0.016 | 0.00 % | 0.01 | 0.82 |
| … | | | | | | | | | |
| 33.995 | 30.8 % | 3.81 | 1.4e−05 | rs28777 | 1 | −0.591 | 12.62 % | 37.00 | 6.1e−39 |
| 33.999 | 31.2 % | 3.93 | 1.0e−05 | rs28117 | 1 | −0.585 | 10.14 % | 27.16 | 4.9e−29 |
| 34.000 | 31.3 % | 3.96 | 9.7e−06 | rs26722 | 2 | −0.371 | 2.34 % | 6.26 | 7.9e−08 |
| 34.000 | 31.3 % | 3.97 | 9.5e−06 | rs183671 | 2 | −0.570 | 11.69 % | 34.11 | 4.9e−36 |
| 34.000 | 31.4 % | 3.97 | 9.4e−06 | rs6451049 | 2 | −0.597 | 4.51 % | 11.17 | 7.4e−13 |
| 34.001 | 31.9 % | 4.13 | 6.5e−06 | rs35410 | 1 | −0.472 | 4.12 % | 11.02 | 1.0e−12 |
| 34.004 | 32.1 % | 4.32 | 4.1e−06 | rs6451050 | 2 | −0.763 | 2.52 % | 6.02 | 1.4e−07 |
| 34.005 | 32.1 % | 4.32 | 4.1e−06 | rs35414 | 2 | −0.264 | 4.61 % | 11.13 | 8.2e−13 |
| 34.005 | 32.1 % | 4.32 | 4.1e−06 | rs3756464 | 2 | −0.166 | 1.88 % | 4.15 | 1.2e−05 |
| 34.007 | 32.1 % | 4.34 | 3.9e−06 | rs3822466 | 1 | −0.306 | 1.52 % | 3.47 | 6.3e−05 |
| 34.010 | 32.1 % | 4.36 | 3.8e−06 | rs16892049 | 2 | −1.282 | 1.56 % | 4.57 | 4.4e−06 |
| 34.016 | 31.9 % | 4.32 | 4.1e−06 | rs6414862 | 1 | −0.018 | 0.02 % | 0.04 | 0.68 |
| … | | | | | | | | | |

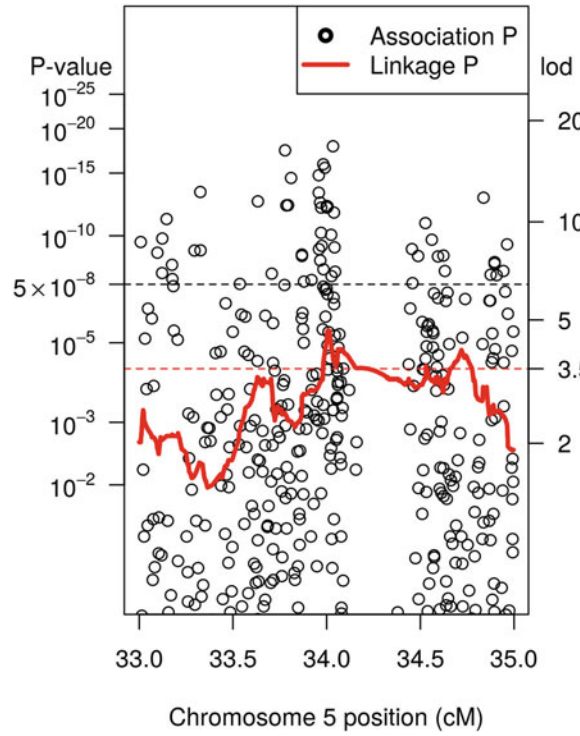



**Fig. 1** Linkage and Association results for a region of human chromosome 5 around the SLC45A2 gene

Extracting and plotting the positions and *P*-values from the above output (*see* Fig. 1) shows again that both linkage and association signals peak very closely, in both cases exceeding the threshold for genome-wide significance.

## 4 Note

1. There are a large number of computer packages for QTL analysis, and the number continues to increase. For example, our group has recently moved from carrying out QTL association analysis using *MERLIN* to *RAREMETALWORKER* [25], which uses the same algorithms as *MERLIN*, but reads *VCF* files natively and uses very little memory. Just within the R statistical computing environment [26] alone, there are the *bqtl*, *boss*, *BayHap*, *dlmap*, *eqtl*, *gap*, *GenABEL*, *hapassoc*, *haplo.stats*, *ibdreg*, *ldlasso*, *JAGUAR*, *MatrixEQTL*, *mqtl*, *multic*, *qtl* (*R/QTL*), *qtlhot*, *qtlmt*, *qtlnet*, *QTLRel*, *SNPassoc*, *snpstats*, *strum*, *wgaim*, and *WCQ* packages. A certain amount of functionality is common, so one should expect consistent answers from different packages [27], but I routinely perform duplicate analyses using different packages, at least for genomic regions

containing significant QTLs. In my own GWAS analyses, I usually deal with closely related individuals (either from nuclear or large extended pedigrees), and so the common approach of removing relatives from the dataset is not feasible. This is one reason I have concentrated in this review on association methods using empirical kinship matrices. Unfortunately, these matrices can be estimated using different approaches, which sometimes causes test results for a given locus to fluctuate significantly. Several recent papers have explored methods of stabilizing this effect by shrinkage estimation, regularization, or combination of multiple estimates (e.g., pedigree and GWAS based) [28, 29].

## References


1. Henderson C (1984) Applications of linear models in animal breeding. University of Guelph, Guelph, ON
2. Astle W, Balding DJ (2009) Population structure and cryptic relatedness in genetic association studies. Stat Sci 24:451–471
3. Box GE, Cox DR (1964) An analysis of transformations. J R Stat Soc B 26:211–252
4. Nelder JA, Nedderburn RW (1972) Generalized linear models. J R Stat Soc Ser A 135:370–384
5. Eisenhart C (1947) The assumptions underlying the analysis of variance. Biometrics 3:1–21
6. VanRaden PM (2008) Efficient methods to compute genomic predictions. J Dairy Sci 91:4414–4423
7. Choi Y, Wijsman EM, Weir BS (2009) Case-control association testing in the presence of unknown relationships. Genet Epidemiol 33:668–678
8. Yang J, Benyamin B, McEvoy BP, Gordon S, Henders AK, Nyholt DR, Madden PA, Heath AC, Martin NG, Montgomery GW, Goddard ME, Visscher PM (2010) Common SNPs explain a large proportion of the heritability for human height. Nat Genet 42:565–569
9. Bradbury PJ, Zhang Z, Kroon DE, Casstevens TM, Ramdoss Y, Buckler ES (2007) TASSEL: software for association mapping of complex traits in diverse samples. Bioinformatics 23:2633–2635
10. Zhang Z, Ersoz E, Lai C-Q, Todhunter RJ, Tiwari HK, Gore MA, Bradbury PJ, Yu J, Arnett DK, Ordovas JM, Buckler ES (2010) Mixed linear model approach adapted for genome-wide association studies. Nat Genet 42:355–360
11. Yang J, Lee SH, Goddard ME, Visscher PM (2011) GCTA: a tool for genome-wide complex trait analysis. Am J Hum Genet 88:76–82
12. Lander E, Kruglyak L (1995) Genetic dissection of complex traits: guidelines for interpreting and reporting linkage results. Nat Genet 11:241–247
13. Lange K, Cantor R, Perola M, Sabatti C, Sinsheimer J, Sobel E (2001) MENDEL version 4.0: a complete package for the exact genetic analysis of discrete traits in pedigrees and population data sets. Am J Hum Genet 69(Suppl): A1886
14. Hu H, Roach JC, Coon H, Guthery SL, Voelkerding KV, Margraf RL, Durtschi JD, Tavtigian SV, Shankaracharya, Wu W, Scheet P, Wang S, Xing J, Glusman G, Hubley R, Li H, Garg V, Moore B, Hood L, Galas DJ, Srivastava D, Reese MG, Jorde LB, Yandell M, Huff CD (2014) A unified test of linkage analysis and rare-variant association for analysis of pedigree sequence data. Nat Biotechnol 32:663–669
15. Galesloot TE, van Steen K, Kiemeney LALM, Janss LL, Vermeulen SH (2014) A comparison of multivariate genome-wide association methods. PLoS One 9:e95923
16. Yi H, Breheny P, Imam N, Liu Y, Hoeschele I (2015) Penalized multimarker vs. single-marker regression methods for genome-wide association studies of quantitative traits. Genetics 199:205–222
17. Aulchenko YS, Ripke S, Isaacs A, van Duijn CM (2007) GenABEL: an R library for genome-wide association analysis. Bioinformatics 23:1294–1296
18. Hadfield JD (2010) MCMC methods for multi-response generalized linear mixed models: the MCMCglmm R package. J Stat Softw 33:1–22
19. Abecasis GR, Cherny SS, Cookson WO, Cardon LR (2002) Merlin-rapid analysis of dense genetic maps using sparse gene flow trees. Nat Genet 30:97–101

20. Almasy L, Blangero J (1998) Multipoint quantitative-trait linkage analysis in general pedigrees. Am J Hum Genet 62:1198–1211
21. Zhou X, Stephens M (2014) Efficient multivariate linear mixed model algorithms for genome-wide association studies. Nat Methods 11:407–409
22. Cheng R, Parker CC, Abney M, Palmer AA (2013) Practical considerations regarding the use of genotype and pedigree data to model relatedness in the context of genome-wide association studies. G3 (Bethesda) 3:1861–1867
23. Hiersche M, Rühle F, Stoll M (2013) Postgwas: advanced GWAS interpretation in R. PLoS One 8:e71775
24. Broman KW, Wu H, Sen S, Churchill GA (2003) R/qtl: QTL mapping in experimental crosses. Bioinformatics 19:889–890
25. Feng S, Liu D, Zhan X, Wing MK, Abecasis GR (2014) RAREMETAL: fast and powerful meta-analysis for rare variants. Bioinformatics 30:2828–2829
26. R Core Team (2013) R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria, URL http://www.R-project.org/
27. Eu-Ahsunthornwattana J, Miller EN, Fakiola M, WTCCC, Jeronimo SMB, Blackwell JM, Cordell HJ (2014) Comparison of methods to account for relatedness in genome-wide association studies with family-based data. PLoS Genet 10:e1004445
28. Crossett A, Lee AB, Klei L, Devlin B, Roeder K (2013) Refining genetically inferred relationships using treelet covariance smoothing. Ann Appl Stat 7:669–690
29. Tucker G, Loh P-R, McLeod IM, Hayes BJ, Goddard ME, Berger B, Price AL (2015) Two variance component model improves genetic prediction in family data sets. Am J Hum Genet 97(5):677–690

# Chapter 12

# High-Dimensional Profiling for Computational Diagnosis

**Claudio Lottaz, Wolfram Gronwald, Rainer Spang, and Julia C. Engelmann**

## Abstract

New technologies allow for high-dimensional profiling of patients. For instance, genome-wide gene expression analysis in tumors or in blood is feasible with microarrays, if all transcripts are known, or even without this restriction using high-throughput RNA sequencing. Other technologies like NMR finger printing allow for high-dimensional profiling of metabolites in blood or urine. Such technologies for high-dimensional patient profiling represent novel possibilities for molecular diagnostics. In clinical profiling studies, researchers aim to predict disease type, survival, or treatment response for new patients using high-dimensional profiles. In this process, they encounter a series of obstacles and pitfalls. We review fundamental issues from machine learning and recommend a procedure for the computational aspects of a clinical profiling study.

**Key words** Microarrays, Gene expression profiles, RNA sequencing, Metabolite analysis, NMR finger printing, Statistical classification, Supervised machine learning, Feature selection, Model assessment

## 1 Introduction

In clinical microarray studies, tissue samples from patients are examined using microarrays measuring gene expression levels of as many as 50,000 transcripts. Such high-dimensional data, possibly complemented by additional information about patients, provide novel opportunities for molecular diagnostics through automatic classification.

For instance, Roepman et al. [1] describe a study on head and neck squamous cell carcinomas. In this disease, treatment strongly depends on the presence of metastases in lymph nodes near the neck. However, diagnosis of metastases is difficult. With standard diagnosis, more than half of the patients undergo surgery unnecessarily, while 23 % remain undertreated. Roepman et al. show that treatment based on microarray prediction can be significantly more accurate: in a validation cohort, undertreatment would have been completely avoided while the rate of unnecessary surgery would have dropped from 50 % to 14 %.

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_12, 

From a statistical point of view, the major characteristic of microarray studies is that the number of genes is orders of magnitude larger than the number of patients. For classification this leads to problems involving overfitting and saturated models. When blindly applying classification algorithms, a model rather adapts to noise in the data than to the molecular characteristics of a disease. Thus, the challenge is to find molecular classification signatures that are valid for entire disease populations.

In the following, we briefly describe the machine learning theory, as it is needed for computational diagnostics using high-dimensional data. We suggest software solutions in Subheading 2 and a procedure for a clinical profiling study in Subheading 3. In the last section, we mention alternative data sources for high-dimensional data, point out some alternative classification and evaluation strategies, and describe pitfalls in the analysis and interpretation of high-dimensional clinical studies.

### 1.1 The Classification Problem

Classification is a well-investigated problem in machine learning. This chapter gives a brief overview of the most fundamental issues baring microarray gene expression analysis in mind as a prominent example for the measurement of high-dimensional data. We refer to [2–6] for further reading on the more theoretical concepts of machine learning and to [7, 8] for an in-depth description of their application to microarray data.

The task in classification is to determine classification rules which enable discrimination between two or more classes of objects based on a set of features. Supervised learning methods construct classification rules based on training data with known classes. They deduce rules by optimizing a classification quality criterion, for example, by minimizing the number of misclassifications. In microarray-based computational diagnostics, features are gene expression levels, objects are tissue samples from patients, and object classes are phenotypic characteristics of patients. Phenotypes can include previously defined disease entities, as in the Microarray Innovations in LEukemia study [9] and lymphoma-related studies [10, 11]; risk groups, as in the breast cancer studies of van't Veer et al. [12]; treatment response, as in the leukemia study by Cheok et al. [13]; or disease outcome, as in the breast cancer study of West et al. [14]. In this context, classification rules are called *diagnostic signatures*.

Study cases are always samples from a larger disease population. Such a population comprises all patients who had a certain disease, have it now, or will have it in the future. We aim for a diagnostic signature with good performance not only on the patients in the study but also in future clinical practice. That is, we aim for a classification rule that generalizes beyond the training set. Different learning algorithms determine signatures of different complexity. We illustrate signature complexity using a toy example in which

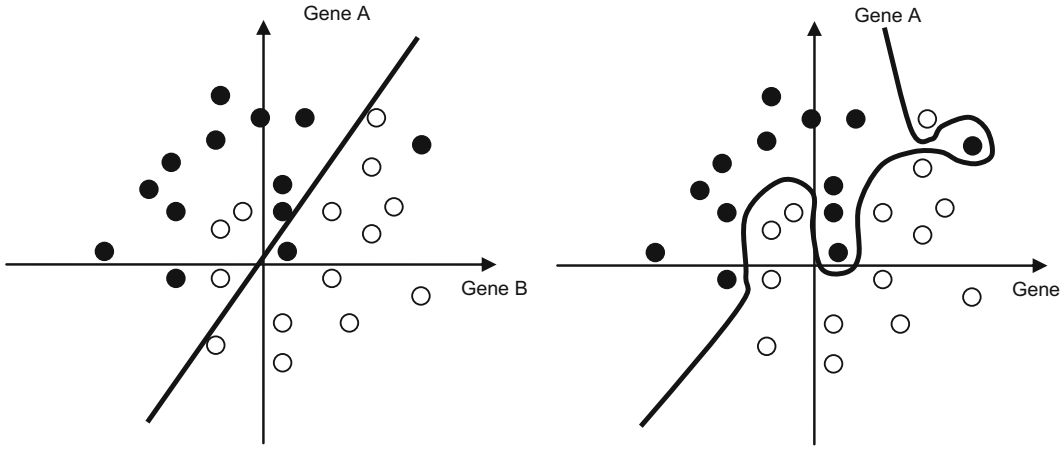



**Fig. 1** Overfitting. The linear boundary (*left-hand side*) reflects characteristics of the disease population, including an overlap of the two classes. The complex boundary (*right-hand side*) does not show any misclassifications but adapts to noise. It is not expected to perform well on future patient data

diagnosis of treatment response is based on the expression levels of only two genes (Fig. 1). A linear signature corresponds to a straight line, which separates the space defined by the two genes into two parts, holding good and bad responders, respectively. When expression levels of many genes are measured, linear signatures correspond to hyperplanes separating a high-dimensional space into two parts. Other learning algorithms determine more complex boundaries. In microarray classification, however, the improvement achieved by sophisticated learning algorithms is controversial [15]. Complex models are more flexible to adapt to the data. However, they also adapt to noise more easily and may thus miss the characteristic features of the disease population. Consider Fig. 1, where black data points represent bad responders and white data points represent good responders. A linear signature is shown in the left panel of Fig. 1, while a complex boundary is drawn in the right panel. The linear signature reflects the general tendency of the data but is not able to classify perfectly. On the other hand, the complex boundary never misclassifies a sample, but it does not appear to be well supported by the data. When applying both signatures to new patients, it is not clear whether the complex boundary will outperform the linear signature. In fact, experience shows that complex signatures often do not generalize well to new data, and hence break down in clinical practice. This phenomenon is called *overfitting*.

### *1.2 The Curse of Dimensionality*

In microarray studies, the number of genes is always orders of magnitude larger than the number of patients. In this situation, overfitting also occurs with linear signatures. To illustrate why this is a problem, we use another simplistic toy example: two genes are

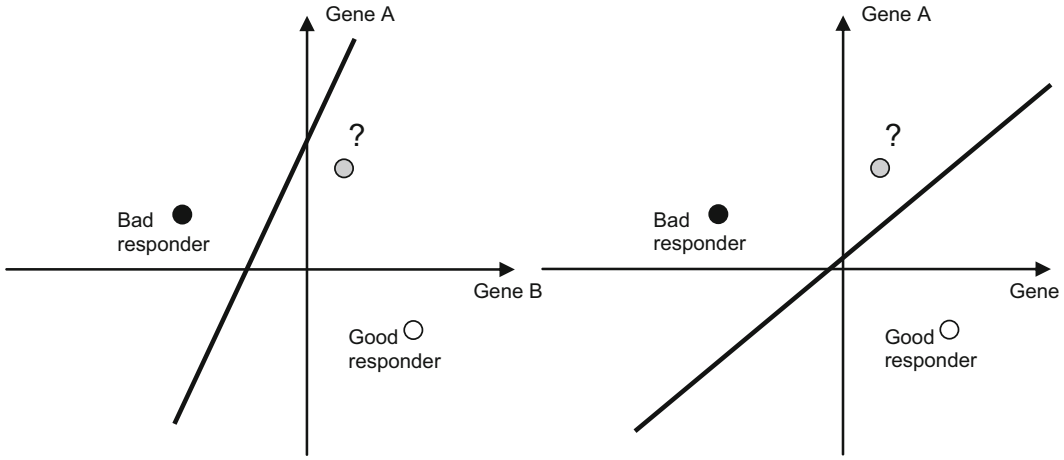



**Fig. 2** Ill-posed classification problem. Even linear signatures are underdetermined when the number of patients does not exceed the number of genes. The *black* and *white data points* represent the training data. The linear signatures in both panels are equally valid but classify the novel data point (*gray*) differently

measured in only two patients. This is the simplest scenario where the number of patients in a study does not exceed the number of genes. We want to construct a linear signature that can discriminate between the two classes, say good and bad responders, each represented by one patient. This is the same problem as finding a straight line separating two points in a plane. Clearly, there is no unique solution (*see* Fig. 2). Next, think about a third point, which does not lie on the line going through the first two points. Imagine it represents a new patient with unknown diagnosis. The dilemma is that it is always possible to linearly separate the first two points such that the new one lies on the same side as either one of them. No matter where the third point lies, there is always one signature with zero training error, which classifies the new patient as a good responder, and a second signature, equally well supported by the training data, which classifies the new patient as a bad responder. The two training patients do not contain sufficient information to diagnose the new patient. We are in this situation whenever there are at least as many genes as training patients. Due to the large number of genes on microarrays this problem is inherent in gene expression studies.

The way out of the dilemma is *regularization*. Generally speaking, regularization means imposing additional criteria in the signature building process. A prominent method of regularization is *gene selection*, which restricts the number of genes contributing to the signature. This can be biologically motivated, since not all genes carry information on disease states. In many cases, gene selection improves the predictive performance of a signature [16]. Furthermore, adequate selection of genes opens the opportunity to design smaller and hence cheaper diagnostic microarrays or marker panels [17].

Often genes are selected independently of the classification algorithm according to univariate selection criteria. This is called *gene filtering* [18]. In the context of classification, selection criteria reflecting the correlation with the class labels are generally used. Hence one favors genes with low expression levels in one class and high expression levels in the other. Popular choices are variants of the *t*-statistic or the nonparametric *Wilcoxon rank sum statistic*.

### 1.3 Calibrating Model Complexity

As illustrated in the previous sections, regularization is essential in the process of determining a good diagnostic signature. Aggressive regularization, however, can be as harmful as too little of it. Hence regularization needs to be calibrated. One way to do so is to vary the number of genes included in the signature: weak regularization means that most genes are kept, whereas strong regularization removes most genes.

With little regularization, classification algorithms fit very flexible decision boundaries to the data. This results in few misclassifications on the training data. Nevertheless, due to overfitting, this approach can have poor predictive performance in clinical practice. With too much regularization, the resulting signatures are too restricted. They have poor performance on both the study patients and in future clinical practice. We refer to this situation as *underfitting*. The problem is schematically illustrated in Fig. 3, where two error rates are compared. First there is the training error, which is the number of misclassifications observed on data from which the signature was learned. In addition, there is a test error, which is observed on an independent test set of additional patients randomly drawn from the disease population. Learning algorithms minimize the training error, but the test error measures whether a signature generalizes well.

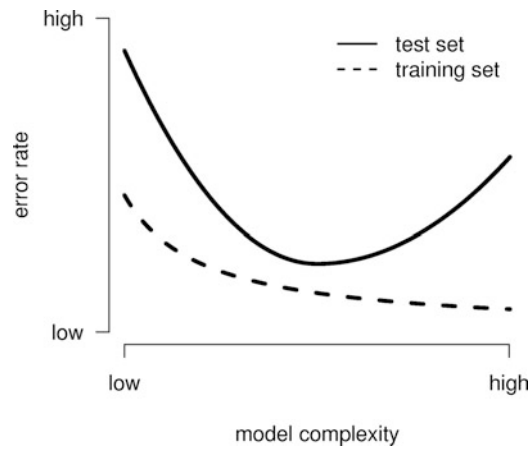



**Fig. 3** Overfitting–underfitting trade-off. The *x*-axis codes for model complexity and the *y*-axis for error rates. The *dashed line* displays the training error, the *solid line* the test error. Low complexity models (strong regularization) produce high test errors (underfitting) and so do highly complex models (weak regularization, overfitting)

In order to learn signatures that generalize well, we adapt model complexity so that test errors are minimized. To this end, we have to estimate test errors on an independent set of patients, the calibration set, which is not used in signature learning. To calibrate regularization, we learn signatures of varying complexity, evaluate them on the calibration set, and pick the signature that performs best.

### 1.4 Evaluation of Diagnostic Signatures

Validation of a diagnostic signature is important, because the errors on a training set do not reflect the expected error in clinical practice. In fact, the validation step is most critical in computational diagnosis studies and several pitfalls are involved. Estimators can be overly optimistic (biased) or they can have high sample variances. It also makes a difference whether we are interested in the performance of a fixed signature (which is usually the case in clinical studies), or whether we are interested in the power of the learning algorithm that builds the signatures (which is usually the case in methodological projects). The performance of a fixed signature varies due to the random sampling of the test set, while the performance of a learning algorithm varies due to sampling of both training and test set.

In computational diagnostics, we are usually more interested in the evaluation of a fixed diagnostic signature. The corresponding theoretical concept is the *conditional error rate*, also called *true error*. It is defined as the probability of misclassifying new patients given the training data used in the study. The true error is not obtainable in practice, since its computation involves the unknown population distribution. Estimates of the true error rate, however, can be obtained by evaluating the diagnostic signature on independent test samples.

## 2 Software

Implementations and corresponding documentation for most computational methods we describe here can be found in packages contributed to the statistical computing environment R [19, 20] and are available from http://cran.R-project.org. In addition, the Bioconductor project [21] containing many R-packages related to the life sciences is found at the link http://www.bioconductor.org.

### 2.1 Classification Software

LDA, QDA, and many other classification methods are implemented in the R-package *MASS* from the *VR* bundle; DLDA is contained in the R-package *supclust*. An implementation of support vector machines (SVMs) is part of the package *e1071*; *svmpath* can compute the entire regularization path for SVMs. The nearest centroid method is implemented in the Bioconductor package *pamr*. The package *MCRestimate* from the Bioconductor project implements many helpful functions for nested cross-validation.

### 2.2 Additional Software Related to RNA Microarrays

The Bioconductor package *oligo* or *aroma.affymetrix* can be used to generate logarithmic data on an additive scale from microarray measurements. These packages focus on measurements from Affymetrix microarrays and can also be used to bring patient profiles to a common scale. A normalization alternative is provided in the Bioconductor package *vsn*.

### 2.3 Additional Software Related to RNA Sequencing

*TopHat2* is one of many suggested tools independent of R to map RNA sequence reads to genomes or transcriptomes for generating count data. Bioconductor packages *edgeR* and *DESeq2* can be used to derive normalized expression values from RNA-seq count data. The R package *PoiClaClu* implements sparse Poisson Linear Discriminant Analysis (sPLDA) and is available from CRAN.

### 2.4 Additional Software Related to NMR Metabolite Measurements

For the analysis of NMR spectra and binning of these the software package *AMIX* (Bruker Biospin GmbH, Rheinstetten, Germany) may be used. Normalization methods developed for microarrays can also normalize data from NMR bins. Suitable choices are variance stabilization and normalization (from the package *vsn*) or quantile normalization (from the package *limma*). In metabolite classification, random forests and support vector machines as implemented in the R-packages *randomForest* and *e1071*, respectively, performed particularly well.

## 3 Methods

Here we describe our choice of methods for the development of a diagnostic signature when data is generated using microarrays. The suggested methods, however, can also be applied and used likewise on other high-dimensional data (*see* **Note 1**). In addition to normalized expression profiles, patients have an attributed class label reflecting a clinical phenotype. The challenge is to learn diagnostic signatures on gene expression data that enable prediction of the correct clinical phenotype for new patients.

### 3.1 Notation

We measure $p$ genes on $n$ patients. The data from the microarray corresponding to patient $i$ is represented by the *expression profile* $x^{(i)} \in \mathfrak{R}^p$. We denote the *label* indicating the phenotype of patient $i$ by $y_i \in \{-1, +1\}$. In this setting, the class labels are binary and we restrict our discussion to this case. However, the discussed methods can be extended to multiclass problems dealing with more than two clinical phenotypes. The profiles are arranged as rows in a *gene expression matrix* $X \in \mathfrak{R}^{n \times p}$. All labels together form the vector $y = (y_i)_{i=1,\ldots,n}$. The pair $(X, y)$ is called a dataset $D$. It holds all data of a study in pairs of observations $\{(x^{(i)}, y_i)\}$, $i = 1, \ldots, n$.

The computational task is to generate a mathematical model $f$ relating $x$ to $y$.

Although we never have access to the complete disease population, it is convenient to make it part of the mathematical formalism. We assume that there is a data-generating distribution $P(X, Y)$ on $\Re^p \times \{-1, +1\}$. $P(X, Y)$ is the joint distribution of expression profiles and associated clinical phenotypes. The patients who enrolled for the study, as well as new patients who need to be diagnosed in clinical practice, are modeled as independent samples $\{(x^{(i)}, y_i)\}$ drawn from $P$. We denote a diagnostic signature, or *classification rule,* by $f : \Re^p \to \{-1, +1\}$.

### 3.2 Diagonal Linear Discriminant Analysis (DLDA)

Various learning algorithms have been suggested for microarray analysis. Many of them implement rather sophisticated approaches to model the training data. However, Wessels et al. [15] report that in a comparison of the most popular algorithms and gene selection methods, simple algorithms perform particularly well. They have assessed the performance of learning algorithms using six datasets from clinical microarray studies. *Diagonal linear discriminant analysis* (DLDA) combined with univariate gene selection achieved very good results. This finding is in accordance with other authors (e.g., [16]). Thus, we recommend and describe this combination of methods.

*Diagonal linear discriminant analysis* (DLDA) is based on a comparison of multivariate Gaussian likelihoods for two classes. The conditional density $P(x|y = c)$ of the data given membership to class $c \in \{-1, +1\}$ is modeled as a multivariate normal distribution $N(\mu^c, \Sigma_c)$ with class mean $\mu^c$ and covariance matrix $\Sigma_c$. The two means and covariance matrices are estimated from the training data. A new point is classified to the class with higher likelihood. Restrictions on the form of the covariance matrix control model complexity: *Quadratic discriminant analysis* (QDA) allows different covariance matrices in both classes; *linear discriminant analysis* (LDA) assumes that they are the same, and diagonal linear discriminant analysis (DLDA) additionally restricts the covariance matrix to diagonal form. The parameters needed by DLDA are therefore class-wise mean expression values for each gene and one pooled variance per gene.

To derive the decision rule applied in DLDA, consider the multivariate Gaussian log-likelihood $N(\mu^c, \Sigma)$ for class $c$ with diagonal covariance matrix $\Sigma$. The vector $\mu^c$ contains the mean gene expression values $\mu_i^c$ for class $c$. It is also called the *class centroid*. The covariance matrix $\Sigma$ contains pooled gene-wise variances $\sigma_i^2$. The log-likelihood $L_c(x)$ can then be written in the form:

$$L_c(x) = -\tfrac{1}{2} \cdot \sum_{i=1}^{p} \log\left(2\pi\sigma_i^2\right) - \tfrac{1}{2} \cdot \sum_{i=1}^{p} \frac{\left[x_i - \mu_i^c\right]^2}{\sigma_i^2}.$$

The first term of $L_c(x)$ does not depend on the class and is therefore neglected in DLDA. DLDA places patients into the class for which the absolute value of the second term is minimized.

### 3.3 Univariate Gene Selection

Diagonal linear discriminant analysis can be directly applied to microarray data. Nevertheless, *gene selection* considerably improves its performance. In gene selection, we impose additional regularization by limiting the number of genes in the signature. A simple way to select informative genes is to rank them according to a univariate criterion measuring the difference in mean expression values between the two classes. We suggest a regularized version of the *t-statistic* also used for detecting differential gene expression. For gene $i$, it is defined as

$$d_i = \frac{\mu_i^{-1} - \mu_i^{+1}}{\sigma_i + \sigma_o},$$

where $\sigma_o$ denotes the statistic's regularization parameter, the so-called *fudge factor*. It is typically set to the median of all $\sigma_i$ and ensures that the statistic does not grow exceedingly when low variances are underestimated. Only top-ranking genes in the resulting list are chosen for classification.

### 3.4 Generation of Diagnostic Signatures

In this section, we suggest a simple procedure to generate and validate a diagnostic signature within a clinical microarray study. It is illustrated in Fig. 4.

#### 3.4.1 Preprocess Your Data

First, normalize your microarray data in order to make the expression values comparable.Various methods for microarray data preprocessing and normalization have been suggested and are equally valid in computational diagnostics. From now on, we assume that

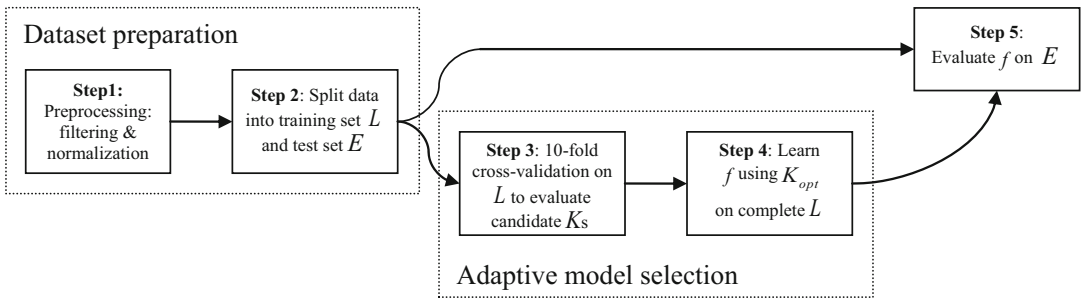



**Fig. 4** Overview of the suggested procedure to generate a diagnostic signature. *K* represents the regularization level (e.g., number of genes selected), $K_{opt}$ denotes the best regularization level found in cross-validation. *L* represents the training set and *E* the test set

gene expression profiles $x^{(i)}$ are normalized and on an additive scale (log transformed).

### 3.4.2 Divide Your Data into a Training and a Test Set

In order to allow an unbiased evaluation of the generated diagnostic signature, we suggest separating the study data right from the start into two parts: the training set $L$ for learning and the test set $E$ for evaluation. The entire learning process must be executed on the training data only. The evaluation of the resulting diagnostic signature must be performed afterward using only the test set.

The training set's exclusive purpose is to learn a diagnostic signature. Unclear or controversial cases can and should be excluded from learning. You may consider focusing on extreme cases. For instance, Liu et al. improve their outcome prediction by focusing the learning phase on very short and very long-term survivors [22].

The test set's exclusive purpose is to evaluate the diagnostic signature. Since you want to estimate the performance of a signature in clinical practice, the test set should reflect expected population properties of the investigated disease. For example, if the disease is twice as common in women as in men, the gender ratio should be close to 2:1 in the test set too.

The size of the training and test sets has an impact on both the performance of the signature and the accuracy of its evaluation. Large training sets lead to a better performance of the signature, while large test sets lead to more accurate estimates of the performance. Actually, small test sets result in unreliable error estimation due to sampling variance (*see* **Note 2**). We recommend splitting the data in a ratio of 2:1.

### 3.4.3 Find the Best Regularization Level

Search for the best regularization level exclusively using the training data. Apply univariate gene filtering and DLDA to learn signatures with varying complexity and estimate their performance on independent data. Your best choice for the regularization level is the one leading to the best performance. *See* **Note 3** for alternative learning algorithms and feature selection schemes.

To estimate performance on independent data we recommend tenfold cross-validation. Partition the training set into 10 bins of equal size. Take care to generate bins that are balanced with respect to the classes to be discriminated. That is, classes should have the same frequency in the bins as in the complete training data. Use each bin in turn as the calibration set, and pool the other bins to generate the learning set. In each iteration, select genes according to the regularized t-statistics $d_i$ and the regularization level $K$ to be evaluated, learn a signature by applying DLDA on the restricted learning set, and compute the number of misclassifications in the calibration bin. The cross-validation error is the sum of these errors. Use this estimate for performance on independent data to

1. Choose regularization level $K$.
2. Separate the training set $L$ into 10 bins $L_1,...,L_{10}$
3. For $i$ in 1 to 10 do
    a. Select features $S$ according to $K$ from $D[L - L_i]$
    b. Learn $f_i(K)$ on $D[L - L_i, S]$
    c. Evaluate $f_i(K)$ on $D[L_i, S]$
4. Average error rates determined in step 3.c

**Fig. 5** Algorithmic representation of the tenfold cross-validation procedure. *K* is the regularization level to be evaluated. *D*[*L*] denotes the study data restricted to patients in the set *L*. *D*[*L*, *S*] represents the study data restricted to the patient set *L* and the gene set *S*. The operator "−" is used for set difference

determine the optimal amount of regularization [23, 24]. *See* Fig. 5 for an algorithmic representation of the cross-validation procedure.

A straightforward method for finding a good set of candidate values for $K$ is *forward filtering*. Starting with the most discriminating gene, include additional genes one by one until the cross-validation error reaches an optimum. Stop iterating and set the optimal regularization level $K_{\text{opt}}$ to the value of $K$ that produced the smallest cross-validation error.

*3.4.4 Learn the Diagnostic Signature*

For the optimal level of regularization $K_{\text{opt}}$, compute the diagnostic signature $f$ on the complete training set $L$. This is the final signature.

*3.4.5 Evaluate Your Signature*

Apply your final signature $f$ to the test set $E$ to estimate the misclassification rate. Note that the result is subject to sampling variance (*see* **Note 4** for more information on sampling variance).

*3.4.6 Document Your Signature*

Diagnostic signatures eventually need to be communicated to other healthcare centers. It should be possible for your signature to be used to diagnose patients worldwide, at least if the same measurement technology and platform is used to profile them. You should therefore provide a detailed description of the signature that you propose. The list of genes contributing to the signature is not enough. The mathematical form of both the classification and the preprocessing models needs to be specified together with the values of all parameters. For DLDA, communication of the centroid for each group and the variance for each gene is necessary to specify the signature completely. While exact documentation of the

classification signature is crucial, involvement of genes in a signature should not be interpreted biologically (*see* **Note 5**).

## 4 Notes

1. *Alternative High-Dimensional Data for Classification*: So far, we have focused on classification based on microarray data. Various new technologies have been developed to generate high-dimensional multivariate data for profiling in the life sciences. Data generated from these new technologies can often be used for classification in a similar manner as just described for microarray data.

   *Custom microarrays*: General purpose genome-wide microarrays are *too complex* and expensive for routine diagnosis in hospitals. Therefore, clinicians are interested in cost and time effective, but officially approved tools to reliably diagnose patients using high-dimensional gene expression measurements. For instance, the custom *AmpliChip* Leukemia microarray (by Roche Molecular Systems) was specifically designed for the classification of various types of leukemia [25]. The chip contains 1480 distinct probe sets with 1457 of them used for generating normalized signal intensities of disease-related genes; the remaining probe sets interrogate control sequences and housekeeping genes. The genes on the *AmpliChip* are selected based on the MILE study on 2096 leukemia patients using the large genome-wide standard *Affymetrix* microarray *HG-U133 plus 2.0* [9]. The *AmpliChip* is intended to identify the correct type of leukemia for new patients. Thereby, it is limited to a preselected set of leukemias. Because the *AmpliChip* is based on the *Affymetrix* microarray technology, a classifier could be generated and applied for classification as discussed earlier. At the same time, the AmpliChip comes with the advantages of the *Affymetrix* microarray technology such as good reproducibility, as well as drawbacks including complex handling in the laboratory.

   *Digital multiplexed gene expression*: One platform for digital gene expression measurement by capturing and counting individual mRNA transcripts is the *NanoString nCounter* gene expression system. The developers report that the *nCounter* system is limited to the measurement of about 500 human genes, has a detection limit between 0.1 fM and 0.5 fM, and a linear dynamic range of over 500-fold [26]. Hence, this platform is limited to a subset of genes but can measure very small amounts of material without amplification. Therefore, the *nCounter* technology is more robust to partially degraded RNA, which is expected in formalin-fixed paraffin-embedded (FFPE) material.

In the vast majority of hospitals, patient material is only used and stored as FFPE material. Researchers have confirmed microarray-based signatures for classification of diffuse large b-cell lymphomas with the *nCounter* technology in FFPE material [27, 28]. In both projects, researchers have measured less than 100 genes and therefore needed particular protocols for normalization. Masqué-Soler et al. [27] performed batch-to-batch correction with sample-wise normalization factors generated by individually averaging the internal controls. Samples were then normalized independently with quantile normalization. Scott et al. [28] normalized each array by dividing the observed expression values by the geometric mean of five housekeeping genes. In addition, a reference array of synthetic oligonucleotides of fixed concentration ratios was included in each run, to adjust for potential batch effects between runs. In both cases, the expression values were finally log2-transformed. Masqué-Soler et al. as well as Scott et al. emphasize the simple and efficient protocol for the gene expression measurement in only 24–36 hours.

*Classification on High-Throughput RNA Sequencing Data*: High-throughput RNA sequencing is on its way to replace microarray transcriptome analyses for many applications. However, fundamental differences between the data delivered from microarrays and data from RNA sequencing experiments must be respected during data analysis. For high-throughput RNA sequencing, first a cDNA library is prepared from RNA input material, which is then PCR amplified and sheared into fragments of similar size. These fragments are then sequenced from one or both sides, giving rise to millions of single- or paired end sequence reads. Strand-specific protocols can recover whether the transcript originated from the plus or minus DNA strand, while strand-unspecific protocols can not. Sequence reads are then aligned to the genome with a splicing-aware mapping tool such as *TopHat2* [29], or de novo assembled if sequencing depth is sufficient. For most gene expression analyses of well-annotated species, the first-align-then-annotate strategy that uses genomic sequence information and respects splicing events is preferred, and a number of mappers have been proposed for this task. From the mapped sequence reads, so-called count tables that summarize the number of observed reads for all features of interest (e.g., exons or genes) can be generated. While expression values from microarrays are on a continuous scale, reflecting fluorescence intensities with a distribution that is close to log-normal, count data is Poisson or negative-binomial distributed. To be able to apply classification algorithms to sequence data, one must either (1) adapt the data to use models that assume continuous and log-normal

high-throughput data, or (2) adapt the models such that they assume Poisson or negative binomial distributed data. In the following paragraphs, we discuss these two approaches.

*Deriving expression values from sequencing data*: When high-throughput sequencing technology first emerged, this was considered the end of the microarray era. It was hoped that RNA sequencing would deliver 'digital' gene expression estimates in an unbiased and transcriptome-wide fashion that is independent of annotation [30]. But from many experiments conducted so far, it became evident that the sequence library preparation as well as the fact that short sequence reads (formerly 25 nucleotides, now up to 150 nucleotides in single or paired end mode, only for some technologies several hundred or even a few thousand nucleotides, albeit with lower numbers of total reads) are sequenced, can introduce substantial biases in observed read counts [31]. A substantial fraction of the short sequence reads are often not uniquely mappable to a genomic region due to sequence similarities caused by, e.g., gene families. But even for the uniquely mapped reads it might not be obvious from which transcript they originated, when the locus gives rise to more than one transcript. Therefore, estimating transcript and gene expression levels from RNA sequencing data is not trivial. Simplistic approaches count only uniquely mapped reads falling in exonic regions, disregarding the rest. These counts can then be used to derive differentially expressed genes and exons but hold no information on the transcript level. Resolution on the transcript level requires allocation of reads with multiple mapping sites—both within the same and across genes—to specific isoforms. Many algorithms have been proposed, the majority assign multimapping reads to isoforms in an iterative process, which has first been described in Xing et al [32]. These models allow inference about relative transcript abundance.

Numerous gene expression normalization methods for RNA-sequencing data have been proposed, the most popular (but also criticized) being RPKM (Reads per Kilobase of transcript per Million mapped reads) [33] and FPKM (Fragments Per Kilobase of transcript per Million mapped reads) [34], the later counting fragments from which the reads originated, to avoid double counting when having paired-end sequencing data. Modifications such as TPM (Transcripts Per Million) [35] claim to better represent relative molar RNA concentration in the sample, the initial aim of the RPKM value. Software packages written for finding differentially expressed genes from RNA sequencing data usually work directly with the count data and often use Negative Binomial models and some form of dispersion shrinkage, as in *DESeq2* [36] and *edgeR* [37].

However, both *DESeq2* and *edgeR* offer functionality to retrieve expression values. In *DESeq2*, one can apply a ridge regularization to transform the count data to log2 scale, reflecting a normalized expression value. This rlog transformation uses library size factors and minimizes differences between samples for rows with small counts. In *edgeR*, fitted values from the log-link negative binomial or Poisson generalized linear model (GLM) that are used for differential expression analysis can be extracted as moderated log counts per million (CPM). Once gene or transcript expression levels are at hand, the classification algorithms and procedures described earlier in this chapter can be applied.

*Classification with count data*: An alternative approach to first deriving expression estimates from RNA sequencing data is to directly work on the count data but adapt the classification procedure accordingly. For the identification of differentially expressed genes, methods have adapted models for count data, but classification has been lagging behind so far, with currently only one proposed method for RNA sequencing data, namely, the Poisson linear discriminant analysis (PLDA) classifier [38]. For classification using PLDA, the authors propose to normalize the count data with an estimate of the size factor, which represents the sequencing library size, followed by a power transformation to account for overdispersion relative to the Poisson model, meaning that the variance is larger than the mean. Instead of assuming normally distributed data as in standard linear discriminant analysis (LDA), Poisson linear discriminant analysis assumes independent features from which the data follow a Poisson distribution. The classification rule that assigns test samples to a specific class is then linear. Since it is not desirable to use all features of a high-dimensional dataset for classification, sparse PLDA (sPLDA) uses shrinkage similar to Nearest Shrunken Centroids (NSC) classifiers that are also used by PAM for classification of microarray data (*see* **Note 3**). sPLDA has been shown to perform well on slightly overdispersed data relative to the Poisson model, but performance decreases on severely overdispersed data. The authors of sPLDA propose to investigate whether negative-binomial models might increase performance for later case datasets.

*Metabolomics*: The aim of metabolomics is primarily the comprehensive analysis of the flow of small organic compounds through bioenergetic and biosynthetic pathways by their quantitative analysis in cells, tissues, organs, biological fluids, and whole organisms. Typical compounds include amino acids, sugars, organic acids, bases, lipids, vitamins, and various conjugates of substances of exogenous origin. Fields of application include such diverse topics as investigating the health status of

dairy cows [39] or analysis of chronic kidney diseases [40]. Metabolomic investigations are mainly conducted by employing hyphenated mass spectrometry or nuclear magnetic resonance (NMR) spectroscopy. Here, we will focus on the application of solution NMR spectroscopy. NMR is a versatile and powerful method for metabolite identification and quantification, as it allows the simultaneous detection of all proton-containing metabolites present at least micromolar concentrations in a given sample. NMR signal volumes scale linearly with concentration enabling accurate quantification of analytes. Furthermore, NMR requires very little sample pretreatment and, typically, no prior chemical derivatization of molecules. On the other hand, a disadvantage of NMR spectroscopy when compared to mass spectrometry, for example, is its relatively poor sensitivity.

*NMR spectroscopy*: The theory of NMR spectroscopy is well established. For a comprehensive description, see for example Ernst et al. [41]. For observing an NMR signal nuclei such as protons possessing a spin unequal to zero are brought into a static magnetic field, where only certain orientations, i.e., certain states of the magnetic moments of the nuclei are allowed. By applying an additional electromagnetic field of appropriate frequency, transitions between the different states can be induced. This will lead to a time-dependent change of the macroscopic magnetization of the nuclei which will be recorded in the time domain as the NMR signal. By application of Fourier transform to the NMR signal the final NMR spectrum will be obtained. The positions of the signals in an NMR spectrum depend on the electronic environment of the nuclei. As the electronic environment depends among other factors on the type of molecule and on the position of a nucleus within a molecule, in the final NMR spectrum each signal corresponds to a certain nucleus or group of nuclei of a certain molecule. By comparison with reference spectra obtained from pure compounds, identification of metabolites becomes feasible. Signal intensities scale linearly with metabolite abundance and can thus be quantified relative to a given reference compound. An example of a typical NMR spectrum of human urine, in which the signals of up to several hundred different compounds may be detected, is shown in Fig. 6.

*Preprocessing*: Acquisition of NMR spectra is generally followed by multivariate statistical data analysis. Here, the first step involves correction for variation in signal position across spectra due to differences in pH, salt concentration, and/or sample temperature. A widely used and robust method to compensate for these effects is spectral binning, where an NMR spectrum is split into a number of segments or bins. Equally sized bins are

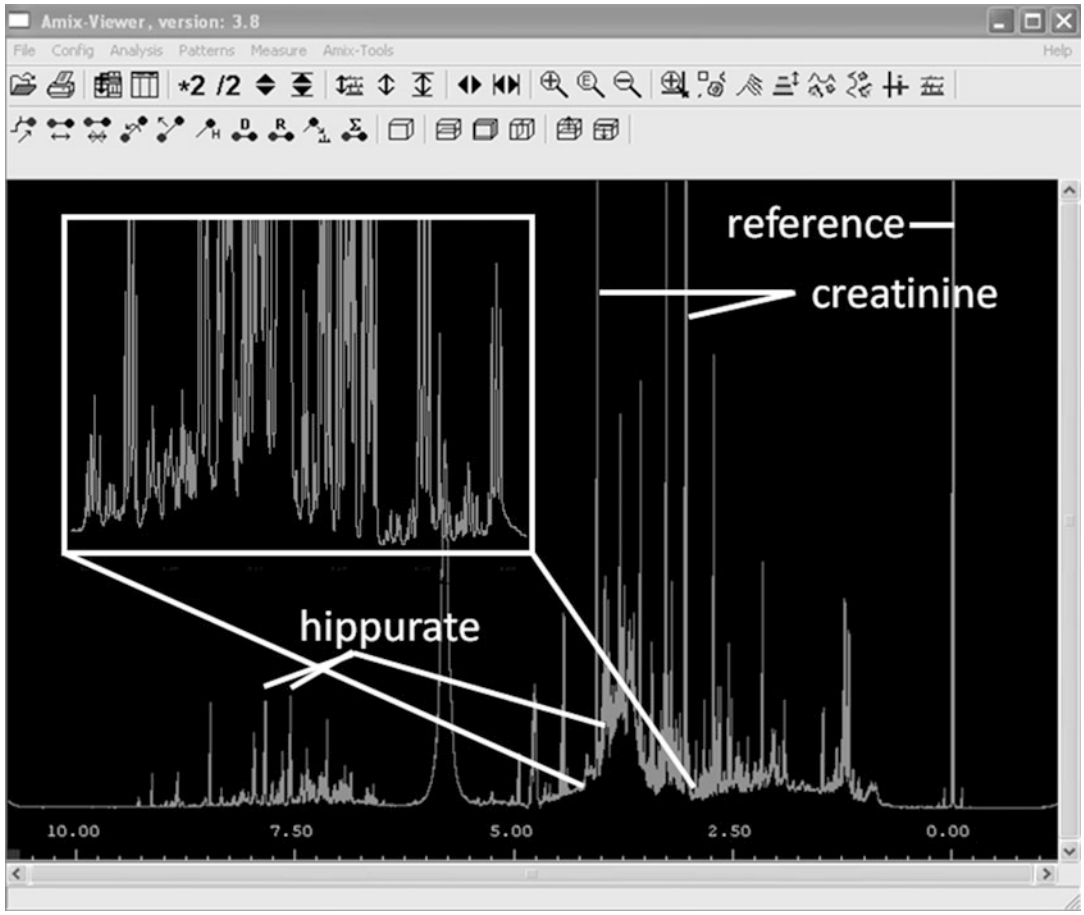



**Fig. 6** 1D $^{1}$H NMR spectrum of human urine. As an example for metabolite identification the signals of hippurate and creatinine are marked. The enlarged region demonstrates the high complexity of the spectrum

mostly used, albeit other schemes such as adaptive binning have been proposed. Data points inside every bin are integrated so that the whole spectrum is then represented as a vector of bucket integrals. Alternative approaches include, for example, signal alignment techniques [42].

*Data normalization*: Generally, metabolomic datasets are prone to unwanted experimental and/or biological variances and biases. To minimize these disturbing factors, data normalization and scaling approaches may be used. The different strategies can be grouped into methods that either adjust the variance of metabolites by variance stabilization and variable scaling strategies, or remove unwanted sample to sample variations. A simple but often effective approach for reducing inter-sample variations of urinary data is scaling relative to the signal of creatinine. For other sample matrices scaling of every spectrum to a total sum of one may be used. Under the condition

that only a relatively small proportion of metabolites is regulated in approximately equal shares up and down so that no large systematic differences in total spectral area between biological groups exist, Variance Stabilization and Normalization [43] as well as Quantile Normalization [44] give reliable results according to our experience [45].

*Sample classification*: Classification of an unknown sample to known classes of disease (e.g., healthy or diseased) is a typical application of metabolomics. Generally, classification algorithms are trained on a training dataset where the class label of each sample is known, followed by application of the trained algorithm to additional independent test data. In case that test data are difficult to obtain, performance evaluation is often performed within a cross-validation setting, where the whole dataset is iteratively split into training and test sets. By employing a nested cross-validation scheme, where parameters relevant for the algorithm are optimized within inner loops, it is ensured that results are not biased by training or parameter optimization (*see* **Note 4**). In our experience, especially, Random Forests [46] and Support Vector Machines [47–49] are particularly suited for the analysis of high-dimensional NMR data [50].

*Conclusion:* Many of the aforementioned methods were originally developed for analysis of gene expression micro-array data and they might as well be applied to other high-dimensional data such as metabolomic data generated by means of hyphenated mass spectrometry as well as to proteomic datasets.

2. *Pitfalls in Signature Evaluation*: There are several pitfalls leading to overly optimistic estimations, for instance, using too small or unbalanced validation sets [51]. When using a single independent test set for evaluation of diagnostic signatures, only training data is used for gene selection, classifier learning, and adaptive model selection. The final signature is then evaluated on the independent test set. Unfortunately, this estimator can have a substantial sample variance, due to the random selection of patients in the test set. This is especially the case if the test set is small. Thus, good performance in small studies can be a chance artifact. For instance, Ntzani et al. [52] have reviewed 84 microarray studies carried out before 2003 and observed that positive results were reported strikingly often on very small datasets.

   Another prominent problem is the selection bias caused by improperly combining cross-validation with gene selection [14, 51, 53]. Gene selection has a strong impact on the predictive performance of a signature. It is an essential part of the signature-building algorithm. There are two possible ways to combine gene selection with cross-validation: either apply gene

selection to the complete dataset and then perform cross-validation on the reduced data, or perform gene selection in every single step of cross-validation anew. We call the first alternative *out-of-loop gene selection* and the second *in-loop gene selection*. In-loop gene selection gives the better estimate of generalization performance, while the out-of-loop procedure is overoptimistic and biased toward low error rates. In out-of-loop gene selection, the genes selected for discriminative power on the whole dataset bear information on the samples used for testing. In-loop gene selection avoids this problem. Here, genes are only selected on the training data of each cross-validation iteration and the corresponding test sets are independent.

The impact of overoptimistic evaluation through out-of-loop gene selection can be very prominent. For example, Reid et al. [54] report the reevaluation of two public studies on treatment response in breast cancer. They observe classification errors of 39 % and 46 %, respectively, when applying in-loop gene selection. Using the overoptimistic out-of-loop gene selection, error rates are underestimated at 25 % and 24 %, respectively. Similarly, Simon et al. [51] describe a case in breast cancer outcome prediction where the out-of-loop cross-validation method estimates the error rate to 27 % while the in-loop cross-validation method estimates an error rate of 41 %. Nevertheless, Ntzani et al. [52] report that 26 % of 84 reviewed microarray studies published before 2003 provide overoptimistic error rates due to out-of-loop gene selection. Seven of these studies have been reanalyzed by Michiels et al. [55]. They determine classification rates using nested cross-validation loops and average over many random cross-validation partitionings. In five of the investigated datasets, classification rates no better than random guessing are observed.

3. *Alternative Classification Algorithms*: Several authors have observed that complex classification algorithms quite often do not outperform simple ones like diagonal linear discrimination on clinical microarray data [15, 16, 29, 56]. We report two more algorithms, however, which have shown good performance on microarray data.

   The first one is a variant of DLDA called Prediction Analysis of Microarrays *(PAM*, [57, 58]). An outstanding feature of PAM is *gene shrinkage*. Filtering uses a hard threshold: when selecting $k$ genes, gene $k$+1 is thrown away even if it bears as much information as gene $k$. Gene shrinkage is a smoother, continuous, soft-thresholding method. PAM is an application of nearest centroid classification, in which the class centroids are shrunken in the direction of the overall centroid. For each

gene $i$ the value $\delta_{ic}$ measures the distance of the centroid for class $c$ to the overall centroid in units of its standard deviation. Each $\delta_{ic}$ is then reduced by an amount $\Delta$ in absolute value and is set to zero if its value becomes negative. With increasing $\Delta$, all genes lose discriminative power and more and more of them will fade away. Genes with high variance vanish faster than genes with low variance. A link between PAM and classical linear models is discussed in Huang et al. [59].

*Support vector machines* (*SVMs,* [2, 47–49]) avoid the ill-posed problem shown in Fig. 2 in Subheading 1.2 by fitting a maximal (soft) margin hyperplane between the two classes. In high-dimensional problems there are always several perfectly separating hyperplanes, but there is only one separating hyperplane with maximal distance to the nearest training points of either class. *Soft margin SVMs* trade off the number of misclassifications with the distance between hyperplane and nearest data points in the training set. This trade-off is controlled by a tuning parameter $C$. The maximal margin hyperplane can be constructed by means of inner products between training examples. This observation is the key to the second building block of SVMs: the inner product $x_i^T x_j$ between two training examples $x_i$ and $x_j$ is replaced by a nonlinear *kernel function*. The use of kernel functions implicitly maps the data into a high-dimensional space, where the maximal margin hyperplane is constructed. Thus, in the original input space, boundaries between the classes can be complex when choosing nonlinear kernel functions. In microarray data, however, choosing a linear kernel and thus deducing linear decision boundaries usually performs well.

A gene selection algorithm tailored to SVMs is *recursive feature elimination* [60]. This procedure eliminates the feature (gene) contributing least to the normal vector of the hyperplane before retraining the support vector machine on the data excluding the gene. Elimination and retraining are iterated until maximal training performance is reached. In high-dimensional microarray data, eliminating features one by one is computationally expensive. Therefore, usually several features are eliminated in each iteration.

4. *Alternative Evaluation Schemes*: The signature evaluation suggested in Subheading 3.4 yields an estimate of the signature's misclassification rate but does not provide any information on its variability. In order to investigate this aspect, you can apply methods designed to evaluate signature generating algorithms. In order to evaluate the performance of a learning algorithm, the sampling variability of the training set has to be taken into account. One approach is to perform the partitioning into training and test set ($L$ and $E$) many times randomly and

execute the complete procedure illustrated in Fig. 4 for each partitioning. This yields a distribution of misclassification rates reflecting sample variability of training and test sets. More effective use of the data can be made via cross-validation. Together with the procedure described in Subheading 3.4, two nested cross-validation loops are needed [24, 61]. Braga-Neto and Dougherty [62] advise averaging cross-validation errors over many different partitionings. Ruschhaupt et al. [61] and Wessels et al. [15] implement such complete validation procedures and compare various machine learning methods.

In the leave-one-out version of cross-validation, each sample is used in turn for evaluation while all other samples are attributed to the learning set. This evaluation method estimates the expected error rate with almost no bias. For big sample sizes, it is computationally more expensive than tenfold cross-validation and suffers from high variance [5, 7, 63]. Efron et al. [64] apply *bootstrap smoothing* to the leave-one-out cross-validation estimate. The basic idea is to generate different *bootstrap replicates*, apply leave-one-out cross-validation to each and then average results. Each bootstrap replicate contains $n$ random draws from the original dataset (with replacement so that samples may occur several times). A result of this approach is the so-called *0.632+ estimator*. It takes the possibility of overfitting into account and reduces variance compared to the regular cross-validation estimates. Ambroise et al. [53] have found it to work well with gene expression data.

Cross-validation and bootstrap smoothing error rates, as well as error rates determined by repeated separation of data into training and test set, refer to the classification algorithm, not to a single signature. In each iteration, a different classifier is learnt based on different training data. The cross-validation performance is the average of the performance of different signatures. Nevertheless, cross-validation performance can be used as an estimate of a signature's expected error rate.

5. *Biological Interpretation of Diagnostic Signatures*: The methodology earlier was presented in the classification context only. In addition, you might be tempted to interpret the genes driving the models biologically, but this is dangerous. First, it is unclear how exactly regularization biases the selection of signature genes. While this bias is a blessing from the diagnostic perspective, this is not the case from a biological point of view. Second, signatures are generally not unique: While outcome prediction for breast cancer patients has been successful in various studies [65–67], the respective signatures do not overlap at all. Moreover, Ein-Dor et al. [68] derived a large number of almost equally performing nonoverlapping signatures from a

single dataset. Also, Michiels et al. [55] report highly unstable signatures when resampling the training set.

This is not too surprising considering the following: the molecular cause of a clinical phenotype might involve only a small set of genes. This primary event, however, has secondary influences on other genes, which in turn deregulate more genes and so on. In clinical microarray analysis we typically observe an avalanche of secondary or later effects, often involving thousands of differentially expressed genes. While complicating biological interpretation of signatures, such an effect does not compromise the clinical usefulness of predictors. On the contrary, it is conceivable that only signals enhanced through propagation lead to a well generalizing signature.

## References


1. Roepman P, Wessels LF, Kettelarij N, Kemmeren P, Miles AJ, Lijnzaad P, Tilanus MG, Koole R, Hordijk GJ, van der Vliet PC, Reinders MJ, Slootweg PJ, Holstege FC (2005) An expression profile for diagnosis of lymph node metastases from primary head and neck squamous cell carcinomas. Nat Genet 37:182–186
2. Schölkopf B, Smola AJ (2001) Learning with kernels. MIT Press, Cambridge, MA
3. Ripley BD (1996) Pattern recognition and neural networks. Cambridge University Press, Cambridge
4. Devroye L, Györfi L, Lugosi L (1996) A probabilistic theory of pattern recognition. Springer, New York
5. Hastie T, Tibshirani R, Friedman J (2001) The elements of statistical learning: data mining, inference, and prediction. Springer, New York
6. Duda RO, Hart PE, Stork DG (2001) Pattern classification. Wiley, New York
7. McLachlan GJ, Do KA, Ambroise C (2004) Analyzing microarray gene expression data. Wiley, New York
8. Speed T (2003) Statistical analysis of gene expression microarray data. Chapman & Hall/CRC, Boca Raton, FL
9. Kohlmann A, Kipps TJ, Rassenti LZ, Downing JR, Shurtleff SA, Mills KI, Gilkes AF, Hofmann WK, Basso G, Dell'orto MC, Foà R, Chiaretti S, De Vos J, Rauhut S, Papenhausen PR, Hernández JM, Lumbreras E, Yeoh AE, Koay ES, Li R, Liu WM, Williams PM, Wieczorek L, Haferlach T (2008) An international standardization programme towards the application of gene expression profiling in routine leukaemia diagnostics: the Microarray Innovations in Leukemia study prophase. Br J Haematol 142 (5):802–807
10. Bacher U, Kohlmann AI, Haferlach T (2009) Perspectives of gene expression profiling for diagnosis and therapy in haematological malignancies. Brief Funct Genomics 8(3):184–193
11. Haferlach T, Kohlmann A, Schnittger S, Dugas M, Hiddemann W, Kern W, Schoch C (2005) A global approach to the diagnosis of leukemia using gene expression profiling. Blood 106:1189–1198
12. van 't Veer LJ, Dai H, van de Vijver MJ, He YD, Hart AA, Mao M, Peterse HL, van der Kooy K, Marton MJ, Witteveen AT, Schreiber GJ, Kerkhoven RM, Roberts C, Linsley PS, Bernards R, Friend SH (2002) Gene expression profiling predicts clinical outcome of breast cancer. Nature 415:530–536
13. Cheok MH, Yang W, Pui CH, Downing JR, Cheng C, Naeve CW, Relling MV, Evans WE (2003) Treatment-specific changes in gene expression discriminate in vivo drug response in human leukemia cells. Nat Genet 34:85–90
14. West M, Blanchette C, Dressman H, Huang E, Ishida S, Spang R, Zuzan H, Olson JA, Marks JR, Nevins JR (2001) Predicting the clinical status of human breast cancer by using gene expression profiles. Proc Natl Acad Sci U S A 98:11462–11467
15. Wessels LF, Reinders MJ, Hart AA, Veenman CJ, Dai H, He YD, Veer LJ (2005) A protocol for building and evaluating predictors of disease state based on microarray data. Bioinformatics 21:3755–3762
16. Dudoit S, Fridlyand J, Speed T (2002) Comparison of discrimination methods for the classification of tumors using gene expression data. J Am Stat Assoc 97:77–87
17. Jäger J, Weichenhan D, Ivandic B, Spang R (2005) Early diagnostic marker panel

determination for microarray based clinical studies. SAGMB 4, Art 9

18. John GH, Kohavi R, Pfleger K (1994) Irrelevant features and the subset selection problem. In: International conference on machine learning. Morgan Kaufmann Publishers, San Francisco, CA, USA, pp 121–129
19. Ihaka R, Gentleman RC (1996) R: a language for data analysis and graphics. J Comput Graph Stat 5:299–314
20. R Development Core Team (2006) R: a language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria
21. Gentleman RC, Carey VJ, Bates DM, Bolstad B, Dettling M, Dudoit S, Ellis B, Gautier L, Ge Y, Gentry J, Hornik K, Hothorn T, Huber W, Iacus S, Irizarry R, Leisch F, Li C, Maechler M, Rossini AJ, Sawitzki G, Smith C, Smyth G, Tierney L, Yang JY, Zhang J (2004) Bioconductor: open software development for computational biology and bioinformatics. Genome Biol 5:R80
22. Liu H, Li J, Wong L (2005) Use of extreme patient samples for outcome prediction from gene expression data. Bioinformatics 21 (16):3377–3384
23. Stone M (1974) Cross-validatory choice and assessment of statistical predictions. J R Stat Soc Ser B Methodol 36:111–147
24. Geisser S (1975) The predictive sample reuse method with applications. J Am Stat Assoc 70:320–328
25. Kohlmann A, Haschke-Becher E, Wimmer B, Huber-Wechselberger A, Meyer-Monard S, Huxol H, Siegler U, Rossier M, Matthes T, Rebsamen M, Chiappe A, Diemand A, Rauhut S, Johnson A, Liu WM, Williams PM, Wieczorek L, Haferlach T (2008) Intraplatform reproducibility and technical precision of gene expression profiling in 4 laboratories investigating 160 leukemia samples: the DACH study. Clin Chem 54(10):1705–1715
26. Geiss GK, Bumgarner RE, Birditt B, Dahl T, Dowidar N, Dunaway DL, Fell HP, Ferree S, George RD, Grogan T, James JJ, Maysuria M, Mitton JD, Oliveri P, Osborn JL, Peng T, Ratcliffe AL, Webster PJ, Davidson EH, Hood L, Dimitrov K (2008) Direct multiplexed measurement of gene expression with color-coded probe pairs. Nat Biotechnol 26(3):317–325
27. Masqué-Soler N, Szczepanowski M, Kohler CW, Spang R, Klapper W (2013) Molecular classification of mature aggressive B-cell lymphoma using digital multiplexed gene expression on formalin-fixed paraffin-embedded biopsy specimens. Blood 122(11):1985–1986
28. Scott DW, Wright GW, Williams PM, Lih C-J, Walsh W, Jaffe ES, Rosenwald A, Campo E, Chan WC, Connors JM, Smeland EB, Mottok A, Braziel RM, Ott G, Delabie J, Tubbs RR, Cook JR, Weisenburger DD, Greiner TC, Glinsmann-Gibson BJ, Fu K, Staudt LM, Gascoyne RD, Rimsza LM (2014) Determining cell-of-origin subtypes of diffuse large B-cell lymphoma using gene expression in formalin-fixed paraffin-embedded tissue. Blood 123 (8):1214–1217
29. Kim D, Pertea G, Trapnell C, Pimentel H, Kelley R, Salzberg SL (2013) TopHat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions. Genome Biol 14(4):R36
30. Wang Z, Gerstein M, Snyder M (2009) RNA-Seq: a revolutionary tool for transcriptomics. Nat Rev Genet 10(1):57–63
31. Rehrauer H, Opitz L, Tan G, Sieverling L, Schlapbach R (2013) Blind spots of quantitative RNA-seq: the limits for assessing abundance, differential expression, and isoform switching. BMC Bioinformatics 14:370
32. Xing Y, Yu T, Wu YN, Roy M, Kim J, Lee C (2006) An expectation-maximization algorithm for probabilistic reconstructions of full-length isoforms from splice graphs. Nucleic Acids Res 34(10):3150–3160
33. Mortazavi A, Williams BA, McCue K, Schaeffer L, Wold B (2008) Mapping and quantifying mammalian transcriptomes by RNA-Seq. Nat Methods 5:621–628
34. Trapnell C, Williams BA, Pertea G, Mortazavi A, Kwan G, van Baren MJ, Salzberg SL, Wold BJ, Pachter L (2010) Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. Nat Biotechnol 28:511–515
35. Wagner GP, Kin K, Lynch VJ (2012) Measurement of mRNA abundance using RNA-seq data: RPKM measure is inconsistent among samples. Theory Biosci 131(4):281–285
36. Love MI, Huber W, Anders S (2014) Moderated estimation of fold change and dispersion for RNA-Seq data with DESeq2. Genome Biol 15(12):550
37. Robinson MD, McCarthy DJ, Smyth GK (2010) edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26 (1):139–140
38. Witten DM (2011) Classification and clustering of sequencing data using a Poisson model. Ann Appl Stat 5(4):2493–2518

39. Klein MS, Buttchereit N, Miemczyk SP, Immervoll AK, Louis C, Wiedemann S, Junge W, Thaller G, Oefner PJ, Gronwald W (2012) NMR metabolomic analysis of dairy cows reveals milk glycerophosphocholine to phosphocholine ratio as prognostic biomarker for risk of ketosis. J Proteome Res 11 (2):1373–1381
40. Gronwald W, Klein MS, Zeltner R, Schulze BD, Reinhold SW, Deutschmann M, Immervoll AK, Böger CA, Banas B, Eckardt KU, Oefner PJ (2011) Detection of autosomal dominant polycystic kidney disease by NMR spectroscopic fingerprinting of urine. Kidney Int 79:1244–1253
41. Ernst RR, Bodenhausen G, Wokaun A (1987) Principles of nuclear magnetic resonance in one and two dimensions. Oxford University Press, London
42. Savorani F, Tomasi G, Engelsen SB (2010) Icoshift: a versatile tool for the rapid alignment of 1D NMR spectra. J Magn Reson 202:190–202
43. Huber W, Heydebreck AV, Sültmann H, Poustka A, Vingron M (2002) Variance stabilisation applied to microarray data calibration and to the quantification of differential expression. Bioinformatics 18:96–104
44. Bolstad BM, Irizarry RA, Astrand M, Speed TP (2003) A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. Bioinformatics 19:185–193
45. Kohl SM, Klein MS, Hochrein J, Oefner PJ, Spang R, Gronwald W (2012) State-of-the art data normalization methods improve NMR-based metabolomic analysis. Metabolomics 8:146–160
46. Breiman L (2001) Random forests. Mach Learn 45:5–32
47. Burges CJC (1998) A tutorial on support vector machines for pattern recognition. Data Min Knowl Disc 2:121–167
48. Vapnik V (1998) Statistical learning theory. Wiley, New York
49. Vapnik V (1995) The nature of statistical learning theory. Springer, New York
50. Hochrein J, Klein MS, Zacharias HU, Li J, Wijffels G, Schirra HJ, Spang R, Oefner PJ, Gronwald W (2012) Performance evaluation of algorithms for the classification of metabolic 1H-NMR fingerprints. J Proteome Res 11:6242–6251
51. Simon R, Radmacher MD, Dobbin K, McShane LM (2003) Pitfalls in the use of DNA microarray data for diagnostic and prognostic classification. J Natl Cancer Inst 95:14–18
52. Ntzani EE, Ioannidis JPA (2003) Predictive ability of DNA microarrays for cancer outcomes and correlates: an empirical assessment. Lancet 362:1439–1444
53. Ambroise C, McLachlan GJ (2002) Selection bias in gene extraction on the basis of microarray gene-expression data. Proc Natl Acad Sci U S A 99:6562–6566
54. Reid JF, Lusa L, De Cecco L, Coradini D, Veneroni S, Daidone MG, Gariboldi M, Pierotti MA (2005) Limits of predictive models using microarray data for breast cancer clinical treatment outcome. J Natl Cancer Inst 97:927–930
55. Michiels S, Koscielny S, Hill C (2005) Prediction of cancer outcome with microarrays: a multiple random validation strategy. Lancet 365:488–492
56. Dudoit S (2003) Introduction to multiple hypothesis testing. Biostatistics Division, California University, Berkeley CA, USA
57. Tibshirani R, Hastie T, Narasimhan B, Chu G (2003) Class prediction by nearest shrunken centroids, with applications to DNA microarrays. Stat Sci 18:104–117
58. Tibshirani R, Hastie T, Narasimhan B, Chu G (2002) Diagnosis of multiple cancer types by shrunken centroids of gene expression. Proc Natl Acad Sci U S A 99:6567–6572
59. Huang X, Pan W (2003) Linear regression and two-class classification with gene expression data. Bioinformatics 19:2072–2078
60. Guyon I, Weston J, Barnhill S, Vapnik V (2002) Gene selection for cancer classification using support vector machines. Mach Learn 46:389–422
61. Ruschhaupt M, Huber W, Poustka A, Mansmann U (2004) A compendium to ensure computational reproducibility in high-dimensional classification tasks. Stat Appl Genet Mol Biol 3:37
62. Braga-Neto UM, Dougherty ER (2004) Is cross-validation valid for small-sample microarray classification? Bioinformatics 20:374–380
63. Kohavi R (1995) A study of cross-validation and bootstrap for accuracy estimation and Model Selection. In International joint conference on artificial intelligence, Montreal, Quebec, Canada, pp. 1137–1145
64. Efron B, Tibshirani R (1997) Improvements on cross-validation: the 632+ bootstrap method. J Am Stat Assoc 92:548–560
65. van de Vijver MJ, He YD, van't Veer LJ, Dai H, Hart AA, Voskuil DW, Schreiber GJ, Peterse JL, Roberts C, Marton MJ, Parrish M, Atsma

D, Witteveen A, Glas A, Delahaye L, van der Velde T, Bartelink H, Rodenhuis S, Rutgers ET, Friend SH, Bernards R (2002) A gene-expression signature as a predictor of survival in breast cancer. N Engl J Med 347:1999–2009

66. Sorlie T, Tibshirani R, Parker J, Hastie T, Emr-ron JS, Nobel A, Deng S, Johnsen H, Pesich R, Geisler S, Demeter J, Perou CM, Lonning PE, Brown PO, Borresen-Dal AL, Botstein D (2003) Repeated observation of breast tumor subtypes in independent gene expression data sets. Proc Natl Acad Sci U S A 100:8418–8423
67. Ramaswamy S, Ross KN, Lander ES, Golub TR (2003) A molecular signature of metastasis in primary solid tumors. Nat Genet 33:49–54
68. Ein-Dor LE, Kela I, Getz G, Givol D, Domany E (2005) Outcome signature genes in breast cancer: is there a unique set? Bioinformatics 21:171–178

# Chapter 13

# Molecular Similarity Concepts for Informatics Applications

Jürgen Bajorath

## Abstract

The assessment of small molecule similarity is a central task in chemoinformatics and medicinal chemistry. A variety of molecular representations and metrics are applied to computationally evaluate and quantify molecular similarity. A critically important aspect of molecular similarity analysis in chemoinformatics and pharmaceutical research is that one is typically not interested in quantifying the degree of structural or chemical similarity between compounds per se, but rather in extrapolating from molecular similarity to property similarity. In other words, one assumes that there is a correlation between calculated similarity and specific properties of small molecules including, first and foremost, biological activities. Although similarity is a priori a subjective concept, and difficult to quantify, it must computationally be assessed in a formally consistent manner. Otherwise, there is little utility of similarity calculations. Consistent treatment requires approximations to be made and the consideration of alternative computational similarity concepts, as discussed herein.

**Key words** Molecular similarity and dissimilarity, Similarity-property principle, Similarity functions, Molecular descriptors, Fingerprints, Structure–activity relationships

## 1 Introduction

The assessment of molecular similarity is a key task in chemoinformatics [1–3] and also of central relevance for medicinal chemistry [3, 4]. Importantly, molecular similarity is qualitatively or quantitatively evaluated as an indicator of activity similarity. However, similarity is principally a subjective concept and no commonly applicable similarity criteria or rules exist [3]. In evaluating similarity relationships, many cognitive aspects play a role that often unconsciously determine human judgment. Regardless of whether similarity is assessed by humans or computationally, pattern recognition plays a central role in this process. To illustrate and further specify this key point, let us consider a quote from a recent work on molecular similarity analysis [3]: "*More than anything else, the recognition of molecular patterns, based on human or computational exploration, provides a basis for arriving at decisions as to whether two compounds are similar to each other or not. Since data complexity*

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_13, © Springer Science+Business Media New York 2017

*generally scales with the number of patterns that can be discovered, it quickly becomes impossible for humans to consider them in a comprehensive manner. Therefore, humans intuitively, and often unconsciously, reduce patterns to simpler ones that contain the essential feature(s) of the original pattern. But unlike applications of computational pattern recognition, the precise nature of these key patterns in human pattern recognition is unknown.*" Importantly, the computational assessment of molecular similarity stringently requires a consistent application of molecular representations and patterns derived from these representations as well as a consistent quantification of pattern correspondence in compounds under comparison. Furthermore, to quote one more time, we should consider the following [3]: "*The key patterns used by humans or computers will generally vary from individual to individual or from algorithm to algorithm, a situation that most likely will yield results with varying degrees of agreement for the same set of data. This follows because the representations used by humans and by computers, which most likely are significantly different, are crucial components in determining what can be understood about relationships of objects to each other, whether they are physical objects, concepts, ideas—or compounds.*" Moreover, human perception of molecules is strongly context- and order-dependent [5], i.e., dependent on the order in which we view compounds and their structural context, different conclusions about molecular patterns and associated properties are usually drawn. However, computational similarity methods must account for molecular representations and patterns derived from such representations in a constant and context-independent manner. Hence, for computational analysis, the inherent subjectivity of the similarity concept must be formalized in a predefined and transparent manner, which requires approximations to be made. The introductory section concludes with the definition (and differentiation) of a few key terms to provide a basis for the following discussion of alternative computational similarity concepts.

### 1.1 Substructure Matching vs. Similarity Analysis

It is important to distinguish between substructure searching/matching and similarity calculations. Substructure search methods [6] are used to detect the presence or absence of a substructure (fragment) in a compound. By definition, substructure matching provides a binary (yes/no) result. In a substructure search, all compounds are retrieved from a database that contains a prespecified substructure (query) or a combination of substructures. By contrast, similarity analysis must differentiate between different degrees of molecular similarity and hence capture a continuum of similarity relationships. Importantly, the question if two compounds are similar to each other and what their degree of similarity is cannot be answered from first principles.

### 1.2 Molecular vs. Chemical Similarity

These terms are often synonymously used, which is not entirely correct. Chemical similarity primarily considers reaction information, the presence or absence of specific functional groups, and physicochemical properties. By contrast, the assessment of molecular similarity is mostly based upon structural and topological features of compounds. In chemoinformatics, one typically attempts to extrapolate from molecular similarity to biological/activity similarity, and not from chemical similarity (although the boundaries are often fluid). Hence, herein the focus is on molecular similarity.

### 1.3 Molecular Similarity, Dissimilarity, and Diversity

It is also important to distinguish between similarity, dissimilarity, and diversity. Dissimilarity is the inverse of similarity [7]. Molecular similarity and dissimilarity are best rationalized at the level of compounds pairs (i.e., on the basis of pairwise compound comparisons). By contrast, diversity is a property of a compound set, which is closely related to chemical space coverage [8]. The major goal of diversity analysis is the generation of a compound set of limited size that best (evenly) covers a given chemical reference space [8]. In this context, chemical space is best understood as a computational construct, i.e., an $n$-dimensional reference space obtained from $n$ preselected descriptors [9] (*see* **Note 1**). Such descriptors are generally defined as mathematical models of molecular structure and/or properties and their complexity greatly varies [9].

## 2 Similarity Concepts

In the following, alternative concepts for molecular similarity analysis are presented. Key aspects of such similarity concepts are illustrated in Fig. 1. First, fundamental methodological requirements are specified.

### 2.1 Key Components of Computational Similarity Analysis

Regardless of the specifics of molecular similarity analysis, the calculation of similarity values (*see* **Note 2**) requires two basic components including (1) a *molecular representation* to capture (similarity-relevant) molecular features and (2) a *similarity function* (often called similarity coefficient) to quantitatively compare chosen representations. In addition, a *weighting scheme* can be introduced (and might be considered as a third basic component) to differentially weigh (scale) individual features of a molecular representation for similarity calculations (if all features are equally considered, no weighting is required). Given this methodological framework, similarity calculations mostly (but not exclusively) rely on pairwise molecular comparisons, i.e., a pairwise assessment of molecular similarity relationships. For many similarity functions/coefficients, calculated similarity fall within the interval [0, 1]. A similarity value of '0' reflects the presence of completely distinct

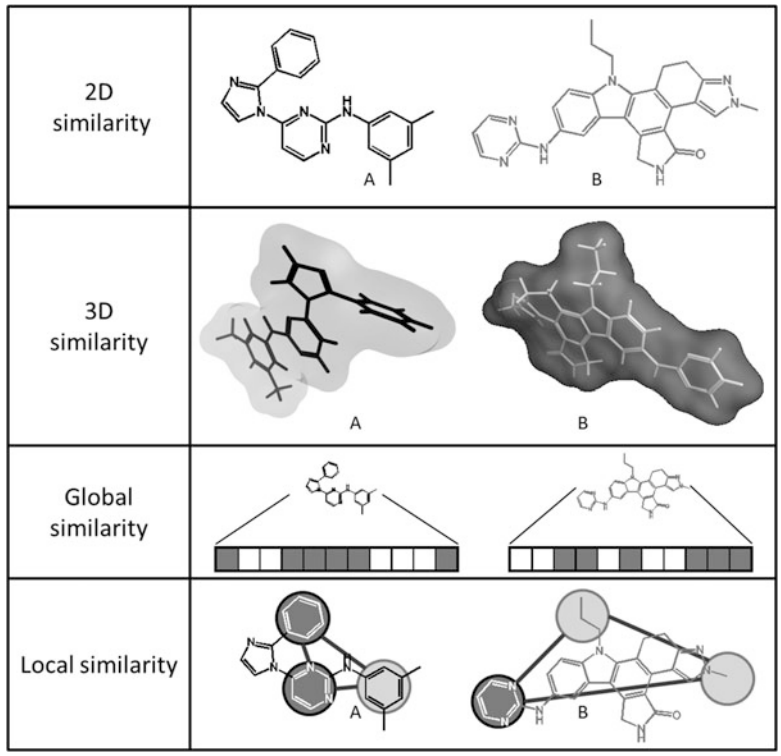



**Fig. 1** Different concepts for molecular similarity analysis are schematically illustrated. 2D similarity is assessed on the basis of molecular graphs and 3D similarity on the basis of compound conformations. Furthermore, global similarity methods compare representations of entire molecules (such as a structure- and/or property-based bit string representation), whereas local similarity methods compare subgraphs or predefined geometrical features of compounds

representations and a value of '1' the presence of identical representations (*see* **Note 3**). This also gives rise to the fundamental numerical relationship between similarity and dissimilarity:

$$\text{Dissimilarity} = 1 - \text{Similarity}$$

Although many similarity methods rely on pairwise compound comparisons, this is not strictly required. For example, partitioning algorithms operate by assigning compounds to subsets of similar ones or, alternatively, to subsections of chemical reference space (also termed cells) on the basis of descriptor (coordinates) [10]. In such cases, no pairwise compound comparisons are carried out. Rather, proximity in chemical reference space, e.g., mapping to the same cell, is applied as a similarity criterion.

### 2.2 Two- vs. Three-Dimensional Similarity

Either 2D or 3D molecular representations can be utilized as a basis for similarity calculations. In general, 2D representations are derived from information provided by molecular graphs. Popular 2D representations for similarity analysis include 'fingerprints' that

capture, for example, molecular fragments and structural patterns [11], topological pathways through compounds [12], or topological atom environments [13]. Fingerprints encode this information either as bit strings [11, 12] or feature sets [13]. Such 2D fingerprints are often of very different design. For example, the molecular access system structural key fingerprint (MACCS) [11] consists of 166 structural fragments with 1–10 nonhydrogen atoms. If a compound contains a specific feature, the corresponding bit position is set to '1'; otherwise, it is set to '0.' This represents a common procedure for many (but not all) binary fingerprint representations. Different from MACCS, the extended connectivity fingerprint with bond diameter four (ECFP4) [13] captures local bond topologies as atom environments that specify the connectivity of atoms in the neighborhood of each nonhydrogen atom in a compound. The size of the neighborhood depends on the bond diameter. Following the ECFP design, many different atom environment features are generated in a molecule-specific manner. Thus, this fingerprint does not have a fixed format but consists of a feature set. Furthermore, approaches for similarity analysis that are based on 3D representations include different approaches to compare molecular conformations [14], shape matching algorithms [15], or 3D fingerprint methods [16]. Such 3D fingerprints encode conformation-dependent molecular properties or geometric compound features. Because compounds are active in specific 3D conformations (so-called bioactive conformations), 3D similarity methods should in principle have higher information content than 2D approaches. However, this does not mean that 3D methods necessarily perform better than methods based upon 2D representations. Since molecular similarity is typically assessed as an indicator of activity similarity, as further discussed later, 3D similarity methods can only produce meaningful results if they use information from bioactive conformations. However, given the uncertainties associated with identifying bioactive conformations of ligands in the absence of experimental 3D structures, 2D approaches are often less error prone and more robust than 3D methods and produce superior results in activity predictions made on the basis of similarity analysis.

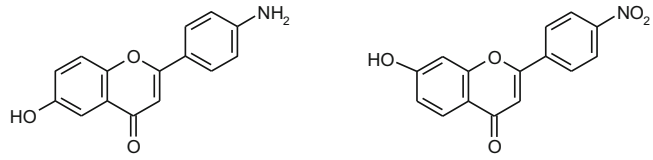
### 2.3 Global vs. Local Similarity

Molecular similarity assessment can either focus on entire compounds, applying a global view of similarity, or parts of compounds, applying a local view. A prime example for local similarity assessment is the 'pharmacophore' concept [17]. A pharmacophore is generally defined as the (spatial) arrangement of those atoms or groups in a compound that are responsible for its biological activity. Thus, pharmacophore approaches apply a strictly local view of similarity by attempting to directly focus on activity determinants. As such, pharmacophore modeling is often hypothesis driven. In a pharmacophore search, database compounds are selected as candidates that match a given pharmacophore model, but structurally

depart from known reference molecules (from which the pharmacophore was derived) in regions outside the pharmacophore [17]. Accordingly, pharmacophore searching is essentially a matching procedure producing a binary (yes/no) readout, analogously to substructure searching. By contrast, other computational approaches to similarity evaluation apply a global, whole-molecule view. For example, if compounds are translated into structural fingerprints, as discussed earlier, global molecular representations are obtained, the comparison of which results in global similarity assessment. Such global views of similarity based upon more or less abstract molecular representations are a hallmark of many chemoinformatics methods. These considerations directly lead us to a fundamental similarity principle.

### 2.4 Similarity-Property Principle

From a book publication [18], which has been a milestone event for the chemoinformatics field, the *similarity-property principle* (SPP) emerged. The SPP simply states that *similar compounds should have similar properties*, with biological activity representing the most important property. This principle clearly reflects a central issue of molecular similarity research (as already discussed previously), i.e., the extrapolation from computed molecular similarity to activity similarity, without taking activity data directly into account. Despite its apparent simplicity, the SPP has profound and complex methodological consequences. First and foremost, it requires the application of a global molecular view and a consistent definition and computational assessment of similarity. In contrast to pharmacophore modeling, the SPP does not make any assumptions about substructures or functional groups in compounds that are activity relevant. Rather, it implies that gradual changes in molecular structure are accompanied by gradual changes in activity. By contrast, small structural modifications of compounds that greatly affect or abolish biological activity, which are often observed in chemical optimization (and best accounted for using local similarity concepts), fall outside the applicability domain of the SPP, as illustrated in Fig. 2. Despite the fact that the SPP cannot account for all structure–activity relationships, it represents a paradigm for similarity searching where one generally attempts to identify structurally increasingly diverse compounds having biological activities similar to known reference molecules [19]. Similarity searching represents one of the most popular applications of molecular similarity analysis. In the following, fingerprint similarity searching is discussed as an example, which includes all key components of similarity analysis and also illustrates major opportunities and limitations of similarity calculations.




$IC_{50} = 1.2\ \mu M$ $IC_{50} > 2000\ \mu M$

**Fig. 2** A protein kinase inhibitor (*left*) and a structural analog (*right*) are shown. These compounds are very similar and only distinguished by the substitution of an amino with a nitro group. However, the inhibitor on the left is active, whereas the analog on the right is essentially inactive. On the basis of the SPP, these compounds would not be distinguished and would be assumed to have similar activity if the compound on the left was known to be active. As activity measurements, $IC_{50}$ values are reported that give the compound concentration at half-maximal inhibition. The figure was adapted from [2]

## 3 Fingerprint Searching

Fingerprint similarity searching is conceptually based on the SPP. Fingerprints including the exemplary MACCS and ECFP4 designs introduced previously represent global molecular representations. They are compared using similarity functions to obtain a numerical value that quantifies fingerprint (bit string or feature set) overlap, which is used as a measure of molecular similarity. In a typical fingerprint search, one or more known active compounds are selected as reference molecules and their fingerprint representations are searched against fingerprints of database compounds [19]. The outcome of a similarity search is a ranking of database compounds according to decreasing similarity to the reference(s). The general goal is the identification of compounds from the ranking that differ structurally from the known reference(s) but have similar activity, as further discussed later.

### 3.1 Similarity Functions

A variety of similarity functions/coefficients have been introduced for molecular similarity calculations (and were often adapted from other research fields) [20, 21]. In chemoinformatics, the Tanimoto coefficient (Tc) [21, 22] represents the most popular similarity function. For two vectors of real values, $A$ and $B$, the general Tc ($Tc_G$) is defined as:

$$Tc_G(A, B) = \frac{\sum_{i=1}^{n} A_i B_i}{\sum_{i=1}^{n} A_i^2 + \sum_{i=1}^{n} B_i^2 - \sum_{i=1}^{n} A_i B_i}$$

For binary vectors such as the MACCS fingerprint, this formulation is reduced to:

$$\mathrm{Tc}(A, B) = \frac{c}{a + b - c}$$

Here, $a$ and $b$ are the number of features present in the fingerprints of compounds $A$ and $B$, respectively, (represented by bit positions set to 1) and $c$ is the number of features that are common to $A$ and $B$. $A_i$ and $B_i$ represent the $i$th instances of compounds that are compared and $n$ is the total number of compounds. Thus, the binary Tc quantifies fingerprint overlap by producing similarity values between 0 and 1, which is the case for many similarity coefficients, as mentioned earlier.

The Tc is symmetric because the similarity of $A$ with respect to $B$ is the same as the similarity of $B$ with respect to $A$, which represents a characteristic feature of many (but not all) similarity coefficients that are used.

From the Tc, a dissimilarity measure is derived by calculating the complement known as the Soergel distance (Sg) [20]:

$$\mathrm{Sg}(A, B) = 1 - \mathrm{Tc}(A, B) = 1 - \frac{c}{a + b - c}$$

Another similarity function that can be used to introduce asymmetry in similarity calculations is the Tversky coefficient (Tv) [21, 23] defined as:

$$\mathrm{Tv}_{\alpha,\beta}(A, B) = \frac{c}{\alpha(a - c) + \beta(b - c) + c}$$

Here, $a$, $b$, and $c$ correspond to the Tc formalism. The two additional parameters $\alpha$ and $\beta$ (typically representing values between 0 and 1) are introduced to weigh the number of features that are unique to $A$ or $B$, i.e., $(a - c)$ and $(b - c)$, respectively. If $A$ and $B$ are fingerprints of a reference and database compound, respectively, the larger $\alpha$ becomes relative to $\beta$, the more weight is put on the unique bit settings of $A$ and the less weight on the bit settings of $B$ (and vice versa). This introduces asymmetry in the similarity calculations and makes it possible to emphasize unique features of reference or database compounds. For the special case $\alpha = \beta = 1$, features of $A$ and $B$ are equally weighted and Tv is identical to Tc. Furthermore, in the case $\alpha = \beta = 0.5$, Tv is transformed into the Dice coefficient (Dc) [20]:

$$\mathrm{Dc}(A, B) = \frac{c}{\frac{1}{2}(a + b)}$$

Here, the denominator represents the arithmetic mean of the number of features in $A$ and $B$.

These similarity functions illustrate the variety of coefficients (there are many more) that have been adapted for quantifying the similarity of molecular fingerprints (and other representations). Although the Tc is currently the most popular coefficient, it is not

a priori superior to others that yield different similarity values. As further discussed later, the relative ranking of compounds is critical for outcome of similarity searching, but not the absolute magnitude of similarity values. Regardless of chosen coefficients, it is of critical importance for similarity analysis that a chosen similarity function quantifies similarity relationships in a consistent manner.

However, a general statistically grounded complication of similarity searching that principally affects all coefficients is that increasing size or topological complexity of compounds typically increases the feature (bit) density of binary fingerprints, which causes a tendency to produce higher similarity values for larger compounds [19]. Such molecular size or complexity effects in fingerprint searching can be overcome by equally considering bits set to '1' and '0' (the latter reflect feature absence and are usually not taken into account) [24] or by merging fingerprints with their bit complements [25], thus producing representations of constant bit density for all compounds (*see* **Note 4**).

### 3.2 Search Strategies

If only a single reference compound is available, its fingerprint is compared to the fingerprints of all database compounds in a pairwise manner and the database compounds are ranked accordingly. If multiple references are available, which usually increases the information content of similarity searching, different strategies can be applied to take this information into account. For example, following the centroid approach [26], an average fingerprint is calculated for all reference molecules and compared to individual fingerprints of database compounds (*see* **Note 5**). Alternatively, following a nearest neighbor approach [26], similarity values of a given database compound are separately calculated for all reference molecules. Then, the largest value is assigned to the database compound or the top$k$ values are averaged to yield a final similarity score. Such nearest neighbor approaches are currently most widely used to combine contributions from multiple reference molecules.

### 3.3 Significance of Similarity Values

If calculated similarity values are to be used as indicators of biological activity, key questions include (1) whether similarity values are statistically significant and (2) whether similarity threshold values exist that firmly indicate the presence of activity relationships between reference and test compounds. Answering such questions is a nontrivial task. First of all, similarity values typically change for different combinations of fingerprints and similarity coefficients [3, 27], as illustrated in Fig. 3. Hence, they must be considered individually for given representations and similarity function. From global similarity value distributions, statistical significance of similarity values can be calculated using conventional $p$-values [3]. For example, a Tc threshold at a significance level of $p = 0.01$ reflects a probability of 1 % that the Tc value calculated for two randomly chosen compounds meets or exceeds this

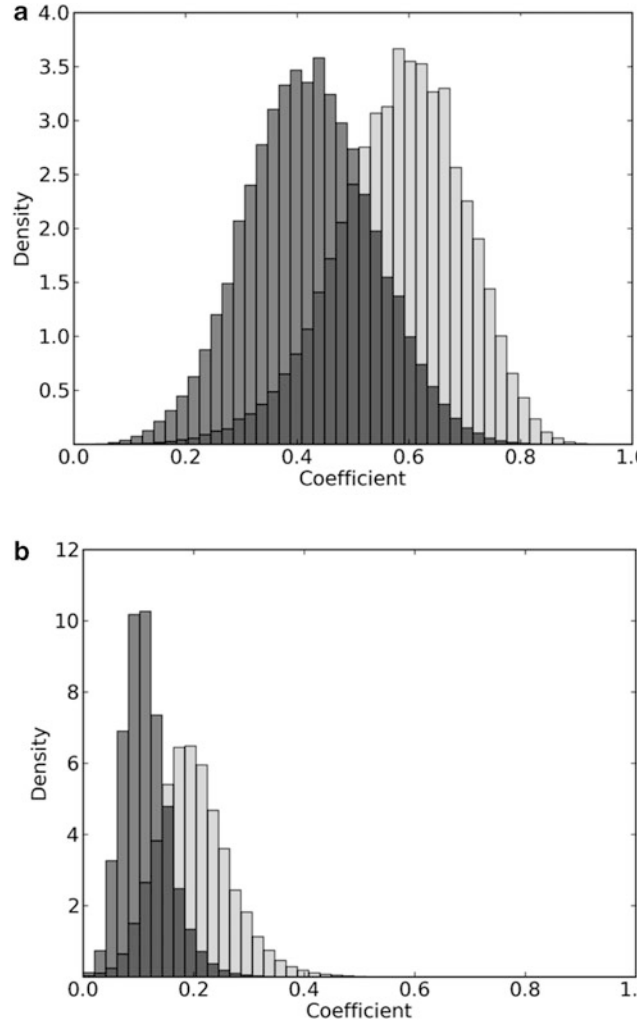



**Fig. 3** Reported are Tc (*dark gray*) and Dc (*light gray*) similarity value distributions for 10 million comparisons of randomly chosen small molecules using the (**a**) MACCS and (**b**) ECFP4 fingerprint

threshold. Importantly, however, significance levels are only based upon the distributions of similarity values and do not take biological activity as an associated property into account. Hence, one ultimately needs to determine whether similarities of compounds sharing the same activity occur by chance or if calculated similarity values at a certain level are indicative of similar activity. Therefore, different compound activity classes were used to calculate MACCS Tc similarity values for all pairs of compounds sharing the same activity [3]. Depending on the activity class, median MACCS Tc values varied from ~0.3 to ~0.75. Thus, the similarity values also showed strong compound class dependence. Furthermore, a MACCS Tc value of ~0.65 was found to correspond to a significance level of $p = 0.01$. Thus, many compounds sharing the

same activity yielded similarity values that not only varied but were not statistically significant. Corresponding observations were made for combinations of other fingerprints and similarity functions. It follows that generally applicable fingerprint similarity threshold values as potential indicators of specific biological activities cannot be determined with certainty, which substantially complicates similarity searching. Therefore, similarity-based compound rankings must be carefully analyzed.

### 3.4 Compound Rankings

The output of fingerprint similarity searching is a ranking of database compounds according to decreasing similarity to reference molecules. If we search a database for new active compounds, it is a priori clear that most database compounds cannot be specifically active. Hence, rankings are expected to mostly consist of inactive compounds. A ranking begins with those compounds that are most similar to the references(s) typically including structural analogs. Such analogs have the highest probability to be active but are not very interesting candidates for selection (they can be readily identified through a substructure search using a given core structure as a query). Rather, one would like to identify compounds that structurally depart from known references but retain similar activity, in accord with the SPP. Figure 4 shows examples of structurally diverse compounds having similar activity that were successfully identified on the basis of similarity search calculations. In order to identify such candidate compounds, one must proceed further down the database ranking where more distant structural relationships occur, without considering the magnitude of similarity values. It has also been shown that fingerprint Tc threshold values that would indicate a significant enrichment of specifically active compounds in a database ranking cannot be determined [27], for the

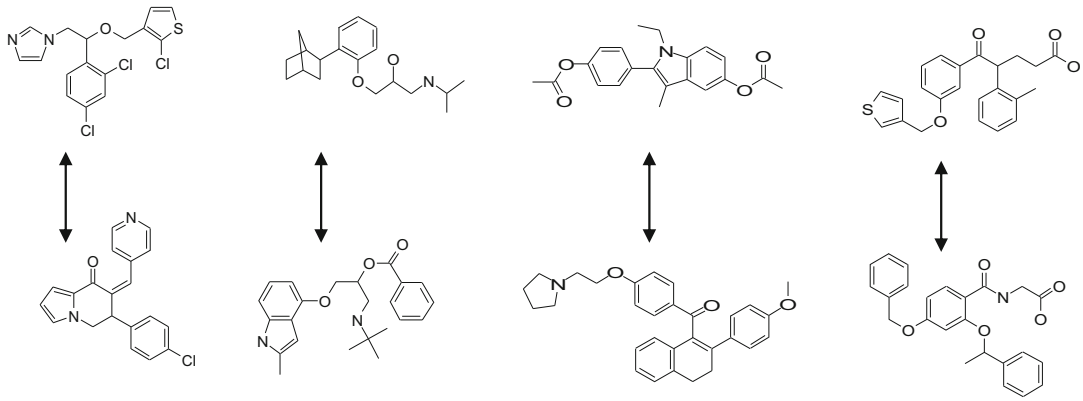



**Fig. 4** Shown are examples of compounds with limited (remote) similarity that share the same specific biological activity and were identified using similarity searching. The figure was adapted from [2]

reasons discussed earlier. However, systematic fingerprint search calculations have revealed that a few structurally novel active compounds are usually found at relatively high rank positions [27, 28], which reflects an early enrichment characteristic for small subsets of available active compounds in similarity search calculations. For example, one or more novel active compounds are frequently detected among the ~100 top-ranked database compounds [28]. These candidate compounds cannot be identified on the basis of calculated similarity values but by inspecting the continuum of similarity relationships captured by compound rankings. One does not know where exactly these active compounds are ranked but often has a good chance of identifying at least some of them by considering ~100 highly ranked candidates. Hence, despite the difficulties associated with comparing absolute similarity values for given fingerprints, similarity functions, and compound classes, the results of similarity search calculations become meaningful when focusing on individual compound rankings. Early enrichment characteristics of fingerprint search calculations provide an opportunity to identify small numbers of novel active compounds. However, there is also global enrichment detectable, although one cannot determine generally applicable similarity threshold values indicating such enrichment. As a rule of thumb, it has been shown that there typically is a statistically significant enrichment of available active compounds in database rankings when at least ~1 % of all database compounds are selected [27]. For a search database of today's size containing a million or more compounds, this fraction still corresponds to 10,000 or more candidates, by far too many for inspection and individual selection. Nonetheless, such global enrichment characteristics offer substantial opportunities for focusing of compound libraries on individual targets. Even if 10 % of database compounds would be selected on the basis of similarity search calculations to ensure likely coverage of available active compounds, significant progress would be made to limit the magnitude of experimental efforts for compound screening.

## 4 Conclusions

Molecular similarity analysis is a core task in chemoinformatics. Herein, alternative similarity concepts have been introduced and discussed that are relevant for the comparison of small molecules and evaluation of their similarity relationships. A key aspect of similarity analysis is that one is typically not interested in evaluating molecular similarity per se but in considering computed similarity relationships as an indicator of activity similarity, as best exemplified by the similarity-property principle. Molecular similarity analysis is also becoming increasingly relevant for bioinformatics applications such as the analysis of gene expression profiles of pharmaceutically

relevant compounds or the systematic study of ligand–target interactions and prediction of ligand-based target relationships. In such cases, boundaries between chemo- and bioinformatics become rather fluid. In order to review key aspects of (global) molecular similarity analysis in context, fingerprint similarity searching has been discussed, which highlights general opportunities and limitations of similarity calculations. Fingerprints are bit string representations of molecular structure (and associated properties) that are relatively simplistic in their design. Using similarity coefficients, fingerprint overlap is quantified as a measure of molecular similarity from which one extrapolates to activity similarity (without taking activity data or parameters explicitly into account). It has been emphasized that absolute similarity values have little, if any meaning for biological activity and that similarity threshold values that might be relevant for specific activities cannot be determined. In fact, the strong molecular representation and compound class dependence of similarity calculations continues to represent a major conundrum in chemoinformatics that is just beginning to be addressed in a comprehensive manner. Nonetheless, similarity search calculations have value and practical utility in the identification of novel active compounds. Computed similarity values are informative on a relative scale and represent a continuum of similarity relationships that can be further explored. Similarity-based database rankings produced by fingerprint searching often display an early enrichment of small numbers of active compounds. In practical applications where a limited number of candidate compounds are selected for testing, this is often sufficient to identify novel active molecules. Regardless of any methodological considerations and computational concepts, it should be understood that similarity is in its essence a subjective concept and that any attempts to quantify similarity relationships between molecules, or any other objects, will have intrinsic shortcomings. Being aware of such limitations will help to avoid pitfalls associated with (mis-) interpretation of calculated similarity values and focus on opportunities of the approach. After all, a consistent computational assessment of molecular similarity relationships is an absolute must, despite principal limitations, given that our ability to subjectively judge about similarity relationships is limited to rather small numbers of compounds and clearly insufficient considering current data volumes.

## 5 Notes

1. For informatics applications, chemical space is usually approximated using molecular descriptor-based reference spaces, which typically vary depending on the specific requirements of a

computational application. An a priori representation of chemical space does not exist.

2. Most but not all similarity methods calculate numerical similarity values to quantify a similarity relationship between two compounds. This is a major attraction of similarity analysis because complex molecular relationships are ultimately reduced to a simple numerical score. However, a caveat is that calculated similarity values are often over- or mis-interpreted, as discussed in the text.
3. It should be noted that a similarity coefficient value of '1' resulting from the comparison of identical molecular representations does not necessarily mean that the compared compounds are also identical. This is the case because molecular representations often abstract from compounds to varying degrees (i.e., they are essentially compound models).
4. Merging a fingerprint with its complement effectively doubles its length (which might lead to an increase in background noise of search calculations in the absence of complexity effects) but produces a constant bit density of 50 % for all test compounds, irrespective of their size and complexity. It follows that this modification renders the fingerprint representation independent of molecular size and complexity effects (due to constant bit density).
5. Even for binary fingerprints, the reference centroid represents a real-valued vector, which requires the application of the general Tc ($Tc_G$) to compare the centroid vector with binary fingerprints of database compounds.

## References


1. Bender A, Glen RC (2004) Molecular similarity: a key technique in molecular informatics. Org Biomol Chem 2:3204–3218
2. Auer J, Bajorath J (2008) Molecular similarity concepts and search calculations. Methods Mol Biol 453:327–347
3. Maggiora G, Vogt M, Stumpfe D, Bajorath J (2014) Molecular similarity in medicinal chemistry. J Med Chem 57:3186–3204
4. Kubinyi H (1998) Similarity and dissimilarity: a medicinal chemist's view. Perspect Drug Discov Des 9–11:225–232
5. Lajiness MS, Maggiora GM, Shanmugasundaram V (2004) Assessment of the consistency of medicinal chemists in reviewing sets of compounds. J Med Chem 47:4891–4896
6. Barnard JM (1993) Substructure searching methods. Old and new. J Chem Inf Comput Sci 33:532–538
7. Willett P (1999) Dissimilarity-based algorithms for selecting structurally diverse sets of compounds. J Comput Biol 6:447–457
8. Martin YC (2001) Diverse viewpoints on computational aspects of molecular diversity. J Comb Chem 3:231–250
9. Bajorath J (2001) Selected concepts and investigations in compound classification, molecular descriptor analysis, and virtual screening. J Chem Inf Comput Sci 41:233–245
10. Stahura FL, Bajorath J (2003) Partitioning methods for the identification of active molecules. Curr Med Chem 10:707–715
11. MACCS Structural Keys; Accelrys: San Diego, CA
12. James CA, Weininger D. Daylight theory manual. Daylight Chemical Information Systems, Inc., Irvine, CA

13. Rogers D, Hahn M (2010) Extended-connectivity fingerprints. J Chem Inf Model 50:742–754
14. Good AC, Richards WG (1998) Explicit calculation of 3D molecular similarity. Perspect Drug Discov Des 9–11:321–338
15. Rush TS, Grant JA, Mosyak L, Nicholls A (2005) A shape-based 3D scaffold hopping method and its application to a bacterial protein–protein interaction. J Med Chem 48:1489–1495
16. Bradley EK, Beroza P, Penzotti JE, Grootenhuis PDJ, Spellmeyer DC, Miller JL (2000) A rapid computational method for lead evolution: description and application to alpha(1)-adrenergic antagonists. J Med Chem 43:2770–2774
17. Gund P (1977) Three-dimensional pharmacophore pattern searching. In: Hahn FE (ed) Progress in molecular and subcellular biology, vol 5. Springer, Berlin, pp 117–142
18. Johnson MA, Maggiora GM (eds) (1990) Concepts and applications of molecular similarity. Wiley, New York
19. Stumpfe D, Bajorath J (2011) Similarity searching. Wiley Interdiscip Rev Comput Mol Sci 1:260–282
20. Willett P, Barnard JM, Downs GM (1998) Chemical similarity searching. J Chem Inf Comput Sci 38:983–996
21. Maggiora GM, Shanmugasundaram V (2004) Molecular similarity measures. Methods Mol Biol 275:1–50
22. Tanimoto TT (1957) IBM internal report. Nov 17
23. Tversky A (1977) Features of similarity. Psychol Rev 84:327–352
24. Fligner M, Verducci J, Blower PA (2002) Modification of the Jaccard-Tanimoto similarity index for diverse selection of chemical compounds using binary strings. Technometrics 44:110–119
25. Nisius B, Bajorath J (2010) Rendering conventional molecular fingerprints for virtual screening independent of molecular complexity and size effects. ChemMedChem 5:859–868
26. Schuffenhauer A, Floersheim P, Acklin P, Jacoby E (2003) Similarity metrics for ligands reflecting the similarity of the target proteins. J Chem Inf Comput Sci 43:391–405
27. Vogt M, Stumpfe D, Geppert H, Bajorath J (2010) Scaffold hopping using two-dimensional fingerprints: true potential, black magic, or a hopeless endeavor? Guidelines for virtual screening. J Med Chem 53:5707–5715
28. Heikamp K, Bajorath J (2011) Large-scale similarity search profiling of CHEMBL compound data sets. J Chem Inf Model 51:1831–1839

# Chapter 14

# Compound Data Mining for Drug Discovery

Jürgen Bajorath

## Abstract

In recent years, there has been unprecedented growth in compound activity data in the public domain. These compound data provide an indispensable resource for drug discovery in academic environments as well as in the pharmaceutical industry. To handle large volumes of heterogeneous and complex compound data and extract discovery-relevant knowledge from these data, advanced computational mining approaches are required. Herein, major public compound data repositories are introduced, data confidence criteria reviewed, and selected data mining approaches discussed.

**Key words** Compound activity data, Public databases, Confidence criteria, Structure–activity relationships, Matched molecular pairs, Activity cliffs, Activity profiles

## 1 Introduction

The number of compounds and associated activity data available in public domain databases currently increases at unprecedented rates. Compound activity data provide an important knowledge base for drug discovery if the data can be effectively mined [1]. Specifically, structure–activity relationships (SARs) can be systematically extracted for compounds active against current targets and utilized in compound design and optimization. Historically, most compound activity data have originated from the pharmaceutical industry and, for the most part, have been kept proprietary. However, with the changing drug discovery landscape, mergers and acquisitions, increasing discovery activities in academia, and more emphasis on discovery collaborations between biotechnology companies, the pharmaceutical industry, and academic environments, the situation has changed over the past decade. Consequences of structural changes in traditional drug discovery settings, the advent of academic drug discovery initiatives, and much more frequent and dynamic interactions between academia and pharma include, among others, data publication and release at significantly increasing rates. Chemical data are still not comparable in magnitude to

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_14, 

biological data that continue to challenge bioinformatics. However, in addition to massively growing compound data volumes, there also is rapidly increasing heterogeneity and complexity of chemical data. Taken together, these developments continuously increase the "big data" character of compound activity data, similar to biological data [2]. The "big data" nature in terms of volumes and complexity substantially challenges data organization, curation, and mining activities, not only in the pharmaceutical industry but also in the public domain. While public compound and data repositories have become essential foundations of drug discovery research in academia, it is also being recognized in pharmaceutical settings that one can no longer afford to ignore publicly available data as a source of knowledge to complement and further advance in-house research and development activities. Hence, there clearly is increasing focus on public domain compound data.

## 2 Compounds, Structures, and Activity Data

### 2.1 Public Domain Repositories

Current major publicly accessible databases for compounds and activity data include ChEMBL [3, 4], BindingDB [5], PubChem [6, 7], Open PHACTS [8], and DrugBank [9]. In addition, there are a number of more specialized smaller public databases (and also large commercial databases) that are not discussed herein. ChEMBL has become the major repository for compound activity data from medicinal chemistry sources and has recently also added patent information [4], which is of high relevance for drug discovery. The ChEMBL database has originated from a small company environment and later become a part of the European Bioinformatics Institute Outstation of the European Molecular Biology Laboratory where it is further developed [3, 10]. BindingDB was founded in academia where it continues to be advanced and maintained. It was initially designed to collect data for compounds active against targets for which three-dimensional structural information was available and has then increasingly incorporated compound activity data from the medicinal chemistry literature and other sources [4, 10]. In addition, as a repository for the Molecular Libraries Initiative of the US National Institutes of Health, PubChem has become the major public domain resource for biological screening data [6] and also maintains large compound and substance collections [7]. Open PHACTS resulted from a joint venture of a variety of academic institutions, small companies, and large pharmaceutical companies to provide pharmacological information for drug discovery in both the public and private sector via semantic web technologies [8]. The basic data unit is a so-called pharmacological record. An Open PHACTS pharmacology record reports a biological target and associated information and/or the activity of a given compound. Moreover, DrugBank, which also originated

from an academic setting, is one of the major resources for approved and experimental drugs as well as drug target information [9]. There is ongoing exchange of data between major public repositories including ChEMBL, BindingDB, and PubChem. It is fair to say that ChEMBL and PubChem currently represent the major public sources of compounds and activity data from medicinal chemistry and biological screening, respectively.

### 2.2 Data Volumes

At the beginning of 2014, the ChEMBL database (release 17) alone contained more than 1.3 million compounds with unique structures associated with more than 12 million activity annotations for ~9300 biological targets. In addition, PubChem's Compound [6], Substance [6], and BioAssay [7] collections contained ~49 million compounds, ~128 million substances, and activity data from ~740,000 assays, respectively. Furthermore, there were 3846 confirmatory bioassays available in PubChem involving 2533 biological targets. The availability of such compound data volumes could not have been imagined just a few years ago. Data volumes in these public repositories further increase on a daily basis. For example, in the current version of ChEMBL (release 18) compound activity data for ~100 additional targets have become available (compared to release 17).

### 2.3 Data Complexity

In addition to growing volumes, heterogeneity and complexity of compound activity data continuously increase [2]. Different types of assays, activity measurements, and target annotations at varying confidence levels are reported. In addition, structural information is often represented and organized in different ways. Information provided for active compounds or drugs in different databases is usually overlapping but distinct. This is best illustrated using an example. Figure 1 shows the kinase inhibitors lapatinib and imatinib that are approved drugs used in cancer treatment. For these drugs, one can readily compare the activity and target information available in different databases [2]. Early in 2014, DrugBank recorded eight targets for lapatinib, ChEMBL reported three targets for which high-confidence activity data was available, and BindingDB 814 records with defined activity measurements for this drug. Imatinib, on the other hand, was annotated with 24 targets in DrugBank and 30 high-confidence targets in ChEMBL. Furthermore, in PubChem, lapatinib was assayed 1556 times and active in 311 assays and imatinib was tested in 2467 assays and active in 469 of these. Hence, even for established and well-characterized drugs, many different measurements and target annotations are available, which are often difficult to reconcile.

### 2.4 Confidence Criteria

In light of the above, it is of critical importance to carefully consider data curation and confidence criteria. This can also be illustrated using an example [11]. Figure 2 shows two similar drugs,

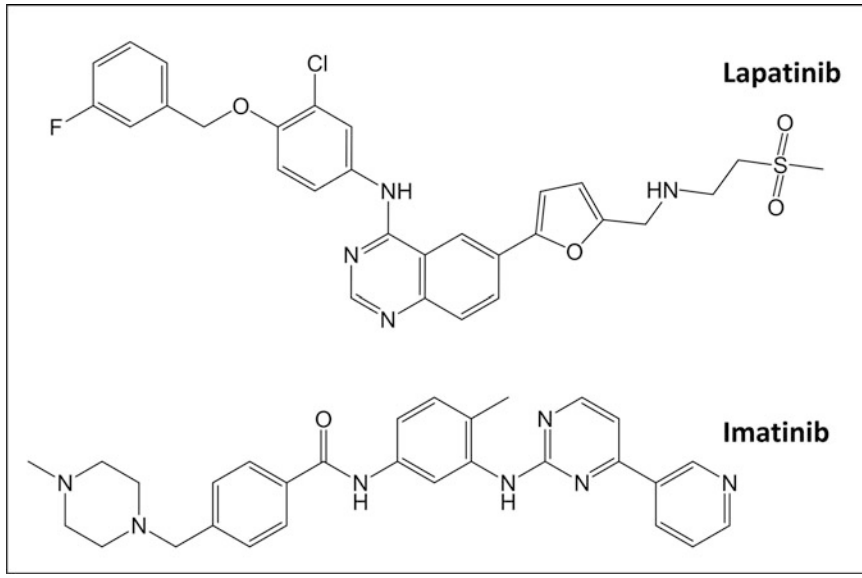



**Fig. 1** Shown are the two marketed protein kinase inhibitors lapatinib und imatinib

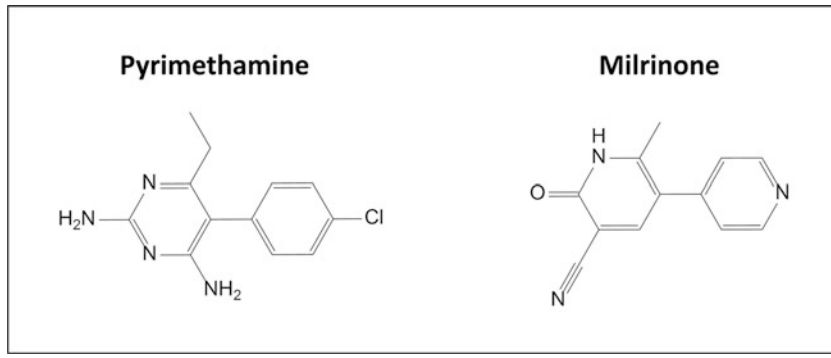



**Fig. 2** Shown are two structurally distantly related drugs, pyrimethamine and milrinone, which are used for different therapeutic indications

pyrimethamine and milrinone, which are used for different therapeutic indications, i.e., pyrimethamine is an antimalarial compounds and milrinone an inotropic cardiotonic agent. In DrugBank, pyrimethamine and milrinone were annotated with two targets and one target, respectively. By contrast, the protein target summary function of ChEMBL provided 22 and 42 targets for pyrimethamine and milrinone, respectively, hence indicating a large potential inconsistency. However, one needs to take into consideration that the protein target summary lists all potential targets for an active compound or drug, regardless of the assays used, the type of activity measurements, and the confidence level of target annotations. Thus, when stringent data selection criteria

were applied to search ChEMBL (*see* **Note 1**) only one target remained for each pyrimethamine and milrinone, which closely matched the target annotations reported in DrugBank. Thus, care must be taken to critically evaluate activity data and be aware of database-specific organization schemes, data acceptance criteria, reported measurements, and confidence levels to avoid drawing premature conclusions.

## 3 Data Mining

Given the volumes, heterogeneity, and complexity of compound activity data, there is a need for clearly defined data selection criteria and the development of advanced data mining concepts [1]. In the following, examples of advanced data mining strategies are discussed that focus the exploration of compound activity data in different ways on systematic SAR analysis and knowledge extraction for compound design and optimization. On the other hand, virtual screening aims to identify new active compounds rather than explore activity and SAR information.

### 3.1 Virtual Compound Screening

For more than two decades, virtual screening has been one of the most popular approaches in chemoinformatics and computational medicinal chemistry [12]. Ligand-based virtual screening aims at the identification of novel active compounds on the basis of known active reference molecules [12]. Here, the main goal is the identification of structurally diverse compounds having activity similar to the references, often referred to as "scaffold hopping" [13]. For this purpose, similarity-based computational screening methods are applied [14]. Compound potency is typically not taken into account as a search parameter in virtual screening. Rather, one attempts to computationally extrapolate from active reference molecules by applying principles of molecular similarity and dissimilarity [14], regardless of the potency levels of the references. Because virtual screening aims at the identification of novel active compounds, it is often not carried out in biologically annotated databases such as ChEMBL, but rather in compound databases such as ZINC [15], which currently contains ~35 millions of small molecules that are typically not biologically annotated. If virtual screening campaigns are carried out in biologically annotated databases, they mostly try to identify additional targets for known active compounds. Other exemplary data mining approaches discussed in the following focus much more on the large-scale assessment of SAR information, rather than the identification of novel hits.

### 3.2 Matched Molecular Pairs

The concept of matched molecular pairs (MMP) [16] has become increasingly important in medicinal chemistry and compound data

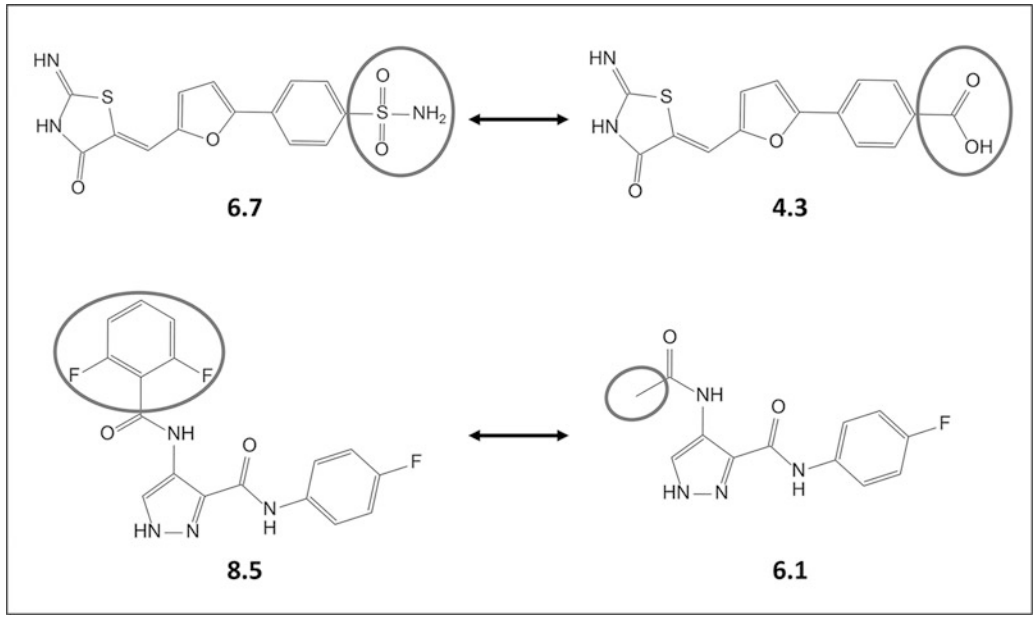



**Fig. 3** Two pairs of structurally analogous inhibitors of cyclin-dependent kinase 2 (CDK2) are shown that form matched molecular pairs (MMPs) and activity cliffs (MMP-cliffs). Substructures that distinguish the compounds in MMPs are encircled and logarithmic potency ($pIC_{50}$) values of compounds are reported. $IC_{50}$ values give the compound concentration at half-maximal inhibition

mining. An MMP is defined as a pair of compounds that only differ by a structural change at a single site, i.e., the exchange of a substructure [16]. Exemplary MMPs are shown in Fig. 3. MMPs can be algorithmically effectively generated from large compound sets [17]. This renders the MMP formalism applicable to large-scale compound data mining. A major attraction of this approach is that changes in activity or other molecular properties associated with MMP formation can be readily identified and compared for systematically generated MMPs. Hence, well-defined structural changes encoded by MMPs can be directly related to changes in biological activity or other drug-discovery relevant properties [16]. This provides a basis for the prediction of property effects in compound design and optimization. For example, structural modifications can be identified that consistently increase the potency of compounds active against a specific target. In addition, compound series can be detected that represent SAR transfer events [18]. SAR transfer involves series of pairwise corresponding structural analogs (i.e., pairs of compounds with corresponding chemical modifications) that contain distinct core structures but display similar potency progression. Hence, the identification of SAR transfer series makes it possible to replace one compound series, which might be toxic or exhibit other liabilities, with another series having similar (desired) SAR behavior, which is affected by such liabilities.

Therefore, the exploitation of SAR transfer events is highly attractive for compound optimization efforts.

### 3.3 Activity Cliffs

The activity cliff concept is also highly relevant for SAR analysis and compound optimization and amenable to large-scale data mining. An activity cliff is generally defined as a pair of structurally similar or analogous compounds that are active against the same target but display a large difference in potency [19]. Accordingly, structural modifications of active compounds can be deduced that cause significant biological effects, which rationalizes the relevance of activity cliffs for SAR analysis and compound design. For data mining, similarity and potency difference criteria for activity cliff formation must be clearly defined and consistently applied [19]. Compound similarity can be computationally assessed in a variety of ways [14] including the formation of MMPs, as discussed above (*see* **Note 2**). Thus, MMP-cliffs have been introduced as a structurally conservative and generally applicable representation of activity cliffs [20]. An MMP-cliff is formed by an MMP encoding a small structural modification (similarity criterion) if the participating compounds are active against the same target and display a potency difference of at least two orders of magnitude (potency difference criterion) [20]. Hence, the pairs of active compounds in Fig. 3 also represent MMP-cliffs. Applying alternative criteria for activity cliff formation (including MMP-cliffs), all activity cliffs formed by public domain active compounds have recently been extracted from ChEMBL and organized by targets [21]. It was found that ~10–20 % of active compounds formed activity cliffs in most target-based compound data sets (depending on the data set and the applied activity cliff definition). Thus, a significant proportion of currently available bioactive compounds form activity cliffs, which provide a large knowledge base for SAR exploration and compound optimization. From activity cliffs, SAR determinants can often be deduced. Utilizing the MMP and activity cliff concepts, compound data mining has already provided a large body of information for practical medicinal chemistry and drug discovery applications.

### 3.4 Activity Profiles

Computational methods can also be applied to systematically extract all high-confidence target annotations of compounds from activity data and generate compound activity profiles. This makes it possible to systematically assess the promiscuity of bioactive compounds [22] (*see* **Note 3**). Similarity relationships between compounds do not need to be considered to assess their promiscuity. However, compounds can also be structurally organized, for example on the basis of their scaffolds (*see* **Note 4**). Through data mining, activity profiles of compounds and corresponding scaffolds have been systematically determined and organized according to their degree of promiscuity [23]. Activity profiles of promiscuous

scaffolds can then be used to aid in the design of compounds with multi-target activities, which addresses another increasingly attractive objective in drug discovery research.

### 3.5 Conclusions

Compound activity data currently grow at unprecedented rates in the public domain and provide a valuable resource for drug discovery in academia and the pharmaceutical industry. Compound data are not the only source of discovery-relevant information. Biological, pharmacological, and clinical data are equally, if not more important, depending on the stage of drug discovery and development efforts. However, compound activity data are most relevant for medicinal chemistry and early-phase compound development. Compound data growth is accompanied by substantially increasing data heterogeneity and complexity, which challenges data mining and knowledge extraction. Herein, major public compound data repositories have been introduced and data volume, complexity, and confidence issues discussed. Since there is a clear need for advanced computational approaches and data mining strategies, selected concepts have also been reviewed that focus on large-scale SAR exploration with utility for compound design and optimization. It is anticipated that additional computational methods will be developed that closely link large-scale data mining efforts and predictive modeling to systematically generate experimentally testable SAR hypothesis and facilitate automated compound design.

## 4 Notes

1. The following specifies a protocol for the selection of high-confidence activity data from ChEMBL that we routinely apply:

   "*Only compounds with direct interactions (i.e., ChEMBL target relationship type "D") at the highest confidence level (i.e., ChEMBL target confidence score 9) are extracted. Two different types of potency measurements are separately considered including (assay-independent) equilibrium constants ($K_i$ values) and (assay-dependent) $IC_{50}$ values. Furthermore, approximate measurements such as ">", "<", or "~" are disregarded. For compounds with multiple $K_i$ or $IC_{50}$ values for the same target, the geometric mean of all potency values is calculated to yield the final potency annotation, provided all potency measurements fall within the same order of magnitude. If this is not the case, the measurements are discarded.*"

   The application of these selection criteria typically eliminates experimental inconsistencies, focuses on high-confidence data, and leads to reliable target annotations (which are separately

considered for $K_i$ or $IC_{50}$ measurements that cannot be directly compared).

2. Alternative similarity criteria for activity cliff assessment include the calculation of similarity values on the basis of molecular descriptors such as fingerprints that exceed a predefined threshold value. Such similarity values can also be consistently calculated and are often used for activity cliff analysis. A potential drawback of their use is that similarity relationships calculated on the basis of molecular descriptors are often more difficult to interpret chemically than substructure relationships established, for example, on the basis of MMPs. Principles of molecular similarity analysis and similarity calculations are discussed in more detail in the accompanying chapter by the same author.
3. Promiscuity refers here to the presence of specific interactions between a bioactive compound and multiple targets (as opposed to nonspecific binding events). The so-defined promiscuity provides the molecular basis of polypharmacology, which is an emerging theme in drug discovery. It is being recognized that many active compounds elicit therapeutically relevant effects through interactions with multiple targets and the ensuing pharmacological consequences (for example, by activating or interfering with multiple signaling pathways).
4. A scaffold essentially represents the core structure of a compound. Scaffolds can be generated in different ways, for example, by removing all substituents from a molecule and retaining the substructure containing all rings. An activity profile of a given scaffold is obtained by calculating the union of the activity profiles of all compounds the scaffold represents. This profile can be further refined by weighting individual activities by their frequency of occurrence in the activity profiles of different compounds represented by the scaffold.

## References


1. Bajorath J (2014) Improving data mining strategies for drug design. Future Med Chem 6:255–257
2. Hu Y, Bajorath J (2014) Learning from 'big data': compounds and targets. Drug Discov Today 19:357–360
3. Gaulton A, Bellis LJ, Bento AP, Chambers J, Davies M, Hersey A, Light Y, McGlinchey S, Michalovich D, Al-Lazikani B, Overington JP (2011) ChEMBL: a large-scale bioactivity database for drug discovery. Nucleic Acids Res 40:D1100–D1107
4. Bento AP, Gaulton A, Hersey A, Bellis LJ, Chambers J, Davies M, Krüger FA, Light Y, Mak L, McGlinchey S, Nowotka M, Papadatos G, Santos R, Overington JP (2014) The ChEMBL bioactivity database: an update. Nucleic Acids Res 42:D1083–D1090
5. Liu T, Lin Y, Wen X, Jorrisen RN, Gilson MK (2007) BindingDB: a web-accessible database of experimentally determined protein-ligand binding affinities. Nucleic Acids Res 35: D198–D201
6. Bolton EE, Wang Y, Thiessen PA, Bryant SH (2008) PubChem: integrated platform of small molecules and biological activities. Annu Rep Comput Chem 4:217–241
7. Wang Y, Xiao J, Suzek TO, Zhang J, Zhou Z, Han L, Karapetyan K, Dracheva S, Shoemaker BA, Bolton EE, Gindulyte A, Bryant SH

(2012) PubChem's BioAssay database. Nucleic Acids Res 42:D400–D412

8. Williams AJ, Harland L, Groth P, Pettifer S, Chichester C, Willighagen EL, Evelo CT, Blomberg N, Ecker G, Goble C, Mons B (2012) Open PHACTS: semantic interoperability for drug discovery. Drug Discov Today 17:1188–1198
9. Knox C, Law V, Jewison T, Liu P, Ly S, Frolkis A, Pon A, Banco K, Mak C, Neveu V, Djoumbou Y, Eisner R, Guo AC, Wishart DS (2011) DrugBank 3.0: a comprehensive resource for 'omics' research on drugs. Nucleic Acids Res 39:D1035–D1041
10. Wassermann AM, Bajorath J (2011) BindingDB and ChEMBL—online compound databases for drug discovery. Expert Opin Drug Discov 6:683–687
11. Hu Y, Bajorath J (2012) Many structurally related drugs bind different targets whereas distinct drugs display significant target overlap. RSC Adv 2:3481–3489
12. Geppert H, Vogt M, Bajorath J (2010) Current trends in ligand-based virtual screening: molecular representations, data mining methods, new application areas, and performance evaluation. J Chem Inf Model 50:205–216
13. Schneider G, Neidhart W, Giller T, Schmid G (1999) "Scaffold-hopping" by topological pharmacophore search: a contribution to virtual screening. Angew Chem Int Ed 38:2894–2896
14. Maggiora G, Vogt M, Stumpfe D, Bajorath J (2014) Molecular similarity in medicinal chemistry. J Med Chem 57:3186–3204
15. Irwin JJ, Shoichet BK (2005) ZINC—a free database of commercially available compounds for virtual screening. J Chem Inf Model 45:177–182
16. Griffen E, Leach AG, Robb GR, Warner DJ (2011) Matched molecular pairs as a medicinal chemistry tool. J Med Chem 54:7739–7750
17. Hussain J, Rean C (2010) Computationally efficient algorithm to identify matched molecular pairs (MMPs) in large data sets. J Chem Inf Model 50:339–348
18. Zhang B, Wassermann AM, Bajorath J (2012) Systematic assessment of compound series with SAR transfer potential. J Chem Inf Model 52:3138–3143
19. Stumpfe D, Bajorath J (2012) Exploring activity cliffs in medicinal chemistry. J Med Chem 55:2932–2942
20. Hu X, Hu Y, Vogt M, Stumpfe D, Bajorath J (2012) MMP-cliffs: systematic identification of activity cliffs on the basis of matched molecular pairs. J Chem Inf Model 52:1138–1145
21. Stumpfe D, Bajorath J (2012) Frequency of occurrence and potency range distribution of activity cliffs in bioactive compounds. J Chem Inf Model 52:2348–2353
22. Hu Y, Bajorath J (2013) Compound promiscuity: what can we learn from current data? Drug Discov Today 18:644–650
23. Hu Y, Bajorath J (2010) Polypharmacology directed data mining: identification of promiscuous chemotypes with different activity profiles and comparison to approved drugs. J Chem Inf Model 50:2112–2118

# Chapter 15

# Studying Antibody Repertoires with Next-Generation Sequencing

**William D. Lees and Adrian J. Shepherd**

## Abstract

Next-generation sequencing is making it possible to study the antibody repertoire of an organism in unprecedented detail, and, by so doing, to characterize its behavior in the response to infection and in pathological conditions such as autoimmunity and cancer. The polymorphic nature of the repertoire poses unique challenges that rule out the use of many commonly used NGS methods and require tradeoffs to be made when considering experimental design.

We outline the main contexts in which antibody repertoire analysis has been used, and summarize the key tools that are available. The humoral immune response to vaccination has been a particular focus of repertoire analyses, and we review the key conclusions and methods used in these studies.

**Key words** Antibodies, Antibodyome, Repertoire analysis, Rep-Seq, Next generation sequencing

## 1 Introduction

The adaptive immune system embodies huge diversity, and current methods are not capable of determining the entire antibodyome—the complete set of antibodies—of a mammalian species. Nevertheless, with high-throughput sequencing, it is now possible to sample at a sufficient level to gain an overview of the molecular response to a pathogen. This response is generally targeted towards a restricted set of antigens (molecules that induce an immune response). For example, the surface glycoproteins hemagglutinin and neuraminidase are the main targets of antibodies directed against the influenza A virus.

Vaccination, with its origins in medieval China and India, is probably the single most important public health measure of all time, and yet there are many pathogens, among them HIV, for which no successful vaccine has yet been developed. Even among well-established vaccines, some, such as those against influenza and tuberculosis, have limited effectiveness and breadth. By studying the change in antibody repertoire during a course of vaccination and

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_15, 

during the course of a disease, we can determine the molecular impact of vaccination on immune memory, and compare it with the memory elicited by the disease itself. By doing so, we can identify the best antigens and presentations to use in vaccines, and verify their ability to raise a lasting and broadly neutralizing response across the population as a whole [1]. We can also improve our understanding of the applicability and limits of animal models, which are often used in the surveillance of human infection as well as in the development of new treatments. As well as their use in the defense against external pathogens, antibodyome studies are used in cancer research, both to understand the natural immune response, and to establish how it could be changed or steered through vaccination, either before or after the cancer is established [2].

What information can computational repertoire analysis provide in pursuit of these goals? The questions asked in a study typically include the following:

- Which *antibody germlines* (*see* Subheading 2.2) are active in the response to a particular antigen, and how prevalent are they in the population as a whole?
- What *antibody clonotypes* (*see* Subheading 2.2) are elicited, how abundant are they, and how broad-spectrum is their response?
- Is there evidence of convergence, with antibodies originating from different germlines directed at the same antigenic target?
- Is there appropriate *isotype switching* (*see* Subheading 2.3)?
- How long is the development pathway to a given antibody of interest, and what are the key development steps?
- Is the initial response converted into long-lasting *immune memory* (*see* Subheading 2.4)?
- Are there potential obstacles to the development of an effective vaccine, for example undue focus on the development of non-neutralizing antibodies?

Computational techniques are mandated by the sheer volume of information obtained from sequencing studies. The challenges imposed by the antibodyome, in particular the high degree of sequence polymorphism and the particular mutation characteristics of somatic hypermutation, have driven the development of specialist tools, which we will highlight in this chapter. The current generation of tools tend to have limitations in terms of species coverage, sequencing requirements, performance, and so on, meaning that a careful match must be made for a particular experimental analysis.

Repertoire studies frequently bring together other sources of data, besides that available from next-generation sequencing. Antibody germline libraries, such as those available from the IMGT databases [3], are used to determine germline ancestry. Isolated

antibodies of interest, such as those that are identified as binding to the target antigen, are often sequenced by low-throughput methods. Crystallographic studies of antibody–antigen complexes may be available, and can be used to inform studies of clonotype evolution. Finally, other high-throughput tools, such as molecular mass spectrography, may be employed to characterize particular isolates [4].

## 2 Background Concepts

Here we briefly describe the key immunological concepts that are relevant to this work, and provide references for further information. We focus on human immunity, although the overall mechanisms and principles are broadly applicable. A more detailed introduction to the topics discussed can be found in Murphy [5].

### 2.1 Antigen Recognition

Antibodies are molecules which exist both as the membrane-bound receptors of B cells and as discrete molecules secreted by plasma cells. The monomeric form has a Y-shaped structure, in which the two identical arms of the Y contain the antigen receptors (Fig. 1). The molecule is made up of two identical heavy chains and two identical light chains, bound by disulfide bonds. At the N terminus of the four chains are the variable regions, which bind to antigen.

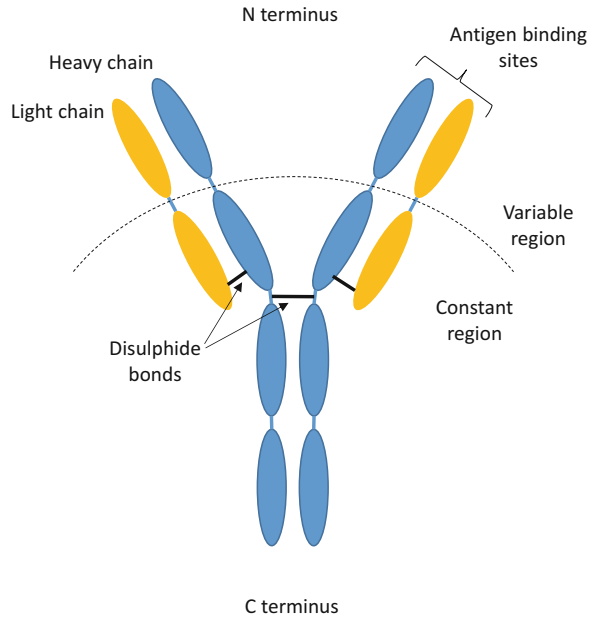



**Fig. 1** Antibody structure. The antibody is composed of four chains: two identical heavy chains, and two identical light chains. The chains are joined by disulfide bonds. The variable regions of each chain, at the N terminal end, contain the antigen receptors. The constant regions at the C terminal end contain the effectors, which determine the antibody function

Light chains have in addition a single constant region, while heavy chains have three constant regions. The C terminal constant region of the heavy chain, CH3, contains the antibody effector, which determines the antibody function and hence its class, or isotype.

The variable regions are each composed of three hypervariable loops, or complementary-determining regions, CDRs 1–3. Interspersed with these are four framing regions, FRs 1–4. During the development of receptor specificity, mutations occur primarily in the CDRs, and it is mainly residues in the CDRs that make contact with antigen, although evidence that non-CDR residues play a crucial role is growing [6, 7].

### 2.2 Receptor Development

The variable region of the heavy chain is encoded by three DNA segments. At the 3′ end is the V segment, which encodes framing regions FRs 1–3, CDRs 1 and 2, and a component of CDR3. The next is a short D segment, which encodes a part of CDR3, and finally the J segment encodes the 5′ end of the CDR3 and FR4. Multiple sequentially diverse copies of each segment exist in the germline. In an antibody-producing cell, specific V, D, and J genes are brought together to form a complete sequence by a process of gene rearrangement. A similar process is followed for the light chain, except that there are only two segments, V and J [8, 9].

Part of the diversity of antibodies comes from the random combination of gene segments, as described above. Further diversity comes from the process by which the segments come together to form the "junction" of which CDR3 is composed. The combination process is noisy, allowing for gene segments to be truncated and also for additional nucleotides to be inserted. Insight can be gained from determining the germline origin of an antibody (i.e., the particular segments from which it was derived), but the process of recombination can make it difficult to determine the origin of all segments with certainty. Because recombination involves the insertion and deletion of nucleotides it can lead to frame shifts. Resulting DNA sequences are classified as productive or nonproductive, depending on whether they can encode a functional protein. Further development and diversity follows through somatic hypermutation, a process through which mutations are introduced into the variable region during transcription through the action of activation-induced cytidine deaminase (AID). Through a process known as affinity maturation, B cells expressing antibody with high affinity to a target antigen are selected [10].

B-cells that share a common rearrangement (and hence descend from an identical naïve B-cell) are said to share the same clonotype. The human response to a single specific antigen is estimated from experiment to elicit $<100$ clonotypes [11]. Examination of the somatic hypermutation of clonotypes can cast light on the evolution of antigen specificity. As the phylogenetic evolution of distinct clonotypes proceeds independently, clonotypes provide

the best basis for the understanding of maturation pathways. However, just as germline attribution cannot in most cases be determined conclusively, there is scope for error in the attribution of clonotypes.

While the focus of this chapter is on B cells, it should be noted that T cell receptor structure and development follows a similar course, except that T cell receptors do not undergo somatic hypermutation. Software tools developed for T cell analysis may be useful for B cell analysis also, but may require modification to account for the greater sequence diversity.

### 2.3 Isotypes

There are five main isotypes: IgA, IgD, IgE, IgG, and IgM. IgM is the isotype produced initially by maturing B cells, and is therefore the isotype seen first in an immune response. As well as being present on the cellular membrane it can be secreted as a discrete molecule, where it has a pentameric structure and is found almost exclusively in the bloodstream. Through the process of isotype switching, again mediated by AID, mature B cells can switch irreversibly to another isotype [12]. The two types of most interest in vaccine-induced repertoire studies are IgA and IgG. Both can be secreted as discrete molecules. IgA is monomeric or dimeric, and is the principal class in mucosal secretions. IgG is always monomeric, and is the principal class in serum. A strong immune response will feature class switching from IgM to an appropriate combination of isotypes for the infection: as an example, the best response to a respiratory infection might be expected to contain a component of IgA: however this could be challenging to elicit with inoculation, which the immune system will recognize as a blood-borne pathogen.

### 2.4 Immunological Memory

While antibody-secreting plasma cells generally have a brief lifetime, estimated to range from several days to several months, a subset migrate to survival niches in which they can survive and continue to secrete antibodies for sustained periods that can last for many years [13]. Memory is also preserved by memory B cells, which can last for the lifetime of an individual, without the need for repeated exposure to an antigen for reinforcement. Memory B cells develop towards the end of an infection, populating the spleen and lymph nodes and circulating at low levels in the blood [14].

## 3 High-Throughput Determination of Antibody Repertoires

The total number of antibody rearrangements made possible by the mechanisms of gene rearrangement and somatic hypermutation is thought, in humans, to exceed $10^{11}$. The number of unique rearrangements in an individual at any one time is lower but still considerable: one study estimating the number of unique B-cells

as at least $3.5 \times 10^{10}$ [15]. Often, for experimental or ethical reasons, only a limited sample, such as a sample of peripheral blood, is available for analysis, although it has been estimated that only 2 % of B-cell diversity is present in peripheral blood [16]. Even where the entire organism is available, the diversity in mammals is several orders of magnitude above our current sequencing capabilities, and the cautions of working with small samples apply.

High-throughput sequencing of antibody repertoires, which has become known as Rep-Seq [17, 18], starts with the isolation of genomic DNA (gDNA) or messenger RNA (mRNA) from cells of interest. mRNA is considered to be more informative of the repertoire, as apparently viable gDNA sequences may be nonfunctional as a result of monoallelic gene expression or other mechanisms, but as levels of mRNA may vary from cell to cell, there is a risk that mRNA read counts will not correlate well with cell populations. However, a recent study did find high correlation between functional gDNA and mRNA sequence frequencies in human peripheral blood samples [19].

V-region mRNA is typically isolated for sequencing by nested RT-PCR. Multiplexed primers are required because of the degree of polymorphism. For full V-region amplification, 3′ primers are usually selected to be complementary to a section of the constant region, making them independent of the variable region germline. 5′ primers, on the other hand, are often complementary to a section of the V-region FR1, and a set of V-gene germline-dependent primers must therefore be chosen. This raises the possibility of germline-dependent amplification bias, which must be checked for if sequence counts are used to infer germline abundance, for example by comparing V-germline abundance inferred from PCR/NGS data with that inferred by single-cell analysis, or by checking for correlation in gene utilization calculations derived from two independent primer sets. An alternative approach to PCR amplification is to use 5′ RACE (rapid amplification of cDNA ends), which does not require a 5′ primer: however, RACE protocols are susceptible to nonspecific amplification and can have low efficiency.

Because of the high diversity of V-gene sequences, exacerbated by the concentration of that diversity into short CDRs separated by relatively constant FRs, it is not possible to employ sequence assembly to join the short reads typically generated by current high-throughput sequencers. The sequencers typically employed for Rep-Seq are the Roche 454 the Illumina MiSeq and the Illumina HiSeq. These are capable of covering the entire V-gene (Table 1), although the HiSeq only acquired this capability in late 2014, and many HiSeq-based studies have been limited to a specific region, typically the CDR3. New sequencing technologies, that will provide increased depth-of-coverage at reduced cost, are under development, but the overall tradeoff between entire V-gene

**Table 1**
**Characteristics of sequencers typically employed for Rep-Seq studies**

| Sequencer | Read length (nt) | Max. reads per sample | Approx. per-base error rate |
|---|---|---|---|
| Roche 454 | 700–1000 | ~$1 \times 10^6$ | ~$10^{-4}$ |
| Illumina MiSeq | 600 ($2 \times 300$) | ~$2.5 \times 10^6$ | ~$10^{-4}$ |
| Illumina HiSeq | 500 ($2 \times 250$) | ~$3 \times 10^8$ | ~$10^{-3}$ |

The V-gene can extend in some cases to >400 nt in length. With suitable PCR primers, the sequencers listed can sequence entire V-genes. The number of reads per sample is taken from manufacturers' data sheets, and numbers obtained in practice are often an order of magnitude lower

coverage on the one hand and maximum depth of coverage on the other is likely to persist for some time.

Because of the high degree of polymorphism in V-regions, separating sequencing read errors and PCR amplification errors from genuine diversity presents a challenge. An approach taken in many studies is to eliminate from consideration all V-sequences which are only observed once, on the assumption that a repeated error at the same location will be rare. Greater discrimination can be gained by adding a randomized barcode to each RNA molecule prior to PCR amplification. All reads with the same barcode should share the same sequence, and the error-free sequence can therefore be determined by consensus [20, 21]. Sequence abundance can be determined from unique barcodes, removing the risk of PCR amplification bias. Three analysis pipelines that can process barcoded reads have been published: Migec [22], Presto [23] and IgRepertoireConstructor [24]. Migec and IgRepertoireConstructor also have the capability to parse the V(D)J junction (see next section).

The heavy and light chains of an antibody are joined by disulfide bonds, and the sequencing techniques discussed above are not capable of determining which heavy and light chains are paired in particular antibodies. To determine this pairing, mRNA or gDNA products from individual cells must be isolated and identified prior to sequencing. A number of enhanced throughput techniques have been developed for this [25–27] but these techniques remain highly specialized and none have been reported that will determine the pairing at the high volumes at which sequencing is possible. In the absence of a method to determine the actual pairing, likely pairing of functionally active antibodies may be inferred via combinatorial phage display in which the chain of interest is paired with many possible chains derived from the sample [28].

## 4 Sequence Analysis and Inference of Germline Ancestry

Once sequences have been obtained, the next step will usually be to determine the boundaries of the complementary-determining and framework regions, by reference to the antibody germline sequences of the organism concerned. Determining the sequences and boundaries of CDR1 and CDR2, and the framework regions adjacent to them, is relatively straightforward as the entirety of these regions is coded in the V-gene. Assuming that a reference set of germline V-gene sequences is available, and prealigned against a reference numbering scheme such as that provided by the IMGT numbering scheme [29], the sequence of interest can be aligned against the closest-matching germline, and its constituent frames determined from the alignment. While this approach has been used successfully in many analyses, and forms the backbone of the online analysis tools described later in this section, it should be noted that V-sequences may be formed by an alternate process of gene conversion, in which sections of the ancestral V-gene are replaced by sections from alternate genes. This form of rearrangement has been observed most notably in the rabbit, but can also occur in other organisms including humans [30–32]. In B-cells, affinity maturation through the mechanism of somatic hypermutation also gives rise in some cases to highly diverged V-sequences where the germline ancestry may not be readily deduced.

A similar analysis of the V(D)J junction, based on the ancestry of the constituent genes, is more challenging for a number of reasons. The relatively small size of the D- and J-genes can make a definitive determination of ancestry difficult. The insertion and deletion of nucleotides at the junction between segments can make it difficult or impossible to infer with certainty exactly which nucleotides at the boundary were attributable to which gene. In a small number of cases—estimated at ~0.5 % of recombinations in humans—two D-genes can combine in tandem, to form a V-D-D-J junction [33]. The overall approach adopted by the majority of published tools relies on the presence of a conserved cysteine at the 5′ end of the junction and a conserved residue at the 3′ end—tryptophan for heavy chains and phenylalanine for light chains and T-cell receptor chains. In this approach, the algorithm first considers nucleotides at the 5′ end, starting from the conserved cysteine, comparing them to the nucleotides expected for the germline V-gene. The V-gene boundary is identified at the point that nucleotide divergence from the germline exceeds a defined threshold. The same process is then initiated at the 3′ end, working downwards through the junction and comparing observed nucleotides against those of the parent J-gene in order to determine the J-gene boundary. The residual sequence lying between the V-gene and J-gene boundaries is then scored against all possible D-genes.

Finally, heuristics are applied to determine inserted palindromic sequences and inserted N-nucleotides [34, 35]. Another important component of junction analysis is the identification of nonproductive recombinations, containing frame shifts or stop codons.

Where it is not critical to establish the germline ancestry of the V(D)J region, the sequence can be determined using pattern-matching methods that take advantage of the conserved residues. An advantage of this approach is that, because it does not rely on the determination of V-gene ancestry, it can be applied to short reads that do not extend substantially into the V-region [36, 37].

Both online tools and downloadable analysis tools are available (Table 2). The most frequently cited online tool for high-volume analysis is IMGT/High V-Quest [38]. While High V-Quest is supported by germline libraries for a growing number of species, it is not possible to use the tool with a customized or user-supplied germline library, limiting its use in certain applications. It is only available as an online service, and analysis completion times are dependent on system load. Another high-volume online service is

**Table 2**
**Software tools for analysis of the V(D)J junction and associated ancestry**

| Tool | Online version? | Local version? | Source code? | Custom germlines? |
|---|---|---|---|---|
| IMGT V-Quest, High V-Quest | Yes | No | No | No |
| IgBLAST | Yes | Yes | No | Yes |
| iHMMune-align [42] | Yes | Yes | Yes | Yes |
| Ab-origin [43] | No | Yes | No | Yes |
| JOINSOLVER [44] | Yes | No | No | No |
| SoDA2 [45] | Yes | No | No | No |
| VDJSolver [46] | Yes | Yes | Yes | Yes |
| Vidjil [47] | No | Yes | Yes | No |
| ARPP [48] | No | Yes | No | No |
| Decombinator [49] | No | Yes | Yes | No |
| Migec [22] | No | Yes | Yes | No |
| MiTCR [50] | No | Yes | Yes | No |
| VDJ [28] | No | Yes | Yes | No |
| VDJFasta [15][a] | No | Yes | Yes | No |
| IgSCUEAL [40] | Yes | Yes | Yes | Yes |
| IgRepertoireConstructor [24] | No | Yes | Yes | Yes |

Tools which do not allow a custom germline library to be defined through the user interface are indicated: these tools are provided with a built-in library and the ease with which it could be replaced has not been assessed. [a]VDJFasta is available at http://sourceforge.net/vdjfasta. The download location of other tools is documented in the referenced article

NCBI IgBlast [39], which employs the BLAST algorithm to determine germline matches. IgBlast does support the use of customized germline libraries, and is available as a standalone program to run locally. IgSCUEAL [40], available both online and to run locally, takes a phylogenetic approach to germline assignment. It is hosted by the HyPhy genetic sequence analysis program [41]. While it is scalable, the approach is relatively compute intensive, requiring a high-performance compute cluster for high-volume analyses. IgRepertoireConstructor [24] is an open-source package for the construction of repertoires from Illumina sequence sets. It incorporates a novel approach to sequence error correction, and supports barcoded reads and the integration of proteomics analyses.

A number of approaches have been taken to identify clonotypes from high-volume sequencing data [51]. Strictly speaking, one would wish to establish that the V, D, and J genes of clonotypes have identical ancestry, and that they share a common pattern of N-insertions, however there is uncertainty in each of these determinations, particularly where the sequence is highly diverged. A starting point is to cluster sequences with the same inferred V, D, and J germline ancestry. Subsequent clonotype clustering may then proceed on the basis of amino acid similarity [52] or nucleotide similarity [53]. While the latter approach appears a priori to conform more closely to the underlying mechanism of rearrangement, and in particular is more likely to respond to differences in "N" insertions, our own experience is that the two approaches provide broadly similar results, with both yielding well-differentiated clusters. A somewhat different approach is described by Giraud et al. [47] and implemented in Vidjil (*see* Table 2): in this approach, junction similarity is determined heuristically without full germline analysis. The Immunoglobulin Analysis Tool (IgAT) is a Microsoft Excel-based tool which provides extended analysis of IMGT results sets, including clonotype analysis [54].

Selective pressure in V-region development can be determined by comparing the ratio of observed non-silent to silent mutations: however there are specific biases in sequence specificity and base substitution in somatic hypermutation [55]. BASELINe is an online tool, also available as public domain software, which calculates the selective pressure in CD and framework regions. It can be used, for example, to compare selective pressure in different regions and in different isotypes [56, 57].

## 5 Monitoring the Humoral Immune Response to Vaccination

High-volume sequencing studies have been conducted with the aim of examining the B-cell response to vaccination or infection, and in particular to understand the process and pathways of somatic hypermutation. This is driven in large part by interest in the

development of vaccines that will elicit broad-spectrum responses to highly polymorphic viruses such as influenza and HIV. It is of particular interest in HIV, where the germline ancestors of broad spectrum antibodies are found to have little or no reactivity, suggesting that their elicitation may require a particular development path to be followed involving exposure to multiple antigens [58]. In this section, we describe the analyses that are typically conducted in these studies, and discuss their relevance to the overall problem.

A number of Rep-Seq studies have confirmed that the B-cell response to viral antigens is associated with clonal expansion and isotype switching from IgM to IgA/IgG [20, 59–61]. Clonal expansion has been observed also in patients with cancer [62]. A subset of the plasmablasts generated during peak response will survive as long-lived plasma cells. Booster vaccination studies with influenza vaccine and tetanus toxoid vaccine have found peak plasmablast production occurring 6–7 days after vaccination [60, 63]. Clonal analysis of samples taken at multiple time points, one around the peak period of plasmablast production, and one or more at times several weeks or months into the future, will facilitate the understanding of these two processes of rapid diversification and incorporation into long-term immune memory. An understanding of relative IgM levels at the different timepoints may be helpful in understanding the extent to which the response is based on the recall of immune memory [61].

Phylogenetic analysis can be used to characterize the development of antibodies of interest, but the combined effects of antibody recombination and NGS sequence read errors make this challenging. Specific tools have been developed for the analysis of variable region ancestry and inference of intermediates [48, 64–66]. Phylogenies that depict descent from a specific V-gene germline should be treated with care, as they are likely to combine descents from multiple V(D)J recombinations. Within specific clonotypes, sequence logo diagrams [67] provide a depiction that illustrates the residues explored by the clonotype, and the potential key residues that remain conserved.

Identity/divergence plots (Fig. 2) (also known as divergence-mutation scatter plots) have been used by a number of authors to investigate the relationship between an antibody of interest (typically an antibody known to be broadly or strongly neutralizing) and its germline, together with other antibodies that have the same germline ancestor [65, 68]. These plots tend to show a small number of sequences that are similar to the target antibody, and a large mass of sequences that are not. It is worth noting that this large mass of sequences is itself quite diverse, and this can be illustrated by coloring the points by clonotype [65]—an approach that will also draw out possible convergence between clonotypes towards the target antibody.

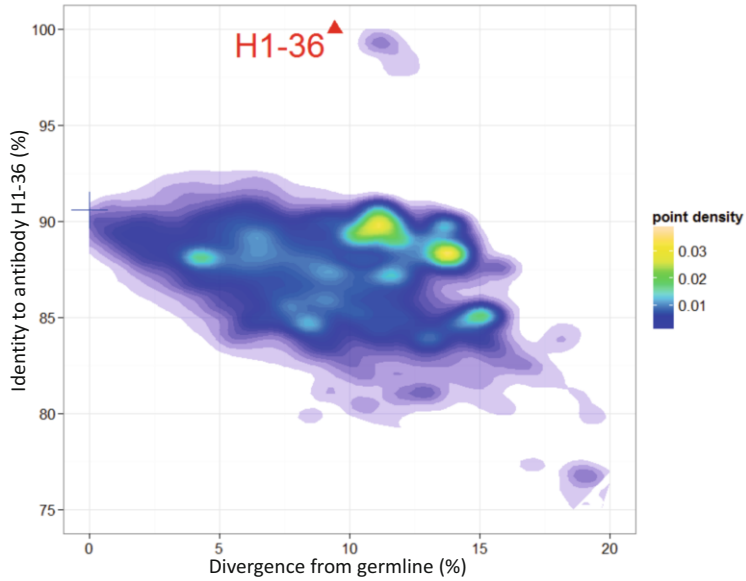



**Fig. 2** Example of an identity/divergence plot, in which V-gene sequences matching the germline of an antibody of interest (marked in *red*) are plotted in terms of their identity to the antibody and their divergence from the germline (marked with a *blue cross*). As over 200,000 sequences are represented in this plot, they are represented by means of a contour plot indicating density

## References

1. Sallusto F, Lanzavecchia A, Araki K, Ahmed R (2010) From vaccines to memory and back. Immunity 33:451–463
2. Lollini P-L, Nicoletti G, Landuzzi L et al (2011) Vaccines and other immunological approaches for cancer immunoprevention. Curr Drug Targets 12:1957–1973
3. Lefranc M-P, Lefranc G (2001) The immunoglobulin FactsBook, 1st edn. Academic, San Diego
4. Cheung WC, Beausoleil SA, Zhang X et al (2012) A proteomics approach for the identification and cloning of monoclonal antibodies from serum. Nat Biotechnol 30:447–452
5. Murphy KM (2012) Janeway's immunobiology, 8th edn. Garland Science, UK
6. Collis AVJ, Brouwer AP, Martin ACR (2003) Analysis of the antigen combining site: correlations between length and sequence composition of the hypervariable loops and the nature of the antigen. J Mol Biol 325:337–354
7. Sela-Culang I, Kunik V, Ofran Y (2013) The structural basis of antibody-antigen recognition. Front Immunol 4:302
8. Schatz DG (2004) V(D)J recombination. Immunol Rev 200:5–11
9. Schatz DG, Ji Y (2011) Recombination centres and the orchestration of V(D)J recombination. Nat Rev Immunol 11:251–263
10. Rajewsky K (1996) Clonal selection and learning in the antibody system. Nature 381:751–758
11. Poulsen TR, Jensen A, Haurum JS, Andersen PS (2011) Limits for antibody affinity maturation and repertoire diversification in hypervaccinated humans. J Immunol 187:4229–4235
12. Stavnezer J, Guikema JEJ, Schrader CE (2008) Mechanism and regulation of class switch recombination. Annu Rev Immunol 26:261–292
13. Radbruch A, Muehlinghaus G, Luger EO et al (2006) Competence and competition: the challenge of becoming a long-lived plasma cell. Nat Rev Immunol 6:741–750
14. Schittek B, Rajewsky K (1990) Maintenance of B-cell memory by long-lived cells generated from proliferating precursors. Nature 346:749–751
15. Glanville J, Zhai W, Berka J et al (2009) Precise determination of the diversity of a combinatorial antibody library gives insight into the human immunoglobulin repertoire. Proc Natl Acad Sci U S A 106:20216–20221

16. Trepel F (1974) Number and distribution of lymphocytes in man. A critical analysis. Klin Wochenschr 52:511–15
17. Fischer N (2011) Sequencing antibody repertoires. MAbs 3:17–20
18. Georgiou G, Ippolito GC, Beausang J et al (2014) The promise and challenge of high-throughput sequencing of the antibody repertoire. Nat Biotechnol 32:158–168
19. Bashford-Rogers R, Palser AL, Idris SF et al (2014) Capturing needles in haystacks: a comparison of B-cell receptor sequencing methods. BMC Immunol 15:29
20. Vollmers C, Sit RV, Weinstein JA et al (2013) Genetic measurement of memory B-cell recall using antibody repertoire sequencing. Proc Natl Acad Sci U S A 110:13463–13468
21. Schmitt MW, Kennedy SR, Salk JJ et al (2012) Detection of ultra-rare mutations by next-generation sequencing. Proc Natl Acad Sci U S A 109:14508–14513
22. Shugay M, Britanova OV, Merzlyak EM et al (2014) Towards error-free profiling of immune repertoires. Nat Methods 11:653–655
23. Vander Heiden JA, Yaari G, Uduman M et al (2014) pRESTO: a toolkit for processing high-throughput sequencing raw reads of lymphocyte receptor repertoires. Bioinformatics 30:1930–1932
24. Safonova Y, Bonissone S, Kurpilyansky E et al (2015) IgRepertoireConstructor: a novel algorithm for antibody repertoire construction and immunoproteogenomics analysis. Bioinformatics 31:i53–i61
25. Meijer P-J, Andersen PS, Haahr Hansen M et al (2006) Isolation of human antibody repertoires with preservation of the natural heavy and light chain pairing. J Mol Biol 358:764–772
26. DeKosky BJ, Ippolito GC, Deschner RP et al (2013) High-throughput sequencing of the paired human immunoglobulin heavy and light chain repertoire. Nat Biotechnol 31:166–169
27. Turchaninova MA, Britanova OV, Bolotin DA et al (2013) Pairing of T-cell receptor chains via emulsion PCR. Eur J Immunol 43:2507–2515
28. Laserson U, Vigneault F, Gadala-Maria D et al (2014) High-resolution antibody dynamics of vaccine-induced immune responses. Proc Natl Acad Sci U S A 111:4928–4933
29. Lefranc M-P, Pommié C, Ruiz M et al (2003) IMGT unique numbering for immunoglobulin and T cell receptor variable domains and Ig superfamily V-like domains. Dev Comp Immunol 27:55–77
30. Knight KL (1992) Restricted VH gene usage and generation of antibody diversity in rabbit. Annu Rev Immunol 10:593–616
31. Darlow JM, Stott DI (2006) Gene conversion in human rearranged immunoglobulin genes. Immunogenetics 58:511–522
32. Duvvuri B, Wu GE (2012) Gene conversion-like events in the diversification of human rearranged IGHV3-23*01 gene sequences. Front Immunol 3:158
33. Larimore K, McCormick MW, Robins HS, Greenberg PD (2012) Shaping of human germline IgH repertoires revealed by deep sequencing. J Immunol 189:3221–3230
34. Yousfi Monod M, Giudicelli V, Chaume D, Lefranc M-P (2004) IMGT/JunctionAnalysis: the first tool for the analysis of the immunoglobulin and T cell receptor complex V-J and V-D-J JUNCTIONs. Bioinformatics 20(Suppl 1):i379–i385
35. Jiang N, Weinstein JA, Penland L et al (2011) Determinism and stochasticity during maturation of the zebrafish antibody repertoire. Proc Natl Acad Sci U S A 108:5348–5353
36. Warren RL, Freeman JD, Zeng T et al (2011) Exhaustive T-cell repertoire sequencing of human peripheral blood samples reveals signatures of antigen selection and a directly measured repertoire size of at least 1 million clonotypes. Genome Res 21:790–797
37. Angelo SD, Glanville J, Ferrara F et al (2014) The antibody mining toolbox, an open source tool for the rapid analysis of antibody repertoires. MAbs 6:160–172
38. Alamyar E, Duroux P, Lefranc M-P, Giudicelli V (2012) IMGT(®) tools for the nucleotide analysis of immunoglobulin (IG) and T cell receptor (TR) V-(D)-J repertoires, polymorphisms, and IG mutations: IMGT/V-QUEST and IMGT/HighV-QUEST for NGS. Methods Mol Biol 882:569–604
39. Ye J, Ma N, Madden TL, Ostell JM (2013) IgBLAST: an immunoglobulin variable domain sequence analysis tool. Nucleic Acids Res 41:W34–W40
40. Frost SDW, Murrell B, Hossain ASMM et al (2015) Assigning and visualizing germline genes in antibody repertoires. Philos Trans R Soc Lond B Biol Sci 370:20140240
41. Kosakovsky Pond SL, Frost SDW, Muse SV (2005) HyPhy: hypothesis testing using phylogenies. Bioinformatics 21:676–679
42. Gaëta BA, Malming HR, Jackson KJL et al (2007) iHMMune-align: hidden Markov model-based alignment and identification of germline genes in rearranged immunoglobulin gene sequences. Bioinformatics 23:1580–1587

43. Wang X, Wu D, Zheng S et al (2008) Ab-origin: an enhanced tool to identify the sourcing gene segments in germline for rearranged antibodies. BMC Bioinformatics 9(Suppl 12):S20
44. Souto-Carneiro MM, Longo NS, Russ DE et al (2004) Characterization of the human Ig heavy chain antigen binding complementarity determining region 3 using a newly developed software algorithm, JOINSOLVER. J Immunol 172:6790–6802
45. Munshaw S, Kepler TB (2010) SoDA2: a Hidden Markov Model approach for identification of immunoglobulin rearrangements. Bioinformatics 26:867–872
46. Ohm-Laursen L, Nielsen M, Larsen SR, Barington T (2006) No evidence for the use of DIR, D-D fusions, chromosome 15 open reading frames or VH replacement in the peripheral repertoire was found on application of an improved algorithm, JointML, to 6329 human immunoglobulin H rearrangements. Immunology 119:265–277
47. Giraud M, Salson M, Duez M et al (2014) Fast multiclonal clusterization of V(D)J recombinations from high-throughput sequencing. BMC Genomics 15:409
48. Kepler TB (2013) Reconstructing a B-cell clonal lineage. I. Statistical inference of unobserved ancestors. F1000Res 2:103
49. Thomas N, Heather J, Ndifon W et al (2013) Decombinator: a tool for fast, efficient gene assignment in T-cell receptor sequences using a finite state machine. Bioinformatics 29:542–550
50. Bolotin DA, Shugay M, Mamedov IZ et al (2013) MiTCR: software for T-cell receptor sequencing data analysis. Nat Methods 10:813–814
51. Hershberg U, Prak ETL (2015) The analysis of clonal expansions in normal and autoimmune B cell repertoires. Philos Trans R Soc B 370:20140239
52. Wine Y, Boutz DR, Lavinder JJ et al (2013) Molecular deconvolution of the monoclonal antibodies that comprise the polyclonal serum response. Proc Natl Acad Sci U S A 110:2993–2998
53. Wu Y-C, Kipling D, Dunn-Walters DK (2012) Age-related changes in human peripheral blood IGH repertoire following vaccination. Front Immunol 3:193
54. Rogosch T, Kerzel S, Hoi KH et al (2012) Immunoglobulin analysis tool: a novel tool for the analysis of human and mouse heavy and light chain transcripts. Front Immunol 3:176
55. Hershberg U, Uduman M, Shlomchik MJ, Kleinstein SH (2008) Improved methods for detecting selection by mutation analysis of Ig V region sequences. Int Immunol 20:683–694
56. Uduman M, Yaari G, Hershberg U et al (2011) Detecting selection in immunoglobulin sequences. Nucleic Acids Res 39:W499–W504
57. Yaari G, Uduman M, Kleinstein SH (2012) Quantifying selection in high-throughput Immunoglobulin sequencing data sets. Nucleic Acids Res 40:e134
58. West AP Jr, Scharf L, Scheid JF et al (2014) Structural insights on the role of antibodies in HIV-1 vaccine and therapy. Cell 156:633–648
59. Frölich D, Giesecke C, Mei HE et al (2010) Secondary immunization generates clonally related antigen-specific plasma cells and memory B cells. J Immunol 185:3103–3110
60. Wrammert J, Smith K, Miller J et al (2008) Rapid cloning of high-affinity human monoclonal antibodies against influenza virus. Nature 453:667–671
61. Jiang N, He J, Weinstein JA et al (2013) Lineage structure of the human antibody repertoire in response to influenza vaccination. Sci Transl Med 5:171ra19
62. Bashford-Rogers R, Palser AL, Huntly BJ et al (2013) Network properties derived from deep sequencing of human B-cell receptor repertoires delineate B-cell populations. Genome Res 23:1874–1884
63. Lavinder JJ, Wine Y, Giesecke C et al (2014) Identification and characterization of the constituent human serum antibodies elicited by vaccination. Proc Natl Acad Sci U S A 111:2259–2264
64. Barak M, Zuckerman NS, Edelman H et al (2008) IgTree©: creating immunoglobulin variable region gene lineage trees. J Immunol Methods 338:67–74
65. Sok D, Laserson U, Laserson J et al (2013) The effects of somatic hypermutation on neutralization and binding in the PGT121 family of broadly neutralizing HIV antibodies. PLoS Pathog 9:e1003754
66. Lees WD, Shepherd AJ (2015) Utilities for high-throughput analysis of B-cell clonal lineages. J Immunol Res, Article ID 323506
67. Schneider TD, Stephens RM (1990) Sequence logos: a new way to display consensus sequences. Nucleic Acids Res 18:6097–6100
68. Wu X, Zhou T, Zhu J et al (2011) Focused evolution of HIV-1 neutralizing antibodies revealed by structures and deep sequencing. Science 333:1593–1602

# Chapter 16

# Using the QAPgrid Visualization Approach for Biomarker Identification of Cell-Specific Transcriptomic Signatures

**Chloe Warren, Mario Inostroza-Ponta, and Pablo Moscato**

## Abstract

In this chapter, we illustrate the use of an integrated mathematical method for joint clustering and visualization of large-scale datasets. In applying these clustering methodologies to biological datasets, we aim to identify differentially expressed genes according to cell type by building molecular signatures supported by statistical scores. In doing so, we also aim to find a global map of highly co-expressed clusters. Variations in these clusters may well indicate other pathological trends and changes.

**Key words** Clustering, Visualization, Neuroscience, Genetics

## 1 Introduction

We have previously developed these methods for the analysis of gene expression microarray (GEM) datasets [1], such as those produced by studies that seek to reverse engineer the functional genomics of model organisms like *Saccharomyces cerevisiae* [2]. We have also applied the methods to the whole-genome analysis of RNA stability [3] and we have created an interactive map based on the results. In addition, our method has also been used to generate a successful redefinition of clinical cases of proctitis syndrome [4]. Interestingly, in all these cases, we have dealt with situations in which the values under consideration (gene expression, symptom measurements) change over time. This has lead us to evaluate this method as a potential tool to provide new insights into our understanding of coherent patterns of co-expression during the progression of neurodegeneration, which occurs in a number of diseases including Alzheimer's, Parkinson's diseases and multiple sclerosis.

However, preliminary investigation of some of the GEM datasets associated with these diseases has led us to prioritize the work we present here in our research agenda. This work came about as a

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_16, 

consequence of one of the shortcomings of GEM technologies. Namely, the biological samples used to generate GEM data can be comprised of whole tissue, meaning that there are multiple cell types present within any one sample. Such sample heterogeneity can complicate the functional biological interpretation of data, as well as the identification of biomarkers [5–7]. Herein we have been inspired to generate methodologies to identify clusters of gene expression signatures that define different cell types.

With this contextual information, we introduce the *QAPgrid integrated clustering and visualization approach* in the field of identification of neurological cell-specific transcriptomes. We use a GEM data set of distinct cell types (neurons, astrocytes, and oligodendrocytes) taken from mouse brains [8]. As previously mentioned, sample heterogeneity can be a major hindrance to the interpretation of biological data. In particular, the brain is renowned for its complex cellular composition. As such, these data were collected using novel cell separation techniques in order to provide a resource for improved understanding of the development, physiology and pathology of the brain [8].

We envisage that further development of these data visualization and analysis methods will lead to the production of a multidimensional resource, wherein graphical representation of expression clusters will be hyperlinked to specific expression data as well as online gene function information.

## 2 Methodology

The GEM data used in these investigations is publically available in the NCBI Gene Expression Omnibus (GEO; http://www.ncbi.nlm.nih.gov/geo/), via the accession number GSE9566. The data were gathered using Affymetrix GeneChip Arrays, and represents three different cell types (neurons; astrocytes; oligodendrocytes) which were acutely isolated from the forebrains of healthy mice. The mice's ages varied between 7 and 17 days. The data were annotated with gene names and symbols using SOURCE (http://smd.stanford.edu/cgi-bin/source/sourceSearch), and normalized such that the sum of all the probe sets values in a sample was equal to 1.

### 2.1 Well Described Biomarker Gene Lists

The lists of well described biomarker genes (*see* Tables 1, 2, and 3) were obtained from the original publication of the data [8], but it is significant that the basis for selection of these markers was not on the data itself, but on previous studies (*see* full list of references, *see* Table 4); it is therefore unbiased in its nature. It should be noted that, in GEMs, genes are often represented by multiple probe sets, which are comprised of 25-base sequences designed to hybridize to various points within the target gene [9]. A common issue with

**Table 1**
**Well described biomarker probe sets for astrocytes, and their location in super clusters and clusters, as specified by application of MST-*k*NN unsupervised clustering algorithm to GEM dataset of pooled cell types (astrocytes, neurons, and oligodendrocytes) isolated from mouse forebrains**

| Gene name | Gene symbol | Probe sets ID | Super cluster | Cluster |
|---|---|---|---|---|
| Solute carrier family 1 (glial high affinity glutamate transporter), member 2 | Slc1a2 | 1438194_at | 1 | 445 |
| | | 1459014_at | 1 | 254 |
| | | 1433094_at | 1 | 445 |
| | | 1457800_at | 1 | 445 |
| | | 1439940_at | 1 | 445 |
| | | 1451627_a_at | 1 | 445 |
| Gap junction protein, beta 6 | Gjb6 | 1448397_at | 1 | 254 |
| Glial fibrillary acidic protein | Gfap | 1440142_s_at | 1 | 254 |
| | | 1426509_s_at | 1 | 254 |
| | | 1426508_at | 1 | 254 |
| Solute carrier family 1 (glial high affinity glutamate transporter), member 3 | Slc1a3 | 1426341_at | 1 | 254 |
| | | 1439072_at | 1 | 445 |
| | | 1440491_at | 1 | 254 |
| | | 1452031_at | 1 | 254 |
| | | 1443749_x_at | 1 | 254 |
| | | 1426340_at | 1 | 254 |
| Aquaporin 4 | Aqp4 | 1447745_at | 1 | 254 |
| | | 1425382_a_at | 1 | 254 |
| | | 1434449_at | 1 | 254 |
| Aldolase C, fructose-bisphosphate | Aldoc | 1424714_at | 1 | 254 |
| | | 1451461_a_at | 1 | 559 |
| Fibroblast growth factor receptor 3 | Fgfr3 | 1421841_at | 1 | 254 |
| | | 1425796_a_at | 1 | 445 |

GEM interpretation is that probe sets for the same gene often have different expression patterns [9, 10]. There are multiple explanations for this; probe sets can hybridize to splice variants of a gene, or can inadvertently hybridize nonspecifically to a different (i.e., wrong) gene due to sequence similarity [9]. In an attempt to limit the affect this issue has on our analysis of the effectiveness of our methods, we have discounted probe sets whose expression is non-cell specific from the lists of well described biomarkers, as seen in Tables 1, 2, and 3.

### 2.2 Clustering

Clustering algorithms applied to gene expression data aim to find groups of related probe sets that share common characteristics, like relative expression values or expression profiles across a sample set. These common characteristics are usually measured using either a similarity or distance metric. The most common of such measures

**Table 2**
**Well described biomarker probe sets for oligodendrocytes, and their location in super clusters and clusters, as specified by application of MST-*k*NN unsupervised clustering algorithm to a GEM dataset of pooled cell types (astrocytes, neurons, and oligodendrocytes) isolated from mouse forebrains**

| Gene name | Gene symbol | Probe sets ID | Super cluster | Cluster |
|---|---|---|---|---|
| SRY-box containing gene 10 | Sox10 | 1451689_a_at | 1 | 1 |
| | | 1424985_a_at | 1 | 1 |
| Platelet derived growth factor receptor, alpha polypeptide | Pdgfra | 1421916_at | 1 | 1 |
| Gap junction protein, gamma 2 | Gjc2 | 1450483_at | 1 | 1 |
| | | 1435214_at | 1 | 1 |
| Myelin basic protein | Mbp | 1451961_a_at | 1 | 1 |
| | | 1419646_a_at | 1 | 11 |
| | | 1436201_x_at | 1 | 11 |
| | | 1454651_x_at | 1 | 11 |
| | | 1456228_x_at | 1 | 11 |
| | | 1433532_a_at | 1 | 11 |
| | | 1425263_a_at | 0 | 5 |
| Myelin oligodendrocyte glycoprotein | Mog | 1448768_at | 1 | 1 |
| Myelin-associated oligodendrocytic basic protein | Mobp | 1433785_at | 1 | 11 |
| | | 1421010_at | 1 | 1 |
| | | 1436263_at | 1 | 1 |
| | | 1450088_a_at | 1 | 1 |
| UDP galactosyltransferase 8A | Ugt8a | 1419064_a_at | 1 | 1 |
| | | 1419063_at | 1 | 1 |
| Galactose-3-O-sulfotransferase 1 Gal3st1 | Gal3st1 | 1454078_a_at | 1 | 1 |
| Myelin-associated glycoprotein | Mag | 1460219_at | 1 | 1 |
| Myelin and lymphocyte protein, T cell differentiation protein | Mal | 1432558_a_at | 0 | 0 |
| | | 1417275_at | 1 | 26 |

are the Euclidean based metric and correlation-based metrics. The choice of metric is a key step, as (1) it defines when two probe sets are going to be considered similar and (2) it has to be relevant for the questions being asked (for instance, the use of robust correlation metrics may be necessary for some problem domains). In order to analyze GEM data and find groups of related probe sets, we use the unsupervised graph-based clustering algorithm, MSTkNN [11, 12]. We use the Pearson correlation distance metric, as shown in Eq. (1). This metric defines a distance between 0, totally uncorrelated probe sets, and 2, totally correlated probe sets. It has the advantage of not being sensitive to the amount of expression, and

**Table 3**
**Well described biomarker probe sets for neurons, and their location in super clusters and clusters, as specified by application of MST-*k*NN unsupervised clustering algorithm to a GEM dataset of pooled cell types (astrocytes, neurons, and oligodendrocytes) isolated from mouse forebrains**

| Gene name | Gene symbol | Probe sets ID | Super cluster | Cluster |
|---|---|---|---|---|
| Neurofilament, light polypeptide | Nefl | 1426255_at | 8 | 561 |
| | | 1454672_at | 8 | 190 |
| Gamma-aminobutyric acid (GABA) A receptor, subunit alpha 1 | Gabra1 | 1455766_at | 8 | 190 |
| | | 1436889_at | 2 | 472 |
| | | 1421281_at | 8 | 190 |
| Synaptotagmin I | Syt1 | 1421990_at | 8 | 190 |
| | | 1431191_a_at | 8 | 561 |
| | | 1433884_at | 2 | 499 |
| Solute carrier family 12, member 5 | Slc12a5 | 1451674_at | 2 | 499 |
| Synaptosomal-associated protein 25 | Snap25 | 1416828_at | 8 | 190 |
| Synaptic vesicle glycoprotein 2 b | Sv2b | 1434800_at | 8 | 190 |
| | | 1435687_at | 8 | 531 |
| Potassium voltage-gated channel, subfamily Q, member 2 | Kcnq2 | 1451595_a_at | 2 | 152 |

being able to identify similar expression profiles between two probe sets.

$$d_{xy} = 1 - \rho_{xy} \tag{1}$$

The *MSTkNN* clustering algorithm is based on the use of two proximity graphs, namely the Minimum Spanning Tree (MST), and $k$ Nearest Neighbor graph ($kNN$). The MST is a connected acyclic graph with the smallest sum of edge's weights, and in the $kNN$ graph, two vertices are connected if either one or the other are among the $k$ closest neighbors of each other according to the edge's weights. The algorithm first builds a complete graph $G(V,E,W)$, with a vertex $v$ for each of the probe sets, and an edge $e_{xy}$ for each of the probe set pairs $(x,y)$, with the edge's weight being the distance between the expression profiles of probe sets. First, the algorithm computes the Minimum Spanning Tree $G_{MST}(V,E_{MST},W_{MST})$, where $E_{MST}$ and $W_{MST}$ are subsets of $E$ and $W$ respectively. Second, the algorithm computes the $k$-Nearest Neighbor graph $G_{kNN}(V, E_{kNN},W_{kNN})$, where again $E_{kNN}$ and $W_{kNN}$ are subsets of E and W respectively. The number of nearest neighbors to be considered is automatically computed using Eq. (2) (*see below*). Then, the algorithm computes the intersection of the edge sets of both graphs ($E_{MST} \cap E_{kNN}$), which will produce a partition of the graph in $c \geq 1$ connected components. If $c = 1$, then the algorithm stops.

**Table 4**
**Literature references for well described marker genes of neurons, oligodendrocytes, and astrocytes**

| Cell type | Well described marker | Symbol | Reference |
|---|---|---|---|
| Neuron | Neurofilament, light polypeptide | Nefl | [19] |
| | Gamma-aminobutyric acid (GABA) A receptor, subunit alpha 1 | Gabra1 | [20] |
| | Synaptotagmin I | Syt1 | [20] |
| | Solute carrier family 12, member 5 | Slc12a5 | [20] |
| | Synaptosomal-associated protein 25 | Snap25 | [21] |
| | Potassium voltage-gated channel, subfamily Q, member 2 | Kcnq2 | [22] |
| | Synaptic vesicle glycoprotein 2 b | Sv2b | [23] |
| Oligodendrocyte | Chondroitin sulfate proteoglycan 4 | Cspg4 | [24] |
| | SRY-box containing gene 10 | Sox10 | [25, 26] |
| | Platelet derived growth factor receptor, alpha polypeptide | Pdgfra | [20, 27] |
| | Gap junction protein, gamma 2 | Gjc2 | [28] |
| | Myelin basic protein | Mbp | [29, 30] |
| | Myelin oligodendrocyte glycoprotein | Mog | [30] |
| | UDP galactosyltransferase 8A | Ugt8a | [31] |
| | Galactose-3-O-sulfotransferase 1/(Cerebroside sulfotransferase) | Gal3st1 | [32] |
| | Myelin-associated oligodendrocytic basic protein | Mobp | [33] |
| | Myelin-associated glycoprotein | Mag | [30] |
| | Myelin and lymphocyte protein, T cell differentiation protein | Mal | [34] |
| Astrocyte | Solute carrier family 1 (glial high affinity glutamate transporter), member 2 | Slc1a2 | [35] |
| | Gap junction protein, beta 6 | Gjb6 | [36] |
| | Glial fibrillary acidic protein | Gfap | [29, 37, 38] |
| | Solute carrier family 1 (glial high affinity glutamate transporter), member 3 | Slc1a3 | [39] |
| | Aquaporin 4 | Aqp4 | [40] |
| | Aldolase C, fructose-bisphosphate | Aldoc | [41] |
| | Fibroblast growth factor receptor 3 | Fgfr3 | [42] |

Otherwise ($c > 1$), the algorithm is recursively applied in each of the connected components. Algorithm 1 shows the pseudocode of the clustering algorithm. It receives a distance matrix and it returns a graph with $c \geq 1$ connected components. A schematic representation of the algorithm is shown in Fig. 1.

$$k = \min\{\ln(n), \min k \;/\; G_k \, is \, connected\} \tag{2}$$

Firstly, the algorithm uses a large number of nearest neighbors to find clusters. It then uses a smaller number depending on the number of probe sets. It is also possible to restrict the number of nearest neighbors to a $k_{max}$ by adding it to Eq. (2): $k = \min\{\ln(n), \min k \;/\; G_k \, is \, connected, \boldsymbol{k_{max}}\}$. This condition forces the algorithm to consider only up to the $k_{max}$ larger similarities (or smallest distances).

**Algorithm 1**

**MSTkNN clustering algorithm. Function *connectedComponents*(*G*) returns the number of connected components in *G*. Function *submatrix*(*D*, $G^i_{CLUSTER}$) get the distance matrix of the elements of the *i*th connected component in $G_{CLUSTER}$**

**MSTkNN**(D: distance matrix)

1. Compute $G$
2. Compute $G_{MST}$
3. Compute $G_{kNN}$, with $k = \min\{\ln(n), \min k \ / \ G_k \ is \ connected\}$
4. $G_{CLUSTER} = \{V_{CLUSTER} = V, E_{CLUSTER} = E_{MST} \cap E_{kNN}\}$
5. c = connectedComponents($G_{CLUSTER}$)
6. **IF** (c > 1) **THEN**
7. $G_{CLUSTER} = \cap_{i=1}^{c} MSTkNN\left(submatrix\left(D, G^i_{CLUSTER}\right)\right)$
8. **ENDIF**
9. **RETURN** $G_{CLUSTER}$

**END MSTkNN**

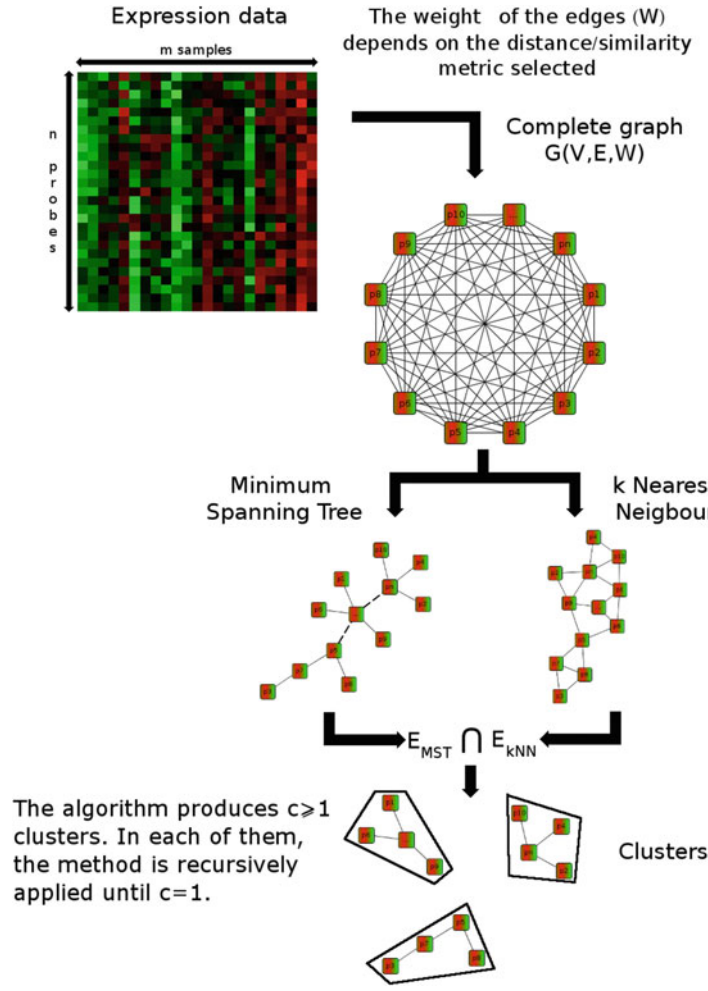



**Fig. 1** Schema of the clustering algorithm MSTkNN used for the analysis of mouse brain tissue. The algorithm automatically decides the number of neighbors to be considered in each step

For this data set, the *MSTkNN* algorithm found 760 clusters of similarly expressed probe sets, with the five largest clusters consisting of 2520 (cluster 252), 2067 (cluster 190), 1581 (cluster 2), 1371 (cluster 50), and 1306 (cluster 68) probe sets. The large number of clusters found makes it hard to analyze the results. To help the analysis, we now group clusters of similarly expressed genes. We apply the clustering algorithm considering that each cluster is now an object, and the distance between two clusters corresponds to the average pairwise distance between all members of two clusters. In this new data set, the MSTkNN algorithm produces 14 "superclusters" (14 clusters of clusters). The largest supercluster consists of 324 clusters (supercluster 1).

### 2.3 QAPgrid

A clustering algorithm will deliver one or more groups that are related according to some criteria. One of the well-known shortcomings of any clustering algorithm is that it can be too sensitive and therefore separate a group of probe sets that should be together. On the other hand, the algorithm might not separate probe sets that should be separated. In order to deal with these situations, we use a layout procedure based on the pairwise relationships of all probe sets. The relationship between any two probe sets can be measured using any similarity/distance measure, in a similar way to that of the clustering algorithm. The layout procedure of the QAPgrid is a combinatorial optimization approach to generating an ordered layout which is mathematically guided in order to place highly related objects in nearby positions in a 2D grid. The layout problem is modeled as an instance of the Quadratic Assignment Problem, and since the QAP belongs to the NP-hard class of problems, we use an ad hoc Memetic algorithm [13, 14] which we have developed to deal with the QAP instances. In QAP, we need to assign a set of $n > 0$ facilities to $m$ locations, given as input to a flow matrix between the facilities and the distance between locations to which these facilities would be assigned. The flow between facilities and distances between locations are represented using matrices $F$ and $L$, respectively. The goal is to minimize the overall flow $C$ in the system shown in Eq. (3) below, where $s(i)$ represents the assigned location of facility $i$ in a solution $s$, and $l_{s(i)s(j)}$ represents the distance between locations of facilities $i$ and $j$, while $f_{ij}$ represents the flow between facilities $i$ and $j$. Then, a good solution for the QAP will put in nearby locations facilities sharing a high flow according to the objective function C given by:

$$C = \sum_{i=1}^{n} \sum_{j=i+1}^{n} f_{ij} l_{s(i)s(j)} \tag{3}$$

In order to create instances of the QAP for the layout problem at hand, each probe set is represented by a facility, and the flow between facilities is created using Eq. (4). Following this, we create

a square grid of $m$ locations, with $m \gg n$, such that the algorithm has no space restrictions to produce the layout. Additionally, we are able to consider other information, such as a graph model that highlights certain probe sets' relationships in the data set. Say we have a graph $H(V,E)$, where each vertex in $V$ represents a probe. In this graph, there would be an edge between two vertices if there was intent that these two vertices may be prioritized to be close in the final layout (i.e., apart from their similarity there might be other specific criteria). We bias the search towards attempting to co-locate these two probe sets together by increasing the flow value between the two given probe sets by a factor of $M \gg n$ as it is shown in Eq. (4).

$$f_{ij} = \begin{cases} 0 & \text{if } i = j \\ \dfrac{M}{d_{ij}} & \text{if } e_{ij} \in E \\ \dfrac{1}{d_{ij}} & \text{otherwise} \end{cases} \tag{4}$$

The QAPgrid algorithm first produces the layout of each cluster independently, before producing a layout of the clusters, as it is shown in Fig. 2. Then, probe sets in a cluster are organized according to their similarity, and clusters are also organized according to the similarities of the clusters' members.

The whole map generated can be seen in Figs. 3, 4 and 5. In these Figures, each element (yellow and purple) represents a cluster of probe sets and they are organized according to the output of the QAPgrid. In Subheading 4, we analyze the results aided by the layout produced.

### 2.4 $CM_2$ Score

In order to investigate the validity of our methodology wherein we can be presented with large quantities of candidate probe sets (some of the clusters generated contain more than 2500), it was seen as appropriate to select ten representative probe sets from each cluster. In order to select the most appropriate probe sets, we have ranked the data by the $CM_2$ value (*see below*), and then selected the top ten.

For each of the probe sets, we compute the difference between expression values between the classes of cells (samples) we are interested in versus the rest of the samples. We call this measure $CM_2$. Probe sets which have the biggest difference in expression between cell types are ranked highest; these are the most likely candidates for biomarkers. Let's say we are looking for candidate biomarkers for samples of class 1 ($c_1$) and we refer to the rest of the samples as class 2 ($c_2$). Then, the $CM_2$ score is the difference between the average value of expression in Class 1 and Class 2 divided by the minimum range observed in Class 1 and Class 2. To compute the range of observed probe set expression values in a

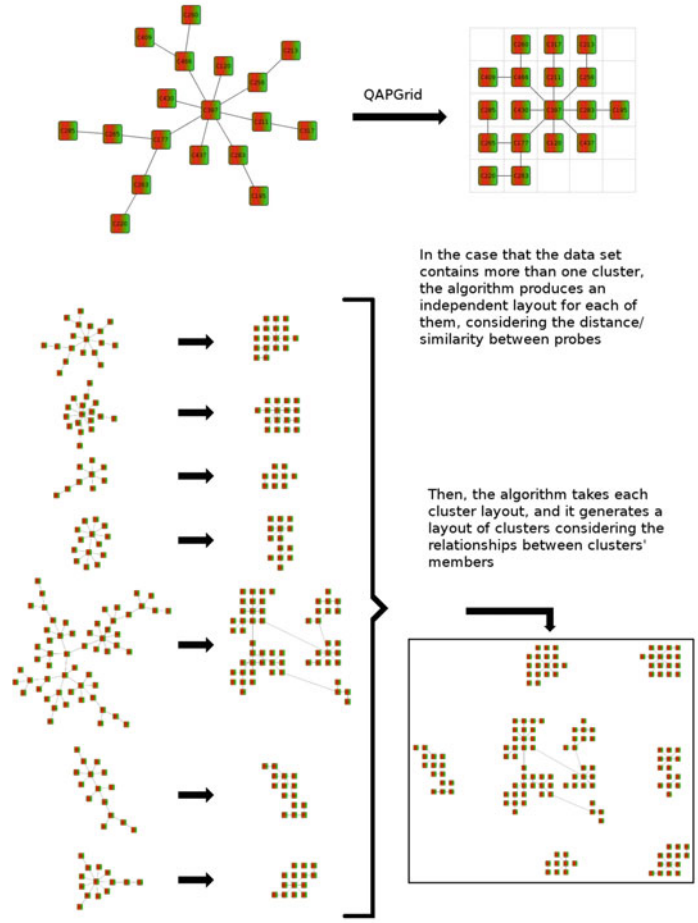



**Fig. 2** Schema of the QAPgrid algorithm. For each cluster of objects, the algorithm produces a layout of the objects. Then, the algorithm produces a second level layout of each of the clusters

class, we need to identify the maximum and subtract the minimum value observed in the class. As a consequence, the $CM_2$ score aims at identifying probe sets which are differentially expressed (on average). It does this by scaling the difference of averages by using the least variable class (the class that has the minimum range value).

## 3 Analysis

By comparing well described biomarker genes for cell type (*see* Tables 1, 2, and 3) to the clustering results, we can identify the

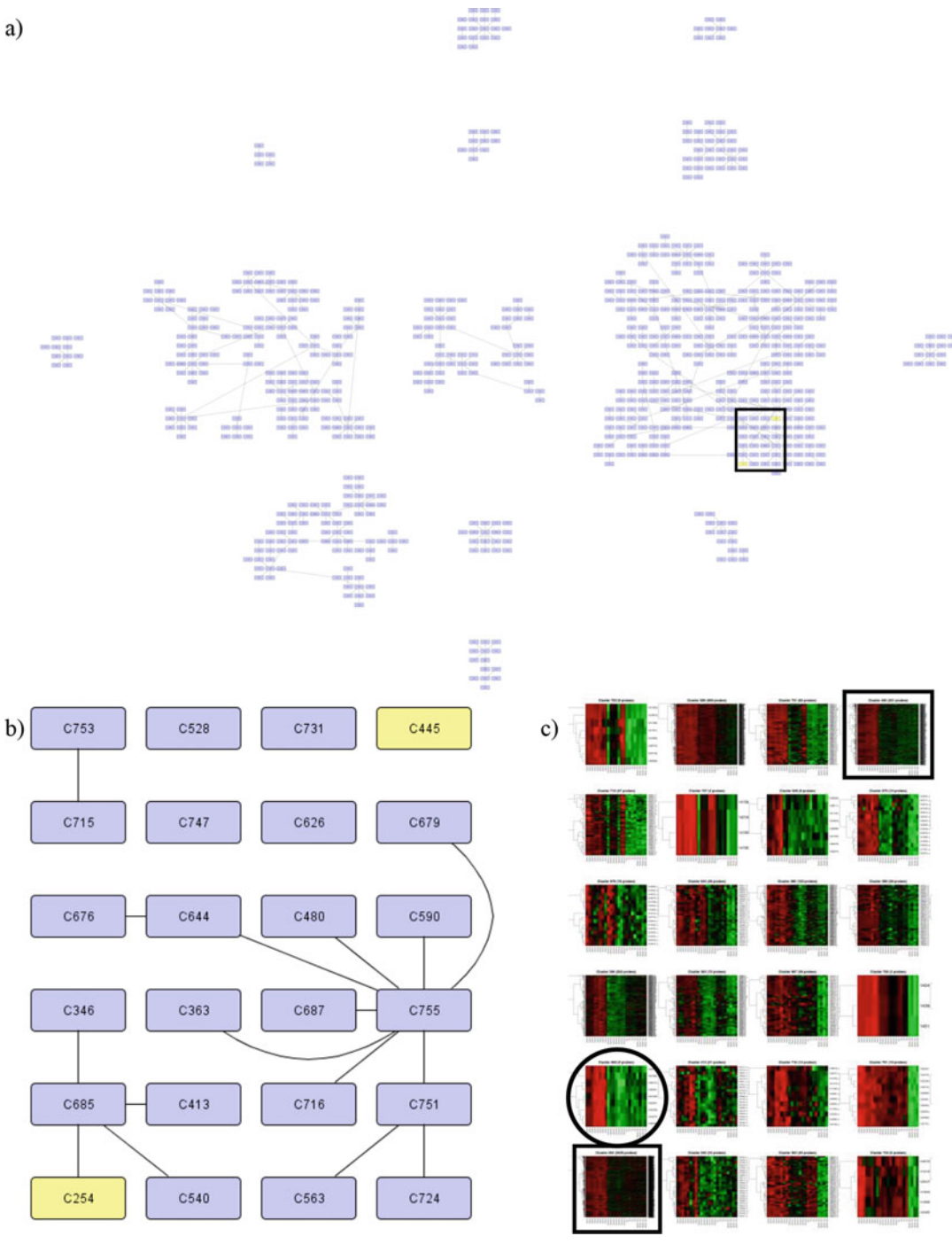



**Fig. 3** (**a**) QAPgrid generated map graphic of gene clusters generated by application of MSTkNN unsupervised clustering algorithm to microarray dataset of pooled cell types (astrocytes, neurons, and oligodendrocytes) isolated from mouse forebrains. The location of clusters 224 and 445 is indicated. (**b**) Location of clusters 224 and 445 at a larger magnification, where the location of neighboring nodes and edges in immediate region are displayed. (**c**) Location of clusters 224 and 445 (indicated by *squares*) at a larger magnification, where representative heat maps of neighboring clusters are also displayed. A candidate cluster for astrocyte markers is indicated by a *circle*

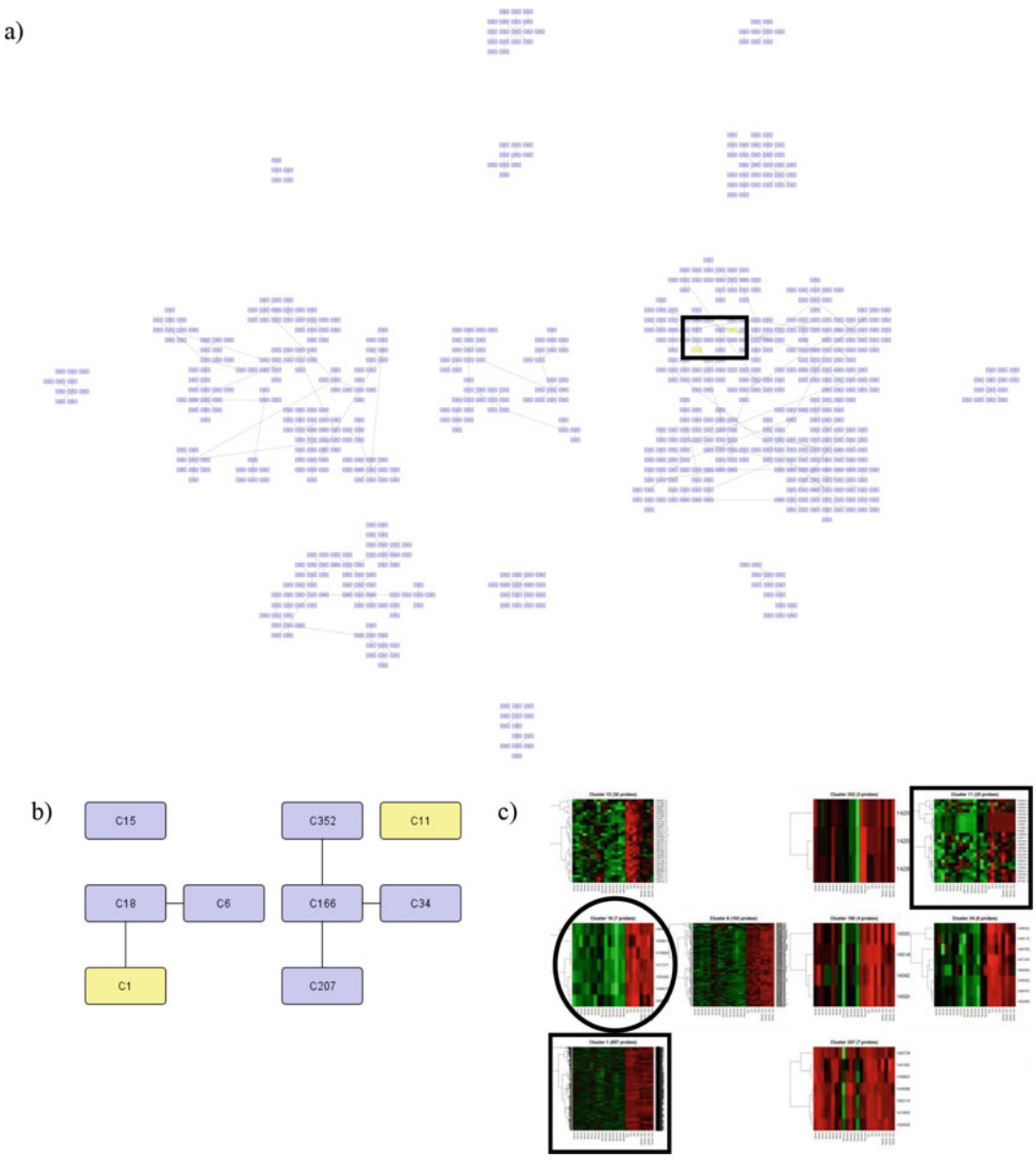



**Fig. 4** (**a**) QAP generated map graphic of gene clusters, as generated by application of MSTkNN unsupervised clustering algorithm to microarray dataset of pooled cell types (astrocytes, neurons, and oligodendrocytes) isolated from mouse forebrains. The location of clusters 1 and 11 is indicated. (**b**) Location of clusters 1 and 11 at a larger magnification, where the location of neighboring nodes and edges of the immediate region are displayed. (**c**) Location of clusters 1 and 11 (indicated by *squares*) at a larger magnification, where representative heat maps of neighboring clusters are also displayed. A candidate cluster for oligodendrocyte markers is indicated by a *circle*

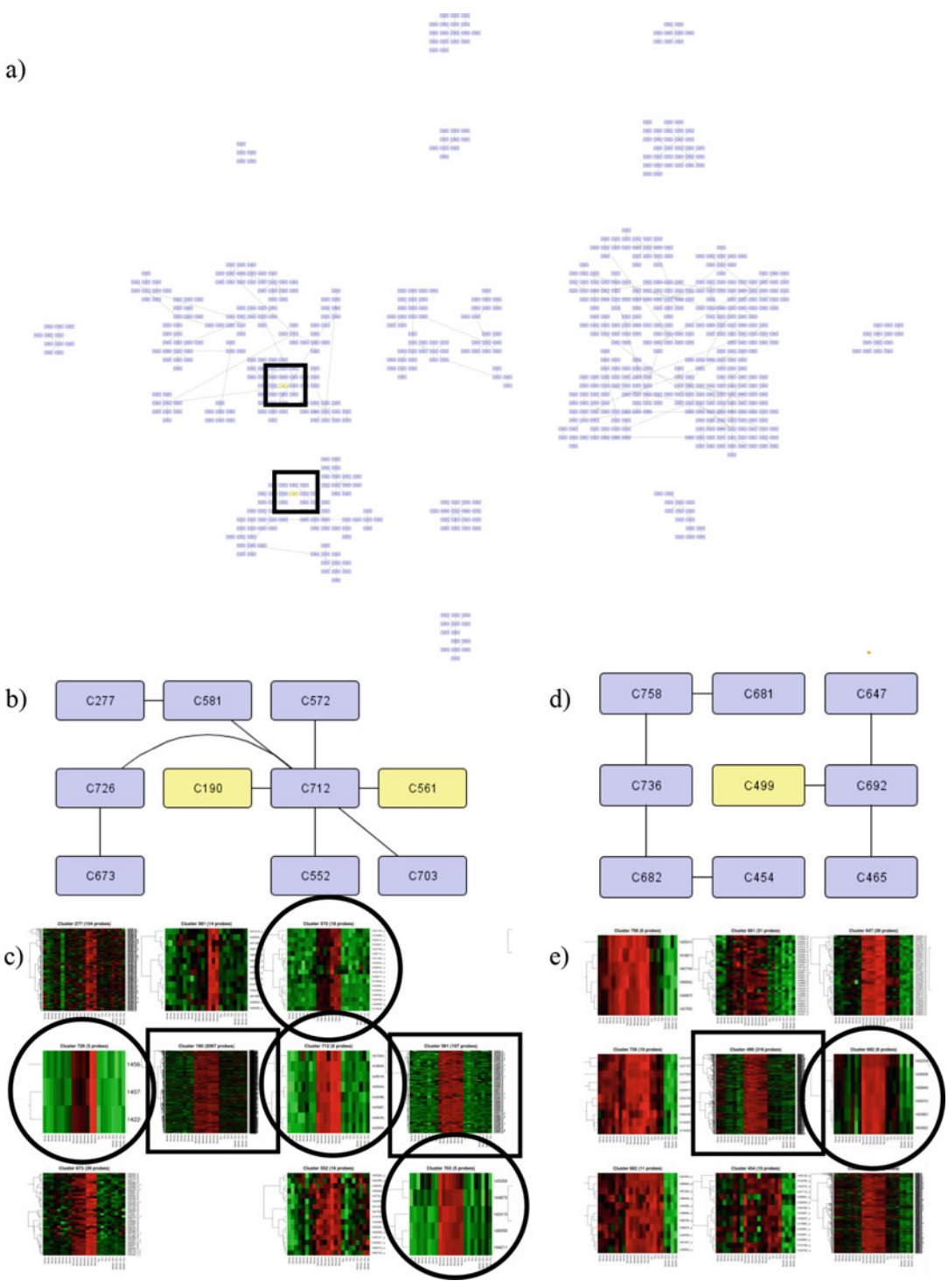



**Fig. 5** (**a**) QAP generated map graphic of gene clusters, as generated by application of MSTkNN unsupervised clustering algorithm to microarray datasets of pooled cell types (astrocytes, neurons, and oligodendrocytes) isolated from mouse forebrains. The location of clusters 190 and 499 are indicated. (**b**) Location of cluster 190 at a larger magnification, where the location of neighboring nodes and edges in immediate region are displayed. (**c**) Location of cluster 190 (*squared*) at a larger magnification, where representative heat maps of neighboring clusters are displayed. (***d**) Location of cluster 499 at a larger magnification, where the location of neighboring modes and edges are displayed. (**e**) Location of cluster 499 (*squared*) at a larger magnification, where representative heat maps of neighboring clusters are displayed. **Circles in (**c**) and (**e**) represent candidate clusters for neuron markers

extent to which the algorithm has successfully clustered together probe sets with similar expression patterns. In addition, we can track these well described biomarker genes in order to find candidate biomarkers within the same cluster.

By researching those probe sets with the highest $CM_2$ score within clusters which are brought to our attention, we are able to demonstrate that these methods are successful in the search for biomarkers; there are multiple genes whose expression profile is notoriously cell specific within these clusters (*see* Tables 5, 6, and 7). Just as significantly, the QAPgrid approach aids with the identification of further clusters wherein further biomarkers may be found, as clusters which consist of probe sets with similar expression profiles are mapped close together (*see* Figs. 4, 5, and 6).

When we locate well described biomarker genes for astrocytes within the clusters and the QAPgrid, we see that their representative probe sets are restricted mainly to just two clusters: 254 and 445 (*see* Table 1), both of which are within super cluster 1 (*see* Table 1 and Fig. 3). All seven of the biomarker genes have representative probe sets localized to cluster 254 (*see* Table 1 and Fig. 6). Upon closer inspection of these clusters (*see* Fig. 6), it is evident that the composite probe sets of both clusters share a similar pattern of expression across the cell types. In order to determine which of the 2520 (cluster 254) and 321 (cluster 445) probe sets may give best candidates for a cell type marker, we can apply the $CM_2$ equation in order to rank the candidate probe sets appropriately. Some of the results (the top ten probe sets) of this approach can be seen in Table 5, wherein a literature review summary indicates the likelihood that we have indeed found a candidate cell marker.

Furthermore, by examining the location of these clusters within the entire QAPgrid map, we can see that some of their neighbors also contain some probe sets whose expression pattern is cell specific, namely cluster 685 (*see* Fig. 3c).

In searching for the location of well described biomarker genes for oligodendrocytes, we can see that the majority of the probe sets are within clusters 1 and 11 (both of which are within super cluster 1) (*see* Fig. 4 and Table 2). All but one of the ten well described biomarker genes for oligodendrocytes have representative probe sets in the same cluster, namely cluster 1 (*see* Table 2). Analysis of the representative heat maps for these clusters reveals an interesting pattern within cluster 11 (consisting of 25 probe sets), wherein there is an observable group of probe sets whose expression profile is highly similar (*see* Fig. 7). When we sort all 25 of the composite probe sets in this cluster by their $\mathbf{CM_2}$ score, it is the constituent probe sets of this observed group (1433785_at, 1433532_a_at,

**Table 5**
**Top ten probe sets as result of ordering by $CM_2$ value for clusters 254 and 445, wherein the majority of well described biomarker genes for astrocytes are located, as a consequence of the application of MSTkNN unsupervised clustering algorithm to a GEM dataset of pooled cell types (astrocytes, neurons, and oligodendrocytes) isolated from mouse forebrains**

| Cluster | $CM_2$ ranking | Gene symbol and probe sets ID | Gene name | Summary of function/marker role in astrocyte |
|---|---|---|---|---|
| 254 | 1 | Vcam1<br>1448162_at | Vascular cell adhesion molecule 1 | Noted for abundant expression in astrocytes, particularly in response to multiple types of cytokine [43, 44] |
| | 2 | Pla2g7<br>1430700_a_at | Phospholipase A2, group VII (platelet-activating factor acetylhydrolase, plasma) | Noted for abundant expression in astrocytes [45] |
| | 3 | Slc1a3<br>1452031_at | Solute carrier family 1 (glial high affinity glutamate transporter), member 3 | Well described marker (*see* Tables 1 and 7) |
| | 4 | Ppp1r3c<br>1433691_at | Protein phosphatase 1, regulatory (inhibitor) subunit 3C | Noted for abundant expression in astrocytes, most likely due to its role in glycogen metabolism, a defining feature of the cell type [46, 47] |
| | 5 | Slc1a3<br>1443749_x_at | Solute carrier family 1 (glial high affinity glutamate transporter), member 3 | Well described marker (*see* Tables 1 and 7) |
| | 6 | Pdk4<br>1417273_at | Pyruvate dehydrogenase kinase, isoenzyme 4 | No previous evidence |
| | 7 | Slc15a2<br>1417600_at | Solute carrier family 15 (H+/peptide transporter), member 2 | No previous evidence |
| | 8 | Slc25a18<br>1428964_at | Solute carrier family 25 (mitochondrial carrier), member 18 | Noted for abundant expression in glial cells [48] |
| | 9 | Ttpa<br>1427284_a_at | Tocopherol (alpha) transfer protein | No previous evidence |
| | 10 | Rfx4<br>1436931_at | Regulatory factor X, 4 (influences HLA class II expression) | No previous evidence |

(continued)

**Table 5**
**(continued)**

| Cluster | $CM_2$ ranking | Gene symbol and probe sets ID | Gene name | Summary of function/marker role in astrocyte |
|---|---|---|---|---|
| 445 | 1 | Gm5089<br>438132_at | Predicted gene 5089 | No previous evidence |
| | 2 | Gli3<br>1455154_at | GLI-Kruppel family member GLI3 | No previous evidence |
| | 3 | Aldh1l1<br>1424400_a_at | Aldehyde dehydrogenase 1 family, member L1 | Noted for abundant expression in astrocytes ; used as antibody marker [49] |
| | 4 | Fgfr3<br>1425796_a_at | Fibroblast growth factor receptor 3 | Well described marker (*see* Tables 1 and 7) |
| | 5 | Sardh<br>1416662_at | Sarcosine dehydrogenase | No previous evidence |
| | 6 | Lonrf3<br>1436200_at | LON peptidase N-terminal domain and ring finger 3 | No previous evidence |
| | 7 | Lfng<br>1420643_at | LFNG O-fucosylpeptide 3-beta-N-acetylglucosaminyltransferase | Noted for abundant expression in glial cells compared to neuronal cells [50] |
| | 8 | Slc1a2<br>1438194_at | Solute carrier family 1 (glial high affinity glutamate transporter), member 2 | Well described marker (*see* Tables 1 and 7) |
| | 9 | Slc27a1<br>1422811_at | Solute carrier family 27 (fatty acid transporter), member 1 | No previous evidence |
| | 10 | Entpd2<br>1418259_a_at | Ectonucleoside triphosphate diphosphohydrolase 2 | Noted for abundant expression in astrocytes [51] |

**Table 6**
**Top ten probe sets as result of ordering by $CM_2$ value for clusters 1 and 11, wherein the majority of well described biomarker genes for oligodendrocytes are located, as a consequence of the application of MSTkNN unsupervised clustering algorithm to a GEM dataset of pooled cell types (astrocytes, neurons, and oligodendrocytes) isolated from mouse forebrains**

| Cluster | $CM_2$ ranking | Gene symbol and probe sets ID | Gene name | Summary of function/marker role in oligodendrocyte |
|---|---|---|---|---|
| 1 | 1 | Adamts4 1452595_at | A disintegrin-like and metallopeptidase (reprolysin type) with thrombospondin type 1 motif, 4 | No previous evidence |
| | 2 | Cldn11 1416003_at | Claudin 11 | Noted for abundant expression in oligodendrocytes, alias "oligodendrocyte specific protein"; used as antibody marker [52] |
| | 3 | AI314604 1442075_at | Expressed sequence AI314604 | No information |
| | 4 | Ugt8a 1419063_at | UDP galactosyltransferase 8A | Well described marker (*see* Tables 3 and 7) |
| | 5 | 1440813_s_at | Plexin B3 | Noted for abundant expression in oligodendrocytes [53] |
| | 6 | Gjc2 1450483_at | Gap junction protein, gamma 2 | Well described marker (*see* Tables 3 and 7) |
| | 7 | Mag 1460219_at | Myelin-associated glycoprotein | Well described marker (*see* Tables 3 and 7) |
| | 8 | Gm98 1439506_at | Predicted gene 98 | Noted for abundant expression in oligodendrocytes, alias "Myelin gene regulatory factor" [54] |
| | 9 | 9630013A20Rik 1439785_at | RIKEN cDNA 9630013A20 gene | No previous evidence |
| | 10 | Mobp 1421010_at | Myelin-associated oligodendrocytic basic protein | Well described marker (*see* Tables 3 and 7) |

(continued)

**Table 6**
**(continued)**

| Cluster | $CM_2$ ranking | Gene symbol and probe sets ID | Gene name | Summary of function/marker role in oligodendrocyte |
|---|---|---|---|---|
| 11 | 1 | Mobp 1433785_at | Myelin-associated oligodendrocytic basic protein | Well described marker (*see* Tables 3 and 7) |
| | 2 | Mbp 1419646_a_at | Myelin basic protein | Well described marker (*see* Tables 3 and 7) |
| | 3 | Mbp 1436201_x_at | Myelin basic protein | Well described marker (*see* Tables 3 and 7) |
| | 4 | Mbp 1454651_x_at | Myelin basic protein | Well described marker (*see* Tables 3 and 7) |
| | 5 | Mbp 1433532_a_at | Myelin basic protein | Well described marker (*see* Tables 3 and 7) |
| | 6 | Mbp 1456228_x_at | Myelin basic protein | Well described marker (*see* Tables 3 and 7) |
| | 7 | Bcl7a 1459416_at | B cell CLL/lymphoma 7A | No previous evidence |
| | 8 | A930024E05Rik 1440206_at | RIKEN cDNA A930024E05 gene | No previous evidence |
| | 9 | Repin1 1457387_at | Replication initiator 1 | Noted for abundant expression in non-neuronal cells [55] |
| | 10 | Atf6 1435444_at | Activating transcription factor 6 | No previous evidence |

**Table 7**
**Top ten probe sets as result of ordering by $CM_2$ value for clusters 561, 499, and 190, wherein the majority of well described neuronal biomarker genes are located, as a consequence of the application of MST-*k*NN unsupervised clustering algorithm to a GEM dataset of pooled cell types (astrocytes, neurons, and oligodendrocytes) isolated from mouse forebrains**

| Cluster | $CM_2$ ranking | Gene symbol | Gene name | Summary of function/marker role in neurons |
|---|---|---|---|---|
| 561 | 1 | D130043K22Rik 1443327_at | SPHK1 interactor, AKAP domain containing | Hypothetical protein |
| | 2 | Nefl 1426255_at | Mitogen-activated protein kinase 8 interacting protein 2 | Well described marker (*see* Tables 3 and 7) |
| | 3 | Tmod3 1455708_at | DNA segment, Chr 3, Brigham and Women's Genetics 0562 expressed | No previous evidence |
| | 4 | Kcns2 1457325_at | ATPase, Na+/K+ transporting, alpha 3 polypeptide | No previous evidence |
| | 5 | Slc6a7 1455469_at | Phosphoglucomutase 2-like 1 | Noted for abundant expression in neurons (role in neurotransmitter transport) [56] |
| | 6 | Dync1i1 1416361_a_at | Dynein cytoplasmic 1 intermediate chain 1 | Noted for neuronal specific expression [57–61] |
| | 7 | Stxbp1 1420505_a_at | Calcium/calmodulin-dependent protein kinase IV | Noted for neuron-specific expression [62] |
| | 8 | Rusc1 1438017_at | Neuron-specific gene family member 2 | Noted for neuron-specific expression [63] |
| | 9 | Epb4.9 1460223_a_at | Transmembrane protein 130 | No previous evidence |
| | 10 | Rgs8 1453060_at | Solute carrier family 12, member 5 | Noted for neuron-specific expression [64] |

(continued)

**Table 7 (continued)**

| Cluster | $CM_2$ ranking | Gene symbol | Gene name | Summary of function/marker role in neurons |
|---|---|---|---|---|
| 499 | 1 | Sphkap 1454926_at | SPHK1 interactor, AKAP domain containing | No previous evidence |
| | 2 | Mapk8ip2 1449225_a_at | Mitogen-activated protein kinase 8 interacting protein 2 | No previous evidence |
| | 3 | D3Bwg0562e 1427247_at | DNA segment, Chr 3, Brigham and Women's Genetics 0562 expressed | Putative amino acid sequence, consistent with identity of PRG1, a neuron-specific calmodulin-binding protein |
| | 4 | Atp1a3 1427481_a_at | ATPase, Na+/K+ transporting, alpha 3 polypeptide | Noted for neuron-specific expression [65] |
| | 5 | Pgm2l1 1456478_at | Phosphoglucomutase 2-like 1 | No previous evidence |
| | 6 | Myt1l 1421175_at | Myelin transcription factor 1-like | Noted for abundant expression in neurons in comparison to glial cells [66] |
| | 7 | Camk4 1439843_at | Calcium/calmodulin-dependent protein kinase IV | Noted for abundant expression in neurons [67] |
| | 8 | Nsg2 1416107_at | Neuron-specific gene family member 2 | Alias "neuron-specific gene member 2" |
| | 9 | Tmem130 1455148_at | Transmembrane protein 130 | No previous evidence |
| | 10 | Slc12a5 1451674_at | Solute carrier family 12, member 5 | Well described marker (*see* Tables 3 and 7) |

| | | | | |
|---|---|---|---|---|
| 190 | 1 | Klc1<br>1417005_at | Kinesin light chain 1 | No previous evidence |
| | 2 | Celf4<br>1426930_at | CUGBP, Elav-like family member 4 | Noted for abundant expression in neurons [68] |
| | 3 | Rbfox1<br>1418314_a_at | RNA binding protein, fox-1 homolog (*C. elegans*) 1 | Noted for abundant expression in neurons [69] |
| | 4 | Gm16532<br>1459717_at | Predicted gene, 16532 | Predicted gene |
| | 5 | Myt1l<br>1436483_at | Myelin transcription factor 1-like | Noted for abundant expression in neurons in comparison to glial cells [66]; used for cellular programming to induce differentiation from fibroblasts to neurons [70] |
| | 6 | Rab15<br>1417829_a_at | RAB15, member RAS oncogene family | *See above*<br>Associated with neuronal differentiation [71] |
| | 7 | Celf4<br>1452240_at | CUGBP, Elav-like family member 4 | Noted for abundant expression in neurons [68]<br>*See above* |
| | 8 | Rbfox3<br>1436450_at | RNA binding protein, fox-1 homolog (*C. elegans*) 3 | Noted for neuron-specific expression; used as antibody marker [72] |
| | 9 | Zrsr2<br>1455727_at | zinc finger (CCCH type), RNA binding motif and serine/arginine rich 2 | No previous evidence |
| | 10 | Tac1<br>1416783_at | Tachykinin 1 | Noted for neuron-specific expression [73] |

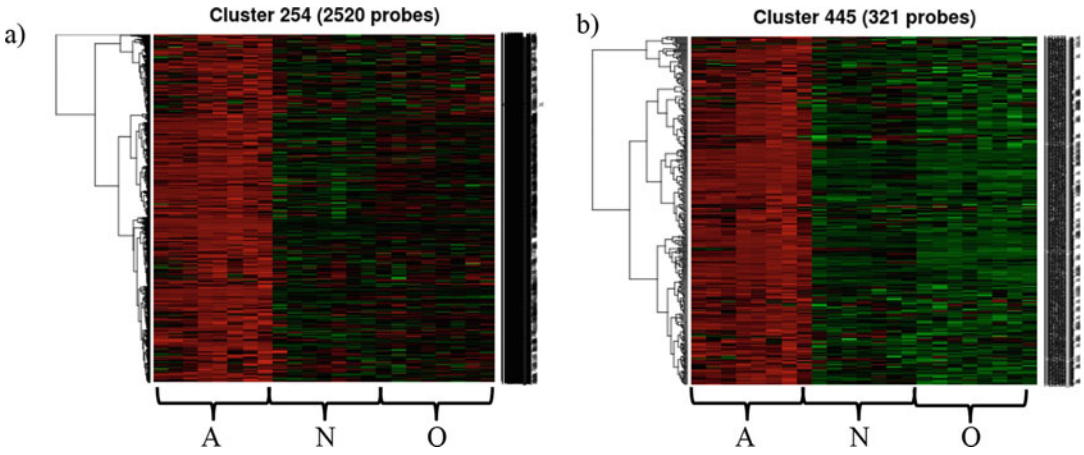



**Fig. 6** (**a**) Cluster 254 and (**b**) Cluster 445; generated by application of MSTkNN unsupervised clustering algorithm to microarray dataset of pooled cell types; astrocytes (A), neurons (N) and oligodendrocyte (O) isolated from mouse forebrains. Both clusters contain representative probe sets for described astrocyte markers (*see* Table 1). (**a**) Cluster 254; Slc1a2, Gjb6, Gfap, Slc1a3, Aqp4, Fgfr3. (**b**) Cluster 445; Slcla2, Slcla3, Fgfr3

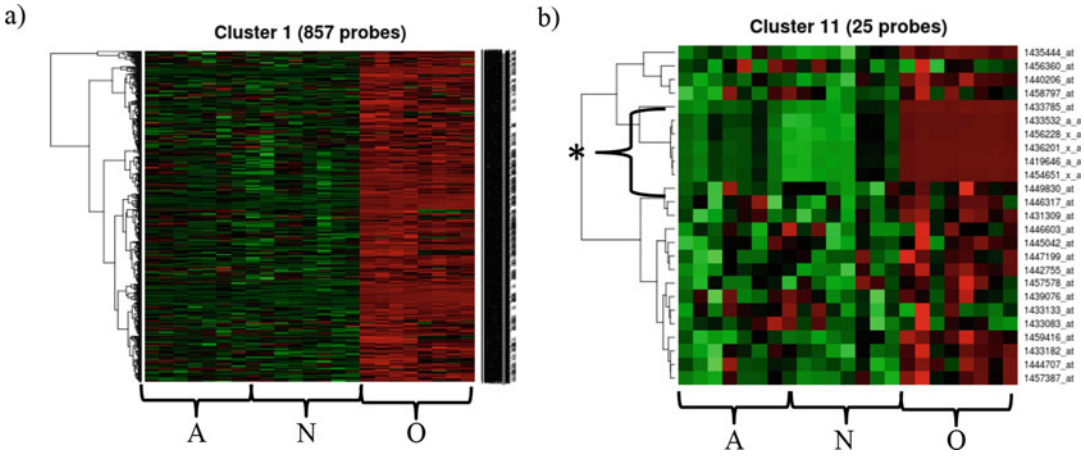



**Fig. 7** (**a**) Cluster 1 and (**b**) Cluster 11 generated by application of MSTkNN unsupervised clustering algorithm to microarray dataset of pooled cell types; astrocytes (A), neurons (N) and oligodendrocyte (O) isolated from mouse forebrains. Both clusters contain representative probe sets for described oligodendrocyte markers (*see* Table 1). (**a**) Cluster 1; Sox10, Pdgfra, Gjc2, Mbp, Mog, Ugt8a, Gal3st1, Mag. (**b**) Cluster 11; Mbp, Mobp. A region of probes sharing highly similar expression is indicated with *asterisk*

1436201_x_at, 1419646_a_at, 1454651_x_at, 1456228_x_at) which have the highest $\mathbf{CM_2}$ ranking (*see* Table 5). These probe sets map to the well described biomarker genes, myelin-associated oligodendrocytic basic protein and myelin basic protein, herein proving the efficacy of our methods wherein application of $\mathbf{CM_2}$

can assist with identifying likely candidates for biomarkers. Indeed, of the 20 probe sets selected by $\mathbf{CM_2}$ filtering, 14 hold previous evidence for cell-specific expression patterns (*see* Table 5).

Looking at the location of these clusters of interest in the QAPgrid, we can identify yet another candidate of interest with highly specific gene expression profiles, namely cluster 18 (*see* Fig. 4c).

Though the probe sets for well described neuronal biomarker genes are not as concentrated into the same clusters as with the other cell types under investigation (Fig. 5), when we look at the other probe sets which are within these particular clusters wherein the majority of the biomarkers lie (190, 499, and 561), we see that there are still plenty of candidate cell markers there. This indicates the usefulness of our methods (*see* Table 7). In addition, the QAPgrid shows that there are multiple neighboring clusters depicting cell-specific expression (*see* Fig. 8).

## 4 Conclusions

We present an integrated approach for clustering, selection, and visualization of patterns of gene expression that constitute molecular signatures of cell-specific transcription. We validate its usefulness by employing a dataset that allowed us to identify both known markers of neuronal, oligodendrocyte, and astrocytic cell types as well as others that warrant further investigation. Our study suggests the following putative novel biomarkers of cell type: for astrocytes, pdk4, slc15a2, Ttpa, Rfx4, Gli3, Sardh, Lonrf3, and Slc27a1; for oligodendrocytes, Adamts4, Bcl7a, and Atf6; and for neurons, Tmod3, Kcns2, Dync1i1, Mapk8ip2, Sphkap, Epb4.9, Zrsr2, Pmg211, Tmem130, Klc1. The validation of these other putative biomarkers of cell type requires wet lab investigations. We believe these will soon follow as some of these genes have already attracted the attention of researchers working in neurodegenerative diseases. For instance Klc1 seems to have a role in amyloid-beta accumulation and intracellular trafficking, thus linking it to Alzheimer's disease [15–17]. We thus expect that researchers would be motivated to identify if all populations of neurons express Klc1 and if, in addition, there are anatomical observable differences. Levels of KLC1 have been observed to be reduced in the frontal cortex, but not in the cerebellar cortex, of Alzheimer's disease patients [18]. Our method allows a comprehensive analysis of all major groups of gene expression patterns across three different cell types and provides a basis for an investigation on disruption of these co-expression patterns in neurodegenerative diseases in model organisms.

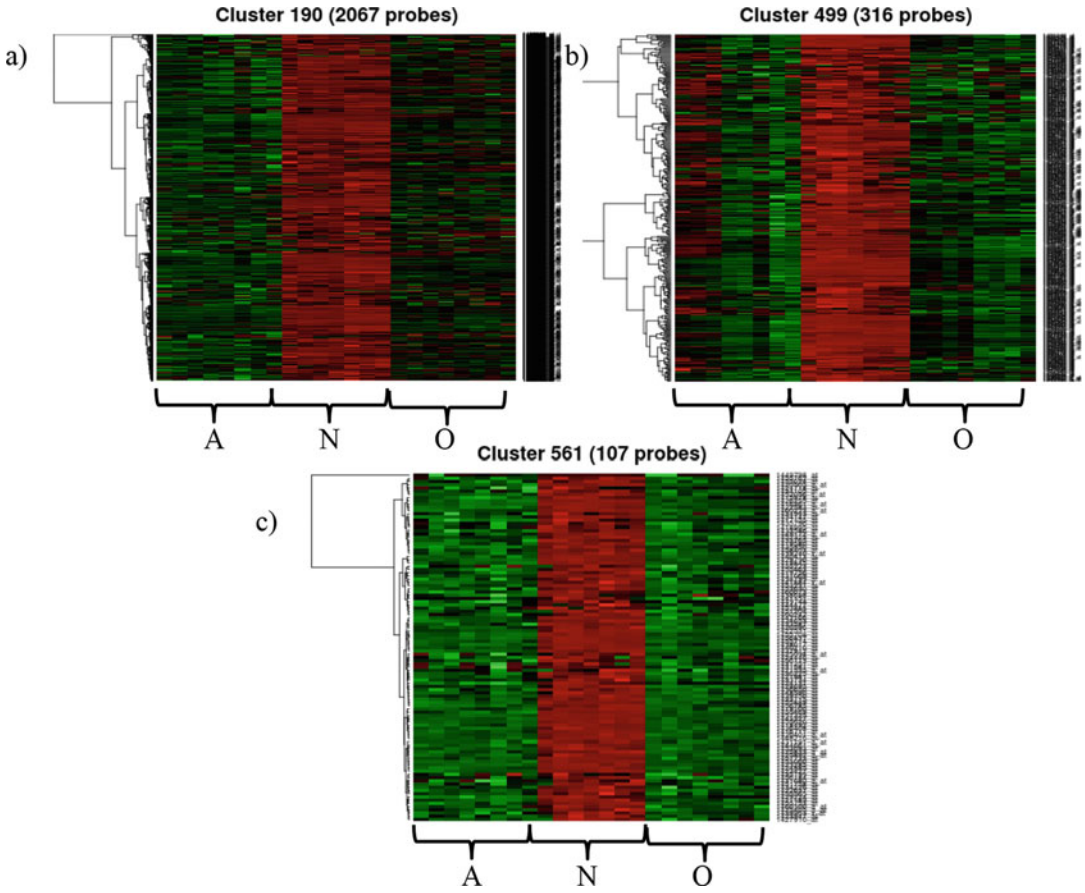



**Fig. 8** (**a**) Cluster 190, (**b**) Cluster 499 and (**c**) Cluster 561; generated by application of MSTkNN supervised clustering algorithm to dataset of pooled cell types; astrocytes (A), neurons (N) and oligodendrocyte (O) isolated from mouse forebrains. Both clusters contain representative probe sets for described neuronal markers (*see* Table 1). (**a**) Cluster 190; Nefl, Gabra1, Syt1, Snap25m, Sv2b. (**b**) Cluster 11; Syt1, Slcl25a. (**c**) Cluster 561; Nefl, Syt1

## Acknowledgments

The authors would like to thank Prof. Manuel Graeber for his suggestion about the datasets to be explored.

### References

1. Schena M, Shalon D, Davis RW, Brown PO (1995) Quantitative monitoring of gene expression patterns with a complementary DNA microarray. Science 270:467–470
2. Inostroza-Ponta M, Berretta R, Moscato P (2011) QAPgrid: a two level QAP-based approach for large-scale data analysis and visualization. PLoS One 6:e14468
3. Clark MB, Johnston RL, Inostroza-Ponta M, Fox AH, Fortini E et al (2012) Genome-wide analysis of long noncoding RNA stability. Genome Res 22:885–898
4. Capp A, Inostroza-Ponta M, Bill D, Moscato P, Lai C et al (2009) Is there more than one proctitis syndrome? A revisitation using data from the TROG 96.01 trial. Radiother Oncol 90:400–407

5. Geschwind DH, Konopka G (2009) Neuroscience in the era of functional genomics and systems biology. Nature 461:908–915
6. Okaty BW, Sugino K, Nelson SB (2011) Cell type-specific transcriptomics in the brain. J Neurosci 31:6939–6943
7. Okaty BW, Sugino K, Nelson SB (2011) A quantitative comparison of cell-type-specific microarray gene expression profiling methods in the mouse brain. PLoS One 6:e16493
8. Cahoy JD, Emery B, Kaushal A, Foo LC, Zamanian JL et al (2008) A transcriptome database for astrocytes, neurons, and oligodendrocytes: a new resource for understanding brain development and function. J Neurosci 28:264–278
9. Stalteri MA, Harrison AP (2007) Interpretation of multiple probe sets mapping to the same gene in Affymetrix GeneChips. BMC Bioinformatics 8:13
10. Upton GJ, Sanchez-Graillet O, Rowsell J, Arteaga-Salas JM, Graham NS et al (2009) On the causes of outliers in Affymetrix GeneChip data. Brief Funct Genomic Proteomic 8:199–212
11. Inostroza-Ponta M, Mendes A, Berretta R, Moscato P (2007) An integrated QAP-based approach to visualize patterns of gene expression similarity. Prog Artif Life Proc 4828:156–167
12. Inostroza-Ponta M, Berretta R, Mendes A, Moscato P (2006) An automatic graph layout procedure to visualize correlated data. Artif Intell Theory Pract 217:179–188
13. Moscato P, Norman MG (1992) A "Memetic" approach for the traveling salesman problem implementation of a computational ecology for combinatorial optimization on message-passing systems. Parallel Comput Transp Appl Pts 1 and 2 28:177–186
14. Moscato P, Mendes A, Berretta R (2007) Benchmarking a memetic algorithm for ordering microarray data. Biosystems 88:56–75
15. Morihara T, Hayashi N, Yokokoji M, Akatsu H, Silverman MA et al (2014) Transcriptome analysis of distinct mouse strains reveals kinesin light chain-1 splicing as an amyloid-beta accumulation modifier. Proc Natl Acad Sci U S A 111:2638–2643
16. Killian RL, Flippin JD, Herrera CM, Almenar-Queralt A, Goldstein LS (2012) Kinesin light chain 1 suppression impairs human embryonic stem cell neural differentiation and amyloid precursor protein metabolism. PLoS One 7: e29755
17. Szpankowski L, Encalada SE, Goldstein LS (2012) Subpixel colocalization reveals amyloid precursor protein-dependent kinesin-1 and dynein association with axonal vesicles. Proc Natl Acad Sci U S A 109:8582–8587
18. Morel M, Heraud C, Nicaise C, Suain V, Brion JP (2012) Levels of kinesin light chain and dynein intermediate chain are reduced in the frontal cortex in Alzheimer's disease: implications for axoplasmic transport. Acta Neuropathol 123:71–84
19. Trojanowski JQ, Walkenstein N, Lee VM (1986) Expression of neurofilament subunits in neurons of the central and peripheral nervous system: an immunohistochemical study with monoclonal antibodies. J Neurosci 6:650–660
20. Riddick G, Fine HA (2011) Integration and analysis of genome-scale data from gliomas. Nat Rev Neurol 7:439–450
21. Geddes JW, Hess EJ, Hart RA, Kesslak JP, Cotman CW et al (1990) Lesions of hippocampal circuitry define synaptosomal-associated protein-25 (SNAP-25) as a novel presynaptic marker. Neuroscience 38:515–525
22. Selyanko AA, Hadley JK, Wood IC, Abogadie FC, Delmas P et al (1999) Two types of K(+) channel subunit, Erg1 and KCNQ2/3, contribute to the M-like current in a mammalian neuronal cell. J Neurosci 19:7742–7756
23. Janz R, Goda Y, Geppert M, Missler M, Sudhof TC (1999) SV2A and SV2B function as redundant Ca2+ regulators in neurotransmitter release. Neuron 24:1003–1016
24. Aguirre A, Dupree JL, Mangin JM, Gallo V (2007) A functional role for EGFR signaling in myelination and remyelination. Nat Neurosci 10:990–1002
25. Komitova M, Eriksson PS (2004) Sox-2 is expressed by neural progenitors and astroglia in the adult rat brain. Neurosci Lett 369:24–27
26. Baer K, Eriksson PS, Faull RL, Rees MI, Curtis MA (2007) Sox-2 is expressed by glial and progenitor cells and Pax-6 is expressed by neuroblasts in the human subventricular zone. Exp Neurol 204:828–831
27. Spassky N, Olivier C, Perez-Villegas E, Goujet-Zalc C, Martinez S et al (2000) Single or multiple oligodendroglial lineages: a controversy. Glia 29:143–148
28. Menichella DM, Goodenough DA, Sirkowski E, Scherer SS, Paul DL (2003) Connexins are critical for normal myelination in the CNS. J Neurosci 23:5963–5973
29. Brunner C, Lassmann H, Waehneldt TV, Matthieu JM, Linington C (1989) Differential ultrastructural localization of myelin basic protein, myelin/oligodendroglial glycoprotein, and 2′,3′-cyclic nucleotide 3′-phosphodiesterase in

the CNS of adult rats. J Neurochem 52:296–304

30. Baumann N, Pham-Dinh D (2001) Biology of oligodendrocyte and myelin in the mammalian central nervous system. Physiol Rev 81:871–927
31. Schulte S, Stoffel W (1993) Ceramide UDP galactosyltransferase from myelinating rat brain: purification, cloning, and expression. Proc Natl Acad Sci U S A 90:10265–10269
32. Hayashi A, Kaneko N, Tomihira C, Baba H (2013) Sulfatide decrease in myelin influences formation of the paranodal axo-glial junction and conduction velocity in the sciatic nerve. Glia 61:466–474
33. Yamamoto Y, Mizuno R, Nishimura T, Ogawa Y, Yoshikawa H et al (1994) Cloning and expression of myelin-associated oligodendrocytic basic protein. A novel basic protein constituting the central nervous system myelin. J Biol Chem 269:31725–31730
34. Schaeren-Wiemers N, Valenzuela DM, Frank M, Schwab ME (1995) Characterization of a rat gene, rMAL, encoding a protein with four hydrophobic domains in central and peripheral myelin. J Neurosci 15:5753–5764
35. Rothstein JD, Martin L, Levey AI, Dykes-Hoberg M, Jin L et al (1994) Localization of neuronal and glial glutamate transporters. Neuron 13:713–725
36. Dermietzel R, Gao Y, Scemes E, Vieira D, Urban M et al (2000) Connexin43 null mice reveal that astrocytes express multiple connexins. Brain Res Brain Res Rev 32:45–56
37. Pekny M, Nilsson M (2005) Astrocyte activation and reactive gliosis. Glia 50:427–434
38. Eng LF, Ghirnikar RS, Lee YL (2000) Glial fibrillary acidic protein: GFAP-thirty-one years (1969–2000). Neurochem Res 25:1439–1451
39. Kondo K, Hashimoto H, Kitanaka J, Sawada M, Suzumura A et al (1995) Expression of glutamate transporters in cultured glial cells. Neurosci Lett 188:140–142
40. Nagelhus EA, Veruki ML, Torp R, Haug FM, Laake JH et al (1998) Aquaporin-4 water channel protein in the rat retina and optic nerve: polarized expression in Muller cells and fibrous astrocytes. J Neurosci 18:2506–2519
41. Staugaitis SM, Zerlin M, Hawkes R, Levine JM, Goldman JE (2001) Aldolase C/zebrin II expression in the neonatal rat forebrain reveals cellular heterogeneity within the subventricular zone and early astrocyte differentiation. J Neurosci 21:6195–6205
42. Balaci L, Presta M, Ennas MG, Dell'Era P, Sogos V et al (1994) Differential expression of fibroblast growth factor receptors by human neurones, astrocytes and microglia. Neuroreport 6:197–200
43. Gimenez MA, Sim JE, Russell JH (2004) TNFR1-dependent VCAM-1 expression by astrocytes exposes the CNS to destructive inflammation. J Neuroimmunol 151:116–125
44. Rosenman SJ, Shrikant P, Dubb L, Benveniste EN, Ransohoff RM (1995) Cytokine-induced expression of vascular cell adhesion molecule-1 (VCAM-1) by astrocytes and astrocytoma cell lines. J Immunol 154:1888–1899
45. Bachoo RM, Kim RS, Ligon KL, Maher EA, Brennan C et al (2004) Molecular diversity of astrocytes with implications for neurological disorders. Proc Natl Acad Sci U S A 101:8384–8389
46. Allaman I, Pellerin L, Magistretti PJ (2000) Protein targeting to glycogen mRNA expression is stimulated by noradrenaline in mouse cortical astrocytes. Glia 30:382–391
47. Brunet JF, Allaman I, Magistretti PJ, Pellerin L (2010) Glycogen metabolism as a marker of astrocyte differentiation. J Cereb Blood Flow Metab 30:51–55
48. Torp R, Danbolt NC, Babaie E, Bjoras M, Seeberg E et al (1994) Differential expression of two glial glutamate transporters in the rat brain: an in situ hybridization study. Eur J Neurosci 6:936–942
49. Yang Y, Vidensky S, Jin L, Jie C, Lorenzini I et al (2011) Molecular comparison of GLT1+ and ALDH1L1+ astrocytes in vivo in astroglial reporter mice. Glia 59:200–207
50. Schwarting GA, Gridley T, Henion TR (2007) Notch1 expression and ligand interactions in progenitor cells of the mouse olfactory epithelium. J Mol Histol 38:543–553
51. Wink MR, Braganhol E, Tamajusuku AS, Lenz G, Zerbini LF et al (2006) Nucleoside triphosphate diphosphohydrolase-2 (NTPDase2/CD39L1) is the dominant ectonucleotidase expressed by rat astrocytes. Neuroscience 138:421–432
52. Sallis ES, Mazzanti CM, Mazzanti A, Pereira LA, Arroteia KF et al (2006) OSP-Immunofluorescent remyelinating oligodendrocytes in the brainstem of toxically-demyelinated Wistar rats. Arq Neuropsiquiatr 64:240–244
53. Worzfeld T, Puschel AW, Offermanns S, Kuner R (2004) Plexin-B family members demonstrate non-redundant expression patterns in the developing mouse nervous system: an anatomical basis for morphogenetic effects of Sema4D during development. Eur J Neurosci 19:2622–2632

54. Koenning M, Jackson S, Hay CM, Faux C, Kilpatrick TJ et al (2012) Myelin gene regulatory factor is required for maintenance of myelin and mature oligodendrocyte identity in the adult CNS. J Neurosci 32:12528–12542
55. Kim MY, Jeong BC, Lee JH, Kee HJ, Kook H et al (2006) A repressor complex, AP4 transcription factor and geminin, negatively regulates expression of target genes in nonneuronal cells. Proc Natl Acad Sci U S A 103:13074–13079
56. Velaz-Faircloth M, Guadano-Ferraz A, Henzi VA, Fremeau RT Jr (1995) Mammalian brain-specific L-proline transporter. Neuronal localization of mRNA and enrichment of transporter protein in synaptic plasma membranes. J Biol Chem 270:15755–15761
57. Zhang J, Twelvetrees AE, Lazarus JE, Blasier KR, Yao X et al (2013) Establishing a novel knock-in mouse line for studying neuronal cytoplasmic dynein under normal and pathologic conditions. Cytoskeleton (Hoboken) 70:215–227
58. Kuta A, Deng W, Morsi El-Kadi A, Banks GT, Hafezparast M et al (2010) Mouse cytoplasmic dynein intermediate chains: identification of new isoforms, alternative splicing and tissue distribution of transcripts. PLoS One 5: e11682
59. Salata MW, Dillman JF 3rd, Lye RJ, Pfister KK (2001) Growth factor regulation of cytoplasmic dynein intermediate chain subunit expression preceding neurite extension. J Neurosci Res 65:408–416
60. Pfister KK, Salata MW, Dillman JF 3rd, Torre E, Lye RJ (1996) Identification and developmental regulation of a neuron-specific subunit of cytoplasmic dynein. Mol Biol Cell 7:331–343
61. Pfister KK, Salata MW, Dillman JF 3rd, Vaughan KT, Vallee RB et al (1996) Differential expression and phosphorylation of the 74-kDa intermediate chains of cytoplasmic dynein in cultured neurons and glia. J Biol Chem 271:1687–1694
62. Bhaskar K, Shareef MM, Sharma VM, Shetty AP, Ramamohan Y et al (2004) Co-purification and localization of Munc18-1 (p67) and Cdk5 with neuronal cytoskeletal proteins. Neurochem Int 44:35–44
63. MacDonald JI, Dietrich A, Gamble S, Hryciw T, Grant RI et al (2012) Nesca, a novel neuronal adapter protein, links the molecular motor kinesin with the pre-synaptic membrane protein, syntaxin-1, in hippocampal neurons. J Neurochem 121:861–880
64. Saitoh O, Masuho I, Itoh M, Abe H, Komori K et al (2003) Distribution of regulator of G protein signaling 8 (RGS8) protein in the cerebellum. Cerebellum 2:154–160
65. Richards KS, Bommert K, Szabo G, Miles R (2007) Differential expression of Na+/K + -ATPase alpha- subunits in mouse hippocampal interneurones and pyramidal cells. J Physiol 585:491–505
66. Kim JG, Armstrong RC, v Agoston D, Robinsky A, Wiese C et al (1997) Myelin transcription factor 1 (Myt1) of the oligodendrocyte lineage, along with a closely related CCHC zinc finger, is expressed in developing neurons in the mammalian central nervous system. J Neurosci Res 50:272–290
67. Frangakis MV, Chatila T, Wood ER, Sahyoun N (1991) Expression of a neuronal Ca2+/calmodulin- dependent protein kinase, CaM kinase-Gr, in rat thymus. J Biol Chem 266:17592–17596
68. Wagnon JL, Mahaffey CL, Sun W, Yang Y, Chao HT et al (2011) Etiology of a genetically complex seizure disorder in Celf4 mutant mice. Genes Brain Behav 10:765–777
69. Kim KK, Kim YC, Adelstein RS, Kawamoto S (2011) Fox-3 and PSF interact to activate neural cell-specific alternative splicing. Nucleic Acids Res 39:3064–3078
70. Ambasudhan R, Talantova M, Coleman R, Yuan X, Zhu S et al (2011) Direct reprogramming of adult human fibroblasts to functional neurons under defined conditions. Cell Stem Cell 9:113–118
71. Pham TV, Hartomo TB, Lee MJ, Hasegawa D, Ishida T et al (2012) Rab15 alternative splicing is altered in spheres of neuroblastoma cells. Oncol Rep 27:2045–2049
72. Kim KK, Adelstein RS, Kawamoto S (2009) Identification of neuronal nuclei (NeuN) as Fox-3, a new member of the Fox-1 gene family of splicing factors. J Biol Chem 284:31052–31061
73. Mar L, Yang FC, Ma Q (2012) Genetic marking and characterization of Tac2-expressing neurons in the central and peripheral nervous system. Mol Brain 5:3

# Chapter 17

# Computer-Aided Breast Cancer Diagnosis with Optimal Feature Sets: Reduction Rules and Optimization Techniques

**Luke Mathieson, Alexandre Mendes, John Marsden, Jeffrey Pond, and Pablo Moscato**

## Abstract

This chapter introduces a new method for knowledge extraction from databases for the purpose of finding a discriminative set of features that is also a robust set for within-class classification. Our method is generic and we introduce it here in the field of breast cancer diagnosis from digital mammography data. The mathematical formalism is based on a generalization of the $k$-Feature Set problem called $(\alpha, \beta)$-$k$-Feature Set problem, introduced by Cotta and Moscato (J Comput Syst Sci 67(4):686–690, 2003). This method proceeds in two steps: first, an optimal $(\alpha, \beta)$-$k$-feature set of minimum cardinality is identified and then, a set of classification rules using these features is obtained. We obtain the $(\alpha, \beta)$-$k$-feature set in two phases; first a series of extremely powerful reduction techniques, which do not lose the optimal solution, are employed; and second, a metaheuristic search to identify the remaining features to be considered or disregarded. Two algorithms were tested with a public domain digital mammography dataset composed of 71 malignant and 75 benign cases. Based on the results provided by the algorithms, we obtain classification rules that employ only a subset of these features.

**Key words** Safe data reduction, Combinatorial optimization, Minimum feature set, Breast cancer diagnostics, Memetic algorithms

## 1 Introduction

Breast cancer is one of the most common types of cancer in women all over the world, for which the most effective detection method is screening mammography analysis. Unfortunately, the method is prone to misjudgments and subjective opinion. Diagnostic error rate ranges vary but have been reported as ranging between 20 % and 43 % [1], and out of all the biopsies performed in suspicious mammograms, between 70 % and 89 % of them will be found benign [2].

This chapter introduces a new approach to improve computer-aided diagnostic methods by selecting, from a given data set, a subset of the features that would allow the identification of a lesion

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_17, 

as an intraductal carcinoma or otherwise, and the different problem of deciding whether a lesion is malignant or benign. The technique is generic and can be applied to some other knowledge extraction methods from available databases [3]. The breast cancer diagnosis issue has been addressed by Kovalerchuk et al. in [4]. In our work we use the same breast cancer data set, which is composed of 149 samples and 16 features. The data is unidentified and was obtained from patients' exams at the Woman's Hospital of Baton Rouge, Louisiana, USA. The features include a number of calcifications, irregularities in shape and size of the calcifications and density of the lesion, among others. We refer to [4] and the supplementary web page for more information on the dataset.

The goal is to find a set of features that can be used to perfectly classify all cases. At this point, we must emphasize that we are not over-fitting and that this method should not be used as a stand-alone procedure. In fact, classifiers with a high generalization ability (i.e., that can correctly classify new samples) might arise when one applies methods such as neural networks, but only using features that are relevant. This work concerns the problem of finding such features.

## 2 The k-Feature Set Problem

The $k$-Feature Set problem has undoubtedly many applications in knowledge extraction from life sciences and medical databases. It appears as a crucial component in several areas, such as gene discovery, disease diagnosis, drug discovery or pharmacogenomics, toxicogenomics, and cancer research. It can be formalized as follows [5]:

### 2.1 k-Feature Set (decision version)

*Input*: A set $\mathbf{X}$ of $m$ examples (which are composed of a binary value specifying the value of the target feature and a vector of $n$ binary values specifying the values of the other features) and an integer $k > 0$.

*Question*: Does there exist a set $S$ of non-target features (i.e., $S \subseteq \{1, \ldots, n\}$) such that:

- $|S| \leq k$
- No two examples in $\mathbf{X}$ that have identical values for all the features in $S$ have different values for the target feature?

Clearly, this problem could be extended to an alphabet which is not binary. This is the case with the data used in this work. Although there are binary values attributed to some of the features, for some of them we have a ternary or quaternary alphabet, and one with rational values. We return to this issue later. An example can be

**Table 1**
**An instance of the *k*-Feature Set problem**

| Example # | F1 | F2 | F3 | F4 | Target |
|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | 1 |
| 2 | 0 | 0 | 1 | 1 | 1 |
| 3 | 1 | 1 | 0 | 0 | 0 |
| 4 | 0 | 1 | 0 | 1 | 0 |
| 5 | 0 | 0 | 1 | 0 | 0 |

Columns represent features; rows are samples. The last column represents the target of each sample. Features have a binary representation: "1" if they are true for that sample; "0" otherwise

seen in Table 1: here we have replaced the value of "1" for "good" (for instance, absence of cancer) and "0" for "bad" (cancer).

Referring to the example given in Table 1, the reader will note three feature sets with cardinality $k = 3$: S1 = {F1; F2; F4}, S2 = {F1; F3; F4}, and S3 = {F2; F3; F4}. The question of interest is if there is a feature set with cardinality $k = 2$.

Computer scientists are always interested in determining the computational complexity of the basic problem. This decision problem has already shown to be NP-complete by a reduction from *k*-Vertex Cover [5]; another problem belonging to this class. NP-completeness is generally viewed as a reflection of the "computational intractability" of a given mathematical problem. What this typically means, in practical terms, is that it is highly improbable that we will be able to find an algorithm that solves this problem efficiently for general instances. Efficiently, in computational terms, means a time which is proportional to a polynomial function of the size of the input.

This does not mean that the problem cannot be addressed by other means. For small instances or for small values of *k*, it may still be possible to solve it with exact algorithms. Although they may have exponential worst-case behavior, it may be possible that such algorithms can provide the optimal solution in reasonable amounts of time. On the other hand, for large instances, there are now powerful new mathematical methods, generically named metaheuristics, which would allow solving these problems in practice for large instances. Some of these methods are based on the evolution of alternative feasible solutions for the optimization versions of the problem. Examples of this type are Genetic [6] and Memetic Algorithms [7]. An important research question, particularly for the analysis of large instances of this problem in the field of Functional Genetics, is if there is a fixed-parameter algorithm for this problem. In 2003, Cotta and Moscato proved that the problem is not only

NP-complete but W[2]-complete [8]. This means it is likely that a fixed-parameter tractable algorithm does not exist for this problem.

The W[2]-completeness of $k$-Feature Set shows that attention should be concentrated on finding efficient rules that do not remove the optimal solution, but can help to reduce the size of the instance. In the parameterized complexity context, this is called reduction to a problem kernel. This will allow a search algorithm to increase the chances of finding the optimal solution and may enable exponential-time exact searches for smaller instances.

We now discuss the characteristics of the application we are using to introduce this methodology. We then discuss the generalization of $k$-Feature Set and how the problem can be formalized as an optimization problem in a particular type of graph.

## 3 The Breast Cancer Data Set—Initial Preprocessing

As mentioned before, the breast cancer data set used in this work is the same as used by Kovalerchuk et al. [9]. What is available in the public domain is a small subset of a larger database related to tumor exams from patients at the Woman's Hospital of Baton Rouge, Louisiana, USA. We refer the reader to the authors' webpage.[1] The reader will identify that there are 149 examples and that each example in the dataset has 17 features (corresponding to the attributes numbered from 2 to 18 inclusive).

With this data, we conducted two independent tests, each with a binary target. The first was to predict an intraductal carcinoma, and the second to identify whether the tumor is malignant or benign. From the 17 features, we only consider the 16 which are represented by a finite alphabet for our study. This departs from [9] and leaves out from our data set the feature #2: "Approximate volume of the lesion in cubic centimeters". This does not mean that we consider the volume of the lesion "irrelevant" at all. When a given feature that can assume any integer value (or any value in an infinite alphabet), a common approach is to determine appropriate thresholds for each of the features. These problems are generally known as optimal thresholding problems and in some cases they lead to other NP-hard problems. In particular, the decision problem for the thresholding for $k$-Feature Set is NP-complete [3].

Considering the 149 examples present in the dataset, there are two samples (#102 and #134, following the original dataset numbering) that have the same values for all features, except for the volume of calcifications, but have differing target values. Even for the volume of calcifications, the difference is minimal—0.048 against 0.072—considering that the observed values range between

[1] http://www.csc.lsu.edu/trianta/ResearchAreas/DigitalMammography/index.html.

0.008 and 218.416. Since the *k*-Feature Set problem does not allow different outcomes for the same feature pattern, we decided to remove both examples.

Another inconsistency found was with sample #140. The value of attribute "ductal orientation" can only take either yes/no values (labeled "A" and "B" respectively), but in example #140 there is an invalid entry "D." Since it was not possible to infer the correct value, we ignored the sample. This leaves us with 146 samples with 16 features. The sixteen features $x_1, \ldots, x_{16}$ we consider are the following:

- $x_1$—number of calcifications per $cm^2$ (A: <10; B: 10 to 20; C: >20)
- $x_2$—total number of calcifications (A: <10; B: 10 to 30; C: >30)
- $x_3$—irregularity in shape of calcifications (A: mild; B: moderate; C: marked)
- $x_4$—variation in the shape of calcif. (A: mild; B: moderate; C: marked)
- $x_5$—irregularity in the size of calcif. (A: mild; B: moderate; C: marked)
- $x_6$—variation in the density of calcif. (A: mild; B: moderate; C: marked)
- $x_7$–$x_{11}$—Le Gal type of lesion (A given lesion may contain several types)
- $x_{12}$—ductal orientation (A: yes; B: no)
- $x_{13}$—density of the calcifications (A: low; B: moderate; C: high)
- $x_{14}$—density of the parenchyma (A: low; B: moderate; C: high)
- $x_{15}$—comparison with previous exam (A: change in the number or character of calcifications; B: not defined; C: newly developed; D: no previous exam)
- $x_{16}$—associated findings (A: multifocality; B: architectural distortion; C: mass; D: none)

The thresholds are the same used in Kovalerchuch et al. [4]. The target feature separates the examples into two main groups. Each example can be classified in the first test as A: intraductal carcinoma or B: everything else (i.e., other cases), totaling 37 and 109 examples for each group respectively. In the second test we classify the examples as A: malignant (71 examples) or B: benign (75 examples).

## 4 The $(\alpha, \beta)$-$k$-Feature Set Problem

In this combinatorial optimization problem the practical objective is to find the minimum set of features that can differentiate every pair of examples belonging to different classes (*see* **Note 1**). In addition, this feature set must also explain every pair of examples that belong to the same class. The greater the parameters $\alpha$ and $\beta$, the greater the reliability of the classification system, at the expense of a larger optimal feature set. Formally, the decision version of the $(\alpha, \beta)$-$k$-feature set problem we are addressing in this chapter is the following:

### 4.1 $(\alpha, \beta)$-$k$-Feature Set (Decision Version)

*Input*: A set of $m$ examples $\mathbf{X} = \{x^{(1)}, \ldots, x^{(m)}\}$, such that for all $i$, $x^{(i)} = \{x_1{}^{(i)}, x_2{}^{(i)}, \ldots, x_n{}^{(i)}, t^{(i)}\} \in \{0,1\}^{n+1}$, and three integers $k > 0$, and $\alpha, \beta \geq 0$.

*Question*: Does there exist an $(\alpha, \beta)$-$k$-feature set $S$, i.e., $S \subseteq \{1, \ldots, n\}$, with $|S| \leq k$ and such that:

- for all pairs of examples $(x_i, x_j)$, $i \neq j$, if $t(i) \neq t(j)$ there exists $S' \subseteq S$ such that $|S'| \geq \alpha$ and for all $l \in S'$ $x_1{}^{(i)} \neq x_1{}^{(j)}$, and
- for all pairs of examples $i \neq j$, if $t(i) = t(j)$ there exists $S' \subseteq S$ such that $|S'| \geq \beta$ and for all $l \in S'$ $x_1{}^{(i)} = x_1{}^{(j)}$ ?

In the definition above the set $S'$ is not fixed for all pairs of examples, but it is a function of the pair of examples chosen, so we mean $S' = S'(i, j)$. The basic idea is to improve robustness of the original method by allowing some redundancy in example discrimination. We seek to have at least $\alpha$ features for differentiating between any two samples of different classes. Similarly, we want to have at least $\beta$ features with consistent values for any two samples of the same class. Note that each feature may have its own distinct alphabet.

Clearly this problem is also NP-complete (the $k$-feature set problem is a special case with $\alpha = 1$ and $\beta = 0$). We also note that this naturally leads to a multiobjective optimization problem in which, for a given input data, we try to maximize the values of $\alpha$, $\beta > 0$ and at the same time minimize the value of $k > 0$.

## 5 The $(\alpha, \beta)$-$k$-Feature Set as an Optimization Problem in Graphs

It is very useful to reformulate the $(\alpha, \beta)$-$k$-feature set as an optimization problem in a graph. This is beneficial as it allows the use of powerful reduction techniques that significantly reduce the computational effort. Next, we define how to create a bipartite graph $G(V, E)$ from an instance of the problem.

Initially, partition the set of nodes of the graph into three disjoint sets $A$, $B$ and $F$ such that $A \cup B \cup F = \mathrm{V}$. We have said that the graph is bipartite so this may seem confusing at first glance, let us note that one of the partitions of $G$ is the set $F$ and the other is $A \cup B$.

We will now proceed to define how we build the graph, starting with the set of vertices. Each node in the first subset of vertices ($A$) represents each unique pair of examples which have different target values. Analogously, we define $B$ as the subset of nodes of $G$ such that each node corresponds to a pair of examples which have the same target value. Each node in the remaining subset $F$ represents a different feature.

We now define the edge set $E$. There are only two types of edges, those that connect nodes in $F$ with nodes in $A$ and those that connect nodes in $F$ with nodes in $B$. There is an edge between a node $f \in F$ and a node $a \in \mathrm{A}$ if and only if the pair of examples that node $a$ represents have different values for feature $f$.

Analogously, there is an edge between a node $f \in F$ and a node $b \in B$ if and only if the pair of examples that node $b$ represents they have the same value for feature $f$. The graph appearance is shown in Fig. 1, labeled as "Original Graph".

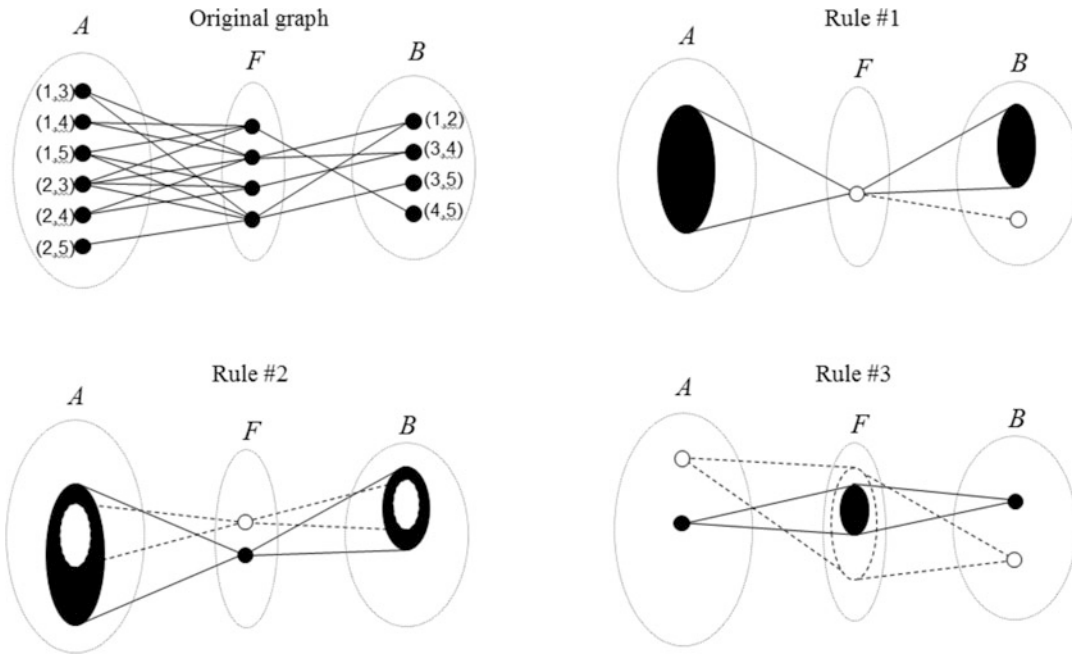



**Fig. 1** Graph representation of the ($\alpha$, $\beta$)-$k$-Feature Set problem. The top-left diagram shows the graph constructed from the example in Table 1. The other three show the action of the reduction rules. Rule #1 searches for pairs of examples that are explained by a single feature. Rule #2 looks for features that explain pairs or examples already covered by another feature. Rule #3 searches for pairs of examples that are irrelevant from the graph domination point-of-view

For the data we are considering, direct application of this graph transformation leads to a large graph, with 10,731 nodes and 87,341 edges. We show how some powerful reduction techniques can then be applied. They will help by selecting "relevant" features (features that must be in the optimal solution), irrelevant features (features that contribute nothing to the optimal solution) and by removing nodes from $A$ and $B$ that do not provide any extra information. These reductions are safe as no information is lost in the process, thus even though we may have a drastic reduction in the size of the graph, we can still find an optimal solution in the graph.

## 6 Reduction Techniques

The application of the reduction rules for the generic problem of knowledge discovery we address here is inspired by the influential work of Weihe [10]. He showed how they can be very useful for solving large, real-world optimization problems. As the graph built from the instance of the problem under study now contains nodes which have degree one (in both $A$ and $B$), we can at most have one feature which explains the differences/similarities of the pairs of examples represented by these nodes, and thus this data set allows at maximum $\alpha = \beta = 1$. For this reason we explain the reduction rules specifically for this condition. If $\alpha$ and $\beta$ could assume larger values, the rules would be slightly different, although the principle remains the same. Illustrations of the following rules can be seen in Fig. 1.

### 6.1 Reduction Rule 1: Relevant and Mandatory Features

This rule searches for features that must be in any minimal cardinality $(1, 1)$-$k$-feature set. If there is a node in either $A$ (or $B$) that has degree one, the feature at the other endpoint of the connecting edge must be in the feature set.

### 6.2 Reduction Rule 2: Irrelevant Features

This rule helps to identify features that can be considered irrelevant. A feature can be deemed irrelevant if it can only explain pairs of examples which are a subset of other pairs already explained by another feature. When we say "explain" we refer to the fact that the presence of a feature can account for the difference or the similarity between a pair of examples. Let $N_{A1}$ and $N_{A2}$ be subsets of pairs of examples belonging to $A$, such that $N_{A1} \subseteq N_{A2}$. Then, consider that every element in $N_{A1}$ is connected to feature $f_i$ and that all elements in $N_{A2}$ are connected to a different feature $f_j$. Moreover, let $N_{B1}$ and $N_{B2}$ be subsets of pairs of examples belonging to $B$, such that $N_{B1} \subseteq N_{B2}$. Then, consider that every element in $N_{B1}$ is connected to $f_i$ and that all elements in $N_{B2}$ are connected to $f_j$. Under such conditions, feature $f_i$ is irrelevant and will not be present in the optimal feature set. If two or more features explain

exactly the same pairs of examples, they are condensed into a single new feature. Then, if such new feature is selected as optimal, the researcher can analyze its components individually and decide which to use.

### 6.3 Reduction Rule 3: Redundant Pairs of Examples

The third reduction rule is motivated by the aim to efficiently find pairs of examples whose differences (or similarities) are already accounted for when dealing with another pair of examples. Let $F1 \subseteq F2 \subseteq F$, where $v_1 \in A \cap B$ is connected to all of $F1$ and $v_2 \in A \cap B$ is connected to all of $F2$. In this case, a suitable set of features in $F1$ that dominates $v_1$ will automatically dominate $v_2$, too. Thus we can delete $v_2$, as it provides no extra information about the features required for the feature set. Here note that we make no distinction between example vertices from $A$ and $B$. This generalization only holds when $\alpha = \beta$. If $\alpha \neq \beta$ we must use the more generic rule presented in [3].

### 6.4 Recursion

For a more general version of these rules, which consider any $\alpha$ and $\beta$ values, please refer to Cotta, Sloper and Moscato [3]. The reduction rules are recursively applied on the original graph in a sequential way, starting with rule 1, until no reduction can be obtained anymore by any of the three.

## 7 Memetic algorithm

As mentioned before, the decision version of the $(\alpha, \beta)$-$k$-feature set problem is NP-complete. Therefore, complete enumeration or exact solution search methods [11] can only be used when the graph is rather small, since the computational complexity grows exponentially with the number of features and samples. Most times, even after the preprocessing step, the graph still remains too large for exact methods to be used in practice.

In such cases, it would be wise to resort to stochastic or informed search algorithms, or more powerful metaheuristics, so as to provide high quality solutions. Towards illustrating this aim, we have implemented a population-based metaheuristic; a memetic algorithm [7, 12, 13], and we give some details of its implementation. We focus on the main aspects, which are important for the Feature Set problem itself.

### 7.1 Representation and Recombination

The representation chosen for the Feature Set problem assigns each feature to a position in a binary array. The positions can assume true/false values that will indicate whether the corresponding feature is in the feature set (effective) or not (ineffective). In a graph context, if a feature becomes ineffective, the corresponding node and all edges connected to it are erased.

Recombination creates new solutions by combining information taken from two original solutions. In this work, the recombination has both deterministic and stochastic aspects. The first phase is deterministic, where features which are effective in both original solutions are set as effective in the new one. The rest of the new solution is completed by randomly choosing an ineffective feature and making it effective, until the solution becomes feasible.

### 7.2 Local Search

Local Search (LS) is applied to all new solutions created through recombination, as is usual in many memetic algorithms. The goal is to improve the solution by testing a series of changes—related to a neighborhood definition—in the solution and keeping the changes that actually improve the solution's quality with regard to the objective function. Since only feasible solutions are generated by the recombination procedure, the only avenue for improvement is to reduce the number of effective features in the solution. In order to do so, three neighborhoods were tested.

The first neighborhood sequentially selects every effective feature and makes it ineffective. If the resulting solution is still feasible, then the feature set size was reduced by one. If the solution became infeasible, then the feature returns to its original state.

The second neighborhood tries to reduce the number of features by removing two features from the feature set and adding only one new feature. It initially selects two effective features, making them ineffective. Then it selects an ineffective feature, different from the two removed, and makes it effective. If the solution remains feasible, the feature set size was reduced, otherwise all features return to their original states.

The last neighborhood is an extension of the second one. Instead of extracting two effective features, it extracts three and adds two, different from those removed. The dimension of this third neighborhood is very large and it could only be applied because the resulting graph after the reduction rules was very small. Depending on the instance size, only the two smaller neighborhoods might be used. The use of local search techniques for such a small instance is not necessary at all. However, when dealing with larger instances, where even after the use of the reduction rules the search space is still considerably large, its use will become imperative.

The three local searches are applied to the solution in a sequential way, until no further improvement can be obtained. When this happens, we conclude the solution has reached a local minimum for all neighborhoods and stop the process.

## 8 Computational Results

The algorithms described above were implemented using the Java Programming Language (JDK 1.7.0) on a Pentium 4 HT PC running at 3.0 GHz, with 1 GB RAM. The CPU time required by the reduction techniques was quite considerable, just over 78 min, but still drastically smaller than many current approaches for solving NP-hard problems. As this was a pilot study on the power of this tandem approach, no systematic effort was conducted towards implementation details that could have delivered a speed up of the reduction rules. Although the CPU time was significant, application of the reduction rules was very beneficial. The reduced graph became an almost trivial task for the memetic algorithm.

The three reduction techniques give very good results for both instances under consideration. In Table 2 we present figures that show the magnitude of the reduction obtained. The decrease in the size of the instances is impressive, both in terms of nodes and edges. Again, we must emphasize that the reduction is not a heuristic, and it is a safe procedure as a minimum cardinality $(\alpha, \beta)$-$k$-feature set is still obtainable from the output of the reductions and the optimal solution of the reduced graph. Concerning the features, for intraductal carcinoma vs. other cases, the reduction rules found that $x_7$, $x_8$, $x_9$, and $x_{15}$ must be in the feature set, and feature $x_{11}$ should be discarded. It is interesting to remark that $x_{11}$ (indicating one of the Le Gal types of the lesion) was not relevant (at least if we include the other three Le Gal type features $x_7$, $x_8$, and $x_9$) and the presence of $x_{15}$ reinforces the relevance of including the comparison with previous exam as an aid to the diagnosis. For malignant vs. benign, the reduction rules found that $x_2$, $x_7$, $x_8$, and $x_{14}$ must be in the feature set, but none of the remaining features were ruled out.

On the reduced graph, the memetic algorithm found an $(\alpha = 1, \beta = 1)$-$k$-feature set with $k = 6$ features for the intraductal

**Table 2**
**Results for the breast cancer-related graph reduction**

| Instance | # of nodes | # of edges | # of features |
|---|---|---|---|
| Intraductal carcinoma test (original) | 10,731 | 87,341 | 16 |
| Intraductal carcinoma test (reduced) | 32 | 93 | 11 |
| Reduction | 99.7 % | 99.9 % | 31.2 % |
| Malignancy test (original) | 10,585 | 88,460 | 16 |
| Malignancy test (reduced) | 31 | 91 | 12 |
| Reduction | 99.7 % | 99.9 % | 25.0 % |

Notice the extreme reduction in the graph's size—always more than 99 % for the number of nodes and edges. The reduction in the number of features was also significant

**Table 3**
**Feature set results for $\alpha = \beta = 1$ for the memetic algorithm**

| Memetic algorithm test | Feature set elements—(# of features) |
|---|---|
| Intraductal carcinoma vs. other | $x_2$, $x_5$, $x_6$, $x_7$, $x_8$, $x_9$, $x_{12}$, $x_{13}$, $x_{15}$, $x_{16}$—(10) |
| Malignant vs. benign | $x_2$, $x_3$, $x_5$, $x_7$, $x_8$, $x_{12}$, $x_{13}$, $x_{14}$, $x_{15}$—(9) |

We show in boldface those features that have been already identified by the reduction rules alone. From the 11 features after the reduction procedure (*see* Table 2), ten are needed to perfectly classify intraductal carcinomas. Similarly, nine features out of the 12 remaining after the reduction are needed to classify malignancy

carcinoma case. After testing all the possible solutions with five or less features—a brute force procedure could be used because of the instance size—none of the 462 sets was found feasible. That means the minimum size of a $(1, 1)$-$k$-feature set that explains the data set is ten—four from the kernel plus six from the memetic algorithm.

Regarding the malignant case, this time the memetic algorithm found a feature set of cardinality five. That translates into nine features to explain the data set—five from the memetic algorithm and four from the kernel. This was also confirmed to be the minimum cardinality feature set possible. The complete enumeration by brute force took just over 10 s while both the MA and the Greedy procedure took a fraction of a second. In Table 3 we present the number of features obtained by the two algorithms.

## 9 Classification Rules

After the determination of the relevant features, the determination of classification rules comes naturally. Considering the intraductal carcinoma case, Table 4 presents the list of rules that classify the data based on the feature set found by the memetic procedure. Table 5 presents the rules associated with the malignant vs. benign cases, attained from the results of the memetic algorithm.

The rules were found using the WEKA data mining software package (http://www.cs.waikato.ac.nz/ml/weka/). WEKA employs several traditional techniques such as ID3 and C4.5 [14]. We have used the PART heuristic from this package, which uses a divide-and-conquer approach to build these rules. It builds a partial C4.5 decision tree in each iteration, and turns the "best" leaf into a rule. The rules should be used in a cascaded manner (i.e., successive if-else statements). Thus, it must be initially checked if the given example satisfies the first rule. If it does not, we proceed checking if it satisfies the second one, and so on, until the last rule. The initial rules are able to better discriminate a larger number of examples, mainly because—at this stage—most of the examples are still unclassified. However, as we approach the end, almost all examples have already been classified by

**Table 4**
**Classification rules for the memetic algorithm—intraductal carcinoma vs. other**

| Rule | Rule's clauses (# of samples covered—intraductal carcinoma? [Y/N]) |
|---|---|
| 1 | $\neg(x_5 = C) \wedge \neg(x_6 = C) \wedge (x_7 = 0) \wedge (x_8 = 2) \wedge (x_9 = 0) \wedge (x_{12} = B) \wedge \neg(x_{15} = B) \wedge \neg(x_{16} = B)$ (24 – N) |
| 2 | $\neg(x_2 = C) \wedge \neg(x_6 = C) \wedge \neg(x_{15} = B) \wedge (x_{16} = B)$ (9 – N) |
| 3 | $\neg(x_6 = C) \wedge (x_7 = 1) \wedge (x_9 = 0) \wedge \neg(x_{15} = B)$ (8 – N) |
| 4 | $\neg(x_2 = C) \wedge \neg(x_5 = B) \wedge \neg(x_6 = C) \wedge (x_7 = 0) \wedge (x_8 = 0) \wedge (x_{12} = B) \wedge \neg(x_{15} = B) \wedge \neg(x_{15} = C) \wedge (x_{16} = D)$ (14 – N) |
| 5 | $\neg(x_5 = A) \wedge \neg(x_6 = C) \wedge \neg(x_{13} = B) \wedge \neg(x_{15} = A) \wedge \neg(x_{15} = B)$ (9 – N) |
| 6 | $(x_6 = C) \wedge (x_{15} = C)$ (6 – Y) |
| 7 | $\neg(x_6 = B) \wedge (x_7 = 1) \wedge \neg(x_{15} = B)$ (4 – N) |
| 8 | $\neg(x_5 = B) \wedge (x_7 = 0) \wedge \neg(x_{15} = B) \wedge (x_{16} = B)$ (3 – Y) |
| 9 | $(x_2 = B) \wedge \neg(x_5 = A) \wedge (x_7 = 0) \wedge (x_8 = 0) \wedge (x_9 = 0) \wedge \neg(x_{13} = A) \wedge \neg(x_{15} = B) \wedge (x_{16} = C)$ (5 – N) |
| 10 | $\neg(x_6 = B) \wedge (x_7 = 0) \wedge (x_8 = 0) \wedge \neg(x_{13} = C) \wedge (x_{15} = C) \wedge \neg(x_{16} = B)$ (4 – Y) |
| 11 | $\neg(x_2 = C) \wedge (x_5 = B) \wedge \neg(x_6 = B) \wedge (x_7 = 0) \wedge \neg(x_{15} = B) \wedge \neg(x_{15} = C) \wedge \neg(x_{16} = B)$ (8 – N) |
| 12 | $\neg(x_2 = C) \wedge (x_7 = 0) \wedge (x_8 = 2) \wedge \neg(x_{15} = B) \wedge \neg(x_{15} = C) \wedge \neg(x_{16} = B)$ (5 – Y) |
| 13 | $(x_2 = B) \wedge \neg(x_6 = A) \wedge (x_7 = 0) \wedge \neg(x_{13} = C) \wedge \neg(x_{15} = B) \wedge \neg(x_{15} = C)$ (6 – Y) |
| 14 | $\neg(x_2 = A) \wedge (x_6 = A) \wedge (x_7 = 0) \wedge (x_{13} = A) \wedge \neg(x_{15} = B) \wedge \neg(x_{16} = C)$ (8 – N) |
| 15 | $(x_7 = 0) \wedge (x_{15} = C)$ (3 – N) |
| 16 | $(x_5 = A) \wedge (x_7 = 0) \wedge \neg(x_{13} = A) \wedge \neg(x_{15} = A) \wedge \neg(x_{15} = B) \wedge \neg(x_{16} = B)$ (2 – Y) |
| 17 | $\neg(x_2 = C) \wedge (x_7 = 0) \wedge \neg(x_{15} = B)$ (4 – N) |
| 18 | $\neg(x_5 = A) \wedge \neg(x_6 = B) \wedge (x_7 = 0) \wedge (x_8 = 2) \wedge \neg(x_{13} = A) \wedge \neg(x_{16} = B)$ (3 – N) |
| 19 | $(x_{13} = A) \wedge \neg(x_{16} = B)$ (2 – Y) |
| 20 | $\neg(x_5 = A) \wedge \neg(x_6 = B) \wedge (x_9 = 3) \wedge \neg(x_{13} = A)$ (2 – N) |
| 21 | $\neg(x_5 = B) \wedge (x_8 = 0) \wedge (x_9 = 0) \wedge (x_{12} = A) \wedge (x_{13} = B)$ (3 – N) |
| 22 | $(x_{12} = A) \wedge \neg(x_{13} = A) \wedge (x_{15} = A) \wedge \neg(x_{16} = C)$ (4 – Y) |
| 23 | $(x_8 = 0) \wedge (x_{15} = A) \wedge \neg(x_{16} = A)$ (4 – N) |
| 24 | $\neg(x_{13} = A)$ (5 – Y) |
| 25 | All the rest (1 – N) |

There are 25 rules in total, which should be read in a cascading way. That is, rule #1 classifies 24 samples as non-intraductal carcinoma. For the remaining samples, rule #2 classifies 9 of them as non-intraductal carcinoma again, and so on

**Table 5**
**Classification rules for the memetic algorithm—Malignant vs. Benign**

| Rule | Rule's clauses (# of samples covered—Malignant? [Y/N]) |
|---|---|
| 1 | $\neg(x_3 = B) \wedge (x_7 = 1)\ (10 - N)$ |
| 2 | $\neg(x_3 = A) \wedge (x_{12} = A) \wedge (x_{13} = B)\ (15 - Y)$ |
| 3 | $(x_2 = C) \wedge \neg(x_3 = A) \wedge (x_5 = C) \wedge (x_{12} = B) \wedge \neg(x_{13} = A)\ (6 - Y)$ |
| 4 | $(x_3 = B) \wedge (x_{15} = C)\ (4 - Y)$ |
| 5 | $\neg(x_5 = C) \wedge (x_8 = 2) \wedge (x_{12} = B) \wedge (x_{13} = C) \wedge \neg(x_{15} = B)\ (8 - N)$ |
| 6 | $(x_2 = B) \wedge \neg(x_3 = B) \wedge \neg(x_5 = C) \wedge (x_7 = 0) \wedge (x_8 = 2) \wedge (x_{12} = B) \wedge \neg(x_{15} = B) \wedge \neg(x_{15} = C)\ (5 - N)$ |
| 7 | $(x_2 = A) \wedge \neg(x_3 = A) \wedge \neg(x_{15} = B)\ (13 - N)$ |
| 8 | $\neg(x_2 = C) \wedge \neg(x_5 = A) \wedge (x_7 = 0) \wedge (x_{12} = B) \wedge (x_{13} = A) \wedge (x_{15} = A)\ (6 - Y)$ |
| 9 | $\neg(x_2 = A) \wedge (x_3 = C) \wedge \neg(x_5 = B) \wedge (x_8 = 0) \wedge (x_{13} = C) \wedge \neg(x_{14} = C) \wedge \neg(x_{15} = B) \wedge \neg(x_{15} = C)\ (3 - Y)$ |
| 10 | $(x_7 = 0) \wedge (x_8 = 0) \wedge (x_{14} = C) \wedge (x_{15} = A)\ (5 - N)$ |
| 11 | $(x_3 = A) \wedge (x_{14} = C) \wedge \neg(x_{15} = B)\ (4 - Y)$ |
| 12 | $\neg(x_2 = B) \wedge (x_8 = 2) \wedge \neg(x_{13} = C) \wedge \neg(x_{15} = A) \wedge \neg(x_{15} = B)\ (10 - N)$ |
| 13 | $(x_3 = A) \wedge (x_7 = 0) \wedge \neg(x_{15} = A) \wedge \neg(x_{15} = B)\ (7 - Y)$ |
| 14 | $(x_5 = B) \wedge (x_7 = 0) \wedge (x_{15} = D)\ (4 - Y)$ |
| 15 | $(x_7 = 0) \wedge \neg(x_{13} = C) \wedge \neg(x_{14} = A) \wedge (x_{15} = D)\ (4 - Y)$ |
| 16 | $\neg(x_2 = A) \wedge \neg(x_3 = A) \wedge (x_5 = A) \wedge \neg(x_{14} = A) \wedge \neg(x_{15} = B)\ (7 - N)$ |
| 17 | $\neg(x_2 = C) \wedge (x_7 = 0) \wedge (x_8 = 2) \wedge \neg(x_{14} = C) \wedge \neg(x_{15} = B)\ (5 - Y)$ |
| 18 | $(x_2 = B) \wedge (x_7 = 0) \wedge (x_{15} = D)\ (3 - N)$ |
| 19 | $(x_2 = C) \wedge \neg(x_3 = B) \wedge (x_7 = 0) \wedge \neg(x_{13} = B) \wedge \neg(x_{15} = B) \wedge \neg(x_{15} = D)\ (5 - N)$ |
| 20 | $\neg(x_2 = A) \wedge \neg(x_3 = C) \wedge (x_5 = B) \wedge (x_7 = 0) \wedge \neg(x_{13} = C) \wedge \neg(x_{14} = C) \wedge \neg(x_{15} = B) \wedge \neg(x_{15} = D)\ (4 - Y)$ |
| 21 | $(x_{12} = A)\ (3 - N)$ |
| 22 | $\neg(x_2 = A) \wedge (x_3 = A)\ (3 - Y)$ |
| 23 | $(x_2 = A)\ (2 - N)$ |
| 24 | $(x_5 = A) \wedge (x_7 = 0) \wedge (x_8 = 0) \wedge (x_{13} = B) \wedge \neg(x_{15} = C)\ (2 - Y)$ |
| 25 | $(x_{13} = B)\ (2 - N)$ |
| 26 | $(x_2 = B) \wedge \neg(x_3 = B) \wedge (x_{14} = B)\ (2 - Y)$ |
| 27 | $(x_2 = B)\ (2 - N)$ |
| 28 | All the rest (2 – N) |

The rules should be interpreted as in Table 4

the previous rules and thus only outliers remained. The consequence is a reduction in the number of samples being classified by the last rules.

Using the features provided by the memetic algorithm for the intraductal carcinoma case, the total number of rules was 25 (*see* Table 4 and **Note 2**). For the malignant vs. benign case (Table 5 and **Note 3**) the PART heuristic returned 28 rules.

One may be inclined to believe that the number of rules created may be a symptom of over-fitting. Again, we must emphasize that we are not over-fitting (at least not in the negative sense of loss of generalization). What we aim to do is to fit exactly in order to find relevant features, in which future classification efforts should be concentrated. The rules presented in Tables 4 and 5 are a contribution to this classification effort, but with no generalization pretensions. Again, we reinforce that the rules must be applied in cascade and no rule can be interpreted stand-alone. For this reason, even though the last rules have fewer features, they are still relevant because all previous rules are required to be false. Following this idea, the last rules might actually rely on information present in as many features as the initial rules.

Concerning the trade-off between the number of features and the number of classification rules, two aspects must be taken into account. The first is that the use of more features than necessary—i.e., a superset of the optimal feature set—does not necessarily improve the reliability of the rules. Indeed, given a certain number of examples to work with, finding the optimal set of features that can explain the data is expected to improve the a priori generalization capability of the generated rules. The second aspect, which may be important in other application areas, not necessarily this one but it would be relevant to mention *en passant*, is related to cost of data collection. Working with more features generally means that more time—and financial resources, particularly in the clinical domain,—is going to be spent on extracting the same information from the data set.

## 10 Discussion

In this section we discuss the results we have obtained in the *malignant* vs. *benign* and the *intraductal carcinoma* vs. *other* classification tasks from the database used. We first note that our conclusions are based only on this data, and we aim at pointing out some conclusions from this study only. We also expect that a larger sample database will lead to the development of other types of classification rules or give more support to the ones obtained here. However, the results need to be put into the perspective of other results in the medical literature and this is our aim.

### 10.1 Malignancy vs. Benign Classification

Considering this classification, one of the most relevant works is from Kovalerchuk et al. [9]. Therein, the authors obtain two rules from radiology experts that are commonly accepted as indicating the possibility of malignant lesions. After translating them into our format to identify the features they become:

- *IF* ($x_1 = C$ *and* $x_2 = C$ *and* $x_3 = C$) *THEN the lesion is highly suspicious for malignancy.*
- *IF* ($x_1 = C$ *and* $x_2 = C$ *and* $x_5 = C$ *and* $x_6 = C$ *and* $x_{13} = C$) *THEN the lesion is highly suspicious for malignancy.*

When the rules were applied to the data set, the result was unsatisfactory. The first rule classified four malignant cases correctly and one benign was classified as malignant. The other 67 malignant cases were missed. The second rule had an even worse performance, classifying correctly only two samples. It misclassified two benign cases as malignant and missed the other 69 malignant cases. The conclusion we might draw is that these types of rules, which we may label as worst-case rules, could be intuitively appealing but are not suitable to help in the diagnosis of this data set.

Another previous work also dealing with malignant/benign diagnosis is Yunus et al. [15]. In their work they show a strong correlation between malignancy and five features: $x_1$, $x_2$, $x_5$, $x_6$, and $x_{13}$. From these, we have three of them ($x_2$, $x_5$, and $x_{13}$) in the optimal feature set found by our algorithm. Also, the dataset used in [15] contains 19 cases with Le Gal type 5 and all of them are malignant. In the database used in this study, this correlation is not so strong but nevertheless very important. From 27 Le Gal type 5 cases, 21 are malignant, corresponding to a ratio of almost 80 %. In our results, the only relevant Le Gal types for optimal discrimination power were 1 and 2 (of course, complemented by the other features in the optimal set). Le Gal type 1 appears only in rule #1 and the rule classifies ten samples as benign. Le Gal type 2 appears in rules #5, #6, #12, and #17. These rules classify 23 samples as benign and five samples as malignant, in total. That gives a 21 % malignancy ratio, which is close to the expected malignancy proportion for Le Gal type 2.

### 10.2 Intraductal Carcinoma

Now we discuss the intraductal carcinoma vs. other cases classification task. For this case our algorithm found eight different feature sets with 10 features. After finding the set of rules for each of them we decided to report the one with the least rules—25 in total. The number of rules varied from 25 to 34. A potential indication of the importance of a feature set, when there is more than one optimal feature set of a given cardinality, is to count the number of times that a feature has appeared in one of the optimal solutions; secondly, the set of rules for all solutions should be generated and their sizes checked.

The number of times each feature not in the kernel appears in the optimal solutions is: $x_1(2)$, $x_2(3)$, $x_3(1)$, $x_4(2)$, $x_5(4)$, $x_6(6)$, $x_{10}(4)$, $x_{12}(6)$, $x_{13}(7)$, $x_{14}(6)$, and $x_{16}(7)$ (of course, those in the kernel appear in all eight optimal solutions). This indicates that, beside the fact we selected the optimum feature set with the lowest number of rules, it could also be good in terms of the number of times its features have appeared in an optimal solution (33 out of a maximum of 36).

A direct comparison with the results of Kovalerchuk et al. [4] is not possible since firstly, they use almost all of the features (16 in total), however, we proved that only ten features are required for the harder task of finding a discriminative feature set which also maximizes within class similarity ($\beta = 1$ which is the maximum in this case). Secondly, their results are also based on a feature (volume of calcifications) which we have initially removed from consideration in this article since we believe there is no support for its inclusion. By including it, while not associating particular thresholds that would quantize it, the feature also seems to be generating a large number of "poor rules" (using the authors' own words) or rules with a small support in the dataset.

In the kernel, we have three features corresponding to Le Gal types (1, 2, and 3) that must be in any optimal solution. This gives some support to the usefulness of this classification system. The other feature in the kernel is the "comparison with the previous exam," which also seems to be well-correlated with current practice and recommendations. It is interesting to remark that the Le Gal type 4 also appears in half of the total number of optimal solutions while Le Gal type 5 does not appear in any of them.

### 10.3 Clustering Using the Optimal Feature Sets

To illustrate other aspects of our discussion, we employ a visualization approach and clustering algorithm based on our work in gene expression data analysis [16, 17]. We refer the reader to those articles to understand the details of the method. In essence, the aim is to arrange, on two dimensions, a large number of one-dimensional arrays (containing the information of interest) in such a way that consecutive, or closely placed arrays, are as similar as possible. This is an NP-hard optimization problem, but our method can obtain either optimal or very close to optimal solutions. We thus provide a permutation of all the samples in the data set (without permuting the feature order) which helps to understand the data.

In Fig. 2 we show two images. The samples were first divided according to their target feature and then clustered within each group. The goal is to visually identify correlations between the features in the optimal solution and the classification itself. If it is possible to find features that assume consistently different values for different classes, the classification problem becomes easier. On the other hand, the absence of any significant difference indicates that

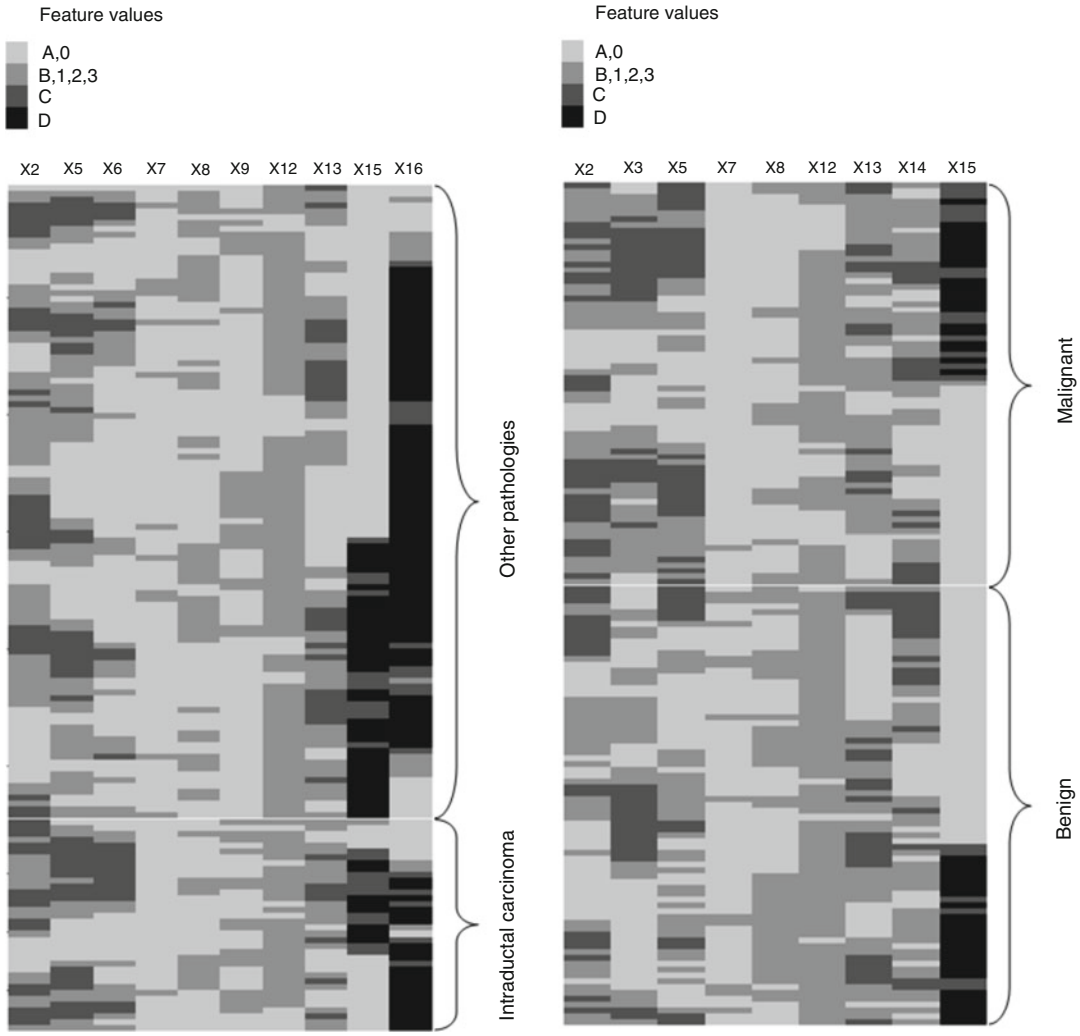



**Fig. 2** Samples separated into two groups, according to their classification, and then clustered within each group using the algorithm of [16]. Intraductal carcinoma vs. other cases (*left*) and Malignant vs. Benign (*right*). The gray scale used helps to identify the feature attributes for each sample. Images were created using the software for data clustering and visualization NBIMiner (Moscato et al. [17])

the classification problem is more difficult and the rules need to be more complex.

Beginning with the left image of Fig. 2, the most visually relevant feature is $x_7$ (Le Gal type 1), which is also in the kernel found by the reduction rules. For the other features we could not see any clear differences between the groups. This indicates that the relations between the features and the tumor classification are very fuzzy, reflecting the practical difficulty of establishing rules for intraductal carcinoma using only individual features. In the right image of the same figure we can see the relation between features and classification better. Excepting features $x_{12}$, $x_{13}$, $x_{14}$, and $x_{15}$,

the others show different average behaviors for both classes. The large number of features with distinct behaviors indicate that classification of malignant vs. benign is easier than intraductal carcinoma vs other cases. This was also reflected in the different sizes of the optimal feature sets found for both cases.

## 11 Conclusion

This work tackled the $(\alpha, \beta)$-$k$-Feature Set problem using a Graph Theoretic approach. One of the main contributions is the new insights derived from a transformation from one problem (computer-aided rule generation for clinical diagnosis) into a constrained graph domination problem. The approach also shows the power of three simple reduction techniques used to shrink the size of the resulting graph, without losing the optimal solution we seek. These reduction techniques were able to eliminate over 99 % of the graph's nodes and edges allowing us to obtain provably optimal solutions.

The reduction rules were applied to two classification problems: intraductal carcinoma vs. others and malignant vs. benign lesions. They found out that $x_7$, $x_8$, $x_9$, and $x_{15}$ (Le Gal types I, II, III and "comparison with previous exam," respectively) must be in the feature set, and feature x11 (Le Gal type V) should be discarded for the intraductal carcinoma case, and that features $x_2$, $x_7$, $x_8$, and $x_{14}$ ("total number of calcifications," Le Gal types I, II and "density of the parenchyma") must be in the feature set for the malignant vs. benign dichotomy classification. The original problem, which would be a challenge for most optimization techniques, became a much more tractable one, in computational terms, after the reduction rules were used.

As well as the reduction techniques, we implemented a memetic algorithm (MA) to find the optimum feature sets for both cases. The MA was able to reach solutions proved to be optimal after an exhaustive search was conducted. The method found feature sets with sizes 10 and 9 for the intraductal carcinoma and malignancy problems, respectively.

Finally, we determined a set of classification rules, obtained from the feature sets returned by the MA. These rules can, in principle, be applied to classify breast cancer tumors, although generalization concerns might arise. A promising direction for future research might arise if the algorithm introduced here is used to determine relevant features that can be used by other classification techniques, such as neural networks. However it is necessary that additional tests with instances composed of many more samples be conducted.

Our method also points out the importance of large-scale combinatorial optimization models and exact and heuristic

techniques coupled with large databases of population-studies of breast cancer to help evolve better computer-aided predictions. The generalization capability is likely to improve when the number of samples increases to larger values, on the order of thousands of samples. The relevance of the reduction rules in analyzing the data suggests that this is an exciting future area of research indicating that computer automated methods for knowledge extraction from large medical databases of population-wide studies are a viable alternative to traditional methods. We also support the availability of these databases on public domain in raw format (instead of expert quantization of the attributes), since different thresholding techniques such as those from [3] could lead to smaller feature sets with higher robustness.

## 12 Notes

1. Since its introduction, the approach of employing data mining techniques in high-dimensional datasets by using the $(\alpha, \beta)$-$k$-feature set problem as a combinatorial model to reduce the dimensionality has found several applications. Its applications in the selection of biomarkers for prediction of Alzheimer's disease [18–20], transcriptomic analyses of brain tissues [21–23], and identification of multiple sclerosis biomarkers in whole blood [24] were of great importance. It also helped to produce the first detection of childhood absence epilepsy using basal clinical EEGs [25], and has been applied in cancer research [26]. Integer programming formulations and current commercial software showed that the approach scales well in practice [11]. Our combinatorial approach offers a new alternative to statistical-only univariate procedures for the detection of biomarkers [26–28]. For clinicians, we think that our work could entice the interest in Le Gal's classification of microcalcifications [29–31] and their role as a possible early marker for pattern recognition panels [32–43]. The use of the PART heuristic leads to one particular approach to generate rules for classification and we have included this particular heuristic in our study as a novelty to the clinical community. We note however, that after employing the $(\alpha, \beta)$-$k$-feature set approach, the information provided by the reduced set of features can be used as an input for ensemble-based classifiers. This active area of research in machine learning has only recently been applied to problems in breast cancer [44–50]. We also expect that techniques based on mining disjunctive closed item sets could be used after feature selection [51, 52]. We expect that the synergies coming from the combination of these techniques will soon translate in improved early tests for the diagnosis of this disease.

2. In this Note we present the first eight rules from Table 4, obtained from the feature set generated with the memetic algorithm for the intraductal carcinoma case in natural language. As noted earlier in the chapter, these rules operate in a cascading fashion, that is, each rule must be applied in order, beginning with rule #1, until a rule applies.
   - *Rule 1*: If the following conditions are true THEN the prediction is NO for intraductal carcinoma (24 samples classified).

     Conditions:
     The irregularity in the size of calcifications is not marked
     and the variation in the density of the calcifications is not marked
     and Le Gal type is not #1
     and Le Gal type is #2
     and Le Gal type is not #3
     and there is no ductal orientation
     and the comparison with previous exam is defined (not "not defined")
     and the associated findings do not show architectural distortion.
   - *Rule 2*: If the following conditions are true, and the previous rule does not apply, THEN the prediction is NO for intraductal carcinoma (9 samples classified).

     Conditions:
     The total number of calcifications is less than (or equal to) 30
     and the variation in the density of the calcifications is not marked
     and the comparison with previous exam is defined
     and the associated findings do show architectural distortion.
   - *Rule 3*: If the following conditions are true, and the previous rules do not apply, THEN the prediction is NO for intraductal carcinoma (8 samples classified).

     Conditions:
     The variation in the density of the calcifications is not marked
     and Le Gal type is #1
     and Le Gal type is not #3
     and the comparison with previous exam is defined.
   - *Rule 4*: If the following conditions are true, and the previous rules do not apply, THEN the prediction is NO for intraductal carcinoma (14 samples classified).

Conditions:
The total number of calcifications is less than (or equal to) 30
and the irregularity in the size of the calcifications is not moderate
and the variation in the density of the calcifications is not marked
and Le Gal type is not #1
and Le Gal type is not #2
and there is no ductal orientation
and the comparison with previous exam is defined
and the comparison with previous exam is not newly developed
and there are no associated findings.

- *Rule 5*: If the following conditions are true, and the previous rules do not apply, THEN the prediction is NO for intraductal carcinoma (nine samples classified).

  Conditions:
  The irregularity in the size of the calcifications is not mild
  and the variation in the density of the calcifications is not marked
  and the density of the calcifications is not moderate
  and the comparison with previous exam does not show a change in the number or character of calcifications
  and the comparison with previous exam is defined.

- *Rule 6*: If the following conditions are true, and the previous rules do not apply, THEN the prediction is YES for intraductal carcinoma (six samples classified).

  Conditions:
  The variation in the density of the calcifications is marked
  and the comparison with previous exam is newly developed.

- *Rule 7*: If the following conditions are true, and the previous rules do not apply, THEN the prediction is NO for intraductal carcinoma (four samples classified).

  Conditions:
  The variation in the density of the calcifications is not moderate
  and Le Gal type is #1
  and the comparison with previous exam is defined.

- *Rule 8*: If the following conditions are true, and the previous rules do not apply, THEN the prediction is YES for intraductal carcinoma (three samples classified).

Conditions:
The irregularity in the size of the calcifications is not moderate
and Le Gal type is not #1
and the comparison with previous exam is defined
and the associated findings show architectural distortion.

3. In this Note we present the first eight rules from Table 5, for the malignant tumor case, generated from the feature set obtained by a Memetic algorithm, in natural language. Once again, these rules are to be used in a cascade fashion. That is, they are to be applied successively, beginning with rule #1, until a rule is satisfied.
   - *Rule 1*: If the following conditions are true THEN the prediction is NO for malignant (ten samples classified).

     Conditions:
     The irregularity in the shape of the calcifications is not moderate
     and Le Gal type is #1.
   - *Rule 2*: If the following conditions are true, and the previous rule does not apply, THEN the prediction is YES for malignant (15 samples classified).

     Conditions:
     The irregularity in the shape of the calcifications is not mild
     and there is ductal orientation
     and the density of the calcifications is moderate.
   - *Rule 3*: If the following conditions are true, and the previous rules do not apply, THEN the prediction is YES for malignant (six samples classified).

     Conditions:
     The total number of calcifications is greater than 30
     and the irregularity in the shape of the calcifications is not mild
     and the irregularity in the size of the calcifications is marked
     and there is no ductal orientation
     and the density of the calcifications is not low.
   - *Rule 4*: If the following conditions are true, and the previous rules do not apply, THEN the prediction is YES for malignant (four samples classified).

     Conditions:
     The irregularity in the shape of the calcifications is moderate
     and the comparison with previous exam is newly developed.

- *Rule 5*: If the following conditions are true, and the previous rules do not apply, THEN the prediction is NO for malignant (eight samples classified).

  Conditions:
  The irregularity in the size of the calcifications is not marked
  and Le Gal type is #2
  and there is no ductal orientation
  and the density of the calcifications is high
  and the comparison with previous exam is defined.

- *Rule 6*: If the following conditions are true, and the previous rules do not apply, THEN the prediction is NO for malignant (five samples classified).

  Conditions:
  The total number of calcifications is between 10 and 30
  and the irregularity in the shape of the calcifications is not moderate
  and the irregularity in the size of the calcifications is not marked
  and Le Gal type is not #1
  and Le Gal type is #2
  and there is no ductal orientation
  and the comparison with previous exam is defined
  and the comparison with previous exam is not newly developed.

- *Rule 7*: If the following conditions are true, and the previous rules do not apply, THEN the prediction is NO for malignant (13 samples classified).

  Conditions:
  The total number of calcifications is less than 10
  and the irregularity in the shape of the calcifications is not mild
  and the comparison with previous exam is defined.

- *Rule 8*: If the following conditions are true, and the previous rules do not apply, THEN the prediction is YES for malignant (six samples classified).

  Conditions:
  The total number of calcifications is less than (or equal to) 30
  and the irregularity in the size of the calcifications is not mild
  and Le Gal type is not #1
  and there is no ductal orientation
  and the density of the calcifications is low

and the comparison with previous exam shows a change in the number or character of calcifications.

## References


1. Bird R, Wallace T, Yankaskas B (1992) Analysis of cancer missed at screening mammography. Radiology 184:613–617
2. Hall F, Storella J, Silverstone D, Wyshak G (1988) Nonpalpable breast lesions: recommendations for biopsy based on suspicion of carcinoma at mammography. Radiology 167:353–358
3. Cotta C, Sloper C, Moscato P (2004) Evolutionary search of thresholds for robust feature set selection: application to the analysis of microarray data. In: Proceedings of EvoBio2004—2nd European workshop on evolutionary computation and bioinformatics, Coimbra, Portugal, 5–7 April 2004, pp 21–30
4. Kovalerchuk B, Triantaphyllou E, Ruiz J, Torvik V, Vityaev E (2000) The reliability issue of computer-aided breast cancer diagnosis. Comput Biomed Res 33:296–313
5. Davies S, Russell S (1994) NP-completeness of searches for smallest possible feature sets. In: Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI) fall symposium on relevance, pp 41–43
6. Goldberg D, Sastry K (2010) Genetic algorithms: the design of innovation, 2nd edn. Springer, New York
7. Moscato P, Cotta C, Mendes A (2004) Memetic algorithms. In: Onwubolu G, Babu B (eds) New optimization techniques in engineering. Springer, New York, pp 53–86
8. Cotta C, Moscato P (2003) The k-Feature Set problem is W[2]-complete. J Comput Syst Sci 67(4):686–690
9. Kovalerchuk B, Vityaev E, Ruiz J (2000) Consistent knowledge discovery in medical diagnosis. IEEE Eng Med Biol 19:26–37
10. Weihe K (1998) Covering trains by stations or the power of data reduction. In: Proceedings of ALEX'98—1st workshop on algorithms and experiments, Trento, Italy, 9–11 February 1998, pp 1–8
11. Berretta R, Mendes A, Moscato P (2007) Selection of discriminative genes in microarray experiments using mathematical programming. J Res Pract Inform Technol 39 (4):287–299
12. Moscato P, Cotta C (2003) A gentle introduction to memetic algorithms. In: Glover F, Kochenberger G (eds) Handbook of metaheuristics. Springer, New York, pp 105–144
13. Neri F, Cotta C, Moscato P (2011) Handbook of memetic algorithms. Springer, New York
14. Witten I, Frank E (2005) Data mining: practical machine learning tools and techniques. Morgan Kaufmann, USA
15. Yunus M, Ahmed N, Masroor I, Yaqoob J (2004) Mammographic criteria for determining the diagnostic value of microcalcifications in the detection of early breast cancer. J Pak Med Assoc 54:24–29
16. Cotta C, Mendes A, Garcia V, Franca P, Moscato P (2003) Applying memetic algorithms to the analysis of microarray data. In: Cagnoni S et al. (eds) Proceedings of EvoBIO2003—1st European workshop on evolutionary bioinformatics, Essex, UK, 14–16 April 2003. Lecture Notes in Computer Science, vol 2611. Springer, Heidelberg, pp 22–32
17. Moscato P, Mendes A, Berretta R (2007) Benchmarking a memetic algorithm for ordering microarray data. Biosystems 88 (1–2):56–75
18. Johnstone D, Milward EA, Berretta R, Moscato P (2012) Multivariate protein signatures of pre-clinical Alzheimer's disease in the Alzheimer's disease neuroimaging initiative (ADNI) plasma proteome dataset. PLoS One 7(4):e34341
19. de Paula MR, Ravetti MG, Berretta R, Moscato P (2011) Differences in abundances of cell-signalling proteins in blood reveal novel biomarkers for early detection of clinical Alzheimer's disease. PLoS One 6(3):e17481
20. Ravetti MG, Moscato P (2008) Identification of a 5-protein biomarker molecular signature for predicting Alzheimer's disease. PLoS One 3 (9):e3111
21. Johnstone D, Graham RM, Trinder D, Delima RD, Riveros C, Olynyk JK et al (2012) Brain transcriptome perturbations in the Hfe(−/−) mouse model of genetic iron loading. Brain Res 1448:144–152
22. Johnstone DM, Graham RM, Trinder D, Riveros C, Olynyk JK, Scott RJ et al (2012) Changes in brain transcripts related to Alzheimer's disease in a model of HFE hemochromatosis are not consistent with increased Alzheimer's disease risk. J Alzheimers Dis 30 (4):791–803
23. Ravetti MG, Rosso OA, Berretta R, Moscato P (2010) Uncovering molecular biomarkers that

correlate cognitive decline with the changes of hippocampus' gene expression profiles in Alzheimer's disease. PLoS One 5(4):e10153

24. Riveros C, Mellor D, Gandhi KS, McKay FC, Cox MB, Berretta R et al (2010) A transcription factor map as revealed by a genome-wide gene expression analysis of whole-blood mRNA transcriptome in multiple sclerosis. PLoS One 5(12):e14176
25. Rosso OA, Mendes A, Berretta R, Rostas JA, Hunter M, Moscato P (2009) Distinguishing childhood absence epilepsy patients from controls by the analysis of their background brain electrical activity (II): a combinatorial optimization approach for electrode selection. J Neurosci Methods 181(2):257–267
26. Mendes A, Scott RJ, Moscato P (2008) Microarrays—identifying molecular portraits for prostate tumors with different Gleason patterns. Methods Mol Med 141:131–151
27. Berretta R, Costa W, Moscato P (2008) Combinatorial optimization models for finding genetic signatures from gene expression datasets. Methods Mol Biol 453:363–377
28. Milward EA, Moscato P, Riveros C, Johnstone DM (2014) Beyond statistics: a new combinatorial approach to identifying biomarker panels for the early detection and diagnosis of Alzheimer's disease. J Alzheimers Dis 39(1):211–217
29. Pastore G, Costantini M, Valentini V, Romani M, Terribile D, Belli P (2002) Clinically nonpalpable breast tumors: global critical review and second look on microcalcifications. Rays 27(4):233–239
30. Bocchi L, Nori J (2007) Shape analysis of microcalcifications using Radon transform. Med Eng Phys 29(6):691–698
31. Resende LM, Matias MA, Oliveira GM, Salles MA, Melo FH, Gobbi H (2008) Evaluation of breast microcalcifications according to Breast Imaging Reporting and Data System (BI-RADS) and Le Gal's classifications. Rev Bras Ginecol Obstet 30(2):75–79
32. Wilson GH 3rd, Gore JC, Yankeelov TE, Barnes S, Peterson TE, True JM et al (2014) An approach to breast cancer diagnosis via PET imaging of microcalcifications using 18F-NaF. J Nucl Med 55(7):1138–1143
33. Boisserie-Lacroix M, Bullier B, Hurtevent-Labrot G, Ferron S, Lippa N, Mac Grogan G (2014) Correlation between imaging and prognostic factors: molecular classification of breast cancers. Diagn Intervent Imaging 95 (2):227–233
34. Scimeca M, Giannini E, Antonacci C, Pistolese CA, Spagnoli LG, Bonanno E (2014) Microcalcifications in breast cancer: an active phenomenon mediated by epithelial cells with mesenchymal characteristics. BMC Cancer 14:286
35. Cox RF, Morgan MP (2013) Microcalcifications in breast cancer: lessons from physiological mineralization. Bone 53(2):437–450
36. Jing H, Yang Y, Nishikawa RM (2012) Retrieval boosted computer-aided diagnosis of clustered microcalcifications for breast cancer. Med Phys 39(2):676–685
37. Baker R, Rogers KD, Shepherd N, Stone N (2010) New relationships between breast microcalcifications and cancer. Br J Cancer 103(7):1034–1039
38. Uematsu T, Kasami M, Yuen S (2009) A cluster of microcalcifications: women with high risk for breast cancer versus other women. Breast Cancer 16(4):307–314
39. Karahaliou A, Skiadopoulos S, Boniatis I, Sakellaropoulos P, Likaki E, Panayiotakis G et al (2007) Texture analysis of tissue surrounding microcalcifications on mammograms for breast cancer diagnosis. Br J Radiol 80 (956):648–656
40. Kamitani T, Yabuuchi H, Soeda H, Matsuo Y, Okafuji T, Sakai S et al (2007) Detection of masses and microcalcifications of breast cancer on digital mammograms: comparison among hard-copy film, 3-megapixel liquid crystal display (LCD) monitors and 5-megapixel LCD monitors: an observer performance study. Eur Radiol 17(5):1365–1371
41. Burnside ES, Rubin DL, Fine JP, Shachter RD, Sisney GA, Leung WK (2006) Bayesian network to predict breast cancer risk of mammographic microcalcifications and reduce number of benign biopsy results: initial experience. Radiology 240(3):666–673
42. Jing H, Yang Y, Nishikawa RM (2012) Regularization in retrieval-driven classification of clustered microcalcifications for breast cancer. Int J Biomed Imaging 2012, id463408
43. Farshid G, Sullivan T, Downey P, Gill PG, Pieterse S (2011) Independent predictors of breast malignancy in screen-detected microcalcifications: biopsy results in 2545 cases. Br J Cancer 105(11):1669–1675
44. Hsieh SL, Hsieh SH, Cheng PH, Chen CH, Hsu KP, Lee IS et al (2012) Design ensemble machine learning model for breast cancer diagnosis. J Med Syst 36(5):2841–2847
45. Djebbari A, Liu Z, Phan S, Famili F (2008) An ensemble machine learning approach to predict survival in breast cancer. Int J Comput Biol Drug Des 1(3):275–294
46. Choi JY, Kim DH, Plataniotis KN, Ro YM (2014) Computer-aided detection (CAD) of

breast masses in mammography: combined detection and ensemble classification. Phys Med Biol 59(14):3697–3719

47. Ali S, Majid A, Khan A (2014) IDM-PhyChm-Ens: intelligent decision-making ensemble methodology for classification of human breast cancer using physicochemical properties of amino acids. Amino Acids 46(4):977–993
48. Krawczyk B, Schaefer G (2013) A pruned ensemble classifier for effective breast thermogram analysis. In: Annual international conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp 7120–7123
49. Luo ST, Cheng BW (2012) Diagnosing breast masses in digital mammography using feature selection and ensemble methods. J Med Syst 36 (2):569–577
50. Takemura A, Shimizu A, Hamamoto K (2010) Discrimination of breast tumors in ultrasonic images using an ensemble classifier based on the AdaBoost algorithm with feature selection. IEEE Trans Med Imaging 29(3):598–609
51. Vimieiro R, Moscato P (2014) Disclosed: an efficient depth-first, top-down algorithm for mining disjunctive closed itemsets in high-dimensional data. Inform Sci 280:171–187
52. Vimieiro R, Moscato P (2014) A new method for mining disjunctive emerging patterns in high-dimensional datasets using hypergraphs. Inform Syst 40:1–10

# Part III

## Computational Methods

# Chapter 18

# Inference Method for Developing Mathematical Models of Cell Signaling Pathways Using Proteomic Datasets

**Tianhai Tian and Jiangning Song**

## Abstract

The progress in proteomics technologies has led to a rapid accumulation of large-scale proteomic datasets in recent years, which provides an unprecedented opportunity and valuable resources to understand how living organisms perform necessary functions at systems levels. This work presents a computational method for designing mathematical models based on proteomic datasets. Using the mitogen-activated protein (MAP) kinase pathway as the test system, we first develop a mathematical model including the cytosolic and nuclear subsystems. A key step of modeling is to apply a genetic algorithm to infer unknown model parameters. Then the robustness property of mathematical models is used as a criterion to select appropriate rate constants from the estimated candidates. Moreover, quantitative information such as the absolute protein concentrations is used to further refine the mathematical model. The successful application of this inference method to the MAP kinase pathway suggests that it is a useful and powerful approach for developing accurate mathematical models to gain important insights into the regulatory mechanisms of cell signaling pathways.

**Key words** Cell signaling pathway, Reverse engineering, Proteomics, Robustness

## 1 Introduction

Proteomics is considered as the next crucial step to study biological systems in the post-genomic era, as it allows large-scale determination of genetic and cellular functions at the proteome level [1, 2]. The proteome is the complete repertoire of proteins, including posttranslational modifications (PTMs) that occur in a particular set of proteins. The purpose of proteomics research is to determine the relative or absolute amount of proteins presented in a biological sample. Advanced proteomic technologies, including mass spectrometry (MS), two-dimensional gel electrophoresis and protein arrays, provide powerful methods for analyzing protein samples. Proteomics technologies have emerged as potent tools for rapid identification of proteins in complex biological samples and characterization of PTMs and protein–protein interactions [3, 4].

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_18, 

An important application of MS-based proteomics is to characterize cell signaling cascades, which involve the binding of extracellular signaling molecules to cell-surface receptors, triggering events inside the cell [5]. Phosphorylation, a key reversible PTM, plays a key role in regulating protein functions and localizations in this process. Phosphoproteomics thus serves as a branch of proteomics with the purpose of identifying and characterizing proteins that contain a phosphate group as a PTM [5]. As a consequence, phosphoproteome studies are able to provide a global and integrated description of cellular signaling networks [6, 7]. However, the complex nature of the cell signaling pathways remains to be fully characterized as to how they are exactly regulated in vivo and what parameters are responsible for determining their dynamics [8]. To address these questions, mathematical modeling is a powerful approach for deducing regulatory principles and interpreting signal transduction mechanisms that underlie various cellular functions [9].

The lack of kinetic rates for mathematical modeling is a major challenge for developing systems biology approaches. These should, in principle, be measured by experiments or estimated from experimental data. However, due to the limited amount of experimental data, a commonly adopted approach in systems biology studies is to collect published experimental data obtained from different cell types under various conditions. Therefore, the progress in proteomics technologies and the rapid accumulation of proteomic data have offered an unprecedented opportunity to better understand how living organisms perform necessary functions at systems levels. From a systems biology perspective, the dynamic temporal data generated by phosphoproteomics experiments represent valuable resources for inferring unknown model parameters and modeling cell signaling networks [10]. However, to date, only limited work has been done to utilize the temporal dynamic proteome datasets for mathematical modeling of biological systems. This chapter presents a computational framework for developing accurate mathematical models using proteomic datasets.

## 2 Review of Modeling for the MAP Kinase Pathway

The mitogen-activated protein (MAP) kinase pathway is one of the most extensively studied signaling pathways. It communicates signals from the growth factor receptors on the cell surface to effector molecules located in the cytoplasm and nucleus. The MAP kinase cascade can be activated by the upstream input signal Ras protein, and comprises a set of three protein kinases: Raf, MEK, and ERK, together with a highly conserved molecular architecture that acts sequentially [11]. The activated MAP kinase is able to phosphorylate

multiple different substrates, including transcription factors, protein kinases, phospholipases, and cytoskeletal proteins, and regulate a wide range of physiological responses, including cell proliferation, differentiation, apoptosis, and tissue development. The signaling downstream of Ras protein has an incredible complexity, which includes positive and negative feedback loops, protein re-localization, signaling complex formation, and cross talk between parallel signaling pathways.

The EGF-regulated MAP kinase pathway is considered as the best-characterized signal transduction pathway. In the last two decades there has been a significant amount of experimental data published regarding signaling entities, regulatory interactions, kinase activities, protein absolute concentrations, and perturbation studies. Moreover, the advances in systems biology have led to the development of a large number of sophisticated mathematical models with various assumptions about the regulatory mechanisms at different levels as well as model parameters inferred from experimental data under various experimental conditions and from different cell types. Although the principal hierarchy of the signaling pathway and its activation sequence are well established, recent experimental studies have provided additional information on critical protein–protein interactions, regulatory loops, and spatiotemporal organization [12].

In the last decade, the MAP kinase pathway has often been used as a testable paradigm for interrogating systems biology approaches. In 1996, Huang and Ferrell developed the first mathematical model by focusing on the Ras-dependent activation of the MAP kinase module. The model could predict highly ultra-sensitive responses of the MAP kinase cascade and was then confirmed by experimentation [13]. The success of this work has stimulated a great deal of interest in developing kinetic models to provide testable predictions and novel insights into signaling events. For example, Bhalla et al. combined experiments and modeling to support the hypothesis that MAP kinase was involved in a bistable feedback loop [14]; Schoeberl et al. developed a mathematical model for the EGF-regulated MAP kinase pathway [15]; we demonstrated a critical function of Ras nanoclusters in generating high-fidelity signal transduction [16]; and a recent study investigated functional cross talks between the MAP kinase pathway and other signaling pathways [17]. In addition, we have developed a mathematical model that contains a nuclear subsystem of ERK kinase activation [18] and studied the robustness property of various kinase modules [19]. Nevertheless, the molecular mechanisms that underlie precise but robust control of MAP kinase signal intensity with a range of activation kinetics and diverse biological outcomes remain poorly understood. Using the MAP kinase pathway as the test system, this chapter discusses how to design a computational framework for developing accurate mathematical models of cell signaling pathway using proteomic datasets.

## 3 Methods

### 3.1 Experimental Data

Olsen et al. have recently applied an integrated phosphoproteomic technology to identify and quantitate the global in vivo phosphoproteome and its temporal dynamics upon growth-factor stimulation in human HeLa cells. In this study, human Hela cells were stimulated with 150 ng/ml of EGF for different time intervals. This proteome dataset includes the quantitative temporal activity ratios of 2244 proteins with a total of 6600 phosphorylation sites, and is available as an excel file in the supplementary information of the reference [6]. However, this dataset includes a proportion of missing values for quite a large number of proteins (*see* **Note 1**).

We used the proteomic data of the ARaf1 protein, the dual specificity mitogen-activated protein kinase kinase 2 (MEK) and the mitogen-activated protein kinase 1 (ERK) in the supplementary table. In this dataset, the kinase activities were measured at 0, 1, 5, 10, and 20 min. The activities of each kinase were further normalized by its activity at 5 min. While the activities of ARaf1 were obtained in the cytosol only, the activities of MEK and ERK were available in both the cytosol and nucleus. Since the kinase activities in the proteomic dataset were mostly available at five time points, we used the linear interpolation to generate kinase activities at another 16 time points during the time interval [0, 20] (min).

Additional experimental data were also available using Western blotting analysis and other experimental techniques in human HeLa cells [20]. Hela cells were stimulated with 50 ng/ml of EGF for different time periods. Therefore, both datasets in previous studies [6, 20] can be combined in our study. The Ras activity in Ref. [20] was used as the signal input of the MAP kinase module. The absolute kinase concentrations and the fractions of the activated kinases (at 5 min) in Ref. [20] were also used and led to the absolute activated kinase concentrations at 5 min shown in Table 1. The relative kinase activities in the proteomic study were then rescaled using the absolute activated kinase concentrations at 5 min. It is noteworthy that the Raf, MEK, and ERK kinase activities in Ref. [20] were utilized only to compare with the simulated kinase activities and served as evidence to validate the feasibility of the proposed modeling framework.

### 3.2 Development of Mathematical Model

Our MAP kinase pathway model comprises a cytosolic subsystem and a nuclear subsystem [18]. In the cytosolic subsystem, the Ras-GTP is the signal input of the MAP kinase cascade and activates Raf molecules in a single step. This activation is followed by sequential activation of the dual-specificity MAP kinase kinase (i.e., MEK) by Raf* (i.e., activated Raf) in a single-step processive module (*see* **Note 2**). The activated MEKpp (i.e., phosphorylated MEK at two residue positions) in turn activates ERK in a two-step distributive

**Table 1**
**Protein concentrations of the pathway models**

| | Initial condition of System 1 | Initial condition of Systems 2 | Max % of the activated kinase at 5 min in System 2 | Activated kinases at 5 min in System 2 |
|---|---|---|---|---|
| [Ras] | 1 | 0.4 [20] | | 0.4 |
| [Raf] | 1 | 0.013 [20] | | 0.013 |
| [Raf-P'ase] | 1 | 0.002 [15] | | |
| [MEK] | 1 | 1.4 [20] | 5 % [20] | 0.07 |
| [MEK-P'ase] | 1 | 0.14 [15] | | |
| [ERK] | 1 | 0.96 [20] | 50 % [20] | 0.48 |
| [ERK-P'ase] | 1 | 0.48 [15] | | |

System 1 is the model based on the proteomic data only with normalized protein concentrations, while System 2 is the model based on both proteomic and other experimental data with absolute protein concentrations. Except the variables in this table, the initial conditions of other variables are set as zeros. The concentrations of three phosphatases are calculated based on both the absolute kinase concentration in Ref. [20] and ratio of phosphatase concentration to the corresponding kinase concentration in Ref. [15]

module [21]. The activated ERKpp (i.e., phosphorylated ERK at two residue positions) is the signal output of the MAK kinase module. Both activated and inactivated MEK and ERK kinases diffuse between the cytosol and nucleus freely. In the nuclear subsystem, the activated MEKpp further activates the ERK kinase via the distributive two-step phosphorylation module. In addition, phosphatases, such as Raf-P'ase, MEK-P'ase, and ERK-P'ase, can respectively deactivate the activated Raf*, MEKpp, and ERKpp kinases at different subcellular locations.

The detailed process of kinase activation is described by a set of chemical reactions [18]. Briefly, the activated kinase (or phosphatase) K binds to its substrate S (or activated kinase Sp) to form a protein complex K-S (or K-Sp), which leads to the activated substrate Sp (or deactivated kinase S). Examples of these reactions are provided below:

1. Processive phosphorylation module of MEK kinase

$$\mathrm{Raf}^{*} + \mathrm{MEK} \underset{d_i}{\overset{a_i}{\rightleftarrows}} \mathrm{Raf}^{*} - \mathrm{MEK} \xrightarrow{k_i} \mathrm{Raf}^{*} + \mathrm{MEKpp} \qquad (1)$$

2. Distributive phosphorylation module of ERK kinase

$$\mathrm{MEKpp} + \mathrm{ERK} \underset{d_j}{\overset{a_j}{\rightleftarrows}} \mathrm{MEKpp} - \mathrm{ERK} \xrightarrow{k_j} \mathrm{MEKpp} + \mathrm{ERKp} \qquad (2)$$

$$\text{MEKpp} + \text{ERKp} \underset{d_j}{\overset{a_j}{\rightleftarrows}} \text{MEKpp} - \text{ERKp} \xrightarrow{k_j} \text{MEKpp} + \text{ERKpp} \quad (3)$$

3. Dephosphorylation reactions of activated ERK kinase

$$\text{ERKp} + \text{ERK-P'ase} \underset{d_j}{\overset{a_j}{\rightleftarrows}} \text{ERKp} - \text{ERK-P'ase} \xrightarrow{k_j} \text{ERK} + \text{ERK-P'ase} \quad (4)$$

$$\text{ERKpp} + \text{ERK-P'ase} \underset{d_j}{\overset{a_j}{\rightleftarrows}} \text{ERKpp} - \text{ERK-P'ase} \xrightarrow{k_j} \text{ERKp} + \text{ERK-P'ase} \quad (5)$$

where $a_i$, $d_i$ and $k_i$ represent protein binding, dissociation and activation rate constants, respectively. The diffusion of MEK kinase, for example, between the cytosolic and nuclear subsystems is represented by

$$\text{MEK} \underset{b_i}{\overset{f_i}{\rightleftarrows}} \text{N-MEK}, \quad (6)$$

where MEK and N-MEK are MEK kinases located in the cytosolic and nuclear subsystems, respectively, $f_i$ and $b_i$ are diffusion rate constants.

A mathematical model has been constructed according to the chemical rate equations of these chemical reactions [18]. For example, Reaction 1 leads to the differential equation for the dynamics of the Raf*-MEK complex, which is given by

$$\frac{d[\text{Raf}^* - \text{MEK}]}{dt} = a_i[\text{Raf}^*][\text{MEK}] - (d_i + k_i)[\text{Raf}^* - \text{MEK}] \quad (7)$$

This mathematical model comprises 33 differential equations which represent the dynamics of 33 variables in the system. To test all the possibilities of molecular mechanisms, we make no assumptions about the model rate constants and as a result there are 57 unknown reaction rate constants. A promising research topic is to develop sophisticated models with less unknown parameters (*see* **Note 3**).

### 3.3 Estimation of Model's Kinetic Rates

A genetic algorithm is used to estimate all model parameters. The MATLAB toolbox developed by Chipperfield et al. [22] is employed to infer the 57 unknown rate constants. It uses MATLAB functions to build a set of versatile routines for implementation of a wide range of genetic algorithms. The genetic algorithm is run over 500 generations for each rate estimate, and a population of 100 individuals in each generation is used. The values of rate constants are taken initially from the uniform distribution in the range of $[0, W_{\max}]$, and the value of $W_{\max}$ is fixed to 1000 for each rate constant.

The estimation error is measured by the weighted distance between the simulated kinase activities and experimental data. The weight of each kinase is determined by its corresponding maximal activity. The total error is calculated by

$$E = \sum_{i=1}^{N} \sum_{j=1}^{M} \frac{\left| x_i(t_j) - x_{ij}^{*} \right|}{max_j \left\{ x_i(t_j) \right\}} \tag{8}$$

where $x_{ij}^{*}$ and $x_i(t_j)$ are the simulated and experimentally measured activities of kinase $x_i$ at time point $t_j$, respectively.

First, we use the genetic algorithm to infer the model kinetic rates based on the proteomic dataset [6]. The corresponding model is termed System 1. The total concentration of each kinase or phosphatase is assumed to be one unit. The initial condition of the differential equation model is given in Table 1. To be consistent with the normalized kinase activities in the proteomic dataset [6], the simulated activity of each kinase is also normalized by its activity at 5 min; and we choose $max_j \left\{ x_i(t_j) \right\} = 1$ in Eq. (8) for calculating the error between the simulation and proteomic data. The parameter set that produces smaller simulation error with respect to the proteomics data is selected as the estimated model rate constants. Due to the local maxima issue of the genetic algorithm, we implement the genetic algorithm with different random seeds that lead to different estimates of the model's kinetic rates. Accordingly, we obtain 20 sets of estimated rate constants and select the top ten estimates with smaller simulation errors when compared to the proteomic data for further analysis. Next, we use the robustness property of the model as an additional criterion to select the optimal rate constants.

Figure 1 provides the simulation results of the MAP kinase pathway using the model that has the smallest estimation error. The corresponding estimated model parameters are given in the Supplementary Table 1 in Ref. [18]. To compare with the proteomic data, simulations are also normalized by the simulated kinase activity at 5 min. The total activity of MEK in Fig. 1c (ERK in Fig. 1d) is also normalized by the corresponding total kinase activity at 5 min. The results show that the simulated kinase activities match the Raf* activities in the cytosol (Fig. 1b) and ERKpp activities in both the cytosol and nucleus (Fig. 1f) very well. However, there is a large difference between the simulated MEK activities and proteomic data in Fig. 1c, possibly because of difference between the MEK kinase proteomic data in the cytosol and nucleus as well as noise in proteomic data (*see* **Note 4**). In addition, the simulated MEK activities in the nucleus are also in good agreement with the proteomic data.

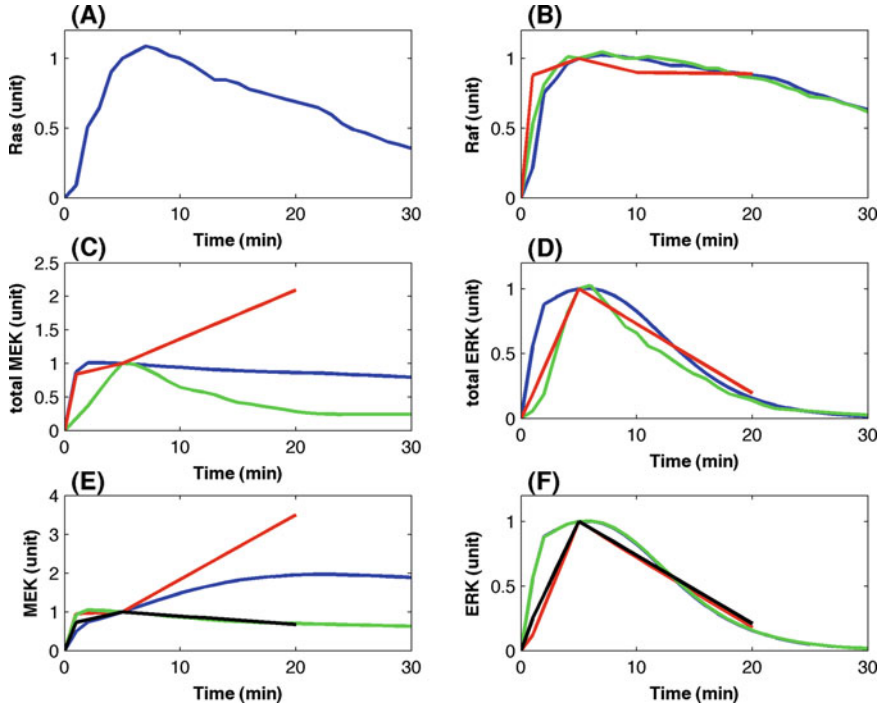



**Fig. 1** Simulations of the normalized kinase activities. (**a**) Normalized Ras activity as the signal input from [20]. (**b**) Raf activity; (**c**) Total MEK activity; and (**d**) Total ERK activity (*blue-line*: simulation; *green-line*: normalized Western blotting data [20]; *red-line*: proteomic data [6]). (**e**) MEK activity and (**f**) ERK activity at different locations (*blue-line*: simulation in the cytosol, *red-line*: proteomic data in the cytosol, *green-line*: simulation in the nucleus, *black-line*: proteomic data in the nucleus)

### *3.4 Robustness Property Analysis*

Robustness can be defined as the ability of a system to function correctly in the presence of both internal and external uncertainty. As robustness is a ubiquitously observed property of biological systems [23, 24], it has been widely used as an important measure to select the optimal network structure or model rate constants from estimated candidates, including the MAP kinase pathway [25, 26]. A formal and abstract definition of the robustness property, given by Kitano [27], is consistent with the general principle of the robustness property of complex systems, and has been widely used in the analysis of robustness properties of biological systems.

Here, we use the concept proposed by Kitano [27] to measure the robustness property of the model. The robustness property of a mathematical model with respect to a set of perturbations $P$ is defined as the average of an evaluation function $D^s_{a,P}$ of the system over all perturbations $p \in P$, weighted by the perturbation probabilities prob($p$), given by

$$R^s_{a,P} = \int_{p \in P} \text{prob}(p) D^s_{a,P} dp \tag{9}$$

Here, the following measure is used to evaluate the average behavior

$$R^{M}_{a,P} = \sum_{i,j} \left[ \int_{p \in P} \text{prob}(p) x_{ij}(p) dp \right] \tag{10}$$

which is the mean of kinase activities that should be close to the simulated kinase activity obtained from the unperturbed rate constants. In addition, the impact of perturbations on nominal behavior is defined by

$$R^{N}_{a,P} = \sum_{i,j} \left[ \int_{p \in P} \text{prob}(p) \left( \overline{x_{ij}(p)} - x_{ij}(p) \right)^2 dp \right] \tag{11}$$

where $x_{ij}(p)$ and $x_{ij}$ are the simulated activities of kinase $x_i$ at time point $t_j$ with perturbed and unperturbed rate constants, respectively, and $\overline{x_{ij}(p)}$ is the mean of $x_{ij}(p)$ over all the perturbed kinetic rates.

For each rate constant $k_i$, the perturbation is set to

$$\overline{k_i} = max\{k_i(1 + \mu(U - 0.5)), 0\} \tag{12}$$

with a uniformly distributed random variable $U(0,1)$ or

$$\overline{k_i} = max\{k_i(1 + \mu N), 0\} \tag{13}$$

with the standard Gaussian random variable $N(0,1)$. Here $\mu$ represents the perturbation strength.

To identify the best set of kinetic rates, we perform the robustness analysis of the mathematical model for the selected ten estimates of kinetic rates. For each set of model rate constants, we first use the estimated kinetic rates without any perturbation to produce a simulation that is used as the standard kinase activity. Then we perturb the value of each parameter using the generated random number. New simulations are obtained using the perturbed rate constants, and we then compare the new simulations with the standard simulation derived from the unperturbed model rate constants. The system with a particular set of rate constants is more stable if the difference between the new simulations and standard simulation is smaller. For each set of estimated rate constants, we generate 10,000 sets of perturbed rate constants using the uniformly distributed random variable and $\mu = 0.5$ in Eq. (12). According to Kitano's definition of robustness [27], we use the average behavior, which is the sum of all the means of each kinase activity as calculated by Eq. (8), and the nominal behavior, which is the sum of all the variances of each kinase activity as calculated by Eq. (9), as the measure of the robustness property.

Figure 2a and b illustrate the average behavior and nominal behavior of the mathematical model with ten different sets of estimated rate constants. We further test the robustness property

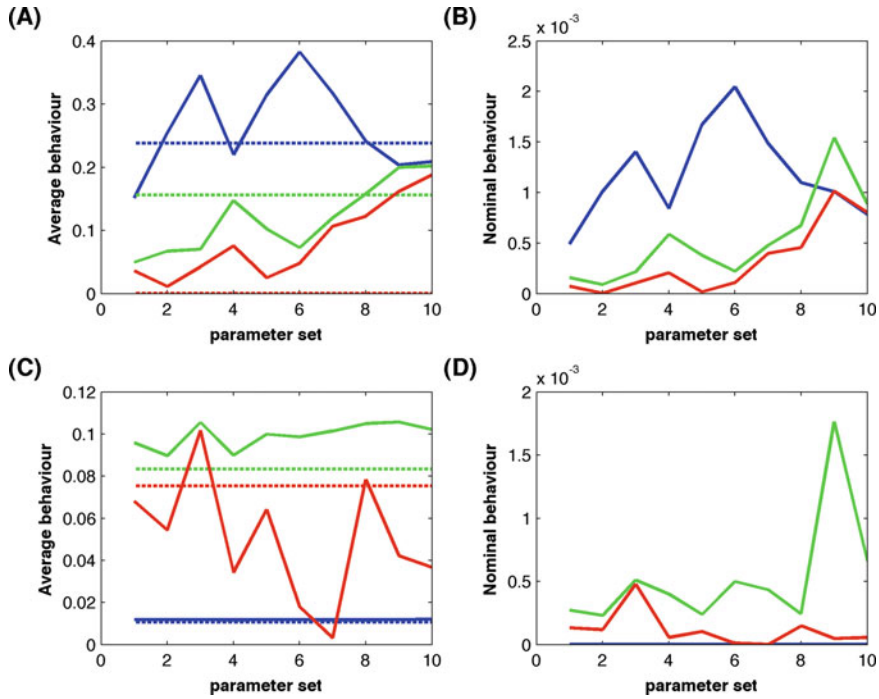



**Fig. 2** Robustness analysis. (**a**, **b**) Robustness analysis of the proposed model with ten sets of estimated kinetic rates derived from the normalized proteomic data. (**a**) The average behavior and (**b**) nominal behavior of the model with perturbed kinetic rates. (**c**, **d**) Robustness analysis of the proposed model with ten sets of estimated kinetic rates that were derived from more resources of experimental data. (**c**) The average behavior and (**d**) nominal behavior of the model with perturbed kinetic rates. (*Blue-line*: Raf; *green-line*: MEK, *red-line*: ERK. The horizontal dash lines in (**a**) and (**c**) are the simulated kinase activities based on the unperturbed model kinetic rates)

of this model in cases where the ten sets of estimated rate constants are perturbed by the Gaussian random variable with strength $\mu = 0.5$ in Eq. (11). In this case, the simulated perturbations of kinase activities are smaller than but still proportional to the corresponding perturbations in Fig. 2a and b (results not shown). In addition, we test the robustness property of the model using the ten sets of the rejected rate constants that generate simulations with larger errors. Simulation results suggest that there is no correlation between the model estimation error and robustness property.

To demonstrate the feasibility of our approach, we compare our simulated kinase activities in Fig. 1 with the kinase activities measured in vivo by Western blotting that are also normalized by its activity at 5 min [20]. Figure 1 shows that our computer simulation matches the Raf activity (Fig. 1b) and ERK activity (Fig. 1d) very well. However, the measured MEK activity in Fig. 1c is different from the proteomic data, and interestingly, the simulated MEK activity is located between the proteomic data and Western blotting data. The simulated MEK activity is smaller,

instead of being larger than the proteomic data, with the increasing time. This suggests that in the cell signaling cascade, the downstream signal activity may be used to calibrate the measurement errors of the upstream signals present in the proteomic datasets.

### 3.5 Model Refinement by Incorporating More Experimental Data

Although the normalized simulation results match the proteomic and experimental data very well in Fig. 1, the robustness analysis shown in Fig. 2 suggest that the percentages of the activated kinases are quite low. In addition, the fraction of the activated MEK kinase is larger than that of the activated ERK, which is contradictory to previous observations [15, 16, 20]. When using the absolute protein concentrations as the initial condition to simulate the model, we find a large difference between the predicted kinase activities and experimentally measured activities [20]. These results suggest that the normalized proteomic data might not be adequate for accurate inference of cell signaling pathway. To achieve better inference results, more experimental data, such as proteomic data, should be incorporated to the model [28] (*see* **Note 5**).

Therefore, to further refine the mathematical model, we use the experimentally measured absolute total concentrations of each kinase, which is also the initial condition of System 2 in Table 1, together with the information on the maximal percentages of MEK and ERK kinases that are activated by EGF stimulation [20], presented in Table 1. Then the normalized proteomic data (with kinase activity of unit one at 5 min) are rescaled by the absolute kinase activities in Table 1. The kinase activity is calculated by

$$[\text{kinase activity}] = [\text{proteomic kinase activity}] * [\text{kinase activity at 5 min in System 2}].$$

Note that the related activities of each kinase remain unchanged. In addition, the absolute concentrations of the three phosphatases, namely Raf-P'ase, MEK-P'ase, and ERK-P'ase, are also included in the model using experimentally measured data [15, 20], which is part of the initial conditions of System 2 in Table 1. Note that the Raf, MEK, and ERK kinase activities in Fig. 7 in Ref. [20] are only used to compare with the simulated kinase activities. As no further information is currently available regarding the distributions of activated MEK and ERK kinases at different subcellular locations, we use the proteomic data to generate normalized kinase activities in the cytosol and nucleus. In summary, the experimental data provide: (1) the absolute concentrations of the activated Raf, total MEK activity and total ERK activity in the first 20 min stimulated by Ras-GTP-binding; (2) the normalized activities of MEK and ERK kinases in the cytosol and nucleus in the first 20 min.

We use these experimental data to infer the model rate constants once again. To balance the errors of different kinases, the weight to scale the error of each kinase in Eq. (8) is the experimentally measured maximal activity of that kinase. However, for the

normalized activities of MEK and ERK in the cytosol and nucleus, the weight in Eq. (8) is set to unit one. In this case, we also derive 20 sets of estimated model rate constants by repeated implementations of the genetic algorithm and select the top ten sets with smaller estimation errors. For the top ten sets of model rate constants, we use the same method previously described to perform the robustness analysis, and select kinetic rates that lead to the best robustness property of the system as our final estimate [18].

The major advantage of incorporating more experimental data is that the mathematical model can realize experimental observations in a much more accurate manner and accordingly computer simulations are able to provide testable predictions regarding the regulatory mechanisms. Figure 3 displays the simulated system dynamics with the absolute kinase activities. We can see that computer simulations match the experimental data very well for the Raf activities in Fig. 3b, and the total ERK kinase activities in Fig. 3d. Moreover, the normalized MEK activity in the cytosol is very close to that in the nucleus, which is consistent with the experimental observation [20]. Another advantage of the refined model is that it has a very good robustness property in response to the perturbations of rate constants. Compared with the results in Fig. 2a and b, the numerical results in Fig. 2c and d suggest that the developed model based on the absolute kinase concentrations has a better robustness property than that based on the normalized kinase concentrations.

## 4 Framework for Developing Mathematical Models

In summary, the flowchart of our proposed modeling framework is illustrated in Fig. 4. The model may also include a graphical schematic structure of the signaling pathway, a list of all chemical reactions involved and a mathematical model that is a system of differential equations. The proteomic data are the time-course quantitative data of kinase activities. The other datasets include data resources obtained by other experimental techniques such as the FRET imaging and Western blotting. Using the genetic algorithm, we can obtain a number of candidate estimates of model parameters. The robustness analysis will be applied to the estimated candidates to identify the parameter set that has the best robustness property as our final parameter estimate. Finally, we can apply the built model to make testable predictions regarding the signal output under various system conditions.

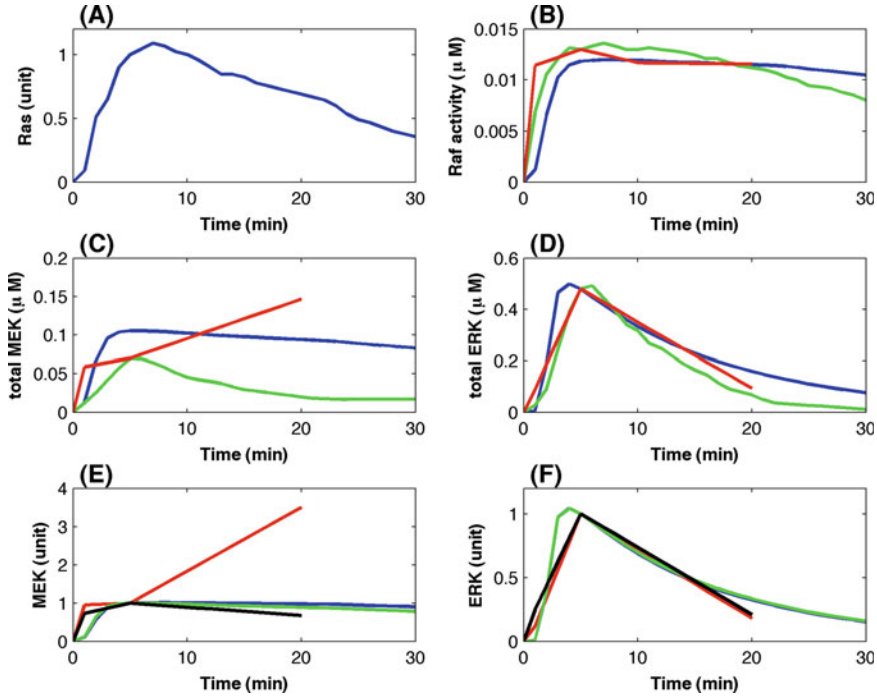



**Fig. 3** Simulated kinase activities based on the incorporation of proteomic data and Western blotting data. (**a**) Normalized Ras activity as the signal input [20]. (**b**) Raf activity; (**c**) Total MEK activity, and (**d**) Total ERK activity (*blue-line*: simulation; *green-line*: Western blotting data [20]; *red-line*: re-scaled proteomic data [6]). (**e**) MEK activity and (**f**) ERK activity at different locations (*blue-line*: simulation in the cytosol, *red-line*: proteomic data in the cytosol, *green-line*: simulation in the nucleus, *black-line*: proteomic data in the nucleus)

## 5 Notes

1. A major challenging issue in using proteomic data is missing values of kinase activities. Although a number of statistical methods have been proposed to estimate the missing value, the implementation of these methods will be extremely difficult, if the activity of a protein is completely unavailable in the proteomics dataset. An example is the Ras protein whose activities are not available in the proteomic dataset at all. Thus, other sources of biological data must be used to fill these data gaps.
2. The MAP kinase cascade comprises a set of three protein kinases, namely MAP kinase kinase kinase (MAPKKK or MAP3K), MAP kinase kinase (MAPKK or MAP2K), and MAP kinase (MAPK), with a highly conserved molecular architecture that acts sequentially [11]. In the MAP kinase pathway discussed in this chapter, these three kinases are Raf, MEK, and ERK proteins.
3. This chapter focuses on the issue of establishing mathematical models from proteomic datasets. However, only small amounts of experimental data are used in this work to refine the

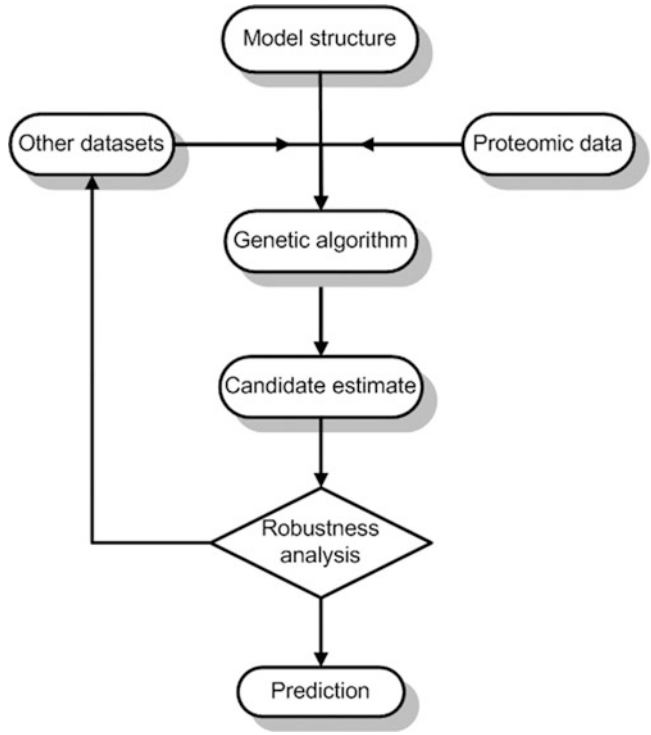



**Fig. 4** Flowchart of the proposed modeling framework for developing mathematical models of cell signaling pathways using proteomic datasets

developed mathematical model. As a result, simulation results suggest that integration of more experimental data could further improve the accuracy of the mathematical model substantially. Future work should thus include the development of more sophisticated models for cell signaling pathways through the combination of large-scale proteomic datasets, more experimental data, more signaling regulatory mechanisms as well as estimated model parameters.

4. Proteomics data suffer from considerable noise, including not only the technical noise arising from repeated experimental processes but also analysis noise [29]. Noise, such as the error of MEK kinase activity in this study, may result in significant variations during the inference of mathematical models. However, compared with the developed stochastic methods for interrogating the role of noise in microarray expression data [30, 31], the study of noise in proteomic data is still at the very early stage of development. More work is required to investigate the influence of noise on the development of mathematical models based on the noisy proteomic datasets.
5. Another issue is the normalization of proteomic data that causes the uncertainty of protein concentrations in mathematical

modeling. In this framework we first use the unified protein concentrations, where the information of the absolute protein concentrations is not known a priori. Although the normalized simulations can match the normalized experimental data very well, the simulated relative protein concentrations do not necessarily reflect the real scenario of signaling pathways, since the concentrations of proteins may also play an important role in modulating signaling transduction. Therefore, it is necessary to enrich the data by integrating more sources of experimental data prior to model simulation.

## Acknowledgments

This work was supported by grants from the Australian Research Council (ARC) (project number FT100100748), the National Health and Medical Research Council of Australia (NHMRC) (490989), and a Monash University Major Interdisciplinary Research Project (201402). TT is an ARC Future Fellow. JS is an NHMRC Peter Doherty Fellow.

### References

1. Aebersold R, Mann M (2003) Mass spectrometry-based proteomics. Nature 422:198–207
2. Hummon AB, Richmond TA, Verleyen P, Baggerman G, Huybrechts J et al (2006) From the genome to the proteome: uncovering peptides in the Apis brain. Science 314:647–649
3. Cox J, Mann M (2011) Quantitative, high-resolution proteomics for data-driven systems biology. Annu Rev Biochem 80:273–299
4. Cravatt BF, Simon GM, Yates JR 3rd (2007) The biological impact of mass-spectrometry-based proteomics. Nature 450:991–1000
5. Choudhary C, Mann M (2010) Decoding signalling networks by mass spectrometry-based proteomics. Nat Rev Mol Cell Biol 11:427–439
6. Olsen JV, Blagoev B, Gnad F, Macek B, Kumar C et al (2006) Global, in vivo, and site-specific phosphorylation dynamics in signaling networks. Cell 127:635–648
7. Oyama M, Kozuka-Hata H, Tasaki S, Semba K, Hattori S et al (2009) Temporal perturbation of tyrosine phosphoproteome dynamics reveals the system-wide regulatory networks. Mol Cell Proteomics 8:226–231
8. Heinrich R, Neel BG, Rapoport TA (2002) Mathematical models of protein kinase signal transduction. Mol Cell 9:957–970
9. Toni T, Stumpf MP (2010) Simulation-based model selection for dynamical systems in systems and population biology. Bioinformatics 26:104–110
10. Ahrens CH, Brunner E, Qeli E, Basler K, Aebersold R (2010) Generating and navigating proteome maps using mass spectrometry. Nat Rev Mol Cell Biol 11:789–801
11. Thomas GM, Huganir RL (2004) MAPK cascade signalling and synaptic plasticity. Nat Rev Neurosci 5:173–183
12. Takahashi K, Tanase-Nicola S, ten Wolde PR (2010) Spatio-temporal correlations can drastically change the response of a MAPK pathway. Proc Natl Acad Sci U S A 107:2473–2478
13. Huang CY, Ferrell JE Jr (1996) Ultrasensitivity in the mitogen-activated protein kinase cascade. Proc Natl Acad Sci U S A 93:10078–10083
14. Bhalla US, Ram PT, Iyengar R (2002) MAP kinase phosphatase as a locus of flexibility in a mitogen-activated protein kinase signaling network. Science 297:1018–1023
15. Schoeberl B, Eichler-Jonsson C, Gilles ED, Muller G (2002) Computational modeling of the dynamics of the MAP kinase cascade activated by surface and internalized EGF receptors. Nat Biotechnol 20:370–375

16. Tian T, Harding A, Inder K, Plowman S, Parton RG, Hancock JF (2007) Plasma membrane nanoswitches generate high-fidelity Ras signal transduction. Nat Cell Biol 9:905–914
17. Chen WW, Schoeberl B, Jasper PJ, Niepel M, Nielsen UB et al (2009) Input-output behavior of ErbB signaling pathways as revealed by a mass action model trained against dynamic data. Mol Syst Biol 5:239
18. Tian T, Song J (2012) Mathematical modelling of the MAP kinase pathway using proteomic datasets. PLoS One 7:e42230
19. Tian T, Harding A (2014) How MAP kinase modules function as robust, yet adaptable, circuits. Cell Cycle 13:2379–2390
20. Fujioka A, Terai K, Itoh RE, Aoki K, Nakamura T et al (2006) Dynamics of the Ras/ERK MAPK cascade as monitored by fluorescent probes. J Biol Chem 281:8917–8926
21. Schilling M, Maiwald T, Hengl S, Winter D, Kreutz C et al (2009) Theoretical and experimental analysis links isoform-specific ERK signalling to cell fate decisions. Mol Syst Biol 5:334
22. Chipperfield AJ, Fleming PJ, Fonseca CM (1994) Genetic algorithm tools for control systems engineering. In: Proc Adap Comput Engin Design Control, pp 128–133
23. Kitano H (2004) Biological robustness. Nat Rev Genet 5:826–837
24. Tian T, Olson S, Whitacre JM, Harding A (2011) The origins of cancer robustness and evolvability. Integr Biol (Camb) 3:17–30
25. Masel J, Siegal ML (2009) Robustness: mechanisms and consequences. Trend Genet 25:395–403
26. Citri A, Yarden Y (2006) EGF-ERBB signalling: towards the systems level. Nat Rev Mol Cell Biol 7:505–516
27. Kitano H (2007) Towards a theory of biological robustness. Mol Syst Biol 3:137
28. Fernandez Slezak D, Suarez C, Cecchi GA, Marshall G, Stolovitzky G (2010) When the optimal is not the best: parameter estimation in complex biological models. PLoS One 5: e13283
29. Karp NA, Lilley KS (2005) Maximising sensitivity for detecting changes in protein expression: experimental design using minimal CyDyes. Proteomics 5:3105–3115
30. Tu Y, Stolovitzky G, Klein U (2002) Quantitative noise analysis for gene expression microarray experiments. Proc Natl Acad Sci U S A 99:14031–14036
31. Tian T (2010) Stochastic models for inferring genetic regulation from microarray gene expression data. Biosystems 99:192–200

# Chapter 19

# Clustering

G.J. McLachlan, R.W. Bean, and S.K. Ng

## Abstract

Clustering techniques are used to arrange genes in some natural way, that is, to organize genes into groups or clusters with similar behavior across relevant tissue samples (or cell lines). These techniques can also be applied to tissues rather than genes. Methods such as hierarchical agglomerative clustering, *k*-means clustering, the self-organizing map, and model-based methods have been used. Here we focus on mixtures of normals to provide a model-based clustering of tissue samples (gene signatures) and of gene profiles, including time-course gene expression data.

**Key words** Clustering of tissue samples, Clustering of gene profiles, Hierarchical agglomerative methods, Partitional methods, *k*-means, Model-based methods, Normal mixture models, Mixtures of factor analyzers, Mixtures of linear mixed-effects models, Time-course data, Autoregressive random effects

## 1 Introduction

DNA microarray technology, first described in the mid-1990s, is a method to perform experiments on thousands of gene fragments in parallel. Its widespread use has led to a huge growth in the amount of expression data available. A variety of multivariate analysis methods has been used to explore these data for relationships among the genes and the tissue samples. Cluster analysis has been one of the most frequently used methods for these purposes. It has demonstrated its utility in the elucidation of unknown gene function, the validation of gene discoveries, and the interpretation of biological processes; *see* [1, 2] for examples.

The main goal of microarray analysis of many diseases, in particular of unclassified cancer, is to identify as yet unclassified cancer subtypes for subsequent validation and prediction, and ultimately to develop individualized prognosis and therapy. Limiting factors include the difficulties of tissue acquisition and the expense of microarray experiments. Thus, often microarray studies attempt to perform a cluster analysis of a small number of tumor samples on

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_19, 

the basis of a large number of genes, and can result in gene-to-sample ratios of approximately 100-fold.

Many researchers have explored the use of clustering techniques to arrange genes in some natural order, that is, to organize genes into clusters with similar behavior across relevant tissue samples (or cell lines). Although a cluster does not automatically correspond to a pathway, it is a reasonable approximation that genes in the same cluster have something to do with each other or are directly involved in the same pathway.

It can be seen there are two distinct but related clustering problems with microarray data. One problem concerns the clustering of the tissues on the basis of the genes; the other concerns the clustering of the genes on the basis of the tissues. This duality in cluster analysis is quite common. In the present context of microarray data, one may be interested in grouping tissues (patients) with similar expression values or in grouping genes on patients with similar types of tumors or similar survival rates.

One of the difficulties of clustering is that the notion of a cluster is vague. A useful way to think about the different clustering procedures is in terms of the shape of the clusters produced [3]. The majority of the existing clustering methods assume that a similarity measure or metric is known a priori; often the Euclidean metric is used. But clearly, it would be more appropriate to use a metric that depends on the shape of the clusters. As pointed out by [4], the difficulty is that the shape of the clusters is not known until the clusters have been found, and the clusters cannot be effectively identified unless the shapes are known.

Before we proceed to consider the clustering of microarray data, we give a brief account of clustering in a general context. For a more detailed account of cluster analysis, the reader is referred to the many books that either consider or are devoted exclusively to this topic; for example, [5–9] and [10, Chapter 7]. A recent review article on clustering is [11].

### 1.1 Brief Review of Some Clustering Methods

Cluster analysis is concerned with grouping a number ($n$) of entities into a smaller number ($g$) of groups on the basis of observations measured on some variables associated with each entity. We let $\boldsymbol{y}_j = \left(y_{1j}, \ldots, y_{pj}\right)^T$ be the observation or feature vector containing the values of $p$ measurements $y_{1j}, \ldots, y_{pj}$ made on the $j$th entity ($j = 1, \ldots, n$) to be clustered. These data can be organized as a matrix,

$$\boldsymbol{Y}_{p\times n} = \left(\left(y_{vj}\right)\right); \tag{1}$$

that is, the $j$th column of $\boldsymbol{Y}_{p\times n}$ is the observation vector $\boldsymbol{y}_j$.

In discriminant analysis (supervised learning), the data are classified with respect to $g$ known classes and the intent is to form

a classifier or prediction rule on the basis of these classified data for assigning an unclassified entity to one of the $g$ classes on the basis of its feature vector. In contrast to discriminant analysis, in cluster analysis (unsupervised learning) there is no prior information on the group structure of the data or, in the case where it is known that the population consists of a number of classes, there are no data of known origin with respect to the classes. The clustering problem falls into two main categories which overlap to some extent [12]:

1. What is the best way of dividing the entities into a given number of groups, where there is no implication that the resulting groups are in any sense a natural division of the data. This is sometimes called dissection or segmentation.
2. What is the best way to find a natural subdivision of the entities into groups. Here by natural clusters, it is meant that the clusters can be described as continuous regions of the feature space containing a relatively high density of points, separated from other such regions by regions containing a relatively low density of points [5]. It is therefore intended that natural clusters possess the two intuitive qualities of internal cohesion and external isolation [13].

Sometimes the distinction between the search for naturally occurring clusters as in (2) and other groupings as in (1) is stressed; *see*, for example, [14]. But often it is not made, particularly as most methods for finding natural clusters are also useful for segmenting the data. Essentially, all methods of cluster analysis attempt to imitate what the eye and brain do so well in $p = 2$ dimensions. For example, in the scatter plot (Fig. 1) of the expression values of two smooth muscle related genes on ten tumors and ten normal tissues from the colon cancer data of [15], it is very easy to detect the presence of two clusters of equal size without making the meaning of the term "cluster" explicit.

Clustering methods can be categorized broadly as being hierarchical or nonhierarchical. With a method in the former category, every cluster obtained at any stage is a merger or split of clusters obtained at the previous stage. Hierarchical methods can be implemented in a so-called agglomerative manner (bottom-up), starting with $g = n$ clusters or in a divisive manner (top-down), starting with the $n$ entities to be clustered as a single cluster. In practice, divisive methods can be computationally prohibitive unless the sample size $n$ is very small. For instance, there are $2^{(n-1)} - 1$ ways of making the first subdivision. Hence hierarchical methods are usually implemented in an agglomerative manner, as to be discussed further in the next section. In [16], a hybrid clustering method was proposed that combines the strengths of bottom-up hierarchical clustering with that of top-down clustering. The first

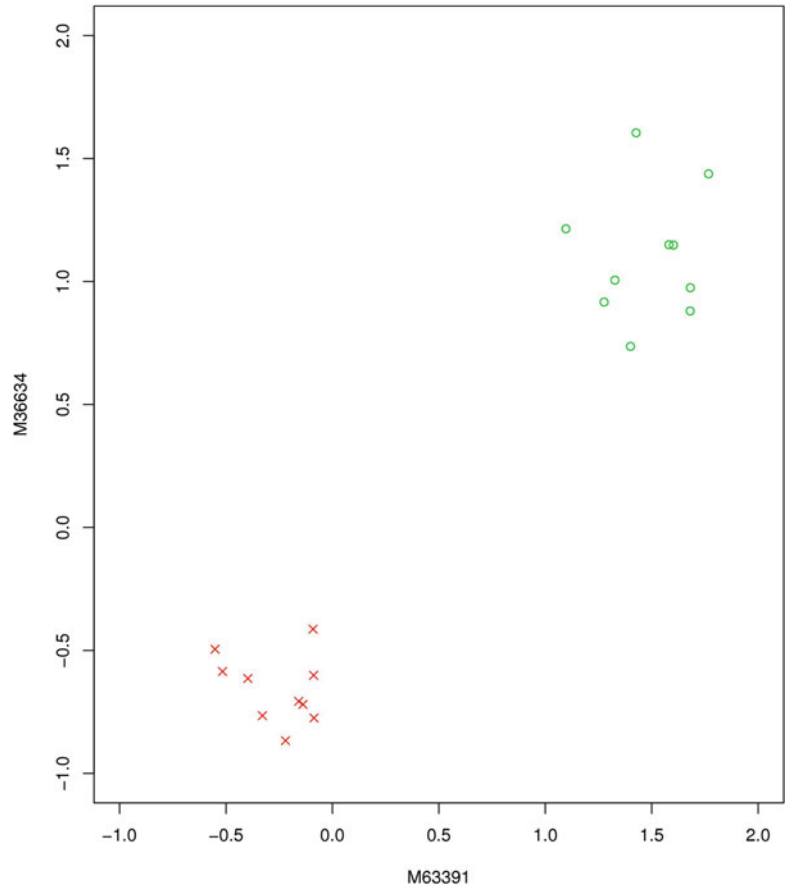



**Fig. 1** Scatter plot of the expression values of two genes on ten colon cancer tumors (*times*) and ten normal tissues (*open circle*)

method is good at identifying small clusters, but not large ones; the strengths are reversed for top-down clustering.

One of the most popular nonhierarchical methods of clustering is $k$-means, where "$k$" refers to the number of clusters to be imposed on the data. It seeks to find $k = g$ clusters that minimize the sum of the squared Euclidean distances between each observation $\boldsymbol{y}_j$ and its respective cluster mean; that is, it seeks to minimize the trace of $\boldsymbol{W}$, tr $\boldsymbol{W}$, where

$$\boldsymbol{W} = \sum_{i=1}^{g} \sum_{j=1}^{n} z_{ij} \left( y_j - \overline{y}_i \right) \left( y_j - \overline{y}_i \right)^T \tag{2}$$

is the pooled within-cluster sums of squares and products matrix, and

$$\bar{\boldsymbol{y}}_i = \sum_{j=1}^{n} \boldsymbol{y}_j / \sum_{j=1}^{n} z_{ij} \tag{3}$$

is the sample mean of the $i$th cluster. Here $z_{ij}$ is a zero-one indicator variable that is one or zero, according as $\boldsymbol{y}_j$ belongs or does not belong to the $i$th cluster ($i = 1, \ldots, g; j = 1, \ldots, n$). It is impossible to consider all partitions of the $n$ observations into $g$ clusters unless $n$ is very small, since the number of such partitions with nonempty clusters is the Stirling number of the second kind,

$$\frac{1}{g!} \sum_{i=0}^{g} (-1)^{(g-i)} \binom{g}{i} i^n \tag{4}$$

which can be approximated by $g^n/g!$; *see* [8]. In practice, $k$-means is therefore implemented by iteratively moving points between clusters so as to minimize tr $\boldsymbol{W}$. In its simplest form, each observation $\boldsymbol{y}_j$ is assigned to the cluster with the nearest center (sample mean) and then the center of the cluster is updated before moving on to the next observation. Often the centers are estimated initially by selecting $k$ points at random from the sample to be clustered.

Other partitioning methods have been developed, including $k$-medoids [8], which is similar to $k$-means, but constrains each cluster center to be one of the observations $\boldsymbol{y}_j$. The self-organizing map [17] is similar to $k$-means, but the cluster centers are constrained to lie on a (two-dimensional) lattice. It is well known that $k$-means tends to lead to spherical clusters since it is predicated on normal clusters with (equal) spherical covariance matrices. One way to achieve elliptical clusters is to seek clusters that minimize the determinant of $\boldsymbol{W}$, $|\boldsymbol{W}|$, rather than its trace, as in [18]; *see also* [19] who derived this criterion under certain assumptions of normality for the clusters.

In the absence of any prior knowledge of the metric, it is reasonable to adopt a clustering procedure that is invariant under affine transformations of the data; that is, invariant under transformations of the data of the form,

$$\boldsymbol{y} \to \boldsymbol{C}\boldsymbol{y} + \boldsymbol{a} \tag{5}$$

where $\boldsymbol{C}$ is a nonsingular matrix. If the clustering of a procedure is invariant under (5) for only diagonal $\boldsymbol{C}$, then it is invariant under change of measuring units but not rotations. But as commented upon in [20], this form of invariance is more compelling than affine invariance. The clustering produced by minimization of $|\boldsymbol{W}|$ is affine invariant.

In the statistical and pattern recognition literature in recent times, attention has been focussed on model-based clustering via mixtures of normal densities. With this approach, each observation vector $\boldsymbol{y}_j$ is assumed to have a $g$-component normal mixture density,

$$f\left(\boldsymbol{y}_j;\boldsymbol{\Psi}\right)=\sum_{i=1}^{g}\pi_i\phi\left(\boldsymbol{y}_j;\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i\right), \tag{6}$$

where $\phi(\boldsymbol{y};\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)$ denotes the $p$-variate normal density function with mean $\boldsymbol{\mu}_i$ and covariance matrix $\boldsymbol{\Sigma}_i$, and the $\boldsymbol{\pi}_i$ denote the mixing proportions, which are nonnegative and sum to one. Here the vector $\boldsymbol{\Psi}$ of unknown parameters consists of the mixing proportions $\boldsymbol{\pi}_i$, the elements of the component means $\boldsymbol{\mu}_i$, and the distinct elements of the component-covariance matrix $\boldsymbol{\Sigma}_i$, and it can be estimated by its maximum likelihood estimate calculated via the EM algorithm; *see* [21, 22]. This approach gives a probabilistic clustering defined in terms of the estimated posterior probabilities of component membership $\tau_i\left(\boldsymbol{y}_j;\hat{\boldsymbol{\Psi}}\right)$, where $\tau_i\left(\boldsymbol{y}_j;\hat{\boldsymbol{\Psi}}\right)$ denotes the posterior probability that the $j$th feature vector with observed value $\boldsymbol{y}_j$ belongs to the $i$th component of the mixture ($i=1,\ldots,g; j=1,\ldots,n$). Using Bayes' theorem, it can be expressed as

$$\tau_i\left(\boldsymbol{y}_j;\boldsymbol{\Psi}\right)=\frac{\pi_i\phi\left(\boldsymbol{y}_j;\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i\right)}{\sum_{h=1}^{g}\pi_h\phi\left(\boldsymbol{y}_j;\boldsymbol{\mu}_h,\boldsymbol{\Sigma}_h\right)}. \tag{7}$$

It can be seen that with this approach, we can have a "soft" clustering, whereby each observation may partly belong to more than one cluster. An outright clustering can be obtained by assigning $\boldsymbol{y}_j$ to the component to which it has the greatest estimated posterior probability of belonging. The number of components $g$ in the normal mixture model (Eq. 6) has to be specified in advance (*see* **Note 1**).

As noted in [23], "Clustering methods based on such mixture models allow estimation and hypothesis testing within the framework of standard statistical theory." Previously, Marriott [12, page 70) had noted that the mixture likelihood-based approach "is about the only clustering technique that is entirely satisfactory from the mathematical point of view. It assumes a well-defined mathematical model, investigates it by well-established statistical techniques, and provides a test of significance for the results." One potential drawback with this approach is that normality is assumed for the cluster distributions. However, this assumption would appear to be reasonable for the clustering of microarray data after appropriate normalization.

One attractive feature of adopting mixture models with elliptically symmetric components such as the normal or its more robust version in the form of the $t$ density [22] is that the implied clustering is invariant under affine transformations in Eq. (5). Also, in the case where the components of the mixture correspond to externally defined subpopulations, the unknown parameter vector $\boldsymbol{\Psi}$ can be estimated consistently by a sequence of roots of the likelihood

equation. Note that this is not the case if a criterion such as minimizing $|\boldsymbol{W}|$ is used.

In the above, we have focussed exclusively on methods that are applicable for the clustering of the observations and the variables considered separately; that is, in the context of clustering microarray data, methods that would be suitable for clustering the tissue samples and the genes considered separately rather than simultaneously. Pollard and van der Laan [24] proposed a statistical framework for two-way clustering; *see also* [25] and the references therein for earlier approaches to this problem. More recently, [26] reported some results on two-way clustering (biclustering) of tissues and genes. In their work, they obtained similar results to those obtained when the tissues and the genes were clustered separately.

## 2 Methods

Although biological experiments vary considerably in their design, the data generated by microarray experiments can be viewed as a matrix of expression levels. For $M$ microarray experiments (corresponding to $M$ tissue samples), where we measure the expression levels of $N$ genes in each experiment, the results can be represented by a $N \times M$ matrix. For each tissue, we can consider the expression levels of the $N$ genes, called its *expression signature*. Conversely, for each gene, we can consider its expression levels across the different tissue samples, called its *expression profile*. The $M$ tissue samples might correspond to each of $M$ different patients or, say, to samples from a single patient taken at $M$ different time points. The $N \times M$ matrix is portrayed in Fig. 2, where each sample

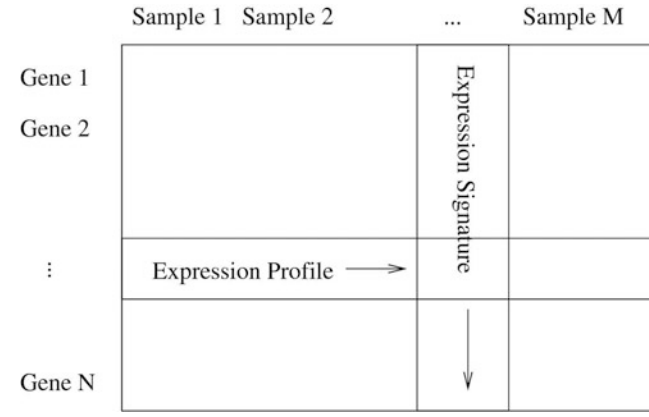



**Fig. 2** Gene expression data from $M$ microarray experiments represented as a matrix of expression levels with the $N$ rows corresponding to the $N$ genes and the $M$ columns to the $M$ tissue samples

represents a separate microarray experiment and generates a set of $N$ expression levels, one for each gene.

Against the above background of clustering methods in a general context as given in the previous section, we now consider their application to microarray data, concentrating on a model-based approach using normal mixtures. But firstly, we consider the application of hierarchical agglomerative methods, given their extensive use for this purpose in bioinformatics.

### 2.1 Clustering of Tissues: Hierarchical Methods

For the clustering of the tissue samples, the microarray data portrayed in Fig. 2 are in the form of the matrix in Eq. (1) with $n = M$ and $p = N$, and the observation vector $\boldsymbol{y}_j$ corresponds to the expression signature for the $j$th tissue sample. In statistics, it is usual to refer to the entirety of the tissue samples as the sample, whereas the biologists tend to refer to each individual expression signature as a sample; we follow the latter practice here.

The commonly used hierarchical agglomerative methods can be applied directly to this matrix to cluster the tissue samples, since they can be implemented by consideration of the matrix of proximities, or equivalently, the distances, between each pair of observations. Thus they require only $O(n^2)$ or at worst $O(n^3)$ calculations, where $n = M$ and the number $M$ of tissue samples is limited usually to being less than 100. The situation would be different with the clustering of the genes as then $n = N$ and the number $N$ of genes could be in the tens of thousands.

In order to compute the pairwise distances between the observations, one needs to select an appropriate distance metric. Metrics that are used include Euclidean distance and the Pearson correlation coefficient, although the latter is equivalent to the former if the observations have been normalized beforehand to have zero means and unit variances. Having selected a distance measure for the observations, there is a need to specify a linkage metric between clusters. Some commonly used metrics include single linkage, complete linkage, average linkage, and centroid linkage. With single linkage, the distance between two clusters is defined by the distance between the two nearest observations (one from each cluster), while with complete linkage, the cluster distance is defined in terms of the distance between the two most distant observations (one from each cluster). Average linkage is defined in terms of the average of the $n_1 n_2$ distances between all possible pairs of observations (one from each cluster), where $n_1$ and $n_2$ denote the number of observations in the two clusters in question. For centroid linkage, the distance between two clusters is the distance between the cluster centroids (sample means). Another commonly used method is Ward's procedure [27], which joins clusters so as to minimize the within-cluster variance (the trace of $\boldsymbol{W}$). Lance and Williams [28] have presented a simple linear system of equations as a unifying framework for these different linkage measures. Eisen et al. [2]

were the first to apply cluster analysis to microarray data, using average linkage with a correlation-based metric. The nested clusters produced by an hierarchical method of clustering can be portrayed in a tree diagram, in which the extremities (usually shown at the bottom) represent the individual observations, and the branching of the tree gives the order of joining together. The height at which clusters of points are joined corresponds to the distance between the clusters. However, it is not clear in general how to choose the number of clusters.

To illustrate hierarchical agglomerative clustering, we use nested polygons in Fig. 3 to show the clusters obtained by applying it to six bivariate points, using single-linkage with Euclidean distance as the distance measure. It can be seen that the cluster of observations 5 and 6 is considerably closer to the cluster of 1 and 2 than observation 4 is.

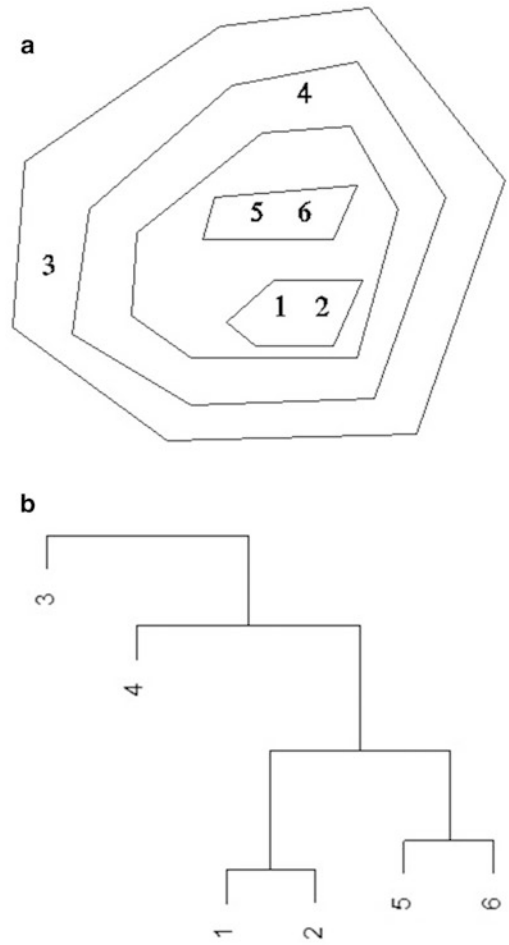



**Fig. 3** An illustrative example of hierarchical agglomerative clustering

There is no reason why the clusters should be hierarchical for microarray data. It is true that if there is a clear, unequivocal grouping, with little or no overlap between the groups, any method will reach this grouping. But as pointed out by [12], "hierarchical methods are not primarily adapted to finding groups." For instance, if the division into $g = 2$ groups given by some hierarchical method is optimum with respect to some criterion, then the subsequent division into $g = 3$ groups is unlikely to be so. This is due to the restriction that one of the groups must be the same in both the $g = 2$ and $g = 3$ clusterings. As explained by [12], this restriction is not a natural one to impose if the purpose is to find a natural grouping of the data. In the sequel, we therefore focus on nonhierarchical methods of clustering. As advocated by [12, page 67), "it is better to consider the clustering problem ab initio, without imposing any conditions."

### 2.2 Clustering of Tissues: Normal Mixtures

More recently, increasing attention is being given to model-based methods of clustering of microarray data [29–32]. However, the normal mixture model (Eq. 6) cannot be directly fitted to the tissue samples if the number of genes $p$ used in the expression signature is large. This is because the component-covariance matrices $\boldsymbol{\Sigma}_i$ are highly parameterized with $\frac{1}{2}p(p+1)$ distinct elements each. A simple way of proceeding in the clustering of high-dimensional data would be to take the component-covariance matrices $\boldsymbol{\Sigma}_i$ to be diagonal. But this leads to clusters whose axes are aligned with those of the feature space, whereas in practice the clusters are of arbitrary orientation. For instance, taking the $\boldsymbol{\Sigma}_i$ to be a common multiple of the identity matrix leads to a soft-version of $k$-means which produces spherical clusters.

Banfield and Raftery [33] introduced a parameterization of the component-covariance matrix $\boldsymbol{\Sigma}_i$ based on a variant of the standard spectral decomposition of $\boldsymbol{\Sigma}_i (i = 1, \ldots g)$. But if $p$ is large relative to the sample size $n$, it may not be possible to use this decomposition to infer an appropriate model for the component-covariance matrices. Even if it were possible, the results may not be reliable due to potential problems with near-singular estimates of the component-covariance matrices when $p$ is large relative to $n$.

Hence, in fitting normal mixture models with unrestricted component-covariance matrices to high-dimensional data, we need to consider first some form of dimension reduction and/or some form of regularization. A common approach to reducing the number of dimensions is to perform a principal component analysis (PCA). However, the latter provides only a global linear model for the representation of the data in a lower-dimensional subspace. Thus it has limited scope in revealing group structure in a data set. A global nonlinear approach can be obtained by postulating a finite mixture of linear (factor) submodels for the distribution of the full observation vector $\boldsymbol{y}_j$ given a relatively small number of

(unobservable) factors. That is, we can provide a local dimensionality reduction method by a mixture of factor analyzers model, which is given by Eq. (6) by imposing on the component-covariance matrix $\boldsymbol{\Sigma}_i$, the constraint

$$\boldsymbol{\Sigma}_i = \boldsymbol{B}_i \boldsymbol{B}_i^T + \boldsymbol{D}_i \quad (i = 1, \ldots, g), \tag{8}$$

where $\boldsymbol{B}_i$ is a $p \times q$ matrix of factor loadings and $\boldsymbol{D}_i$ is a diagonal matrix ($i = 1, \ldots, g$). We can think of the use of this mixture of factor analyzers model as being purely a method of regularization. But in the present context, it might be possible to make a case for it being a reasonable model for the correlation structure between the genes. This model implies that the latter can be explained by the linear dependence of the genes on a small number of latent (unobservable variables) specific to each component.

The EMMIX-GENE program of [31] has been designed for the clustering of tissue samples via mixtures of factor analyzers. In practice we may wish to work with a subset of the available genes, particularly as the fitting of a mixture of factor analyzers will involve a considerable amount of computation time for an extremely large number of genes. Indeed, the simultaneous use of too many genes in the cluster analysis may serve only to create noise that masks the effect of a smaller number of genes. Also, the intent of the cluster analysis may not be to produce a clustering of the tissues on the basis of all the available genes, but rather to discover and study different clusterings of the tissues corresponding to different subsets of the genes [24, 34]. As explained in [35], the tissues (cell lines or biological samples) may cluster according to cell or tissue type (for example, cancerous or healthy) or according to cancer type (for example, breast cancer or melanoma). However, the same samples may cluster differently according to other cellular characteristics, such as progression through the cell cycle, drug metabolism, mutation, growth rate, or interferon response, all of which have a genetic basis.

Therefore, the EMMIX-GENE procedure has two optional steps before the final step of clustering the tissues. The first step considers the selection of a subset of relevant genes from the available set of genes by screening the genes on an individual basis to eliminate those which are of little use in clustering the tissue samples. The usefulness of a given gene to the clustering process can be assessed formally by a test of the null hypothesis that it has a single component normal distribution over the tissue samples (*see* **Note 2**). Even after this step has been completed, there may still be too many genes remaining. Thus there is a second step in EMMIX-GENE in which the retained gene profiles are clustered (after standardization) into a number of groups on the basis of Euclidean distance so that genes with similar profiles are put into the same group. In general, care has to be taken with the scaling of variables

before clustering of the observations, as the nature of the variables can be intrinsically different. In the present context the variables (gene expressions) are measured on the same scale. Also, as noted above, the clustering of the observations (tissues) via normal mixture models is invariant under changes in scale and location. The clustering of the tissue samples can be carried out on the basis of the groups considered individually using some or all of the genes within a group or collectively. For the latter, we can replace each group by a representative (a metagene) such as the sample mean as in the EMMIX-GENE procedure.

To illustrate this approach, we applied the EMMIX-GENE procedure to the colon cancer data of [15]. It consists of $n = 2000$ genes and $p = 62$ columns denoting 40 tumors and 22 normal tissues. After applying the selection step to this set, there were 446 genes remaining in the set. The remaining genes were then clustered into 20 groups, which were ranked on the basis of $-2\log\lambda$, where $\lambda$ is the likelihood ratio statistic for testing $g = 1$ versus $g = 2$ components in the mixture model. The heat map of the second ranked group $G_2$ is shown in Fig. 4. The clustering of the tissues on the basis of the 24 genes in $G_2$ resulted in a partition of the tissues in which one cluster contains 37 tumors (1–29, 31–32, 34–35, 37–40) and 3 normals (48, 58, 60), and the other cluster contains 3 tumors (30, 33, 36) and 19 normals (41–47, 49–57, 59, 61–62). This corresponds to an error rate of 6 out of 62 tissues compared to the "true" classification given in [15]. (This is

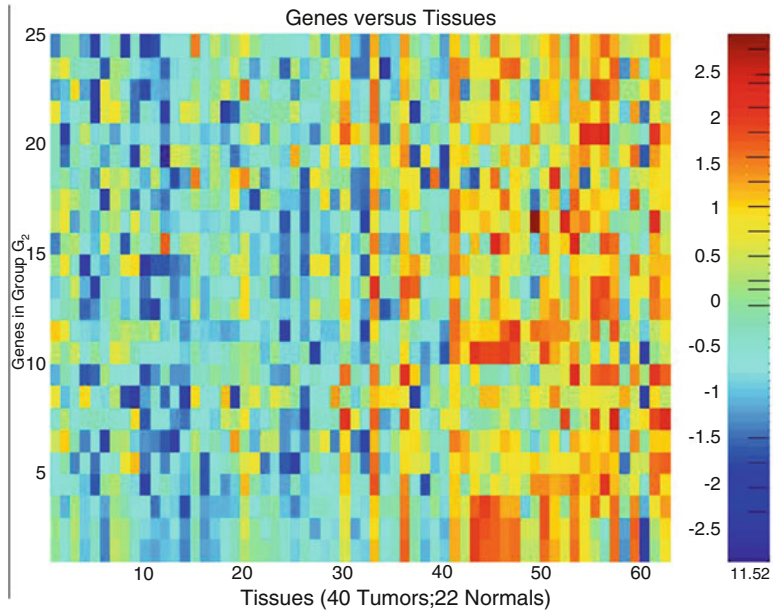



**Fig. 4** Heat map of 24 genes in group $G_2$ on 40 tumor and 22 normal tissues in Alon data

why here we examine the heat map of $G_2$ instead of $G_1$.) For further details about the results of the tissue clustering procedure on this data set, *see* [31].

### 2.3 Clustering of Gene Profiles

In order to cluster gene profiles, it might seem possible just to interchange rows and columns in the data matrix in Eq. (1). But with most applications of cluster analysis in practice it is assumed that

(a) there are no replications on any particular entity specifically identified as such;

(b) all the observations on the entities are independent of one another.

These assumptions should hold for the clustering of the tissue samples, although the tissue samples have been known to be correlated for different tissues due to flawed experimental conditions. However, condition (b) will not hold for the clustering of gene profiles, since not all the genes are independently distributed, and condition (a) will generally not hold either as the gene profiles may be measured over time or on technical replicates. While this correlated structure can be incorporated into the normal mixture model in Eq. (6) by appropriate specification of the component-covariance matrices $\boldsymbol{\Sigma}_i$, it is difficult to fit the model under such specifications. For example, the M-step (the maximization step of the EM algorithm) may not exist in closed form. Accordingly, we now consider the EMMIX-WIRE model of Ng et al. [36], who adopt conditionally a mixture of linear mixed models to specify this correlation structure among the tissue samples and to allow for correlations among the genes. It also enables covariate information to be incorporated into the clustering process.

For a gene microarray experiment with repeated measurements, we have for the $j$th gene ($j = 1,\ldots,n$), when $n = N$, a feature vector (profile vector) $\boldsymbol{y}_j = \left(\boldsymbol{y}_{1j}^T,\ldots,\boldsymbol{y}_{tj}^T\right)^T$, where $t$ is the number of distinct tissues in the experiment and

$$\boldsymbol{y}_{lj} = \left(y_{l1j},\ldots,y_{lrj}\right)^T \qquad (l = 1,\ldots,t)$$

contains the $r$ replications on the $j$th gene from the $l$th tissue. Note that here, the $r$ replications can also be time points. The dimension $p$ of the profile vector $\boldsymbol{y}_j$ is equal to the number of microarray experiments, $p = rt$. Conditional on its membership of the $i$th component of the mixture, the EMMIX-WIRE procedure assumes that $\boldsymbol{y}_j$ follows a linear mixed-effects model (LMM),

$$\boldsymbol{y}_j = \boldsymbol{X}\boldsymbol{\beta}_i + \boldsymbol{U}\boldsymbol{b}_{ij} + \boldsymbol{V}\boldsymbol{c}_i + \boldsymbol{\varepsilon}_{ij}, \tag{9}$$

where the elements of $\boldsymbol{\beta}_i$ (a $m$-dimensional vector) are fixed effects (unknown constants) ($i = 1,\ldots,g$). In Eq. (9), $\boldsymbol{b}_{ij}$ (a $q_b$-dimensional vector) and $\boldsymbol{c}_i$ (a $q_c$-dimensional vector) represent the unobservable

gene- and tissue-specific random effects, respectively, conditional on membership of the $i$th cluster. These random effects represent the variation due to the heterogeneity of genes and tissues (corresponding to $\boldsymbol{b}_i = \left(b_{i1}^T, \ldots, b_{in}^T\right)^T$ and $\boldsymbol{c}_i$, respectively). The random effects $\boldsymbol{b}_{ij}$ and $\boldsymbol{c}_i$, and the measurement error vector $\boldsymbol{\varepsilon}_{ij}$ are assumed to be mutually independent. In Eq. (9), $\boldsymbol{X}$, $\boldsymbol{U}$, and $\boldsymbol{V}$ are known design matrices of the corresponding fixed or random effects. The dimensions $q_b$ and $q_c$ of the random effects terms $\boldsymbol{b}_{ij}$ and $\boldsymbol{c}_i$ are determined by the design matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ which, along with $\boldsymbol{X}$ and $\boldsymbol{H}$, specify the experimental design to be adopted.

With the LMM, the distributions of $\boldsymbol{b}_{ij}$ and $\boldsymbol{c}_i$ are taken, respectively, to be multivariate normal $N_{q_b}\left(\boldsymbol{0}, \theta_{bi}\boldsymbol{I}_{q_b}\right)$ and $N_{q_c}\left(\boldsymbol{0}, \theta_{ci}\boldsymbol{I}_{q_c}\right)$, where $\boldsymbol{I}_{q_b}$ and $\boldsymbol{I}_{q_c}$ are identity matrices with dimensions being specified by the subscripts. The measurement error vector $\boldsymbol{\varepsilon}_{ij}$ is also taken to be multivariate normal $N_p(\boldsymbol{0}, \boldsymbol{A}_i)$, where $\boldsymbol{A}_i = \mathrm{diag}(\boldsymbol{H}\boldsymbol{\varphi}_i)$ is a diagonal matrix constructed from the vector $(\boldsymbol{H}\boldsymbol{\varphi}_i)$ with $\boldsymbol{\varphi}_i = \left(\sigma_{i1}^2, \ldots, \sigma_{iq_e}^2\right)^T$ and $\boldsymbol{H}$ is a known $p \times q_e$ zero-one design matrix. That is, we allow the $i$th component-variance to be different among the $p$ microarray experiments.

The vector $\boldsymbol{\Psi}$ of unknown parameters can be obtained by maximum likelihood via the EM algorithm, proceeding conditionally on the tissue-specific random effects $\boldsymbol{c}_i$. The E- and M-steps can be implemented in closed form. In particular, an approximation to the E-step by carrying out time-consuming Monte Carlo methods is not required. A probabilistic or an outright clustering of the genes into $g$ components can be obtained, based on the estimated posterior probabilities of component membership given the profile vectors and the estimated tissue-specific random effects $\hat{\boldsymbol{c}}_i (i = 1, \ldots, g)$.

To illustrate this method, we report here an example from [37] who extended the EMMIX-WIRE model to incorporate first-order autoregressive AR(1) random effects for clustering some time-course data from the yeast cell-cycle study of Cho et al. [38]. The data consist of the expression levels of 237 genes over two cycles for the yeast cells at $p = 17$ time points, sampling at 10-min intervals, where the raw data were log transformed and normalized by columns and rows. A general form of the first-order Fourier series expansion is adopted to model periodic gene expression [39]. With reference to Eq. (9), the design matrix was taken be an $17 \times 2$ matrix ($m = 2$) with the $(l + 1)$th row ($l = 0, \ldots, 16$)

$$(\cos(2\pi(10l)/\omega + \Phi) \quad \sin(2\pi(10l)/\omega + \Phi)), \tag{10}$$

where the period of the cell cycle $\omega$ was taken to be 85 and the phase offset $\Phi$ was set to zero. The design matrices for the random effects parts were specified as $\boldsymbol{U} = \boldsymbol{I}_{17}$ and $\boldsymbol{V} = \boldsymbol{I}_{17}$. That is, it is assumed that there exist random gene effects $\boldsymbol{b}_{ij}$ with $q_b = 17$ and

random temporal effects $\boldsymbol{c}_i = \left(c_{i1}, \ldots c_{iq_c}\right)^T$ with $q_c = p = 17$. The latter introduce dependence among expression levels within the same cluster obtained at the same time point. Also, $\boldsymbol{H} = \mathbf{1}_{17}$ and $\boldsymbol{\varphi}_i = \sigma_i^2\,(\,q_e = 1\,)$ so that the component variances are common among the $p = 17$ experiments. To account for the time dependent random gene effects, an AR(1) correlation structure is adopted for the gene profiles, so that $\boldsymbol{b}_{ij}$ follows a $N(\boldsymbol{0}, \theta_i \boldsymbol{A}(\rho_i))$ distribution, where

$$\boldsymbol{A}(\rho_i) = \frac{1}{1-\rho_i^2}\begin{pmatrix} 1 & \rho_i & \cdots & \rho_i^{16} \\ \rho_i & 1 & \cdots & \rho_i^{15} \\ \vdots & \vdots & \vdots & \vdots \\ \rho_i^{16} & \rho_i^{15} & \cdots & 1 \end{pmatrix}. \tag{11}$$

The inverse of $\boldsymbol{A}(\rho_i)$ can be expressed as

$$\boldsymbol{A}(\rho_i)^{-1} = \left(1+\rho_i^2\right) I - \rho_i J - \rho_i^2 K, \tag{12}$$

and

$$trace\left(\frac{\partial \boldsymbol{A}(\rho_i)^{-1}}{\partial \rho_i}\boldsymbol{A}(\rho_i)\right) = -\frac{2\rho_i}{\left(1-\rho_i^2\right)}. \tag{13}$$

In Eq. (12), all $\boldsymbol{I}$, $\boldsymbol{J}$, and $\boldsymbol{K}$ are $17 \times 17$ matrices, where $\boldsymbol{I}$ is the identity matrix, $\boldsymbol{J}$ has its sub-diagonal entries ones and zeros elsewhere, and $\boldsymbol{K}$ takes on the value 1 at the first and last elements of its principal diagonal and zeros elsewhere; *see* [37] for detailed derivation.

With this data set, the 237 genes were categorized with respect to the four categories in the MIPS database (DNA synthesis and replication, organization of centrosome, nitrogen and sulfur metabolism, and ribosomal proteins); *see* [40]. The clustering results using the extended EMMIX-WIRE model for $g = 4$ are given in Fig. 5, where the expression profiles for genes in each cluster are presented. The adjusted Rand index [36], for assessing the degree of agreement between the clustering results and the four categories of genes, was 0.6189, which is the best match (the largest index) compared with several model-based and hierarchical clustering algorithms considered in [40]; *see* [37] for more details.

## 3 Notes

1. For both procedures, as with other partitional clustering methods, the number of clusters $g$ needs to be specified at the outset. As both procedures are model-based, we can make a choice as to an appropriate value of $g$ by consideration of the likelihood function. In the absence of any prior information as to the number of clusters present in the data, we monitor the increase

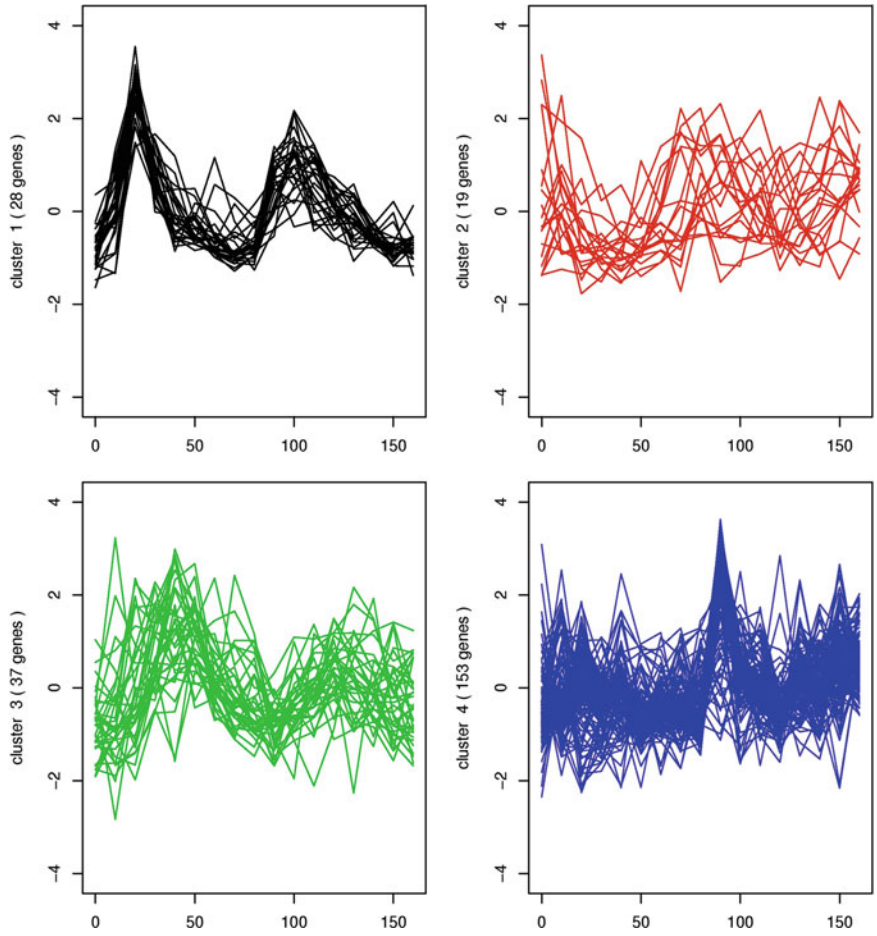



**Fig. 5** Clustering results for Cho's yeast cell cycle data. For all the plots, the *x*-axis is the time point and the *y*-axis is the gene-expression level

in the log likelihood function as the value of $g$ increases. At any stage, the choice of $g = g_o$ versus $g = g_1$, for instance $g_1 = g_o + 1$, can be made by either performing the likelihood ratio test or by using some information-based criterion, such as BIC (Bayesian information criterion). Unfortunately, regularity conditions do not hold for the likelihood ratio test statistic $\lambda$ to have its usual null distribution of chi-squared with degrees of freedom equal to the difference $d$ in the number of parameters for $g = g_1$ and $g = g_o$ components in the mixture models. One way to proceed is to use a resampling approach as in [41]. Alternatively, one can apply BIC, which leads to the selection of $g = g_1$ over $g = g_o$ if $-2\log \lambda$ is greater than $d \log(n)$. The value of d is obvious in applications of EMMIX-GENE, but is not so clear with applications of EMMIX-WIRE, due to the presence of random effects terms [36].

2. The most time-consuming step of the three steps is the gene selection step of EMMIX-GENE. This step is slower than the others as a mixture of two normals is fitted for each gene, instead of a multivariate normal being fitted to a group of genes or a metagene simultaneously. A faster but ad hoc selection step is to make the decision for each gene on the basis of the interquartile range of the gene expression values over the tissues.

## References


1. Alizadeh A, Eisen MB, Davis RE et al (2000) Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. Nature 403:503–511
2. Eisen MB, Spellman PT, Brown PO, Botstein D (1998) Cluster analysis and display of genome-wide expression patterns. Proc Natl Acad Sci U S A 95:14863–14868
3. Reilly C, Wang C, Rutherford R (2005) A rapid method for the comparison of cluster analyses. Stat Sin 15:19–33
4. Coleman D, Dong XP, Hardin J, Rocke DM, Woodruff DL (1999) Some computational issues in cluster analysis with no a priori metric. Comput Stat Data Anal 31:1–11
5. Everitt BS (1993) Cluster analysis, 3rd edn. Edward Arnold, London
6. Hartigan JA (1975) Clustering algorithms. Wiley, New York
7. Hastie T, Tibshirani RJ, Friedman JH (2001) The elements of statistical learning. Springer, New York
8. Kaufman L, Rousseeuw P (1990) Finding groups in data: an introduction to cluster analysis. Wiley, New York
9. Ripley BD (1996) Pattern recognition and neural networks. Cambridge University Press, Cambridge
10. Seber GAF (1984) Multivariate observations. Wiley, New York
11. Kettenring JR (2006) The practice of cluster analysis. J Classif 23:3–30
12. Marriott FHC (1974) The interpretation of multiple observations. Academic, London
13. Cormack RM (1971) A review of classification (with discussion). J R Stat Soc A 134:321–367
14. Hand DJ, Heard NA (2005) Finding groups in gene expression data. J Biomed Biotechnol 2005:215–225
15. Alon U, Barkai N, Notterman DA, Gish K et al (1999) Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. Proc Natl Acad Sci U S A 96:6745–6750
16. Chipman H, Tibshirani R (2006) Hybrid hierarchical clustering with applications to microarray data. Biostatistics 7:286–301
17. Kohonen T (1989) Self-organization and associative memory, 3rd edn. Springer, Berlin
18. Friedman HP, Rubin J (1967) On some invariant criteria for grouping data. J Am Stat Assoc 62:1159–1178
19. Scott AJ, Symons MJ (1971) Clustering methods based on likelihood ratio criteria. Biometrics 27:387–397
20. Hartigan JA (1975) Statistical theory in clustering. J Classif 2:63–76
21. McLachlan GJ, Basford KE (1988) Mixture models: inference and applications to clustering. Marcel Dekker, New York
22. McLachlan GJ, Peel D (2000) Finite mixture models. Wiley, New York
23. Aitkin M, Anderson D, Hinde J (1981) Statistical modelling of data on teaching styles (with discussion). J R Stat Soc A 144:419–461
24. Pollard KS, van der Laan MJ (2002) Statistical inference for simultaneous clustering of gene expression data. Math Biosci 176:99–121
25. Getz G, Levine E, Domany E (2000) Coupled two-way clustering analysis of gene microarray data. Cell Biol 97:12079–12084
26. Ambroise C, Govaert G (2006) Model based hierarchical clustering. Unpublished manuscript
27. Ward JH (1963) Hierarchical grouping to optimize an objective function. J Am Stat Assoc 58:236–244
28. Lance GN, Williams WT (1967) A generalized theory of classificatory sorting strategies: I. Hierarchical systems. Comput J 9:373–380
29. Ghosh D, Chinnaiyan AM (2002) Mixture modelling of gene expression data from microarray experiments. Bioinformatics 18:275–286
30. Yeung KY, Fraley C, Murua A, Raftery AE, Ruzzo WL (2001) Model-based clustering

and data transformations for gene expression data. Bioinformatics 17:977–987

31. McLachlan GJ, Bean RW, Peel D (2002) A mixture model-based approach to the clustering of microarray expression data. Bioinformatics 18:413–422
32. Medvedovic M, Sivaganesan S (2002) Bayesian infinite mixture model based clustering of gene expression profiles. Bioinformatics 18:1194–1206
33. Banfield JD, Raftery AE (1993) Model-based Gaussian and non-Gaussian clustering. Biometrics 49:803–821
34. Friedman JH, Meulman JJ (2004) Clustering objects on subsets of attributes (with discussion). J R Stat Soc B 66:815–849
35. Belitskaya-Levy I (2006) A generalized clustering problem, with application to DNA microarrays. Stat Appl Genet Mol Biol 5, Article 2
36. Ng SK, McLachlan GJ, Wang K, Ben-Tovim Jones L, Ng S-W (2006) A mixture model with random-effects components for clustering correlated gene-expression profiles. Bioinformatics 22:1745–1752
37. Wang K, Ng SK, McLachlan GJ (2012) Clustering of time-course gene expression profiles using normal mixture models with autoregressive random-effects. BMC Bioinformatics 13:300
38. Cho RJ, Huang M, Campbell MJ, Dong H, Steinmetz L, Sapinoso L, Hampton G, Elledge SJ, Davis RW, Lockhart DJ (2001) Transcriptional regulation and function during the human cell cycle. Nat Genet 27:48–54
39. Kim BR, Zhang L, Berg A, Fan J, Wu R (2008) A computational approach to the functional clustering of periodic gene-expression profiles. Genetics 180:821–834
40. Wong DSV, Wong FK, Wood GR (2007) A multi-stage approach to clustering and imputation of gene expression profiles. Bioinformatics 23:998–1005
41. McLachlan GJ (1987) On bootstrapping the likelihood ratio test statistic for the number of components in a normal mixture. Appl Stat 36:318–324

# Chapter 20

# Parameterized Algorithmics for Finding Exact Solutions of NP-Hard Biological Problems

**Falk Hüffner, Christian Komusiewicz, Rolf Niedermeier, and Sebastian Wernicke**

## Abstract

Fixed-parameter algorithms are designed to efficiently find optimal solutions to some computationally hard (NP-hard) problems by identifying and exploiting "small" problem-specific parameters. We survey practical techniques to develop such algorithms. Each technique is introduced and supported by case studies of applications to biological problems, with additional pointers to experimental results.

**Key words** Computational intractability, NP-hard problems, Algorithm design, Exponential running times, Discrete problems, Fixed-parameter tractability, Optimal solutions

## 1 Introduction

Many problems that emerge in bioinformatics require vast amounts of computer time to be solved optimally. An illustrative example, though somewhat oversimplified, would be the following: Given a set of $n$ experiments of which some pairs have conflicting results (that is, at least one result must be wrong), identify a minimum-size subset of experiments to eliminate such that no conflict remains. This problem, while simple to describe, has no known algorithm that solves it efficiently on all inputs. From a theoretical standpoint, such computational hardness can be traced back to the NP-*hardness* of a problem. Assuming a widely believed conjecture in complexity theory, the classification of a computational problem as NP-hard implies that the time needed to solve it grows very quickly (usually exponentially) with the input size [61]. However, the demand to solve NP-hard problems commonly arises in practical settings, including bioinformatics. To obtain solutions to these problems despite their NP-hardness, it is common to sacrifice solution quality for efficiency, for example, by employing heuristic algorithms or approximation algorithms. A different approach is to insist on exact

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_20, 

solutions and accept that the algorithm will not be efficient on all inputs but hopefully on those that arise in the application at hand.

Most theory on computational hardness is based on the assumption that the difficulty of solving an instance of a computational problem is determined by the size of that instance. The crucial observation this chapter is based on is that often it is not the *size* of an instance that makes a problem computationally hard to solve, but rather its *structure*. *Parameterized algorithmics* renders this observation precise by quantifying structural hardness with the so-called *parameters*, typically a nonnegative integer variable denoted by $k$ or a tuple of such variables. A parameterized problem is then called *fixed-parameter tractable* (FPT) if it can be solved efficiently when the parameter is small; the corresponding algorithm is called *fixed-parameter algorithm*. The concept of fixed-parameter tractability thus formalizes and generalizes the concept of "tractable special cases" that are known for virtually all NP-hard problems. For example, as we will discuss in more detail below, our introductory problem can be solved quickly whenever the number of conflicting experiments is small (a reasonable assumption in practical settings, since the results would otherwise not be worth much anyway).

Often, there are many possible parameters to choose from. For example, for solving our introductory problem we could choose the maximum number of conflicts for a single experiment to be the parameter or, alternatively, the size of the largest group of pairwise conflicting experiments. This makes parameterized algorithmics a multipronged attack that can be adapted to different practical applications. Of course, not all parameters lead to efficient algorithms; in fact, parameterized algorithmics also provides tools to classify parameters as "not helpful" in the sense that we cannot expect provably efficient algorithms even when these parameters are small.

Fixed-parameter algorithms have by now facilitated many success stories and several techniques have emerged as being applicable to large classes of problems [81]. This chapter presents several of these techniques, namely kernelization (Subheading 2), depth-bounded search trees (Subheading 3), dynamic programming (Subheading 4), tree decompositions of graphs (Subheading 5), color-coding (Subheading 6), and iterative compression (Subheading 7). We start each section by introducing the basic concepts and ideas, followed by some case studies concerning practically relevant bioinformatics problems. Concluding each section, we survey known applications, implementations, and experimental results, thereby highlighting the strengths and fields of applicability for each technique.

Another commonly used strategy for exactly solving NP-hard problems is to reduce the problem at hand to "general-purpose problems" such as integer linear programming [7, 8] and

satisfiability solving [14, 105, 127]. For these, there exist highly optimized tools with years of algorithm engineering effort that went into their development. Therefore, if an NP-hard problem can be efficiently expressed as one of these general-purpose problems, these tools might be able to find an optimum solution without the need for any further algorithm design. In many application scenarios, it will actually make sense to try and combine these general-purpose approaches and the more problem-specific approach of parameterized algorithmics since the specific advantage of fixed-parameter algorithms is that they are usually crafted directly for the problem at hand and thus may allow a better exploitation of problem-specific features to substantially gain efficiency. In particular, the polynomial-time data reduction techniques that are introduced in Subheading 2 usually combine nicely and productively with the more general solver tools.

Before discussing the main techniques of fixed-parameter algorithms in the following sections, the remainder of this section provides a crash course in computational complexity theory and a few formal definitions related to parameterized complexity analysis. Furthermore, some terms from graph theory are introduced, and we present our running example problem VERTEX COVER.

### 1.1 Computational Complexity Theory

In this survey, we are concerned with efficiently solving computational problems. A standard format for specifying these problems is to phrase them in an "Input/Task" way that formally specifies the input and desired output. A core topic of computational complexity theory is the evaluation and comparison of different algorithms for a given problem [106, 113]. Since most algorithms are designed to work with variable inputs, the efficiency (or *complexity*) of an algorithm is not just stated for some concrete inputs (*instances*), but rather as a function that relates the input length $n$ to the number of steps that are required to execute the algorithm. Generally, this function is given in an asymptotic sense, the standard way being the *big-O notation* where we write $f(n) = O(g(n))$ to express that $f(n)/g(n)$ is upper-bounded by a positive constant in the limit for large $n$ [45, 86, 122]. Since instances of the same size might take different amounts of time, it is implicitly assumed in this chapter that we are considering the *worst-case* running time among all instances of the same size; that is, we deliberately exclude from our analysis the potentially efficient solvability of some specific input instances of a computational problem.

Determining the computational complexity of problems (meaning the best possible worst-case running time of an algorithm for them) is a key issue in theoretical computer science. Herein, it is of central importance to distinguish between problems that can be solved efficiently and those that presumably cannot. To this end, theoretical computer science has coined the notions of *polynomial-time solvability*, on the one hand, and NP-*hardness*, on the

other [61]. Here, polynomial-time solvability means that for every size-$n$ input instance of a problem, an optimal solution can be computed in $n^{O(1)}$ time. In contrast, the (unproven, yet widely believed) working hypothesis of theoretical computer science is that NP-hard problems cannot be solved in $n^{O(1)}$ time. More specifically, typical running times for NP-hard problems are of the form $O(c^n)$ for some constant $c > 1$; that is, we have an exponential growth in the number of computation steps as instances grow larger. In this sense, polynomial-time solvability has become a synonym for efficient solvability.

As there are thousands of known NP-hard optimization problems and their number is continuously growing [106, 114], several approaches have been developed that try to circumvent the assumed computational intractability of NP-hard problems. One such approach is based on polynomial-time approximation algorithms, where one gives up seeking optimal solutions in order to have efficient algorithms [9, 128, 131]. Another common strategy is to use heuristics, where one gives up provable performance guarantees (concerning running time or solution quality) by developing algorithms that behave well in "most" practical applications [104, 106].

### 1.2 Parameterized Complexity

For many applications, the compromises inherent to approximation algorithms and heuristics are not satisfactory. Fixed-parameter algorithms can provide an alternative by providing exact solutions with useful running time guarantees [53, 59, 109]. The core concept is formalized as follows:

**Definition 1.** *A parameterized problem instance consists of a problem instance* I *and a parameter* k. *A parameterized problem is* fixed-parameter tractable *if it can be solved in* $f(k) \cdot |I|^{O(1)}$ *time, where* f *is a (computable) function solely depending on the parameter* k.

For NP-hard problems, $f(k)$ will typically be an exponential function like $2^k$ rather than a polynomial function.

Note the difference between "fixed-parameter tractable" and "polynomial-time solvable for fixed $k$": an algorithm running in $|I|^{f(k)}$ time demonstrates that a problem is polynomial-time solvable for any fixed $k$, but does not show fixed-parameter tractability since the degree of the polynomial depends on $k$; ideally, a fixed-parameter algorithm provides a linear-time algorithm for each fixed $k$ [126].

As an example for this "parameterized perspective," consider again the identification of $k$ faulty experiments among $n$ experiments. We could naively solve this problem in $O(2^n)$ time by trying all possible subsets of the $n$ experiments. However, this would not be practically feasible for $n > 40$. In contrast, a simple fixed-parameter algorithm with running time $O(2^k \cdot n)$ exists for this

problem, which allows it to be solved even for $n > 1000$, as long as $k < 20$ (as we will discuss in Subheading 2.4, real-world instances can often be solved for much larger values of $k$ by an extension of this approach).

Unfortunately, there are parameterized problems for which there is good evidence that they are not fixed-parameter tractable (*see* **Note 1**).

### 1.2.1 A Few Words on the Art of Problem Parameterization

Typically, a problem allows for more than one parameterization [90, 110]. From a theoretical point of view, parameterization is a key to better understand the nature of computational intractability. The ultimate goal here is to learn how parameters influence the computational complexity of problems. The more we know about these interactions, the more likely it becomes to cope with computational intractability. In a sense, it may be considered as an art to find the most useful parameterizations of a computational problem.

From an applied point of view, the identification of parameters for a concrete problem should go hand-in-hand with an extensive data analysis. One natural way for spotting relevant parameterizations of a problem in real-world applications is to analyze the given input data and check which quantifiable aspects of it appear to be small and might thus be suitable as parameters. For example, if the input is a network, one such observable parameter could be the maximum vertex degree. Often, real-world input instances also carry some hidden structure that might be exploited. Again turning to graphs, well-known parameters such as "feedback vertex set number" or "treewidth" measure how tree-like a graph is. These parameters are motivated by the observation that many intractable graph problems become tractable when restricted to trees. For NP-hard string problems, which also occur frequently in bioinformatics, natural parameters are, for example, the size of the alphabet or the number of occurrences of a letter [35].

## 1.3 Graph Theory

Many of the problems we deal with in this work can be formulated in graph-theoretic terms [48, 129]. An undirected graph $G = (V, E)$ is given by a set of *vertices* $V$ and a set of *edges* $E$, where each edge $\{v, w\}$ is an undirected connection of two vertices $v$ and $w$. Throughout this work, we use $n := |V|$ to denote the number of vertices and $m := |E|$ to denote the number of edges. For a set of vertices $V' \subseteq V$, the *induced subgraph* $G[V']$ is the graph $(V', \{\{v, w\} \in E | v, w \in V'\})$, that is, the graph $G$ restricted to the vertices in $V'$. We denote the *open neighborhood* of a vertex $v$ by $N(v) := \{u | \{u, v\} \in E\}$ and its *closed neighborhood* by $N[v] := N(v) \cup \{v\}$.

It is not hard to see that we can formalize our introductory problem of recognizing faulty experiments as a graph problem where vertices correspond to experiments and edges correspond to pairs of conflicting experiments. Thus, we need to choose a small

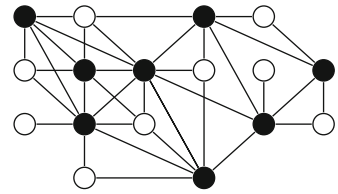


**Fig. 1** A graph with a size-8 vertex cover (cover vertices are marked *black*)

set of vertices (the experiments to eliminate) so that each edge is incident with at least one chosen vertex. This is known as the NP-hard VERTEX COVER problem, which serves as a running example for several techniques in this work.

VERTEX COVER
**Input:** An undirected graph $G = (V, E)$ and a nonnegative integer $k$.
**Task:** Find a set $C \subseteq V$ of at most $k$ vertices such that each edge in $E$ has at least one of its endpoints in $C$.

The problem is illustrated in Fig. 1. VERTEX COVER can well be considered a poster child of fixed-parameter research, as many discoveries that influenced the whole field originated from the study of this single problem.

## 2 Kernelization: Data Reduction with Guaranteed Effectiveness

The idea of data reduction is to quickly presolve those parts of a given problem instance that are easy to cope with, shrinking it to those parts that form its hard core [73, 94]. Computationally expensive algorithms need then only be applied to this core. In some practical scenarios, data reduction may even reduce instances of a seemingly hard problem to triviality. Once an effective (and efficient) reduction rule has been found, it is typically not only useful in the context of parameterized algorithmics, but also in other problem solving contexts, whether they be heuristic, approximative, or exact.

This section introduces the concept of *kernelization*, that is, polynomial-time data reduction with guaranteed effectiveness. Kernelization is closely connected to fixed-parameter tractability and emerges within its framework.

### 2.1 Basic Concepts

There are many examples of combinatorial problems that would not be solvable without employing heuristic data reduction and preprocessing algorithms. For example, commercial solvers for hard combinatorial problems such as the integer linear program solver CPLEX heavily rely on data-reducing preprocessors for their efficiency [15]. Obviously, many practitioners are aware of the general concept of data reduction. Parameterized algorithmics

adds to this by providing a way to use data reduction rules not only heuristically, but also with guaranteed performance quality. These so-called *kernelizations* guarantee an upper bound on the size of the reduced instance, which solely depends on the parameter value. More precisely, the concept is defined as follows:

**Definition 2 ([53, 109]).** *Let* I *be an instance of a parameterized problem with given parameter* k. *A* reduction to a problem kernel *(or* kernelization*) is a polynomial-time algorithm that replaces* I *by a new instance* I′ *and* k *by a new parameter* k′ *such that*

- *the size of* I′ *and the value of* k′ *are guaranteed to only depend on some function of* k*, and*
- *the new instance* I′ *has a solution with respect to the new parameter* k′ *if and only if* I *has a solution with respect to the original parameter* k.

Kernelizations can help to understand the practical effectiveness of some data reduction rules and, conversely, the quest for kernelizations can lead to new and powerful data reduction rules based on deep structural insights.

Intriguingly, there is a close connection between fixed-parameter tractable problems and those problems for which there exists a kernelization—in fact, they are exactly the same [36]. Unfortunately, the running time of a fixed-parameter algorithm directly obtained from a kernelization is usually not practical and, in the other direction, there exists no constructive scheme for developing data reduction rules for a fixed-parameter tractable problem. Nevertheless, this equivalence can establish the fixed-parameter tractability and amenability to kernelization of a problem by knowing just one of these two properties.

## 2.2 Case Studies

In this section, we first illustrate the concept of kernelization by a simple example concerning the VERTEX COVER problem. We then show a more involved kernelization algorithm for the graph clustering problem CLUSTER EDITING. Finally, we discuss the limits of the kernelization approach for fixed-parameter tractable problems and present an extension of the kernelization concept that can be used to cope with the nonexistence of problem kernels.

### 2.2.1 A Simple Kernelization for Vertex Cover

Consider our running example VERTEX COVER. In order to cover an edge in the graph, one of its two endpoints must be in the vertex cover. If one of these is a degree-1 vertex (that is, it has exactly one neighbor), then the other endpoint has the potential to cover more edges than this degree-1 vertex, leading to a first data reduction rule.

Reduction Rule VC1
If there is a degree-1 vertex, then put its neighboring vertex into the cover.

Here, "put into the cover" means adding the vertex to the solution set and removing it and its incident edges from the instance. Note that this reduction rule assumes that we are only looking for *one* optimal solution to the VERTEX COVER instance we are trying to solve; there may exist other minimum vertex covers that do include the reduced degree-1 vertex (*see* **Note 2**).

After having applied Rule VC1, we can further do the following in the fixed-parameter setting where we ask for a vertex cover of size at most $k$.

Reduction Rule VC2
If there is a vertex $v$ of degree at least $k + 1$, then put $v$ into the cover.

The reason this rule is correct is that if we did not take $v$ into the cover, then we would have to take every single one of its $k + 1$ neighbors into the cover in order to cover all edges incident with $v$. This is not possible because the maximum allowed size of the cover is $k$.

After exhaustively performing Rules VC1 and VC2, all vertices in the remaining graph have degree at most $k$. Thus, at most $k$ edges can be covered by choosing an additional vertex into the cover. Since the solution set may be no larger than $k$, the remaining graph can have at most $k^2$ edges if it has a solution. Clearly, we can assume without loss of generality that there are no isolated vertices (that is, vertices with no incident edges) in a given instance. In conjunction with Rule VC1, this means that every vertex has degree at least two. Hence, the remaining graph can contain at most $k^2$ vertices.

Stepping back, what we have just done is the following: After applying two *polynomial-time* data reduction rules to an instance of VERTEX COVER, we arrived at a reduced instance whose size can *be expressed solely in terms of the parameter k*. Hence, considering Definition 2, we have found a kernelization for VERTEX COVER.

### 2.2.2 A Kernelization for Cluster Editing

In the above example kernelization for VERTEX COVER, there is a notable difference between Rules VC1 and VC2: Rule VC1 is based on a local optimality argument whereas Rule VC2 makes explicit use of the parameter $k$. In applications, the first type of data reduction rules is usually preferable, as they can be applied independently of the value of $k$ and this value is only used in the analysis of the power of the data reduction rules. For the NP-hard graph clustering problem CLUSTER EDITING, we now present an efficient kernelization algorithm that is based solely on a data reduction rule of the first type.

CLUSTER EDITING
**Input:** An undirected graph $G = (V, E)$, an edge-weight function $\omega : V^2 \to N^+$, and a nonnegative integer $k$.
**Task:** Find whether we can modify $G$ to consist of vertex-disjoint cliques (that is, fully connected components) by adding or deleting a set of edges whose weights sum up to at most $k$.

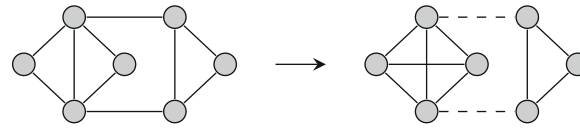


**Fig. 2** Illustration for CLUSTER EDITING with unit weights: By removing two edges from and adding one edge to the graph on the left (that is, $k = 3$), we can obtain a graph that consists of two vertex-disjoint cliques

CLUSTER EDITING can be used, for example, to cluster proteins with high sequence similarity [23] and to identify cancer subtypes [133]; a comprehensive overview of its applications is given by Böcker and Baumbach [20]. Many theoretical studies consider the case in which all edges have weight one, but the weighted version of CLUSTER EDITING is more relevant in biological applications. The positive edge weights describe the cost to delete an existing edge or to insert a missing edge, respectively. Figure 2 shows an instance of the unweighted problem variant together with a solution. A simple kernelization for CLUSTER EDITING uses similar high-degree reduction rules as the VERTEX COVER kernelization described above. These rules yield a kernel with $O(k^2)$ vertices [67]. This bound can be improved to $O(k)$ vertices using reduction rules whose correctness is based on local optimality arguments. We now describe such a kernelization algorithm for CLUSTER EDITING that was developed by Cao and Chen [39].

The idea of this kernelization is to examine for each vertex $v$ of the graph whether its neighborhood is already very dense and only loosely connected to the rest of the graph. If this is the case, then it is optimal to put all neighbors of $v$ in the same cluster as $v$. This knowledge can be used to identify edges that have to be deleted or edges that have to be added. Formally, the algorithm computes the sum of the weights of the missing edges in the neighborhood $N[v]$; this number is denoted by $\delta(v)$. Then it computes the sum of the edge weights between $N[v]$ and $V \setminus N[v]$; this number is denoted by $\gamma(v)$. These two measures are combined to form what is called the stable cost of a vertex $v$ defined as $c(v) = 2\delta(v) + \gamma(v)$. Now a vertex $v$ is called *reducible* if $c(v) < |N[v]|$. The main consequence of being reducible is that if $N[v]$ is reducible, then there is an optimal solution such that $N[v]$ is contained in a single cluster. This implies the correctness of the following data reduction rules; an example application of the first two reduction rules is given in Fig. 3.

The first rule adds missing edges in neighborhoods of reducible vertices.

Reduction Rule CE1
If there is a reducible vertex $v$ and a pair of vertices $u, x$ in $N[v]$ that are not neighbors, then add $\{u, x\}$ to $G$ and decrease $k$ by $\omega(\{u, x\})$.

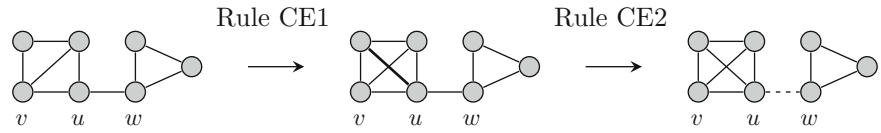



**Fig. 3** The application of Reduction Rules CE1 and CE2 to an instance of CLUSTER EDITING. In the example, the weight of all existing and missing edges is 1. Initially, the vertex $v$ is reducible. Then, Rule CE1 inserts the missing edge in $M[v]$. Subsequently, Rule CE2 deletes the edge between $u$ and $w$, since it is more costly to make $w$ adjacent to all vertices of $M[v]$ than to delete this edge

The next rule finds vertices that have some but only few neighbors in $N[v]$. In an optimal solution, these vertices are never in the same cluster as $N[v]$. Thus, the edges between these vertices and $N[v]$ may be deleted.

Reduction Rule CE2
If there is a reducible vertex $v$ and a vertex $u \notin N[v]$ such that it is more costly to add all missing edges between $u$ and $N[v]$ than to remove all edges between $u$ and $N[v]$, then remove all edges between $u$ and $N[v]$ and decrease $k$ accordingly.

The final rule merges $N[v]$ into one vertex and adjusts the edge weights accordingly.

Reduction Rule CE3
If there is a reducible vertex $v$ to which Rules CE1 and CE2 do not apply, then merge $N[v]$ into a single vertex $v'$. For each vertex $u \in V \setminus N[v]$ set $\omega(\{u, v'\}) := \Sigma_{x \in N[v]} \omega(\{u, x\})$.

As long as the instance contains a reducible vertex, the reduction rules will either modify an edge or merge a vertex. If there are no more reducible vertices, then the last trivial step of the kernelization algorithm is to remove all isolated vertices from the instance. Afterwards, the instance has at most $2k$ vertices. The intuition behind this size bound is the following. Every edge has weight at least one, so a solution contains at most $k$ edges. Since each edge has two endpoints, the modifications can affect at most $2k$ vertices. Now if in a cluster every vertex is affected, then the size of the cluster is at least two times the number of edge modifications within the cluster plus the number of edge modifications between this cluster and other clusters. If there is a vertex $v$ in the cluster that is not affected by the solution, then the same bound on the cluster size holds, in this case because $v$ is not reducible. Summing these size bounds over all clusters, we obtain a sum of edges in which each solution edge appears at most twice. This gives the size bound of $2k$ vertices.

### 2.3 Limits and Extensions of Kernelization

The two example problems VERTEX COVER and CLUSTER EDITING are especially amenable to kernelization since they admit *polynomial-size problem kernels*. That is, the size bound for the kernel is a polynomial function in the parameter $k$. While all fixed-parameter

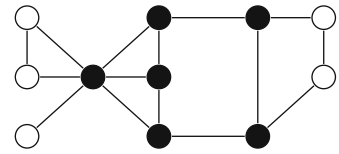


**Fig. 4** A graph with a 2-club of size six (marked *black*)

tractable problems admit a problem kernelization, it is not the case that all fixed-parameter tractable problems admit *polynomial* kernels [53, 94]. It is beyond the scope of this paper to introduce the proof techniques for showing nonexistence of polynomial problem kernels. We will give, however, an example of a biologically motivated graph problem that does not admit a polynomial problem kernel and describe one way of circumventing this hardness result.

2-CLUB
**Input:** An undirected graph $G = (V, E)$ and a nonnegative integer $k$.
**Task:** Find a set $S \subseteq V$ of at least $k$ vertices such that the subgraph induced by $S$ has diameter at most two.

The NP-hard 2-CLUB problem attempts to identify large cohesive subgroups of an input graph. The idea behind the formulation is to relax the overly restrictive definition of the CLIQUE problem which only accepts solutions that are complete graphs or, equivalently, that have diameter one. The 2-CLUB problem finds applications in the detection of protein interaction complexes [115]; an instance of 2-CLUB with a maximum-cardinality solution is shown in Fig. 4.

As we will see, 2-CLUB is fixed-parameter tractable with respect to the parameter solution size $k$. It does not, however, admit a polynomial problem kernel for this parameter [119] (*see* **Note 3** for a brief discussion).

In spite of this hardness result, one can still perform a useful parameterized data reduction for 2-CLUB. The idea is to reduce the problem to many problem kernels instead of just one. This approach is called *Turing kernelization*. In the case of 2-CLUB, the Turing kernelization consists of two simple parts. First, one looks for a trivial solution using the following observation: for every vertex $v$ in a graph, its closed neighborhood $N[v]$ has diameter two.

Reduction Rule 2-C
If there is a vertex $v$ with at least $k - 1$ neighbors, then return $N[v]$.

After this rule has been applied, we have either obtained a solution or the maximum degree of the graph is at most $k - 2$. Now, the Turing kernelization uses only one further observation: To find a largest 2-club it is sufficient to examine for each vertex $v$ of the input graph $G$ the subgraph of $G$ that contains only the vertices which have distance at most two to $v$. We can now use the fact that the maximum degree is bounded: every vertex $v$ has at

most $k-2$ neighbors and each of these has at most $k-3$ further neighbors. Thus, 2-CLUB can be solved by independently solving $n$ small instances with $O(k^2)$ vertices each. Formally, this means that 2-CLUB admits a Turing kernelization with $O(k^2)$ vertices.

### 2.4 Applications and Implementations

Solving VERTEX COVER is relevant in many bioinformatics-related scenarios such as analysis of gene expression data [44] and the computation of multiple sequence alignments [40]. Besides solving instances of VERTEX COVER, another application of VERTEX COVER kernelizations is to search maximum-cardinality cliques (that is, maximum-size complete subgraphs) in a graph. Here, use is made of the fact that an $n$-vertex graph $G$ has a clique of size $(n-k)$ if and only if its complement graph, that is, the graph that contains exactly the edges not contained in $G$, has a size-$k$ vertex cover. The best known kernel for VERTEX COVER (up to minor improvements) has $2k$ vertices [108]. Abu-Khzam et al. [1] studied various kernelization methods for VERTEX COVER and their practical performance on biological networks with respect to running time and resulting kernel size. Experimental results for the computation of large cliques via VERTEX COVER are given, for example, by Abu-Khzam et al. [2].

Several kernelization approaches including the one presented in Subheading 2.2.2 have been implemented for CLUSTER EDITING [25, 76]. The Turing kernelization for 2-CLUB was implemented and experimentally evaluated; it turned out to be a crucial ingredient for obtaining an efficient algorithm for this problem [77]. Another biologically relevant clustering problem where kernelizations have been successfully implemented is the CLIQUE COVER problem. Here, the task is to cover all edges of a graph using at most $k$ cliques (these may overlap). Using data reduction, Gramm et al. [69] solved even large instances with 1 000 vertices and $k \approx 6\ 000$ as long as they are sparse ($m \approx 7\ 000$).

## 3 Depth-Bounded Search Trees

Once data reductions as discussed in the previous section have been applied to a problem instance, we are left with the "really hard" problem kernel to be solved. A standard way to explore the huge search space of a computationally hard problem is to perform a systematic exhaustive search. This can be organized in a tree-like fashion, which is the subject of this section.

### 3.1 Basic Concepts

Search tree algorithms—also known as backtracking algorithms, branching algorithms, or splitting algorithms—certainly are no new idea and have extensively been used in the design of exact algorithms (e.g., *see* [45, 60, 122]). The main contribution of parameterized algorithmics to search tree algorithms is the

consideration of search trees whose *depth is constrained by a function in the parameter*. Combined with insights on how to find useful—and possibly non-obvious—parameters, this can lead to search trees that are much smaller than those of naive brute-force searches. For example, a very naive search tree approach for solving VERTEX COVER is to just take one vertex and branch into two cases: either this vertex is in the vertex cover or not. For an $n$-vertex graph, this leads to a search tree of size $O(2^n)$. As we outline in this section, we can do much better than that and obtain a search tree whose depth is upper-bounded by $k$, giving a size bound of $O(2^k)$. Extending what we discuss here, even better search trees of size $O(1.28^k)$ are possible [43]. Since usually $k \ll n$, this can draw the problem into the zone of feasibility even for large graphs.

Besides depth-bounding, parameterized algorithmics provides additional means to provably improve the speed of search tree exploration, particularly by interleaving this exploration with kernelizations, that is, applying data reduction to partially solved instances during the exploration.

## 3.2 Case Studies

Starting with our running example VERTEX COVER, this section introduces the concept of depth-bounded search trees by three case studies.

### 3.2.1 Vertex Cover Revisited

For many search tree algorithms, the basic idea is to find a small subset of the input instance in polynomial time such that at least one element of this subset must be part of an optimal solution to the problem. In the case of VERTEX COVER, the most simple such subset is any set of two adjacent vertices. By definition of the problem, one of these two vertices has to be part of a solution, or the respective edge would not be covered. Thus, a simple search-tree algorithm to solve VERTEX COVER on a graph $G = (V, E)$ can proceed by picking an arbitrary edge $e = \{v, w\}$ and recursively searching for a vertex cover of size $k - 1$ both in $G[V \setminus \{v\}]$ and $G[V \setminus \{w\}]$, that is, in the graphs obtained by removing either $v$ and its incident edges or $w$ and its incident edges. In this way, the algorithm branches into two subcases knowing one of them must lead to a solution of size at most $k$ (provided that it exists).

As shown in Fig. 5, the recursive calls of the simple VERTEX COVER algorithm can be visualized as a tree structure. Because the depth of the recursion is upper-bounded by the parameter value and we always branch into two subcases, the number of cases that are considered by this tree—its size, so to say—is $O(2^k)$. Independent of the size of the input instance, it only depends on the value of the parameter $k$.

The currently "best" search trees for VERTEX COVER have worst-case size $O(1.28^k)$ [43] and are mainly achieved by elaborate case distinctions. These algorithms consist of several branching rules; for example, the degrees of the endpoints of an edge determine

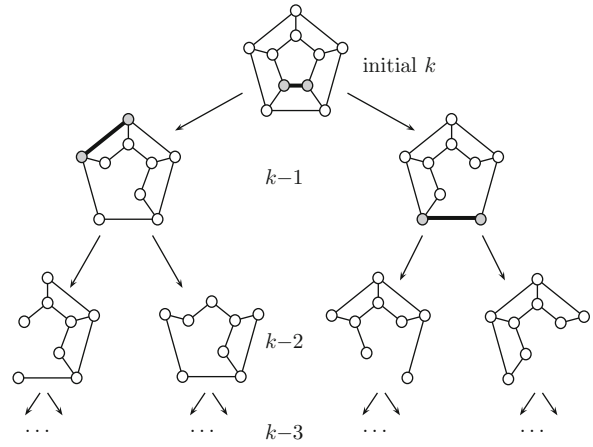



**Fig. 5** Simple search tree for finding a vertex cover of size at most $k$ in a given graph. The size of the tree is $O(2^k)$

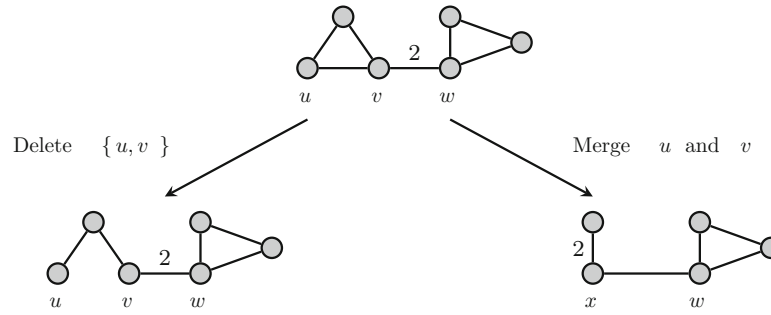



**Fig. 6** The merge branching for CLUSTER EDITING. In the example instance, all edges and missing edges have unit weight, except $\{v, w\}$ which has weight 2. In one branch, the edge $\{u, v\}$ is deleted and $k$ is reduced by 1. In the other branch, $u$ and $v$ are merged and the edge weights are adjusted. For example, edge $\{x, w\}$ obtains weight 1 since the missing edge $\{u, w\}$ had weight 1. Accordingly, $k$ is decreased by 1. All missing edges between $x$ and other vertices have weight 2

which of the branching rules is applied. However, for practical applications it is always concrete implementation and testing that has to decide whether the administrative overhead caused by distinguishing more and more cases pays off. A simpler algorithm with slightly worse search tree size bounds may be preferable.

### *3.2.2 A Search Tree Algorithm for Cluster Editing*

For VERTEX COVER, we have found a depth-bounded search tree by observing that at least one endpoint of any given edge must be part of the cover. A somewhat similar approach can be used to derive a depth-bounded search tree for CLUSTER EDITING (Fig. 6).

Recall that the aim for CLUSTER EDITING is to modify a graph into a cluster graph, that is, a vertex-disjoint union of cliques, by

modifying edges whose weight sums up to at most $k$. Similar to VERTEX COVER, a search tree for CLUSTER EDITING can be obtained by noting that the desired graph of vertex-disjoint cliques forbids a certain structure: If two vertices in a cluster graph are adjacent, then their neighborhoods must be the same. Hence, whenever we encounter two vertices $u$ and $v$ in the input graph $G$ that are adjacent and where one vertex, say $v$, has a neighbor $w$ that is not adjacent to $u$, we are compelled to do one of three things: Either remove the edge $\{u, v\}$, or add the edge $\{u, w\}$, or remove the edge $\{v, w\}$. Note that each such modification incurs a cost of at least one. Therefore, exhaustively branching into three cases, each time decreasing $k$ by one, we obtain a search tree of size $O(3^k)$ to solve CLUSTER EDITING. Using computer-aided algorithm design, this idea can be improved to obtain, for the unit-weight case, a search tree of size $O(1.92^k)$ [66]. The current-best theoretical running time is, however, achieved by exploiting the fact that edge weights make it possible to consider the merging operation in a search tree algorithm. The observation is that in the presence of a conflict as described above, one may either delete the edge $\{u, v\}$ or, otherwise, $u$ and $v$ are in the same cluster of the final cluster graph. Thus, one may merge $u$ and $v$ and adjust the edge weights accordingly. The main trick is that when performing the merging, this still causes some cost: The edge $\{v, w\}$ must be deleted or the edge $\{u, w\}$ must be added.

After merging $u$ and $v$ into a new vertex $x$ one may thus "remember" that the new edge $\{x, w\}$ will incur a cost irrespective of whether this edge is deleted or kept by a solution. With a more refined branching strategy, this idea leads to a search tree of size $O(1.82^k)$ for the general case [23] and of size $O(1.\ 62^k)$ for the unit-weight case [19].

### 3.2.3 The Closest String Problem

The CLOSEST STRING problem is also known as CONSENSUS STRING.

CLOSEST STRING
**Input:** A set of $k$ length-$\ell$ strings $s_1, \ldots, s_k$ and a nonnegative integer $d$.
**Task:** Find a *consensus string* $s$ that satisfies $d_H(s, s_i) \leq d$ for all $i = 1, \ldots, k$.

Here, $d_H(s, s_i)$ denotes the Hamming distance between two strings $s$ and $s_i$, that is, the number of positions where $s$ and $s_i$ differ. Note that there are at least two immediately obvious parameterizations of this problem. The first is given by choosing the "distance parameter" $d$ and the second is given by the number of input strings $k$. Both parameters are reasonably small in various applications; we refer to Gramm et al. [65] for more details. Here, we focus on the parameter $d$.

CLOSEST STRING appears, for example, in *primer design*, where we try to find a small DNA sequence called *primer* that binds to a set of (longer) target DNA sequences as a starting point for replication of these sequences. How well the primer binds to a sequence is mostly

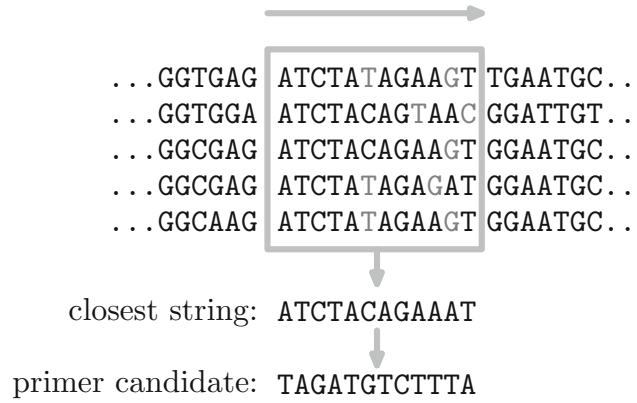



**Fig. 7** Illustration to show how DNA primer design can be achieved by solving CLOSEST STRING instances on length-$\ell$ windows of aligned DNA sequences. The primer candidate is not the computed consensus string but its nucleotide-wise complement

determined by the number of positions in that sequence that hybridize to it. While often done by hand, Stojanovic et al. [124] proposed a computational approach for finding a well-binding primer of length $\ell$. First, the target sequences are aligned, that is, as many matching positions within the sequences as possible are grouped into columns. Then, a "sliding window" of length $\ell$ is moved over this alignment, giving a CLOSEST STRING problem for each window position. Figure 7 illustrates this (*see* [63] for details).

In the remainder of this case study, we sketch a fixed-parameter search tree algorithm for CLOSEST STRING due to Gramm et al. [65], the parameter being the distance $d$. Unlike for VERTEX COVER and CLUSTER EDITING, the central challenge lies in even *finding* a depth-bounded search tree, which is not obvious at a first glance. Once found, however, the derivation of the upper bound for the search tree size is straightforward. The underlying algorithm is very simple to implement.

The main idea behind the algorithm is to maintain a *candidate string* $s^*$ for the center string and compare it to the strings $s_1, \ldots, s_k$. If $s^*$ differs from some $s_i$ in more than $d$ positions, then we know that $s^*$ needs to be modified in at least one of these positions to match the character that $s_i$ has there. Consider the following observation:

> **Observation 1.** Let d be a nonnegative integer. If two strings $s_i$ and $s_j$ have a Hamming distance greater than 2d, then there is no string that has a Hamming distance of at most d to both of $s_i$ and $s_j$.

This means that $s_i$ is allowed to differ from $s^*$ in at most $2d$ positions. Hence, among any $d + 1$ of those positions where $s_i$ differs from $s^*$, at least one must be modified to match $s_i$. This can be used to obtain a search tree that solves CLOSEST STRING.

We start with a string from $\{s_1, \ldots, s_k\}$ as the candidate string $s^*$, knowing that a center string can differ from it in at most $d$ positions. If $s^*$ already is a valid center string, we are done. Otherwise, there exists a string $s_i$ that differs from $s^*$ in more than $d$ positions, but less than $2d$. Choosing any $d + 1$ of these positions, we branch into $(d + 1)$ subcases, each subcase modifying a position in $s^*$ to match $s_i$. This position cannot be changed anymore further down in the search tree (otherwise, it would not have made sense to make it match $s_i$ at that position). Hence, the depth of the search tree is upper-bounded by $d$, for if we were to go deeper down into the tree, then $s^*$ would differ in more than $d$ positions from the original string we started with. Thus, CLOSEST STRING can be solved by exploring a search tree of size $O((d + 1)^d)$ [65]. Combining data reduction with this search tree, we arrive at the following:

**Theorem 1.** *CENTER STRING* can be solved in $O(k \cdot \ell + k \cdot d \cdot (d + 1)^d)$ time.

It might seem as if this result is purely of theoretical interest—after all, the term $(d + 1)^d$ becomes prohibitively large already for $d = 15$. Two things, however, should be noted in this respect: First, for one of the main applications of CLOSEST STRING, primer design, $d$ is very small (often less than 4). Second, empirical analysis reveals that when the algorithm is applied to real-world and random instances, it often beats the proven upper bound by far, solving many real-world instances in less than a second. The algorithm is also faster than a simple integer linear programming formulation of CLOSEST STRING when the input consists of many strings and $\ell$ is small [65].

Unfortunately, many variants of CLOSEST STRING—roughly speaking, these deal with finding a matching *sub*string and distinguish between strings to which the center is supposed to be close and to which it should be distant—are known to be intractable for many standard parameters [56, 68, 103].

### 3.3 Applications and Implementations

In combination with data reduction, the use of depth-bounded search trees has proven itself quite useful in practice, for example, allowing to find vertex covers of more than ten thousand vertices in some dense graphs of biological origin [2]. It should also be noted that search trees trivially allow for a parallel implementation: when branching into subcases, each process in a parallel setting can further explore one of these branches with no additional communication required. Experimental results for VERTEX COVER show linear speedups even for thousands of cores [3].

The merge-based search tree algorithm for CLUSTER EDITING can solve many instances arising in the analysis of protein similarity data [23]; it is part of a software package [132]. A fixed-parameter search tree algorithm was also used to solve instances of the

MINIMUM COMMON STRING PARTITION problem [34]. This NP-hard problem is motivated by applications in comparative genomics; the fixed-parameter algorithm was able to solve the problem on some bacterial genomes. The parameters exploited by the algorithm are the number of breakpoints and the maximum gene copy number in the genomes. Fixed-parameter search tree algorithms have also been applied for solving the MAXIMUM AGREEMENT FOREST problem which arises in the comparison of phylogenetic trees [130]; the fixed-parameter algorithm outperformed two previous approaches for MAXIMUM AGREEMENT FOREST, one using a formulation as integer linear program and another one using a formulation as satisfiability problem. Another example is the search for *k-plexes* in graphs, which can be used, for example, to model functional modules in protein interaction networks. By combining search trees with data reduction, it is often possible to outperform previously used methods [107].

Besides in parameterized algorithmics, search tree algorithms are studied extensively in the area of artificial intelligence and heuristic state space search. There, the key to speedups are *admissible heuristic evaluation functions* which quickly give a lower bound on the distance to the goal. The reason that admissible heuristics are rarely considered by the parameterized algorithmics community in their works (*see* [64] for a counterexample) is that they typically cannot improve the asymptotic running time. Still, the speedups obtained in practice can be quite pronounced, as demonstrated for VERTEX COVER [57].

As with kernelizations, algorithmic developments outside the fixed-parameter setting can make use of the insights that have been gained in the development of depth-bounded search trees in a fixed-parameter setting. One example for this is the MINIMUM QUARTET INCONSISTENCY problem arising in the construction of evolutionary trees. Here, an algorithm that uses depth-bounded search trees was developed by Gramm and Niedermeier [64]. Their insight was used by Wu et al. [134] to develop a faster (non-parameterized) algorithm for this problem.

In conclusion, depth-bounded search trees with clever branching rules are certainly one of the first approaches to try when solving fixed-parameter tractable problems in practice.

## 4 Dynamic Programming

Dynamic programming is one of the most useful algorithm design techniques in bioinformatics; it also plays an important role in developing fixed-parameter algorithms. Since dynamic programming is a classic algorithm design technique covered in many standard textbooks [122], we keep the presentation of "fixed-parameter dynamic programming" short.

### 4.1 Basic Concepts

The general idea is to recursively break down the problem into possibly overlapping subproblems whose optimal solution allows to find an overall optimal solution. The solutions to subproblems are stored in a table, avoiding recalculation. A classic example is sequence alignment of two strings, for instance, using the Needleman–Wunsch algorithm [18]. The dynamic programming technique, however, is not restricted to polynomial-time solvable problems.

The running time of dynamic programming depends mainly on the table size, so the main trick in obtaining fixed-parameter dynamic programming algorithms is to bound the size of the table by a function of the parameter times a polynomial in the input size. Two generic methods for this are tree decompositions and color-coding, described in Subheadings 5 and 6, respectively. In many cases, however, the table size is obviously bounded in the parameter and thus no additional techniques are necessary to obtain a fixed-parameter algorithm.

### 4.2 Case Study

One application of dynamic programming is in the interpretation of mass spectrometry data, which contains mass peaks for a sample molecule and for fragments thereof [22, 27]. The method builds a graph where a vertex corresponds to a possible molecular formula of a peak, and an edge corresponds to a hypothetical fragmentation step. Edges are weighted by the likeliness of the corresponding fragmentation step. The goal is then to calculate a maximum scoring subtree of this graph. In this tree, we must use only one of the molecular formulas of a peak. This is achieved by giving each vertex a corresponding color and asking for a *colorful* subtree.

MAXIMUM COLORFUL SUBTREE
**Input:** A directed graph $D = (V, A)$ with a vertex coloring $c\colon V \to C$ and arc weights $w : A \to \mathbb{Q}_+$.
**Task:** Find a subtree of $G$ that uses each color at most once and has maximum total arc weight.

This NP-hard problem can be solved by dynamic programming [22, 27, 120] by building a table $W(v, S)$ for $v \in V$ and $S \subseteq C$. An entry $W(v, S)$ holds the maximum score of a subtree with root $v$ whose vertex set has exactly the colors of $S$. The table is filled out with the following recurrence:

$$W(v,S) = \max \begin{cases} \max_{u \in V : c(u) \in S \setminus \{c(v)\}} W(u, S \setminus \{c(v)\}) + w(v,u) \\ \max_{\substack{(S_1,S_2) : S_1 \cap S_2 = \{c(v)\} \\ S_1 \cup S_2 = S}} W(v,S_1) + W(v,S_2) \end{cases} \tag{1}$$

with initial condition $W(v, \{c(v)\}) = 0$ and the weight of nonexistent arcs set to $-\infty$. The first line extends a tree by introducing $v$ as

new root and adding the arc $(v, u)$, and the second line merges two trees that have the same root but are otherwise disjoint.

The table $W$ has $n \cdot 2^k$ entries where $k$ is the number of different vertex colors, and filling it out can be done in $O(3^k km)$ time. Thus, MAXIMUM COLORFUL SUBTREE is fixed-parameter tractable with respect to the parameter $k$. In the application, the parameter is the number of peaks in the spectrum which is usually small.

### 4.3 Applications and Implementations

The algorithm described in Subheading 4.2 was found to be fast and accurate in determining glycan structure [27]. There are several further applications of dynamic programming over exponentially sized tables. In phylogenetics, for example, the task of reconciling a binary gene tree with a nonbinary species trees can be solved via a dynamic programming algorithm whose table size is exponential only in the maximum outdegree of the species tree [125]. The implementation solves instances based on cyanobacterial gene trees on average in less than 1 s. In these instances, the parameter value ranges from 2 to 6.

Another application of dynamic programming is in a variant of haplotyping (*see* also Subheading 7.2) which deals with the analysis of genomic fragments. Using dynamic programming, solutions to the *weighted minimum error correction* formulation can be found in a running time of $O(2^k \cdot m)$. Here, $k$ is the maximum coverage of any genome position by the input fragments and $m$ is the number of SNPs per sequencing read [116]. The algorithm scales up to $k \approx 20$.

## 5 Tree Decompositions of Graphs

Many NP-hard graph problems become computationally feasible when they are restricted to cycle-free graphs, that is, *trees* or collections of trees (*forests*). Trees, while potentially simplifying computation, form a very limited class of graphs that seldom suffices as a model for real-life applications. Hence, as a compromise between general graphs and trees, one might want to look at "tree-like" graphs. This tree-likeness can be formalized by the concept of *tree decompositions*. In this section, we survey some important aspects of tree decompositions and their algorithmic use with respect to computational biology and FPT. Surveys on this topic are given by Berger et al. [11] and Bodlaender and Koster [29].

### 5.1 Basic Concepts

There is a very helpful and intuitive characterization of tree decompositions in terms of a robber–cop game in a graph [28]: A robber stands on a graph vertex and, at any time, he can run at arbitrary speed to any other vertex of the graph as long as there is a path connecting both. The only restriction is that he is not permitted to run through a cop. There can be several cops and, at any time, each

of them may either stand on a graph vertex or be in a helicopter (that is, she is above the game board and can move anywhere without being restricted by graph edges). The cops want to land a helicopter on the vertex occupied by the robber. The robber can see a helicopter approaching its landing vertex and he may run to a new vertex before the helicopter actually lands. Thus, the cops want to occupy all vertices adjacent to the robber's vertex, making him unable to move, and to then land one more remaining helicopter on the robber's vertex itself to catch him. The *treewidth* of the graph is the minimum number of cops needed to catch a robber minus one (observe that if the graph is a tree, two cops suffice and trees hence have a treewidth of one) and a corresponding *tree decomposition* is a tree structure that provides the cops with a scheme to catch the robber. Intuitively, the tree decomposition indicates "bottlenecks" (separators) in the graph and thus reveals an underlying scaffold that can be exploited algorithmically.

Formally, tree decompositions and treewidth center around the following somewhat technical definition; Fig. 8 shows a graph together with an optimal tree decomposition of width two.

**Definition 3.** *Let* $G = (V, E)$ *be an undirected graph. A* tree decomposition *of* G *is a pair* $\langle \{X_i | i \in I\}, T \rangle$ *where each* $X_i$ *is a subset of* V, *called a* bag, *and* T *is a tree with the elements of* I *as nodes. The following three properties must hold:*

1. $\bigcup_{i \in I} X_i = V$;
2. *for every edge* $\{u, v\} \in E$, *there is an* $i \in I$ *such that* $\{u, v\} \subseteq X_i$; *and*
3. *for all* $i, j, k \in I$, *if* j *lies on the path between* i *and* k *in* T, *then* $X_i \cap X_k \subseteq X_j$.

*The* width *of* $\langle \{X_i | i \in I\}, T \rangle$ *equals max*$\{| X_i | \, | i \in I\} - 1$. The treewidth *of* G *is the minimum* k *such that* G *has a tree decomposition of width* k.

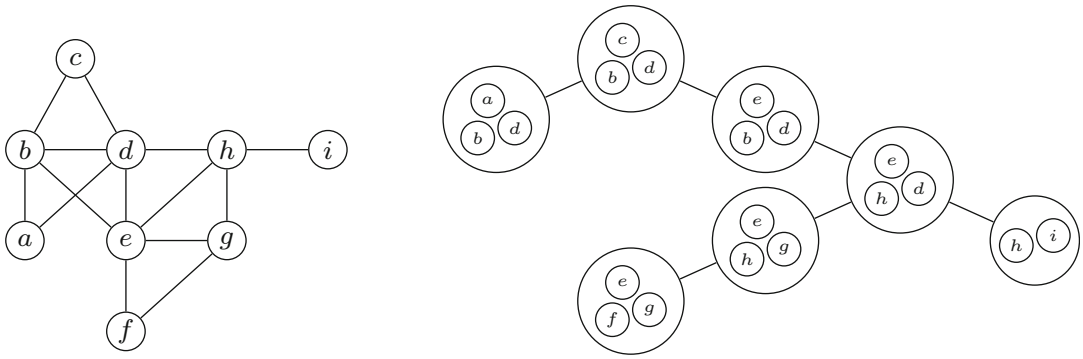



**Fig. 8** A graph together with a tree decomposition of width 2. Observe that—as demanded by the consistency property—each graph vertex induces a subtree in the decomposition tree

The third condition of the definition is often called *consistency property*. It is important in dynamic programming, the main algorithmic tool when solving problems on graphs of bounded treewidth. An equivalent formulation of this property is to demand that for any graph vertex $v$, all bags containing $v$ form a connected subtree.

For trees, the bags of a corresponding tree decomposition are simply the two-element vertex sets formed by the edges of the tree. In the definition, the subtraction of 1 thus ensures that trees have a treewidth of 1. In contrast, a clique of $n$ vertices has treewidth $n - 1$. The corresponding tree decomposition trivially consists of one bag containing all graph vertices; in fact, no tree decomposition with smaller width is attainable since it is known that every complete subgraph of a graph $G$ is completely "contained" in a bag of $G$'s tree decomposition.

Tree decompositions of graphs are connected to another central concept in algorithmic graph theory: *graph separators* are vertex sets whose removal from the graph separates the graph into two or more connected components. Each bag of a tree decomposition forms a separator of the corresponding graph.

Given a graph, determining its treewidth is an NP-hard problem itself. However, several tools and heuristics exist that construct tree decompositions [29–31], and for some graphs that appear in practice, computing a tree decomposition is easy. Here, we concentrate on the algorithmic use of tree decompositions, assuming that they are provided to us.

### 5.2 Case Study

Typically, tree decomposition-based algorithms have two stages:

1. Find a tree decomposition of bounded width for the input graph.
2. Solve the problem by dynamic programming on the tree decomposition, starting from the leaves.

Intuitively speaking, a decomposition tree provides us with a scaffold-structure that allows for efficient and consistent processing through the graph. By design, this scaffold leads to optimal solutions even when the utilized tree decompositions are not optimal; however, the algorithm will run slower and consume more memory in that case.

To exemplify dynamic programming on tree decompositions, we make use of our running example VERTEX COVER and sketch a fixed-parameter dynamic programming algorithm for VERTEX COVER with respect to the parameter treewidth.

**Theorem 2.** For a graph G with a given width-ω tree decomposition $\langle \{X_i | i \in I\}, T \rangle$, an optimal vertex cover can be computed in $O(2^{\omega} \cdot \omega \cdot |I|)$ time.

The basic idea of the algorithm is to examine for each bag $X_i$ all of the at most $2^{|X_i|}$ possibilities to obtain a vertex cover for the subgraph $G[X_i]$. This information is stored in tables $A_i$, $i \in I$. Adjacent tables are updated in a bottom-up process starting at the leaves of the decomposition tree. Each bag of the tree decomposition thus has a table associated with it. During this updating process it is guaranteed that the "local" solutions for each subgraph associated with a bag of the tree decomposition are combined into a "globally optimal" solution for the overall graph $G$. (We omit several technical details here; these can be found in [109, Chapter 10].) The following points of Definition 3 guarantee the validity of this approach:

1. The first condition in Definition 3, that is, $V = \bigcup_{i \in I} X_i$, makes sure that every graph vertex is taken into account during the computation.
2. The second condition in Definition 3, that is, $\forall e \in E \; \exists i \in I : \; e \in X_i$, makes sure that all edges can be treated and thus will be covered.
3. The third condition in Definition 3 guarantees the consistency of the dynamic programming, since information concerning a particular vertex $v$ is only propagated between neighboring bags that both contain $v$.

While the running time of the dynamic programming part can often be improved over a naive approach, there is evidence that known algorithms for some basic combinatorial problems are essentially optimal [101].

One thing to keep in mind for a practical application is that storing dynamic programming tables requires memory space that grows exponentially in the treewidth. Hence, even for "small" treewidths, say, between 10 and 20, the computer program may run out of memory and break down. Some techniques for limiting memory use have been proposed [12, 55, 70].

### 5.3 Applications and Implementations

Tree decomposition-based algorithms are a valuable alternative whenever the underlying graphs have small treewidth. As a rule of thumb, the typical border of practical feasibility lies somewhere below a treewidth of 20 for the underlying graph, although with advantageous data and careful implementation higher values are possible (e.g., [70]). Successful implementations for solving VERTEX COVER with tree decomposition approaches have been reported [4, 12].

A practical application of tree decompositions is found in protein structure prediction, namely the prediction of backbone structures and side-chain prediction. These two problems can be modeled as a graph labeling problem, where the resulting graphs

have a very small treewidth in practice, allowing the problems to be solved efficiently [11].

Besides taking an input graph, computing a tree decomposition for it, and hoping that the resulting tree decomposition has small treewidth, there have also been cases where a problem is modeled as a graph problem such that it can be *proven* that the resulting graphs have a tree decomposition with small treewidth that can efficiently be found. As an example, Song et al. [123] used a so-called conformational graph to specify the consensus sequence-structure of an RNA family. They proved that the treewidth of this graph is basically determined by the structural elements that appear in the RNA. More precisely, they showed that if there is a bounded number of crossing stems, say $k$, in a pseudoknot structure, then the resulting graph has treewidth $(2 + k)$. Since the number of crossing stems is usually small, this yields a fast algorithm for searching RNA secondary structures (*see* also [135]).

Other biological applications include peptide sequencing and spectral alignment [100], molecule bond multiplicity inference [26], charge group partitioning for biomolecular simulations [38], and NMR interpretation [98]. The idea of exploiting the treewidth of an auxiliary structure describing interdependencies of the input also has attracted much attention in artificial intelligence (AI) applications [62, 85].

Besides dynamic programming, a very powerful method to obtain fixed-parameter results for the parameter treewidth is to cast the problem as an expression in *monadic second-order logic* (MSO) [97]. For example, for VERTEX COVER, the expression is

$$\mathrm{vc}(U) := \forall x, y \in V : \neg(\{x, y\} \in E) \vee x \in U \vee y \in U.$$

Since the worst-case running time obtained from this formulation is extremely bad, this approach was thought to be impractical [109, Chapter 10]. However, recently a solver was presented that indeed just requires the user to provide the MSO expression [88, 96, 97]. If the problem at hand admits a formulation in MSO (as most problems that are fixed-parameter tractable for treewidth do), this provides a quick way to evaluate the feasibility of the treewidth approach for the data at hand, with the option to get a quicker algorithm by designing a customized dynamic programming.

Besides treewidth, a number of alternative concepts have been developed to compare the structure of a graph to a tree, including branch-width, rank-width, and hypertree-width [78, 97].

## 6 Color-Coding

The color-coding technique due to Alon et al. [5] is a general method for finding small patterns in graphs. In its simplest form,

color-coding can solve the MINIMUM-WEIGHT PATH problem, which asks for the cheapest path of length $k$ in a graph. This has been successfully employed with protein–protein interaction networks to find signaling pathways [82, 120] and to evaluate pathway similarity queries [121].

### 6.1 Basic Concepts

A naive approach to discover a small structure of $k$ vertices within a graph of $n$ vertices would be to combinatorially try all of the roughly $n^k$ possibilities of selecting $k$ out of $n$ vertices and then testing the selection for the desired structural property. This approach quickly leads to a combinatorial explosion, making it infeasible even for rather small input graphs of a few hundred vertices. The central idea of color-coding is to randomly color each vertex of a graph with one of $k$ colors and to hope that all vertices in the subgraph searched for obtain different colors (that is, the vertex set becomes *colorful*).

When the structure that is searched for becomes colorful, the task of finding it can be solved by dynamic programming in a running time where the exponential part solely depends on $k$, the size of the substructure searched for. Of course, given the randomness of the initial coloring, most of the time the target structure will actually not be colorful. Therefore, we have to repeat the process of random coloring and searching (called a *trial*) many times until the target structure is colorful at least once with sufficiently high probability. As we will show, the number of trials also depends only on $k$ (albeit exponentially). Consequently this algorithm has a fixed-parameter running time. Thus it is much faster than the naive approach which needs $O(n^k)$ time.

### 6.2 Case Study

Formally stated, the problem we consider is the following:

MINIMUM-WEIGHT PATH
**Input:** An undirected graph $G$ with edge weights $w : E \to \mathbb{Q}^+$ and a nonnegative integer $k$.
**Task:** Find a simple length-$k$ path in $G$ that minimizes the sum over its edge weights.

This problem is well known to be NP-hard [61, ND29]. What makes the problem hard is the requirement of *simple* paths, that is, paths where no vertex may occur more than once (otherwise, it is easily solved by traversing a minimum-weight edge $k - 1$ times).

Given a fixed coloring of vertices, finding a minimum-weight path that is colorful can be accomplished by dynamic programming: Assume that for some $i < k$ we have computed a value $W(v, S)$ for every vertex $v \in V$ and every cardinality-$i$ subset $S$ of vertex colors such that $W(v, S)$ denotes the minimum weight of a path that uses each color in $S$ exactly once and ends in $v$. Clearly, the resulting path is simple because no color is used more than once. We can now use this to compute the values $W(v, S)$ for all cardinality-$(i + 1)$ subsets $S$ and vertices $v \in V$, because any colorful

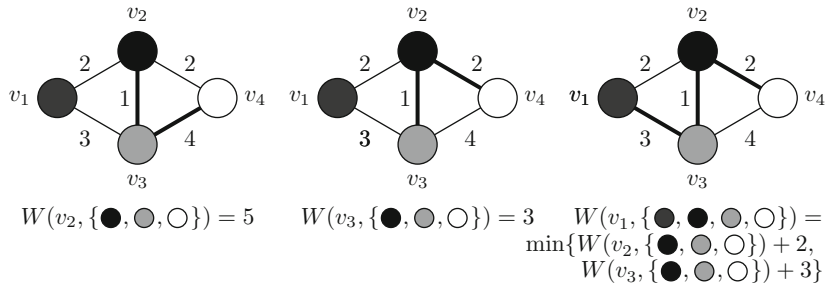



**Fig. 9** Example for solving MINIMUM-WEIGHT PATH using the color-coding technique. Here, using (2) a new table entry (*right*) is calculated using two already known entries (*left* and *middle*)

length-$(i+1)$ path that ends in a vertex $v \in V$ must be composed of a colorful length-$i$ path that does not use the color of $v$ and ends in a neighbor of $v$.

More precisely, we let

$$W(v, S) = \min_{e=\{u,v\} \in E} \Big( W(u, S \backslash \{\text{color}(v)\}) + w(e) \Big). \tag{2}$$

See Fig. 9 for an example.

It is straightforward to verify that on an $m$-edge graph the dynamic programming takes $O(2^k m)$ time. Whenever the minimum-weight length-$k$ path $P$ in the input graph is colored with $k$ different colors (that is, its vertex set is colorful), then the algorithm finds $P$. The problem, of course, is that the coloring of the input graph is random and hence many coloring trials have to be performed to ensure that the minimum-weight path is found with a high probability. More precisely, the probability of any length-$k$ path (including the one with minimum weight) being colorful in a single trial is

$$P_c = \frac{k!}{k^k} > \sqrt{2\pi k} e^{-k} \tag{3}$$

because there are $k^k$ ways to arbitrarily color $k$ vertices with $k$ colors and $k!$ ways to color them such that no color is used more than once. Using $t$ trials, a path of length $k$ is found with probability $1 - (1 - P_c)^t$. Therefore, to ensure that a colorful path is found with a probability greater than $1 - \varepsilon$ (for any $0 < \varepsilon \leq 1$), at least

$$t(\varepsilon) = \left\lceil \frac{\ln \varepsilon}{\ln(1 - P_c)} \right\rceil = -\ln \varepsilon \cdot O(e^k) \tag{4}$$

trials are needed. This bounds the overall running time by $2^{O(k)} \cdot n^{O(1)}$. While the result is only correct with a certain probability, we can specify any desired error probability, say 0.1 %, noting that even very low error probabilities do not incur excessive extra running time costs.

Note that the number of colors chosen poses a trade-off: While using more than $k$ colors increases the chance of a target structure becoming colorful—and thus decreases the number of trials needed to achieve a given error probability—it increases the running time and memory requirements of the dynamic programming step. As a theoretical analysis points out, using $1.3k$ colors instead of just $k$ improves the worst-case running time of the color-coding algorithm. Moreover, in practice it is often beneficial to increase the number of colors even further [82].

### 6.3 Applications and Implementations

Protein interaction networks represent proteins by vertices and mutual protein–protein interaction probabilities by weighted edges. They are a valuable source of information for understanding the functional organization of the proteome. Scott et al. [120] demonstrated that *high-scoring simple paths* in the network constitute plausible candidates for linear signal transduction pathways, *simple* meaning that no vertex occurs more than once and *high-scoring* meaning that the product of edge weights is maximized. To match the above definition of MINIMUM-WEIGHT PATH, one works with the *weight* $w(e) := -\log p(e)$ of an edge $e$ with interaction probability $p(e)$ between $e$'s endpoints. Then minimizing the sum of the weights is equivalent to maximizing the product of the probabilities.

The currently most efficient implementation based on color-coding [82] is capable of finding optimal paths of length up to 20 in seconds within a yeast protein interaction network containing about 4 500 vertices.

A particularly appealing aspect of color-coding is that it can be easily adapted to many practically relevant variations of the problem formulation:

- The set of vertices where a path can start and end can be restricted (such as to force it to start in a membrane protein and end in a transcription factor [120]).
- Not only the minimum-weight path can be computed but rather a collection of low-weight paths (typically, one demands that these paths must differ in a certain amount of vertices to ensure that they are diverse and not small modifications of the global minimum-weight path) [82].
- More generally, pathway queries to a network, that is, the task of finding a pathway in a network that is as similar as possible to a query pathway, can be handled with color-coding [121].

Several other works use color-coding for querying in protein interaction networks. For example, the queries can be trees, allowing for identification of non-exact (homeomorphic) matches [52]. Another application is counting non-induced occurrences of subgraph topologies in the form of trees and bounded treewidth subgraphs [6].

A further use of color-coding is to solve the GRAPH MOTIF problem. In a biological application of GRAPH MOTIF, the query is a set of proteins, and the task is to find a matching set of proteins that are sequence-similar to the query proteins and span a connected region of the network. Bruckner et al. [33] and Betzler et al. [13] provided implementations based on color-coding; they differ in the way insertions and deletions are handled, and are thus not directly comparable.

Further, color-coding has also found applications in string problems: for example, Bonizzoni et al. [32] used it to solve a variant of LONGEST COMMON SUBSEQUENCE that is motivated by a sequence comparison problem. However, to the best of our knowledge no string algorithm using color-coding has been implemented yet.

### 6.3.1 Related Techniques

We mention some techniques that use ideas similar to color-coding. To the best of our knowledge, with one exception none of them has been implemented so far.

Two variants use only two colors to separate the pattern from surrounding vertices (*random separation*) [37] or to divide the graph into two parts for recursion (*divide-and-color*) [87]. Random separation can be used to find small subgraphs with desired properties in sparse graphs. For these problems enumerating connected subgraphs and using color-coding [91] sometimes gives faster algorithms. A further extension known as *chromatic coding* was used to obtain (theoretically) fast algorithms for the DENSE TRIPLET INCONSISTENCY problem motivated from phylogenetics [72].

Algebraic techniques [92, 93] can improve on the worst-case running time of many color-coding approaches; for example, the currently strongest worst-case bound for GRAPH MOTIF is obtained this way [16]. This approach, however, is not as flexible as color-coding, for example, with respect to the handling of large weights. Experiments for the unweighted version of MINIMUM-WEIGHT PATH on random graphs have shown that the approach is feasible for a path length of 16 and 8000 vertices [17].

## 7 Iterative Compression

The main idea of iterative compression is induction: we construct a slightly smaller instance, solve it recursively, and then make use of the solution to solve the actual instance. While induction is a classic algorithmic approach, iterative compression first appeared in a work by Reed et al. in 2004 (*see* also a 2009 survey [75]). Although it is perhaps not quite as generally applicable as data reduction or search trees, it appears to be useful for solving a wide range of problems and has led to significant breakthroughs in showing fixed-parameter tractability results. Iterative compression is typically

used for "minimum obstruction deletion" problems: given a set of items, omit the minimum number of items such that the remaining items exhibit some "nice" structure. Thus, it can sometimes model parsimonious error correction. VERTEX COVER is one example fitting this scheme, and it can be solved with iterative compression [117].

### 7.1 Basic Concepts

The central concept of iterative compression is to employ a so-called *compression routine*.

**Definition 4.** *A* compression routine *is an algorithm that, given a problem instance and a solution of size* k, *either calculates a smaller solution or proves that the given solution is of minimum size.*

With a compression routine, we can find an optimal solution for an instance by recursively solving a smaller instance, using the solution for the smaller instance to find a possibly suboptimal solution for the actual instance, and then using the compression routine to find an optimal solution. For "minimum obstruction deletion" problems, the only nontrivial step is the compression routine.

The main strength of iterative compression is that it allows us to see a problem from a different angle, since the compression routine does not only have the problem instance as input, but also a solution, which carries valuable structural information on the input. Also, the compression routine does not need to find an optimal solution at once, but only any better solution. Therefore, the design of a compression routine can often be simpler than designing a complete algorithm.

Algorithmically, the compression routine is the "complex" step in iterative compression in two regards: First, while the mode of use of the compression routine is usually straightforward, finding the compression routine itself often is not. Second, if the compression routine is a fixed-parameter algorithm with respect to the parameter $k$, then so is the whole algorithm.

### 7.2 Case Studies

The showcase for iterative compression is the VERTEX BIPARTIZATION problem, also known as ODD CYCLE COVER.

VERTEX BIPARTIZATION
**Input:** An undirected graph $G = (V, E)$ and a nonnegative integer $k$.
**Task:** Find a set $D \subseteq V$ of at most $k$ vertices such that $G[V \setminus D]$ is bipartite.

This problem appears as MINIMUM FRAGMENT REMOVAL in the context of SNP haplotyping [112]. When analyzing DNA fragments obtained by shotgun sequencing, it is initially unknown which of the two chromosome copies of a diploid organism a fragment belongs to. We can, however, determine for some pairs of fragments that they cannot belong to the same chromosome copy since they contain conflicting information at some SNP locus.

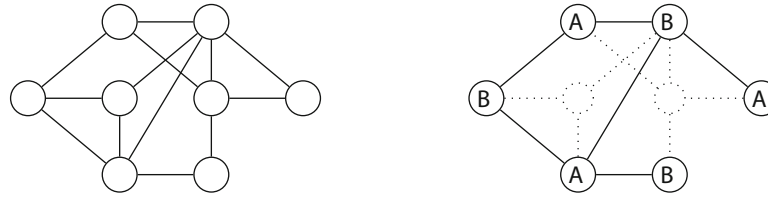



**Fig. 10** A VERTEX BIPARTIZATION instance (*left*), and an optimal solution (*right*): when deleting two fragments (*dashed*), the remaining fragments can be allocated to the two chromosome copies (A and B) such that no conflicting fragments get the same assignment

Using this information, it is straightforward to reconstruct the chromosome assignment. We can model this as a graph problem, where the fragments are the vertices and a conflict is represented as an edge. The task is then to color the vertices with two colors such that no vertices with the same color are adjacent. The problem gets difficult in the presence of errors such as parasite DNA fragments which randomly conflict with other fragments. In this scenario, we ask for the least number of fragments to remove such that we can get a consistent fragment assignment (*see* Fig. 10). Using the number of fragments $k$ to be removed as a parameter is a natural approach, since the result is only meaningful for small $k$ anyway.

Iterative compression provided the first fixed-parameter algorithm for VERTEX BIPARTIZATION with this parameter [118]. We sketch how to apply this to finding an optimal solution (a *removal set*) for a VERTEX BIPARTIZATION instance $(G = (V, E), k)$. Choose an arbitrary vertex $v$ and let $G'$ be $G$ with $v$ deleted. Recursively find an optimal removal set $R'$ for $G'$ (this recursion terminates after $n = |V|$ steps, where we can yield the empty removal set for the empty graph). Clearly, $R' \cup \{v\}$ is a removal set for $G$, although it might not be optimal (it can be too large by one). Now using the compression routine for $G$ and $R' \cup \{v\}$, we can find an optimal solution for $G'$.

The compression routine itself works by examining a number of vertex cuts in an auxiliary graph (that is, a set of vertices whose deletion makes the graph disconnected), a task which can be accomplished in polynomial time by maximum flow techniques. We refer to the literature for details [80, 95, 118]. The running time of the complete algorithm is $O(3^k \cdot mn)$ [80].

### 7.3 Applications and Implementations

The iterative compression algorithm for VERTEX BIPARTIZATION has been employed for a number of biological applications. An implementation, improved by heuristics, can solve all MINIMUM FRAGMENT REMOVAL problems from a testbed based on human genome data within minutes, whereas established methods are only able to solve about half of the instances within reasonable time [80]. The UNORDERED MAXIMUM TREE ORIENTATION, which models inference of signal

transmissions in protein–protein interaction networks based on cause–effect pairs, can be reduced to GRAPH BIPARTIZATION [21]. Also, ordering and orienting contigs produced during genome assembly can be reduced to GRAPH BIPARTIZATION, and this is implemented in the SCARPA scaffolder [51]. Recently, an algorithm with a better worst-case bound of $2.32^k \cdot n^{O(1)}$ based on linear programming was presented [102], which seems like a promising alternative to iterative compression for VERTEX BIPARTIZATION.

EDGE BIPARTIZATION, the edge deletion version of VERTEX BIPARTIZATION, can also be solved by iterative compression [74]. Enhanced with data reduction rules and generalized to the SIGNED GRAPH BALANCING problem, this algorithm was used to analyze gene regulatory networks [83]. It can solve many networks to optimality, but fails for the largest ones [83]. The TANGLEGRAM LAYOUT problem is about drawing two phylogenetic trees on the same species set in order to facilitate analysis; it can be reduced to EDGE BIPARTIZATION [24]. The implementation by Hüffner et al. [83] can find exact solutions for all practically relevant TANGLEGRAM LAYOUT instances within seconds [24]. Finally, computing the minimum number of recombination events for general pedigrees with two sites for all members can also be reduced to EDGE BIPARTIZATION [49].

Another prominent problem amenable to iterative compression is FEEDBACK VERTEX SET, which also has applications for genetic linkage analysis [10]. While initial algorithms based on iterative compression [47, 74] had prohibitive worst-case running times, the currently fastest known approach runs in $3.619^k \cdot n^{O(1)}$ time for finding a feedback vertex set of $k$ vertices [89]. However, these algorithms have not been implemented yet.

The DIRECTED FEEDBACK VERTEX SET problem was also shown to be fixed-parameter tractable by iterative compression [42], solving a long-standing open question. However, the worst-case running time bound is much worse than for the previously mentioned problems. Still, an experimental evaluation on random graphs [58], employing also data reduction, showed encouraging results for very small parameter values. DIRECTED FEEDBACK VERTEX SET has applications in pairwise genome alignment under the duplication-loss model [50] and in the comparison of gene orders [71]. For an application in reconstructing reticulation networks in particular, the authors mention that the parameter could be expected to be very small [99].

Finally, the CLUSTER VERTEX DELETION problem, the "vertex deletion variant" of CLUSTER EDITING, aims to cluster objects by removing objects that do not fit in the cluster structure. It can also be solved by a fixed-parameter algorithm with respect to the number of removed vertices using iterative compression [84].

## 8 A Roadmap Towards Efficient Implementations

Here we try to give some general recommendations on how to go about applying parameterized algorithmics to NP-hard computational problems in practice.

### 8.1 Identification of Parameters

The first task is to identify fruitful parameters. As detailed in Subheading 1.2, it is useful to consider several "structural" parameters, possibly also deduced from a data-driven analysis of the input instances. The usefulness of the parameter clearly depends on whether it is small in the input instances. For graph instances, a tool such as Graphana (http://fpt.akt.tu-berlin.de/graphana/) that calculates a wide range of graph parameters can be helpful. At this point, it is also useful to determine whether the problem is fixed-parameter tractable or W[1]-hard. While a hardness result encourages to look for another parameter or combined parameters, bear in mind that certain techniques such as data reduction can still be effective in practice even without a performance guarantee.

### 8.2 Implementation of Brute-Force Search

The next thing to do is to implement a brute-force search that is as simple as possible. There are several reasons for this: First, it gives some first impression on what solutions look like (for example, can we use their size as parameter?). Second, a simple starting implementation is invaluable in shaking out bugs from later, more sophisticated implementations, in particular if results for random instances are systematically compared. Possibly the best way to get a simple brute-force result is to use an integer linear program (ILP). These sometimes need only a few lines when using a modeling language, but are often surprisingly effective. The second method of choice is a simple search tree (Subheading 3).

### 8.3 Implementation of Data Reduction

Data reduction is valuable in combination with any other algorithmic technique such as approximation, heuristics, or fixed-parameter algorithms. In some cases it can even completely solve instances without further effort; it can be considered as essential for the treatment of NP-hard problems. Thus it should always be the first nontrivial technique to be developed and implemented. When combined with even a naive brute-force approach, it can often already solve instances of notable size. For large instances, an efficient implementation of the data reduction rules is necessary. A rule of thumb is to aim for linear running time for most of the implemented data reduction rules and to apply linear-time data reduction rules first [126].

### 8.4 Tuned Search Trees

After this, the easiest speedups typically come from a more carefully tuned search tree algorithm. Case distinction can help to improve provable running time bounds, although it has often been reported

that a too complicated branching actually leads to a slowdown. Heuristic branching priorities can help, as well as admissible heuristic evaluation functions [57]. Further, interleaving with data reduction can lead to a speedup [111].

### 8.5 Non-traditional Techniques

When search trees are not applicable or too slow, less clear instructions can be given. The best thing to do is to look at other fixed-parameter algorithms and techniques for inspiration: are we looking for a small pattern in the input? Possibly color-coding (Subheading 6) helps. Are we looking for minimum modifications to obtain a nice combinatorial structure? Possibly iterative compression (Subheading 7) is applicable. In this way, using some of the less common approaches of fixed-parameter algorithms, one might still come up with a fixed-parameter algorithm.

Here, one should be wary of exponential-space algorithms as these can often fill the memory within seconds and therefore become unusable in practice. In contrast, one should *not* be too afraid of bad upper bounds for fixed-parameter algorithms—the analysis is worst-case and often much too pessimistic.

### 8.6 Heuristic Speedups

Some of the largest speedups experienced in experiments come from techniques that can be considered heuristic in the sense that they do not improve worst-case time bounds or the kernel size. The general idea of most heuristics is to recognize early that some branches or subcases cannot lead to an optimal solution and to skip those. Their potential effectiveness, even when no performance guarantees can be given, should always be kept in mind when implementing algorithms.

Furthermore, most algorithms will have numerous degrees of freedom concerning their actual implementation, execution order, and the value of some thresholds, for example, concerning the fraction of search tree nodes to which data reduction should be applied. There are tools for algorithm configuration that can exploit this freedom and may yield magnitudes of speedup [79].

## 9 Conclusion

We surveyed several techniques for developing efficient fixed-parameter algorithms for computationally hard (biological) problems. Since many of these problems appear to "carry small parameters," we firmly believe that there will continue to be a strong interaction between parameterized complexity analysis and algorithmic bioinformatics. To make this as fruitful as possible, it is necessary to analyze real-world data in search for "hidden structure" which can be captured by suitable parameterizations. A subsequent parameterized complexity analysis can then determine which of these parameterizations yield field-parameter algorithms.

This data-driven line of algorithmic research is still underdeveloped and should receive increased attention in future research. Moreover, in order to obtain the practically most useful algorithms, it may often be good to combine fixed-parameter algorithms (particularly, data reduction and kernelization) with general-purpose tools for solving computationally hard problems, including SAT solving and integer linear programming. This certainly will need a lot of experimentation going far beyond purely theoretical algorithm design.

## 10 Notes

1. To show that a problem is unlikely to be fixed-parameter tractable, the concept of W[1]-hardness was developed. It is widely assumed that a W[1]-hard problem cannot have a fixed-parameter algorithm (W[$t$]-hardness, $t \geq 2$ has the same implication). For example, the CLIQUE problem to find a clique (complete subgraph) in an undirected graph is W[1]-hard with respect to the parameter "number of vertices in the clique." To show that a problem is W[1]-hard, a *parameterized reduction* from a known W[1]-hard problem can be used (*see*, e.g., [41, 54]).
2. There exist suitable data reduction rules when it is of interest to enumerate *all* minimal vertex covers of a given graph. For example, Damaschke [46] suggests the notion of a *full kernel* that contains all minimal solutions in a compressed form and thereby allows enumeration of them.
3. One technique to show that a polynomial kernel is unlikely is called *composition* [53, 94]. A composition is an algorithm that combines the inputs of many instances of a problem into one "equivalent" instance. For 2-CLUB, the composition is to take the disjoint union of the input graphs of the instances: Any solution to such a combined instance has to live completely inside one of its connected components, which are completely contained in one of the original input instances. Thus, the combined instance has a solution if and only if at least one of the input instances has one. The existence of a composition and a polynomial kernelization leads to an implausible complexity-theoretic collapse. Thus, it is widely assumed that there is no polynomial problem kernel for problems with a composition [53, 94].

## Acknowledgements

This is a completely revised, updated, and significantly expanded version of the previous book chapter "Developing Fixed-Parameter

Algorithms to Solve Combinatorially Explosive Biological Problems" authored by Hüffner, Niedermeier, and Wernicke.
Falk Hüffner was supported by the Deutsche Forschungsgemeinschaft (DFG), project ALEPH (HU 2139/1).

Christian Komusiewicz was supported by the Deutsche Forschungsgemeinschaft (DFG), project MAGZ (KO 3669/4-1).

## References

1. Abu-Khzam FN, Collins RL, Fellows MR, Langston MA, Suters WH, Symons CT (2004) Kernelization algorithms for the vertex cover problem: theory and experiments. In: Proceedings of 6th workshop on algorithm engineering and experiments (ALENEX '04). SIAM, Philadelphia, PA, pp 62–69
2. Abu-Khzam FN, Langston MA, Shanbhag P, Symons CT (2006) Scalable parallel algorithms for FPT problems. Algorithmica 45 (3):269–284
3. Abu-Khzam FN, Daudjee K, Mouawad AE, Nishimura N (2013) An easy-to-use scalable framework for parallel recursive backtracking. Technical Report arXiv:1312.7626, arXiv
4. Alber J, Dorn F, Niedermeier R (2005) Empirical evaluation of a tree decomposition based algorithm for vertex cover on planar graphs. Discret Appl Math 145(2):219–231
5. Alon N, Yuster R, Zwick U (1995) Color-coding. J ACM 42(4):844–856
6. Alon N, Dao P, Hajirasouliha I, Hormozdiari F, Sahinalp SC (2008) Biomolecular network motif counting and discovery by color coding. Bioinformatics 24(13):i241–i249
7. Althaus E, Klau GW, Kohlbacher O, Lenhof H, Reinert K (2009) Integer linear programming in computational biology. In: Efficient algorithms, essays dedicated to Kurt Mehlhorn on the occasion of his 60th birthday. Lecture notes in computer science, vol 5760. Springer, Berlin, pp 199–218
8. Atias N, Sharan R (2012) Comparative analysis of protein networks: hard problems, practical solutions. Commun ACM 55(5):88–97
9. Ausiello G, Crescenzi P, Gambosi G, Kann V, Marchetti-Spaccamela A, Protasi M (1999) Complexity and approximation: combinatorial optimization problems and their approximability properties. Springer, Berlin
10. Becker A, Geiger D, Schäffer A (1998) Automatic selection of loop breakers for genetic linkage analysis. Hum Genet 48(1):49–60
11. Berger B, Singht R, Xu J (2008) Graph algorithms for biological systems analysis. In: Proceedings of the 19th annual ACM-SIAM symposium on discrete algorithms (SODA '08). SIAM, Philadelphia, PA, pp 142–151
12. Betzler N, Niedermeier R, Uhlmann J (2006) Tree decompositions of graphs: saving memory in dynamic programming. Discret Optim 3(3):220–229
13. Betzler N, van Bevern R, Fellows MR, Komusiewicz C, Niedermeier R (2011) Parameterized algorithmics for finding connected motifs in biological networks. IEEE/ACM Trans Comput Biol Bioinform 8(5):1296–1308
14. Biere A, Heule M, van Maaren H, Walsh T (eds) (2009) Handbook of satisfiability. IOS Press, Amsterdam
15. Bixby RE (2002) Solving real-world linear programs: a decade and more of progress. Oper Res 50:3–15
16. Björklund A, Kaski P, Kowalik L (2016) Constrained multilinear detection and generalized graph motifs. Algorithmica 74(2):947–967
17. Björklund A, Kaski P, Kowalik Ł (2014) Fast witness extraction using a decision oracle. In: Proceedings of the 22th annual European symposium on algorithms (ESA '14). Lecture notes in computer science, vol 8737. Springer, Berlin, pp 149–160
18. Böckenhauer H-J, Bongartz D (2007) Algorithmic aspects of bioinformatics. Springer, Berlin
19. Böcker S (2012) A golden ratio parameterized algorithm for cluster editing. J Discrete Algorithms 16:79–89
20. Böcker S, Baumbach J (2013) Cluster editing. In: Proceedings of the 9th conference on computability in Europe (CiE '13). Lecture notes in computer science, vol 7921. Springer, Berlin, pp 33–44
21. Böcker S, Damaschke P (2012) A note on the parameterized complexity of unordered maximum tree orientation. Discret Appl Math 160 (10–11):1634–1638
22. Böcker S, Rasche F (2008) Towards de novo identification of metabolites by analyzing tandem mass spectra. Bioinformatics 24 (16):49–55

23. Böcker S, Briesemeister S, Bui QBA, Truß A (2009) Going weighted: parameterized algorithms for cluster editing. Theor Comput Sci 410(52):5467–5480
24. Böcker S, Hüffner F, Truss A, Wahlström M (2009) A faster fixed-parameter approach to drawing binary tanglegrams. In: Proceedings of the 4th international workshop on parameterized and exact computation (IWPEC '09). Lecture notes in computer science, vol 5917. Springer, Berlin, pp 38–49
25. Böcker S, Briesemeister S, Klau GW (2011) Exact algorithms for cluster editing: evaluation and experiments. Algorithmica 60 (2):316–334
26. Böcker S, Bui QBA, Truß A (2011) Computing bond orders in molecule graphs. Theor Comput Sci 412(12–14):1184–1195
27. Böcker S, Kehr B, Rasche F (2011) Determination of glycan structure from tandem mass spectra. IEEE/ACM Trans Comput Biol Bioinform 8(4):976–986
28. Bodlaender HL (1998) A partial *k*-arboretum of graphs with bounded treewidth. Theor Comput Sci 209:1–45
29. Bodlaender HL, Koster AMCA (2008) Combinatorial optimization on graphs of bounded treewidth. Comput J 51(3):255–269
30. Bodlaender HL, Koster AMCA (2010) Treewidth computations I: upper bounds. Inf Comput 208(3):259–275
31. Bodlaender HL, Fomin FV, Koster AMCA, Kratsch D, Thilikos DM (2012) On exact algorithms for treewidth. ACM Trans Algorithm 9(1):12:1–12:23
32. Bonizzoni P, Vedova GD, Dondi R, Pirola Y (2010) Variants of constrained longest common subsequence. Inf Process Lett 110 (20):877–881
33. Bruckner S, Hüffner F, Karp RM, Shamir R, Sharan R (2010) Topology-free querying of protein interaction networks. J Comput Biol 17(3):237–252
34. Bulteau L, Fertin G, Komusiewicz C, Rusu I (2013) A fixed-parameter algorithm for minimum common string partition with few duplications. In: Proceedings of the 13th international workshop on algorithms in bioinformatics (WABI '13). Lecture notes in computer science, vol 8126. Springer, Berlin, pp 244–258
35. Bulteau L, Hüffner F, Komusiewicz C, Niedermeier R (2014) Multivariate algorithmics for NP-hard string problems. Bull EATCS 114:31–73
36. Cai L, Chen J, Downey RG, Fellows MR (1997) Advice classes of parameterized tractability. Ann Pure Appl Log 84(1):119–138
37. Cai L, Chan SM, Chan SO (2006) Random separation: a new method for solving fixed-cardinality optimization problems. In: Proceedings of the 2nd international workshop on parameterized and exact computation (IWPEC '06). Lecture notes in computer science, vol 4169. Springer, Berlin, pp 239–250
38. Canzar S, El-Kebir M, Pool R, Elbassioni KM, Mark AE, Geerke DP, Stougie L, Klau GW (2013) Charge group partitioning in biomolecular simulation. J Comput Biol 20 (3):188–198
39. Cao Y, Chen J (2012) Cluster editing: kernelization based on edge cuts. Algorithmica 64 (1):152–169
40. Cheetham J, Dehne FKHA, Rau-Chaplin A, Stege U, Taillon PJ (2003) Solving large FPT problems on coarse-grained parallel machines. J Comput Syst Sci 67(4):691–706
41. Chen J, Meng J (2008) On parameterized intractability: hardness and completeness. Comput J 51(1):39–59
42. Chen J, Liu Y, Lu S, O'Sullivan B, Razgon I (2008) A fixed-parameter algorithm for the directed feedback vertex set problem. J ACM 55:(5)
43. Chen J, Kanj IA, Xia G (2010) Improved upper bounds for vertex cover. Theor Comput Sci 411(40–42):3736–3756
44. Chesler EJ, Lu L, Shou S, Qu Y, Gu J, Wang J, Hsu HC, Mountz JD, Baldwin NE, Langston MA, Threadgill DW, Manly KF, Williams RW (2005) Complex trait analysis of gene expression uncovers polygenic and pleiotropic networks that modulate nervous system function. Nat Genet 37:233–242
45. Cormen TH, Leiserson CE, Rivest RL, Stein C (2009) Introduction to algorithms, 3rd edn. MIT Press, Cambridge, MA
46. Damaschke P (2006) Parameterized enumeration, transversals, and imperfect phylogeny reconstruction. Theor Comput Sci 351 (3):337–350
47. Dehne FKHA, Fellows MR, Langston MA, Rosamond FA, Stevens K (2007) An $O(2^{O(k)}n^3)$ FPT algorithm for the undirected feedback vertex set problem. Theory Comput Syst 41(3):479–492
48. Diestel R (2010) Graph theory. Graduate texts in mathematics, vol 173, 4th edn. Springer, Berlin
49. Doan DD, Evans PA (2011) An FPT haplotyping algorithm on pedigrees with a small number of sites. Algorithms Mol Biol 6:8
50. Dondi R, El-Mabrouk N (2013) Aligning and labeling genomes under the duplication-loss model. In: Proceedings of the 9th conference on computability in Europe (CiE '13).

Lecture notes in computer science, vol 7921. Springer, Berlin, pp 97–107
51. Donmez N, Brudno M (2013) SCARPA: scaffolding reads with practical algorithms. Bioinformatics 29(4):428–434
52. Dost B, Shlomi T, Gupta N, Ruppin E, Bafna V, Sharan R (2008) QNet: a tool for querying protein interaction networks. J Comput Biol 15(7):913–925
53. Downey RG, Fellows MR (2013) Fundamentals of parameterized complexity. Texts in computer science. Springer, Berlin
54. Downey RG, Thilikos DM (2011) Confronting intractability via parameters. Comput Sci Rev 5(4):279–317
55. Fafianie S, Bodlaender HL, Nederlof J (2015) Speeding up dynamic programming with representative sets: an experimental evaluation of algorithms for Steiner tree on tree decompositions. Algorithmica 71(3):636–660
56. Fellows MR, Gramm J, Niedermeier R (2006) On the parameterized intractability of motif search problems. Combinatorica 26 (2):141–167
57. Felner A, Korf RE, Hanan S (2004) Additive pattern database heuristics. J Artif Intell Res 21:1–39
58. Fleischer R, Wu X, Yuan L (2009) Experimental study of FPT algorithms for the directed feedback vertex set problem. In: Proceedings of the 17th annual European symposium on algorithms (ESA '09). Lecture notes in computer science, vol 5757. Springer, Berlin, pp 611–622
59. Flum J, Grohe M (2006) Parameterized complexity theory. Springer, Berlin
60. Fomin FV, Kratsch D (2010) Exact exponential algorithms. Texts in theoretical computer science. Springer, Berlin
61. Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of NP-completeness. Freeman, San Francisco, CA
62. Gottlob G, Pichler R, Wei F (2010) Bounded treewidth as a key to tractability of knowledge representation and reasoning. Artif Intell 174 (1):105–132
63. Gramm J (2003) Fixed-parameter algorithms for the consensus analysis of genomic sequences. PhD thesis, WSI für Informatik, Universität Tübingen, Germany
64. Gramm J, Niedermeier R (2003) A fixed-parameter algorithm for minimum quartet inconsistency. J Comput Syst Sci 67 (4):723–741
65. Gramm J, Niedermeier R, Rossmanith P (2003) Fixed-parameter algorithms for closest string and related problems. Algorithmica 37(1):25–42
66. Gramm J, Guo J, Hüffner F, Niedermeier R (2004) Automated generation of search tree algorithms for hard graph modification problems. Algorithmica 39:321–347
67. Gramm J, Guo J, Hüffner F, Niedermeier R (2005) Graph-modeled data clustering: exact algorithms for clique generation. Theory Comput Syst 38(4):373–392
68. Gramm J, Guo J, Niedermeier R (2006) Parameterized intractability of distinguishing substring selection. Theory Comput Syst 39 (4):545–560
69. Gramm J, Guo J, Hüffner F, Niedermeier R (2008) Data reduction and exact algorithms for clique cover. ACM J Exp Algorithmics 13:2.2:1–2.2:15
70. Groër C, Sullivan BD, Weerapurage D (2012) INDDGO: Integrated network decomposition & dynamic programming for graph optimization. Technical Report ORNL/TM-2012/176, Oak Ridge National Laboratory
71. Guillemot S (2011) Parameterized complexity and approximability of the longest compatible sequence problem. Discret Optim 8 (1):50–60
72. Guillemot S, Mnich M (2013) Kernel and fast algorithm for dense triplet inconsistency. Theor Comput Sci 494:134–143
73. Guo J, Niedermeier R (2007) Invitation to data reduction and problem kernelization. ACM SIGACT News 38(1):31–45
74. Guo J, Gramm J, Hüffner F, Niedermeier R, Wernicke S (2006) Compression-based fixed-parameter algorithms for feedback vertex set and edge bipartization. J Comput Syst Sci 72 (8):1386–1396
75. Guo J, Moser H, Niedermeier R (2009) Iterative compression for exactly solving NP-hard minimization problems. In: Algorithmics of large and complex networks. Lecture notes in computer science, vol 5515. Springer, Berlin, pp 65–80
76. Hartung S, Hoos HH (2015) Programming by optimisation meets parameterised algorithmics: A case study for cluster editing. In: Proceedings of the 9th learning and intelligent optimization conference (LION'15). Lecture notes in computer science, vol 8994. Springer, Berlin, pp 43–58
77. Hartung S, Komusiewicz C, Nichterlein A (2015) Parameterized algorithms and

computational experiments for finding 2-clubs. J Graph Algorithms Appl 19(1):155–190

78. Hlinený P, Oum S, Seese D, Gottlob G (2008) Width parameters beyond tree-width and their applications. Comput J 51 (3):326–362
79. Hoos HH (2012) Programming by optimization. Commun ACM 55(2):70–80
80. Hüffner F (2009) Algorithm engineering for optimal graph bipartization. J Graph Algorithms Appl 13(2):77–98
81. Hüffner F, Niedermeier R, Wernicke S (2008) Techniques for practical fixed-parameter algorithms. Comput J 51(1):7–25
82. Hüffner F, Wernicke S, Zichner T (2008) Algorithm engineering for color-coding with applications to signaling pathway detection. Algorithmica 52(2):114–132
83. Hüffner F, Betzler N, Niedermeier R (2010) Separator-based data reduction for signed graph balancing. J Comb Optim 20 (4):335–360
84. Hüffner F, Komusiewicz C, Moser H, Niedermeier R (2010) Fixed-parameter algorithms for cluster vertex deletion. Theory Comput Syst 47(1):196–217
85. Kask K, Dechter R, Larrosa J, Dechter A (2005) Unifying tree decompositions for reasoning in graphical models. Artif Intell 166(1–2):165–193
86. Kleinberg JM, Tardos É (2006) Algorithm design. Addison-Wesley, Reading, MA
87. Kneis J, Mölle D, Richter S, Rossmanith P (2006) Divide-and-color. In: Proceedings of the 32nd international workshop on graph-theoretic concepts in computer science (WG '06). Lecture notes in computer science, vol 4271. Springer, Berlin, pp 58–67
88. Kneis J, Langer A, Rossmanith P (2011) Courcelle's theorem—a game-theoretic approach. Discret Optim 8(4):568–594
89. Kociumaka T, Pilipczuk M (2014) Faster deterministic feedback vertex set. Inf Process Lett 114(10):556–560
90. Komusiewicz C, Niedermeier R (2012) New races in parameterized algorithmics. In: Proceedings of the 37th international symposium on mathematical foundations of computer science (MFCS '12). Lecture notes in computer science, vol 7464. Springer, Berlin, pp 19–30.
91. Komusiewicz C, Sorge M (2015) An algorithmic framework for fixed-cardinality optimization in sparse graphs applied to dense subgraph problems. Discret Appl Math 193:145–161
92. Koutis I (2008) Faster algebraic algorithms for path and packing problems. In: Proceedings of the 35th international colloquium on automata, languages and programming (ICALP '08). Lecture notes in computer science, vol 5125. Springer, Berlin, pp 575–586
93. Koutis I, Williams R (2009) Limits and applications of group algebras for parameterized problems. In: Proceedings of the 36th international colloquium on automata, languages and programming (ICALP '09). Lecture notes in computer science, vol 5555. Springer, Berlin, pp 653–664
94. Kratsch S (2014) Recent developments in kernelization: a survey. Bull EATCS 113:58–97
95. Krithika R, Narayanaswamy NS (2013) Another disjoint compression algorithm for odd cycle transversal. Inf Process Lett 113 (22–24):849–851
96. Langer A, Reidl F, Rossmanith P, Sikdar S (2012) Evaluation of an MSO-solver. In: Proceedings of the 14th workshop on algorithm engineering and experiments (ALENEX '12). SIAM, Philadelphia, PA, pp 55–63
97. Langer A, Reidl F, Rossmanith P, Sikdar S (2014) Practical algorithms for MSO model-checking on tree-decomposable graphs. Comput Sci Rev 13–14:39–74
98. Liberti L, Lavor C, Mucherino A (2013) The discretizable molecular distance geometry problem seems easier on proteins. In: Distance geometry: theory, methods, and applications. Springer, Berlin, pp 47–60
99. Linz S, Semple C, Stadler T (2010) Analyzing and reconstructing reticulation networks under timing constraints. J Math Biol 61 (5):715–737
100. Liu C, Song Y, Yan B, Xu Y, Cai L (2006) Fast de novo peptide sequencing and spectral alignment via tree decomposition. In: Proceedings of the 11th Pacific symposium on biocomputing (PSB '06), pp 255–266
101. Lokshtanov D, Marx D, Saurabh S (2011) Known algorithms on graphs on bounded treewidth are probably optimal. In: Proceedings of the 22nd annual ACM-SIAM symposium on discrete algorithms (SODA '11). SIAM, Philadelphia, PA, pp 777–789
102. Lokshtanov D, Narayanaswamy NS, Raman V, Ramanujan MS, Saurabh S (2014) Faster parameterized algorithms using linear programming. ACM Trans Algorithm 11 (2):15:1–15:31
103. Marx D (2008) Closest substring problems with small distances. SIAM J Comput 38 (4):1382–1410

104. Michalewicz Z, Fogel DB (2004) How to solve it: modern heuristics, 2nd edn. Springer, Berlin
105. Miranda M, Lynce I, Manquinho VM (2014) Inferring phylogenetic trees using pseudo-boolean optimization. AI Commun 27 (3):229–243
106. Moore C, Mertens S (2011) The nature of computation. Oxford University Press, Oxford
107. Moser H, Niedermeier R, Sorge M (2012) Exact combinatorial algorithms and experiments for finding maximum $k$-plexes. J Comb Optim 24(3):347–373
108. Nemhauser GL, Trotter LE (1975) Vertex packings: structural properties and algorithms. Math Program 8(1):232–248
109. Niedermeier R (2006) Invitation to fixed-parameter algorithms. Oxford University Press, Oxford
110. Niedermeier R (2010) Reflections on multivariate algorithmics and problem parameterization. In: Proceedings of the 27th international symposium on theoretical aspects of computer science (STACS '10). Leibniz International Proceedings in Informatics (LIPIcs), vol 5. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Wadern, pp 17–32
111. Niedermeier R, Rossmanith P (2000) A general method to speed up fixed-parameter-tractable algorithms. Inf Process Lett 73:125–129
112. Panconesi A, Sozio M (2004) Fast hare: a fast heuristic for single individual SNP haplotype reconstruction. In: Proceedings of the 4th workshop on algorithms in bioinformatics (WABI '04). Lecture notes in computer science, vol 3240. Springer, Berlin, pp 266–277
113. Papadimitriou CH (1994) Computational complexity. Addison-Wesley, Reading, MA
114. Papadimitriou CH (1997) NP-completeness: a retrospective. In: Proceedings of the 24th international colloquium on automata, languages and programming (ICALP '97). Lecture notes in computer science, vol 1256. Springer, Berlin, pp 2–6
115. Pasupuleti S (2008) Detection of protein complexes in protein interaction networks using $n$-Clubs. In: Proceedings of the 6th European conference on evolutionary computation, machine learning and data mining in bioinformatics (EvoBIO '06). Lecture notes in computer science, vol 4973. Springer, Berlin, pp 153–164
116. Patterson M, Marschall T, Pisanti N, van Iersel L, Stougie L, Klau GW, Schönhuth A (2015) WhatsHap: weighted haplotype assembly for future-generation sequencing reads. J Comput Biol 22(6):498–509
117. Peiselt T (2007) An iterative compression algorithm for vertex cover. Studienarbeit, Institut für Informatik, Friedrich-Schiller-Universität Jena
118. Reed B, Smith K, Vetta A (2004) Finding odd cycle transversals. Oper Res Lett 32 (4):299–301
119. Schäfer A, Komusiewicz C, Moser H, Niedermeier R (2012) Parameterized computational complexity of finding small-diameter subgraphs. Optim Lett 6(5):883–891
120. Scott J, Ideker T, Karp RM, Sharan R (2006) Efficient algorithms for detecting signaling pathways in protein interaction networks. J Comput Biol 13(2):133–144
121. Shlomi T, Segal D, Ruppin E, Sharan R (2006) QPath: a method for querying pathways in a protein–protein interaction network. BMC Bioinf 7:199
122. Skiena SS (2008) The algorithm design manual, 2nd edn. Springer, Berlin
123. Song Y, Liu C, Malmberg RL, Pan F, Cai L (2005) Tree decomposition based fast search of RNA structures including pseudoknots in genomes. In: Proceedings of the 4th international IEEE computer society computational systems bioinformatics conference (CSB 2005). IEEE Computer Society, Washington, DC, pp 223–234
124. Stojanovic N, Florea L, Riemer C, Gumucio D, Slightom J, Goodman M, Miller W, Hardison R (1999) Comparison of five methods for finding conserved sequences in multiple alignments of gene regulatory regions. Nucleic Acids Res 27(19):3899–3910
125. Stolzer M, Lai H, Xu M, Sathaye D, Vernot B, Durand D (2012) Inferring duplications, losses, transfers and incomplete lineage sorting with nonbinary species trees. Bioinformatics 28(18):409–415
126. van Bevern R (2014) Fixed-parameter linear-time algorithms for NP-hard graph and hypergraph problems arising in industrial applications. PhD thesis, TU Berlin
127. Vardi MY (2014) Boolean satisfiability: theory and engineering. Commun ACM 57(3):5
128. Vazirani VV (2001) Approximation algorithms. Springer, Berlin
129. West DB (2000) Introduction to graph theory, 2 edn. Prentice-Hall, Englewood Cliffs, NJ
130. Whidden C, Beiko RG, Zeh N (2016) Fixed-parameter and approximation algorithms for

maximum agreement forests of multifurcating trees. Algorithmica 74(3):1019–1054

131. Williamson DP, Shmoys DB (2011) The design of approximation algorithms. Cambridge University Press, Cambridge
132. Wittkop T, Emig D, Lange S, Rahmann S, Albrecht M, Morris JH, Böcker S, Stoye J, Baumbach J (2010) Partitioning biological data with transitivity clustering. Nat Methods 7(6):419–420
133. Wittkop T, Emig D, Truss A, Albrecht M, Böcker S, Baumbach J (2011) Comprehensive cluster analysis with transitivity clustering. Nat Protoc 6(3):285–295
134. Wu G, You J-H, Lin G (2005) A lookahead branch-and-bound algorithm for the maximum quartet consistency problem. In: Proceedings of the 5th workshop on algorithms in bioinformatics (WABI '05). Lecture notes in computer science, vol 3692. Springer, Berlin, pp 65–76
135. Zhao J, Malmberg RL, Cai L (2008) Rapid ab initio prediction of RNA pseudoknots via graph tree decomposition. J Math Biol 56 (1–2):145–159

# Chapter 21

# Information Visualization for Biological Data

Tobias Czauderna and Falk Schreiber

## Abstract

Visualization is a powerful method to present and explore a large amount of data. It is increasingly important in the life sciences and is used for analyzing different types of biological data, such as structural information, high-throughput data, and biochemical networks. This chapter gives a brief introduction to visualization methods for bioinformatics, presents two commonly used techniques in detail, and discusses a graphical standard for biological networks and cellular processes.

**Key words** Visualization, Data exploration, Heat-maps, Force-based layout, Graph drawing, Systems Biology Graphical Notation

## 1 Introduction

Visualization is the transformation of data, information or knowledge into a visual form such as images and maps. It uses the human ability to take in a large amount of data in a visual form and to detect trends and patterns in pictures easily. Visualization is a helpful method to analyze and explore data or to communicate information; a fact expressed by the common proverb "A picture speaks a thousand words." The visual representation of data or knowledge is not a particularly modern technique, but rather is as old as human society. Rock engravings and cave images can be seen as an early form of visual communication between humans. Molecular biologica information has also been represented visually for a long time. Well-known examples are illustrations in books, such as molecular structures (e.g., DNA and other molecules) or biological processes (e.g., cell cycle, metabolic pathways). Most of the other chapters in this book use visualizations to illustrate concepts or present information.

Nowadays visualization is an increasingly important method in bioinformatics to present very diverse information. Structural information of molecules can be shown in 2D (structural formulae of substances) and 3D space [1–3]. Genome and sequence

Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4_21, 

annotation is often displayed in linear or circular representations with additional annotations [4–6]. Expression and metabolite profiles are high-dimensional data which can be visualized with techniques such as bar-charts, line-graphs, scatter-plot matrices [7], parallel coordinates [8], heat-maps [9], and tree-maps [10]. There are several methods to visualize hierarchical structures (e.g., phylogenetic trees) [11–13] and biochemical networks (e.g., metabolic pathways) [14–16]. Typical examples of visualizations in bioinformatics are shown in Fig. 1, overviews of visualization methods are given, for example, in the "Points of view" series in *Nature Methods* and for omics data in [20].

This chapter presents *heat-maps* and *force-based network layout* in detail and introduces the Systems Biology Graphical Notation, a graphical standard for biological networks and cellular processes.

*Heat-maps* are a standard method to visualize and analyze large-scale data obtained by the high-throughput technologies discussed in previous chapters. These technologies lead to an ever-increasing amount of molecular-biological data, deliver a snapshot of the system under investigation, and allow the comparison of a biological system under different conditions or in different developmental stages. Examples include gene expression data [21], protein data [22], and the quantification of metabolite concentrations [23]. A typical visualization of such data using a heat-map is shown in Fig. 2.

*Force-based network layout* is the main method used to visualize biological networks. Biological networks are important in bioinformatics; *see also* Section VI (Pathways and Networks). Biological processes form complex networks such as metabolic pathways, gene regulatory networks, and protein–protein interaction networks. Furthermore, the data obtained by high-throughput methods and biological networks are closely related. There are two common ways to interpret experimental data: (1) as a biological network and (2) in the context of an underlying biological network. A typical example of the first interpretation is the analysis of interactomics data, for example, data from two-hybrid experiments [25]. The result of these experiments is information as to whether proteins interact pairwise with each other or not. Taking many different protein pairs into account, a protein–-protein interaction network can be directly derived. An example of the second interpretation is the analysis of metabolomics data, such as data from mass spectrometry based metabolome analysis [23]. These experiments give, for example, time series data for different metabolites, which can be mapped onto metabolic networks and then analyzed within the network context. A visualization of a network using force-based layout is shown in Fig. 3. There are many extensions to force-based layout algorithms such as extra forces [26, 27], animation [28], and the consideration of mapped data.

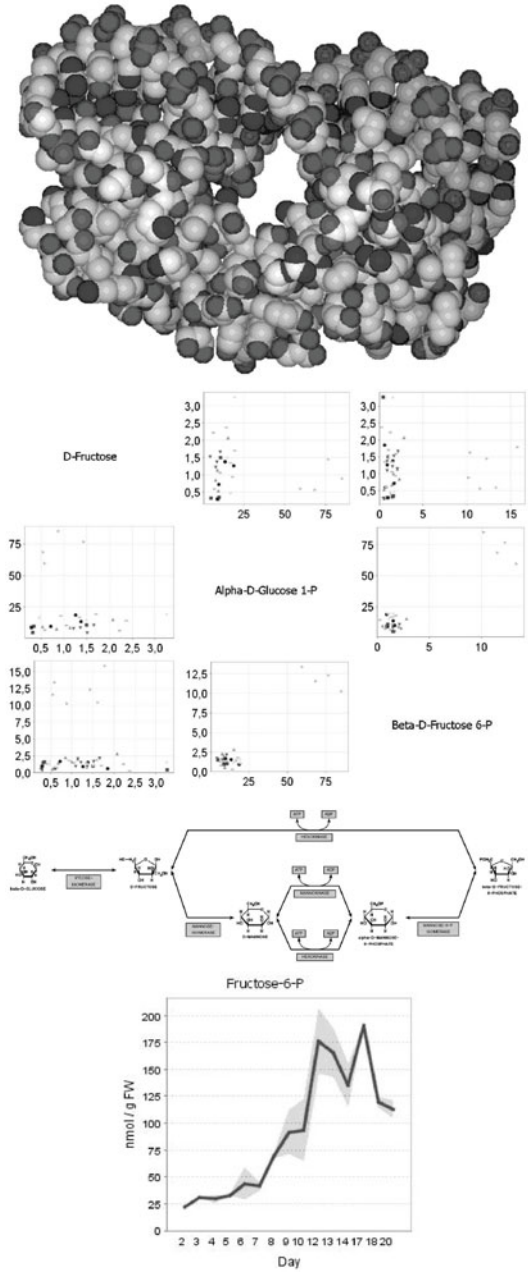



**Fig. 1** Examples of visualizations in bioinformatics (from *top* to *bottom*): 3D structure of a molecule (produced with Molw PDB Viewer [17]), scatter-plot matrix of metabolite profiling data of different lines of an organism (produced with VANTED [18]), layout of a metabolic pathway (produced with BioPath [19]), and line-graph of time series data of the concentration of a metabolite (produced with VANTED [18])

*Systems Biology Graphical Notation (SBGN)* [30] is a standard for the graphical representation of biological networks and cellular processes. It provides an unambiguous and uniform way to present

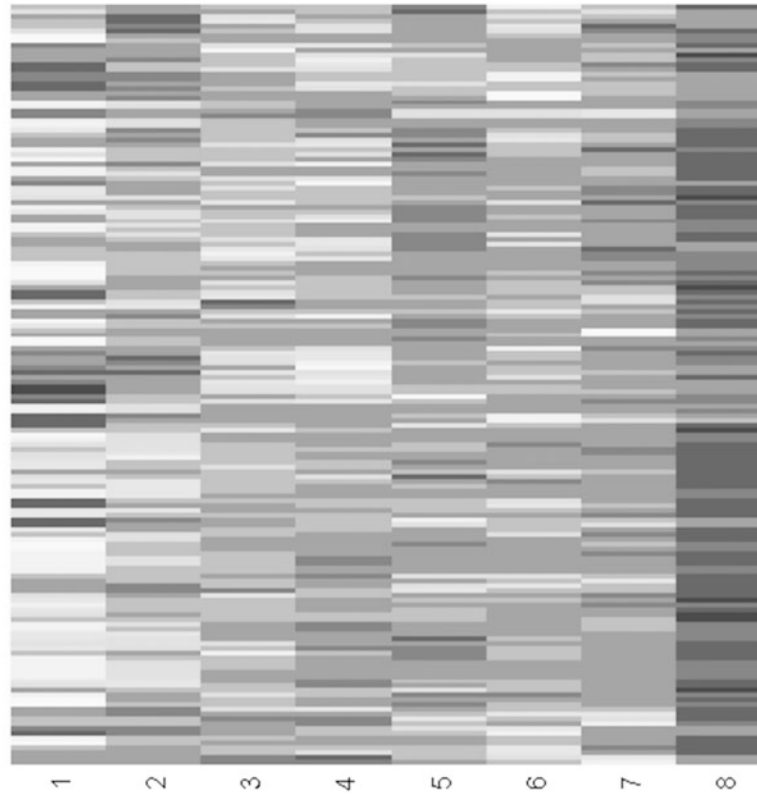



**Fig. 2** A heat-map showing the expression of genes under eight different conditions (produced with R [24])

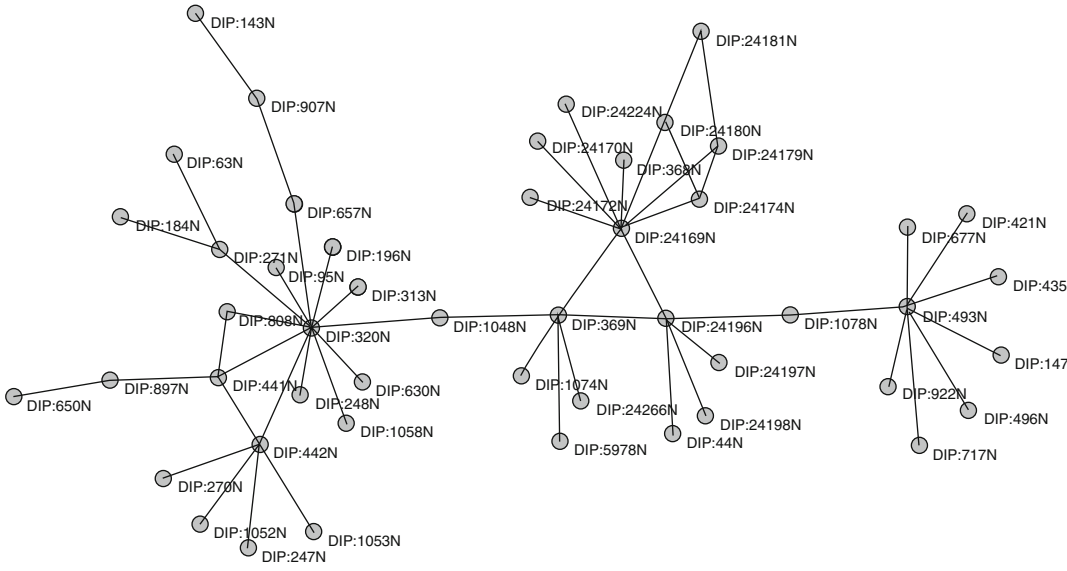



**Fig. 3** A picture of a protein–protein interaction network based on the force-based layout method (produced with CentiBin [29])

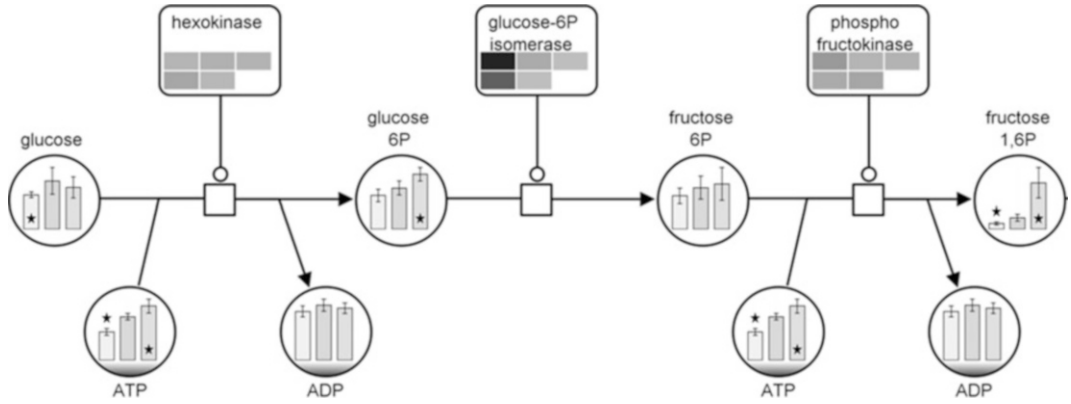



**Fig. 4** A SBGN map showing the first steps of the metabolic pathway glycolysis with additional information. The *circles*, *rectangles*, and *rectangles with rounded corners* represent simple chemicals (metabolites), processes (reactions), and macromolecules (enzymes), respectively. Additional information given as diagrams within simple chemicals and macromolecules represents metabolite measurements and activity of genes related to enzymes (produced with SBGN-ED [40])

information, thereby reducing the risk of misinterpreting maps of biological processes and supporting faster information exchange. SBGN provides three corresponding views of a biological system focusing on different aspects and levels of detail: Process Description maps describe elements and processes of biological systems [31], Entity Relationship maps focus on interactions between biological entities [32], and Activity Flow maps describe information flow between biological activities [33]. Several databases provide information in SBGN, for example, Reactome [34], PANTHER Pathways [35], BioModels Database including Path2-Models [36, 37], MetaCrop [38], and RIMAS [39]. A SBGN map is shown in Fig. 4.

## 2 Methods

### 2.1 Heat-maps

High-throughput data is often represented by a two-dimensional matrix $M$. Usually the rows represent the measured entities (e.g., expression of genes) and the columns represent the different samples (e.g., different time points, environmental conditions or genetically modified lines of an organism). To show patterns in the data it is often useful to rearrange the rows and/or columns of the matrix so that similar rows (columns) are close to each other, for example, to place genes with similar expression patterns close together.

A heat-map is a two-dimensional, colored grid of the same size as the matrix $M$ where the color of each place is determined by the corresponding value of the matrix as shown in Fig. 5.

For a given matrix $M$, the algorithm to produce a heat-map is as follows:

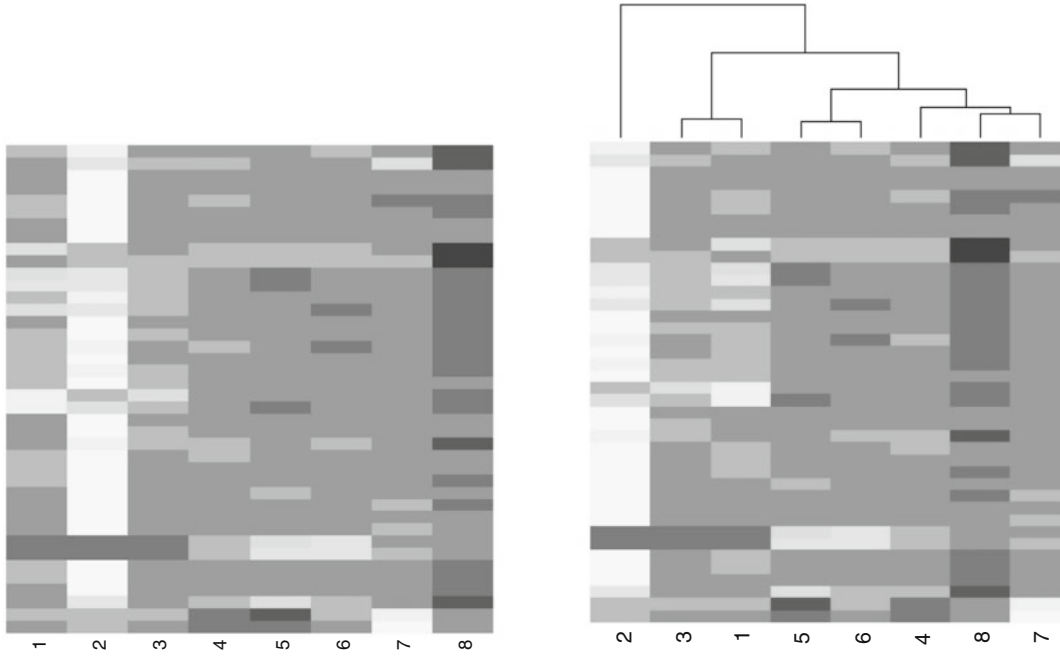



**Fig. 5** (*Left*) A heat-map of the data set in the given order (*x*-axis: conditions, *y*-axis: genes). (*Right*) Rearrangement of columns (conditions) and dendrogram showing a hierarchical clustering of the different conditions (conditions with similarly expressed genes are close together, produced with R [24])

1. (Optional) Rearrange the rows of the matrix as follows: Compute a distance matrix containing the distance between each pair of rows (consider each row as a vector). There are several possible distance measures (e.g., Euclidean distance, Manhattan distance and correlation coefficient). Based on the distance matrix, either rearrange the rows directly such that neighboring rows have only a small distance, or compute a hierarchical clustering (using one of the various methods available, such as complete linkage or single linkage). Rearrange the rows such that a crossing-free drawing of the tree representing the hierarchical clustering is obtained and similar rows are close together. Details of this rearranging step and several variations can be found in Chapter 54 (Combinatorial optimization models for finding genetic signatures from gene expression datasets) and in [7, 41, 42].
2. (Also optional) Rearrange the columns of the matrix similarly.
3. Use a color scheme such that the distances between the colors represent the distances between the values of the elements of the matrix $M$ (*see* **Note 1**). Assign to each matrix element its color and compute a grid visualization and (optional) dendrogram(s) displaying the hierarchical clustering(s) for rows/columns as shown in Fig. 5.

Free software to produce such visualizations is, for example, the R programming package [24].

### 2.2 Force-Based Network Layout

Biological networks are commonly represented as graphs. A graph $G = (V,E)$ consists of a set of vertices $V = \{v_1,\ldots,v_n\}$ representing the biological objects (e.g., proteins) and a set of edges $E \subseteq \{(v_i,v_j) \mid v_i,v_j \in V\}$ representing the interactions between the biological objects (e.g., interactions between proteins). To visualize a graph, a layout has to be computed, that is, coordinates for the vertices and curves for the edges. In the following we present the force-based graph layout approach usually applied to biological networks.

A force-based layout method uses a physical analogy to draw graphs by simulating a system of physical forces defined on the graph. It produces a drawing, which represents a locally minimal energy configuration of the physical system. Such layout methods are popular as they are easy to understand and implement, and give good visualization results. In general, force-based layout methods consist of two parts: (1) a system of forces defined by the vertices and edges, and (2) a method to find positions for the vertices (representing the final layout of the graph) such that for each vertex the total force is zero [43]. There are several frequently used varieties of force-based methods [44–47].

Here we use a force model that interprets vertices as mutually repulsive "particles" and edges as "springs" connecting the particles. This results in attractive forces between adjacent vertices and repulsive forces between nonadjacent vertices. To find a locally minimal energy configuration iterative numerical analysis is used. In the final drawing, the vertices are connected by straight lines.

For a given graph $G = (V,E)$ the algorithm to compute a layout $l(G)$ is as follows (*see also* Fig. 6):

1. Place all vertices on random positions. This gives an initial layout $l_0(G)$ (*see* **Note 2**).
2. Repeat the following steps (**steps 3** and **4**) until a stop criterion (e.g., number of iterations or quality of current layout) is reached.
3. For the current layout $l_i(G)$ compute for each vertex $v \in V$ the force $F(v) = \sum_{(u,v)\in E} f_a(u,v) + \sum_{(u,v)\in V\times V} f_r(u,v)$, which is the sum of all attractive forces $f_a$ and all repulsive forces $f_r$ affecting $v$. For 2D or 3D drawings these force vectors consist of two $(x,y)$ or three $(x,y,z)$ components, respectively. For example, for the $x$ component the forces $f_a$ and $f_r$ are defined as $f_a(u,v) = c_1 * (d(u,v) - l) * \frac{x(v)-x(u)}{d(u,v)}$ and $f_r(u,v) = \frac{c_2}{d(u,v)^2} * \frac{x(v)-x(u)}{d(u,v)}$, respectively, where $l$ is the optimal distance between any pair of adjacent vertices, $d(u,v)$ is the current distance between the vertices $u$ and $v$, $x(u)$ is the $x$-coordinate of vertex $u$, $x(v)$ is the $x$-coordinate of vertex $v$, and $c_1$, $c_2$ are positive constants (*see* **Note 3**). The other components are similarly defined.

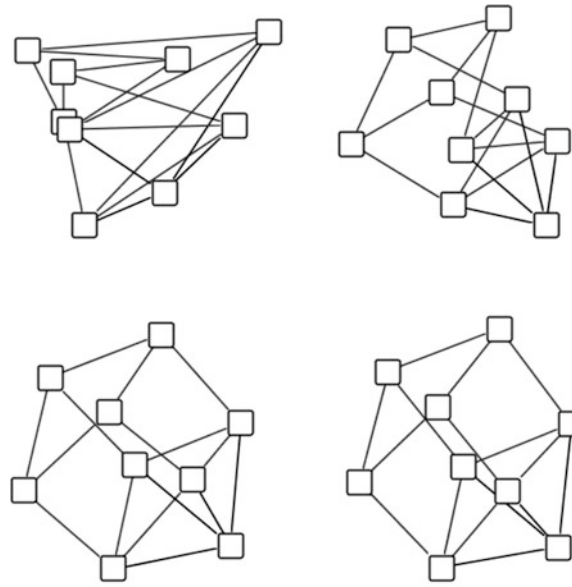


**Fig. 6** Visualization of a graph at different steps of the force-based layout (from *top left clockwise*): initial layout, after 10, 25, and 100 iterations, respectively

4. Move each vertex in the direction of $F(v)$ to produce a new layout $l_{i+1}(G)$ (*see* **Note 4**).

Free software packages to produce such network layouts are, for example, JUNG [48], Gravisto [49], and Vanted [50].

### 2.3 SBGN

Depending on the type of biological information and the level of detail, different SBGN languages are recommended (*see* **Note 5**). Process Description (PD) maps are suitable to represent the transitions of entities from one form or state to another with a high level of detail. Such maps are unambiguous, mechanistic, and sequential. The representation of multistate entities results in a combinatorial explosion leading to large and complex maps. A typical example for a PD map is a metabolic pathway as shown in Fig. 4. Entity Relationship (ER) maps show the relations between entities and the influence of entities upon the behavior of other entities. Such maps are unambiguous, mechanistic, and non-sequential. A typical example for an ER map is a protein interaction network. Activity flow (AF) maps represent the activity flow from one entity to another or within the same entity and provide an abstract view on a biological system where detailed mechanistic information is either not known or omitted. Such maps are ambiguous, conceptual, and sequential. A typical example for an AF map is a signaling pathway.

SBGN defines a number of glyphs for the different entities and how these glyphs can be combined to valid SBGN maps but it does not outline how to embody biological knowledge. Therefore the SBGN bricks [51] have been introduced as a means for the

representation of biological knowledge in SBGN. SBGN bricks are building blocks representing recurring biological patterns in all three SBGN languages, which can be used for quick assembly of SBGN maps. They enable users to draw SBGN maps directly without the need to know all details from the SBGN specifications. An initial set of SBGN bricks in a wiki style format is available at http://sbgnbricks.sourceforge.net covering a number of important biological processes, which can be extended on demand.

Figure 7 shows the assembly of the SBGN map from Fig. 4 using SBGN bricks (without the additional information, *see* **Note 6**). The necessary steps to assemble the SBGN map are as follows:

1. Choose the SBGN bricks "Catalysis—Irreversible reaction with 2 substrates and 2 products" and "Catalysis - Irreversible reaction with 1 substrate and 1 product" and place them on the drawing area.
2. Merge "P1" from the brick on the left with "S1" from the brick in the middle and merge "P1" from the brick in the middle with "S1" from the brick on the right.
3. Change the labels of the three macromolecules to "hexokinase", "glucose-6P isomerase", and "phospho fructokinase" (from left to right).
4. Change the labels of the simple chemicals "S1" and "P1" to "glucose", "glucose 6P", "fructose 6P", and "fructose 1,6P" (from left to right).
5. Change the label of the simple chemicals "S2" to "ATP" and add clone markers to indicate they appear more than once on the map. Change the label of the simple chemicals "P2" to "ADP" and add clone markers to indicate they appear more than once on the map.
6. Adapt the layout of the map.

Freely available software to produce visualizations in SBGN from scratch or by using the SBGN bricks is, for example, SBGN-ED [40] (*see* **Note 7**). A detailed step-by-step description for the creation of visualizations in SBGN enriched by experimental data similar to the SBGN map shown in Fig. 4 can be found in [52].

## 3 Notes

1. Do not use a red-green color scheme as quite a number of people are red-green colorblind and therefore unable to interpret the visualization.
2. The initial positions of the vertices should not be on a line. An alternative to a random placement is to use a given initial

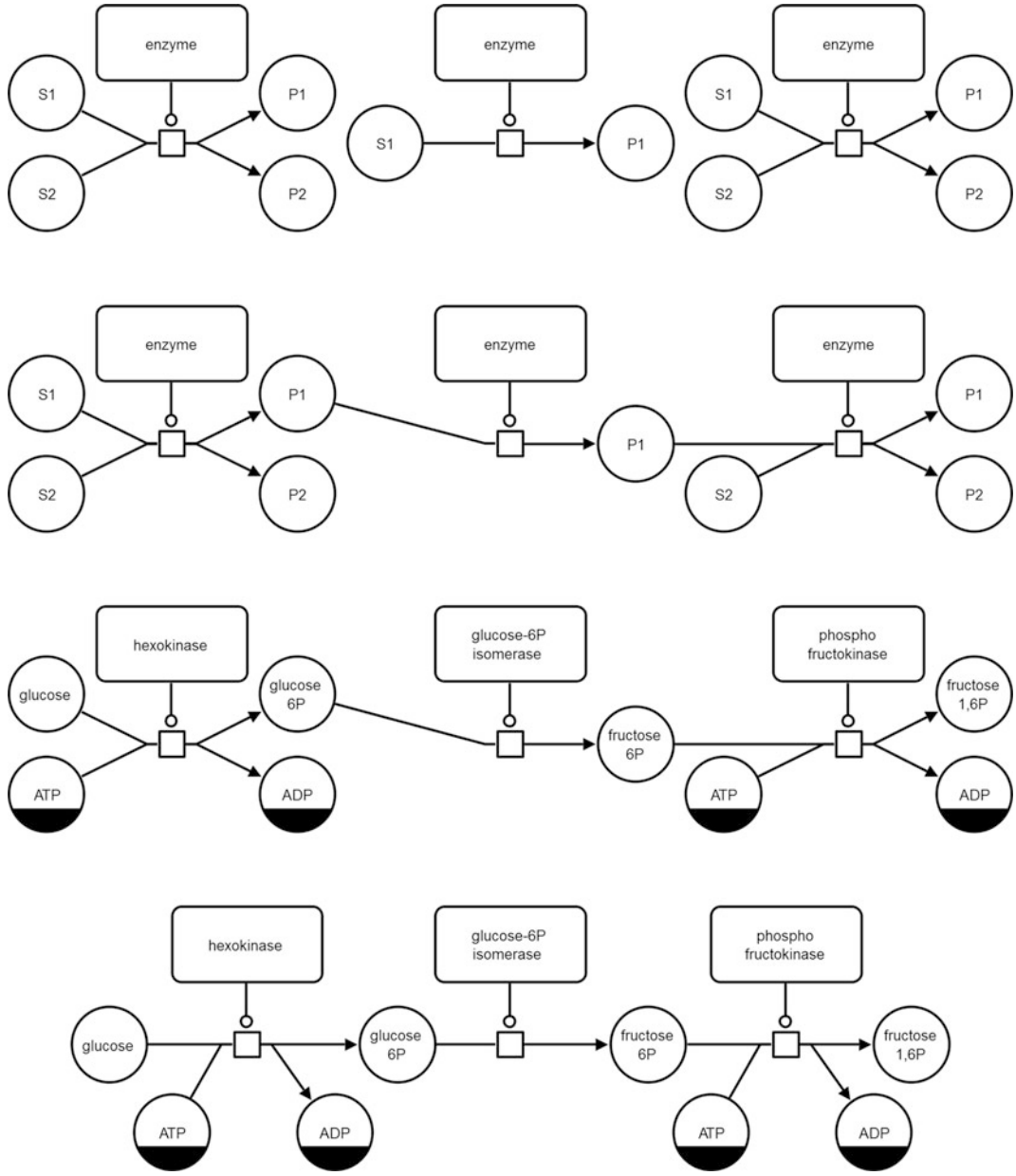



**Fig. 7** Assembly of a SBGN map using SBGN bricks (*see also* Fig. 4). From *top* to *bottom*: SBGN bricks "Catalysis—Irreversible reaction with 2 substrates and 2 products" and "Catalysis—Irreversible reaction with 1 substrate and 1 product" placed on drawing area; "P1" from the brick on the left merged with "S1" from the brick in the middle and "P1" from the brick in the middle merged with "S1" from the brick on the right; labels of macromolecules and simple chemicals changed, clone markers added; layout of the map adapted

layout, which then can be improved by the force-based layout algorithm.

3. The parameters $l$, $c_1$, and $c_2$ greatly affect the final drawing. A good way to find appropriate values for a specific graph is to

interactively change them during the run of the algorithm until appropriate interim results are obtained.

4. It is possible to dampen the force $F(v)$ with an increasing number of iterations to allow large movements of vertices in the beginning and only small movements close to the end of the algorithm. This can help to avoid "oscillation effects" where vertices repeatedly jump between two positions.
5. The SBGN specifications [31–33] provide detailed descriptions of any element of SBGN as well as layout rules. However, to start with SBGN and represent simple information such as metabolic or regulatory pathways, very few symbols are necessary.
6. In SBGN maps, color does not have any meaning, and therefore, color can be used to express user-specific information.
7. SBGN maps can be produced on paper or with tools. Tools such as SBGN-ED [40] support the creation of SBGN maps by validation mechanisms.

## References


1. Tsai CS (2003) Molecular graphics: visualization of biomolecules. In: Introduction to computational biochemistry. John Wiley & Sons, Inc., pp 53–71
2. Can T, Wang Y, Wang YF, Su J (2003) FPV: fast protein visualization using Java3D. Bioinformatics 19(8):913–922
3. Stivala A, Wybrow M, Wirth A, Whisstock JC, Stuckey PJ (2011) Automatic generation of protein structure cartoons with Pro-origami. Bioinformatics 27(23):3315–3316
4. Helt GA, Lewis S, Loraine AE, Rubin GM (1998) BioViews: Java-based tools for genomic data visualization. Genome Res 8(3):291–305
5. Kerkhoven R, van Enckevort FHJ, Boekhorst J, Molenaar D, Siezen RJ (2004) Visualization for genomics: the microbial genome viewer. Bioinformatics 20(11):1812–1814
6. Naquin D, d'Aubenton-Carafa Y, Thermes C, Silvain M (2014) CIRCUS: a package for Circos display of structural genome variations from paired-end and mate-pair sequencing data. BMC Bioinformatics 15:198
7. Andrews DF (1972) Plots of high-dimensional data. Biometrics 29:125–136
8. Inselberg A, Dimsdale B (1990) Parallel coordinates: a tool for visualizing multi-dimensional geometry. In: Proc. Visualization, pp 361–370
9. Eisen MB, Spellman PT, Brown PO, Botstein D (1998) Cluster analysis and display of genome-wide expression patterns. Proc Natl Acad Sci U S A 95(25):14863–14868
10. Baehrecke EH, Dang N, Babaria K, Shneiderman B (2004) Visualization and analysis of microarray and gene ontology data with treemaps. BMC Bioinformatics 5(1):84
11. Hughes T, Hyun Y, Liberles D (2004) Visualising very large phylogenetic trees in three dimensional hyperbolic space. BMC Bioinformatics 5(1):48
12. Rost U, Bornberg-Bauer E (2002) TreeWiz: interactive exploration of huge trees. Bioinformatics 18(1):109–114
13. Huson D, Richter D, Rausch C, Dezulian T, Franz M, Rupp R (2007) Dendroscope: an interactive viewer for large phylogenetic trees. BMC Bioinformatics 8(1):460
14. Schreiber F (2002) High quality visualization of biochemical pathways in BioPath. In Silico Biol 2(2):59–73
15. Sirava M, Schäfer T, Eiglsperger M, Kaufmann M, Kohlbacher O, Bornberg-Bauer E, Lenhof HP (2002) BioMiner—modeling, analyzing, and visualizing biochemical pathways and networks. Bioinformatics 18(suppl 2):S219–S230
16. Kerren A, Schreiber F (2014) Network visualization for integrative bioinformatics. In: Hofestädt R, Chen M (eds) Approaches in Integrative Bioinformatics: towards the virtual cell. Springer, New York, pp 173–202
17. Molw PDB Viewer 4.0
18. Junker BH, Klukas C, Schreiber F (2006) VANTED: a system for advanced data analysis and visualization in the context of biological networks. BMC Bioinformatics 7:109

19. Forster M, Pick A, Raitner M, Schreiber F, Brandenburg FJ (2002) The system architecture of the BioPath system. In Silico Biol 2 (3):415–426
20. Gehlenborg N, O'Donoghue SI, Baliga NS, Goesmann A, Hibbs MA et al (2010) Visualization of omics data for systems biology. Nat Methods 7:S56–S68
21. Duggan D, Bittner B, Chen Y, Meltzer P, Trent J (1999) Expression profiling using cDNA microarrays. Nat Genet 21(Suppl 1):11–19
22. MacBeath G (2002) Protein microarrays and proteomics. Nat Genet 32(Suppl 1):526–532
23. Villas-Boas SG, Mas S, Akesson M, Smedsgaard J, Nielsen J (2005) Mass spectrometry in metabolome analysis. Mass Spectrom Rev 24 (5):613–646
24. R. http://www.r-project.org/
25. Ito T, Chiba T, Ozawa R, Yoshida M, Hattori M, Sakaki Y (2001) A comprehensive two-hybrid analysis to explore the yeast protein interactome. Proc Natl Acad Sci U S A 98:4569–4574
26. Genc B, Dogrusöz U (2003) A constrained, force-directed layout algorithm for biological pathways. In: Liotta G (ed) Proc. International Symposium on Graph Drawing (GD'03), vol 2912. Springer LNCS, Heidelberg, pp 314–319
27. Klukas C, Schreiber F, Schwöbbermeyer H (2006) Coordinated perspectives and enhanced force-directed layout for the analysis of network motifs. In: Misue K, Sugiyama K, Tanaka J (eds) Proc. Asia-Pacific symposium on information visualization (APVis'06), CRPIT 60, pp 39–48
28. Friedrich C, Schreiber F (2003) Visualization and navigation methods for typed protein-protein interaction networks. Appl Bioinformatics 2(3):19–24
29. Junker BH, Koschützki D, Schreiber F (2006) Exploration of biological network centralities with CentiBiN. BMC Bioinformatics 7:219
30. Le Novère N, Hucka M, Mi HY, Moodie S, Schreiber F, Sorokin A et al (2009) The systems biology graphical notation. Nat Biotechnol 27:735–741
31. Moodie S, Le Novère N, Sorokin A, Mi H, Schreiber F (2009) The Systems Biology Graphical Notation: process description language level 1. Nature Precedings 2009.3721
32. Le Novère N, Moodie S, Sorokin A, Schreiber F, Mi H (2009) Systems biology graphical notation: entity relationship language level 1. Nature Precedings 2009.3719
33. Huaiyu M, Schreiber F, Le Novère N, Moodie S, Sorokin A (2009) Systems biology graphical notation: activity flow language level 1. Nature Precedings 2009.3724
34. Croft D, Mundo AF, Haw R, Milacic M, Weiser J, Wu G et al (2014) The Reactome pathway knowledgebase. Nucleic Acids Res 42 (1):D472–D477
35. Mi H, Muruganujan A, Thomas PD (2013) PANTHER in 2013: modeling the evolution of gene function, and other gene attributes, in the context of phylogenetic trees. Nucleic Acids Res 41(1):D377–D386
36. Chelliah V, Laibe C, Le Novère N (2013) BioModels database: a repository of mathematical models of biological processes. Methods Mol Biol 1021:189–199
37. Büchel F, Rodriguez N, Swainston N, Wrzodek C, Czauderna T et al (2013) Large-scale generation of computational models from biochemical pathway maps. BMC Syst Biol 7:116
38. Schreiber F, Colmsee C, Czauderna T, Grafahrend-Belau E, Hartmann A, Junker A, Junker BH, Klapperstück M, Scholz U, Weise S (2012) MetaCrop 2.0: managing and exploring information about crop plant metabolism. Nucleic Acids Res 40(1):D1173–D1177
39. Junker A, Hartmann A, Schreiber F, Bäumlein H (2010) An engineer's view on regulation of seed development. Trends Plant Sci 15 (6):303–307
40. Czauderna T, Klukas C, Schreiber F (2010) Editing, validating, and translating of SBGN maps. Bioinformatics 26(18):2340–2341
41. Bar-Joseph Z, Gifford DK, Jaakkola TS (2001) Fast optimal leaf ordering for hierarchical clustering. Bioinformatics 17(Suppl 1): S22–S29
42. Biedl T, Brejová B, Demaine ED, Hamel AM, Vinař T (2001) Optimal arrangement of leaves in the tree representing hierarchical clustering of gene expression data. Technical report 2001–14, Dept. of Computer Science, University of Waterloo
43. Di Battista G, Eades P, Tamassia R, Tollis IG (1999) Graph drawing. Prentice-Hall, Upper Saddle River, NJ
44. Eades P (1984) A heuristic for graph drawing. Congr Numer 42:149–160
45. Fruchterman T, Reingold E (1991) Graph drawing by force-directed placement. Software Pract Exper 21:1129–1164
46. Kamada T, Kawai S (1989) An algorithm for drawing general undirected graphs. Inf Process Lett 31:7–15
47. Sugiyama K, Misue K (1995) Graph drawing by magnetic spring model. J Vis Lang Comput 6(3):217–231

48. Madadhain J, Fisher D, Smyth P, White S, Boey YB (2005) Analysis and visualization of network data using JUNG. J Stat Softw 10:1–35
49. Bachmaier C, Brandenburg FJ, Forster M, Holleis P, Raitner M (2005) Gravisto: graph visualization toolkit. In: Pach J (ed) Proc. International Symposium on Graph Drawing (GD'04), vol 3383. Springer LNCS, Heidelberg, pp 502–503
50. Rohn H, Junker A, Hartmann A, Grafahrend-Belau E, Treutler H, Klapperstück M, Czauderna T, Klukas C, Schreiber F (2012) VANTED v2: a framework for systems biology applications. BMC Syst Biol 6(1):139
51. Junker A, Sorokin A, Czauderna T, Schreiber F, Mazein A (2012) Wiring diagrams in biology: towards the standardized representation of biological information. Trends Biotechnol 30(11):555–557
52. Junker A, Rohn H, Czauderna T, Klukas C, Hartmann A, Schreiber F (2012) Creating interactive, web-based and data-enriched maps with the Systems Biology Graphical Notation. Nat Protoc 7:579–593

# INDEX

## A



## B




Jonathan M. Keith (ed.), *Bioinformatics: Volume II: Structure, Function, and Applications*, Methods in Molecular Biology, vol. 1526, DOI 10.1007/978-1-4939-6613-4, © Springer Science+Business Media New York 2017

## C

## G



## H

## I



## J



## K



## L

## M

## N



## O



## P

## Q



## R

## S

## T



## U



## V



## W



## Y



## Z