Machine Learning for Quantum Mechanics

Matthias Rupp

Fritz Haber Institute of the Max Planck Society, Berlin, Germany

Hands-on Workshop on Density Functional Theory and Beyond
July 31–August 11 2017, Humboldt University, Berlin, Germany

Outline

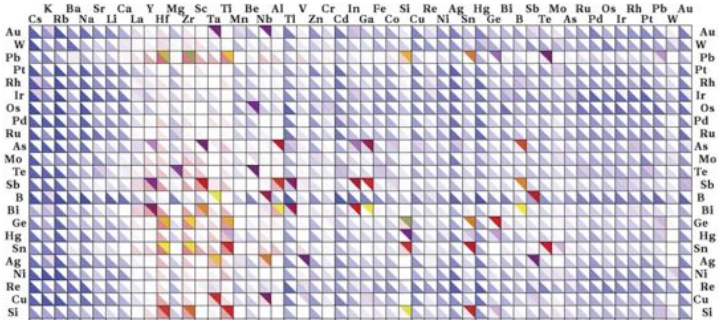
1. Rationale
   quantum mechanics, machine learning
2. Kernel learning
   kernel trick, kernel ridge regression
3. Model building
   overfitting, validation, free parameters
4. Property prediction
   representation, energies of molecules and crystals

Hands-on session
property prediction with `qmmlpack`

Rationale

Challenges in quantum mechanical simulations

High-throughput **screening**



Castelli et al, *Energy Environ Sci* 12, 2013

Large systems

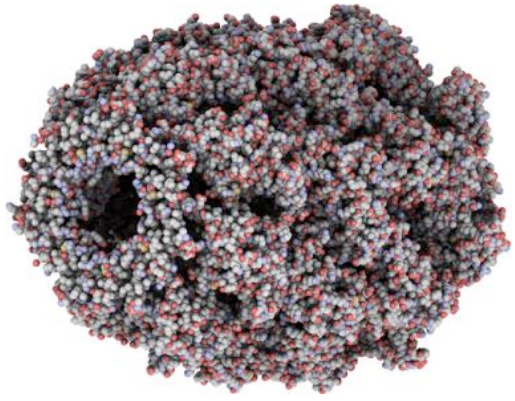
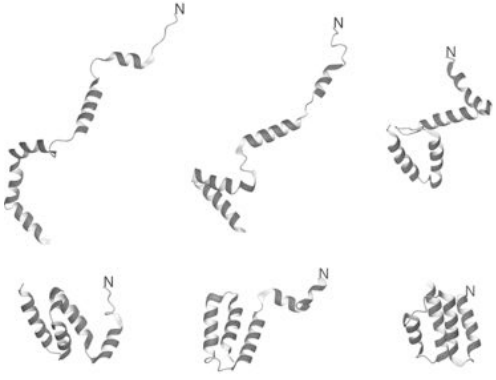


Image: Tarini et al, *IEEE Trans Visual Comput Graph* 2006

Long simulations



Liwo et al, *Proc Natl Acad Sci USA* 102: 2362, 2005

Quantum effects

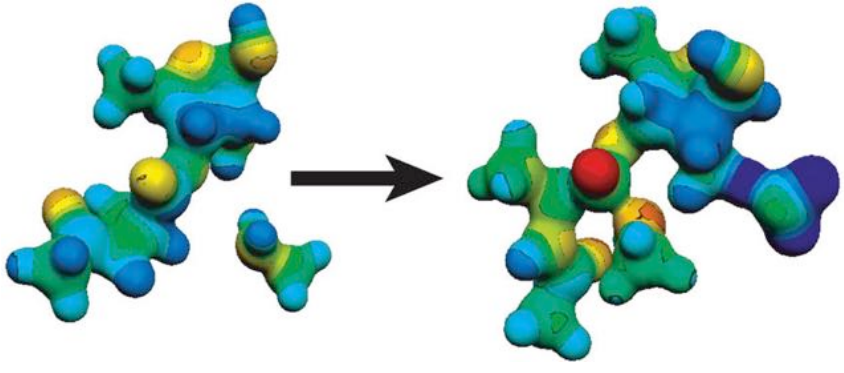


Image: Hiller et al, *Nature* 476: 236, 2011

Approximations

Hierarchy of numerical approximations to Schrödinger's equation:

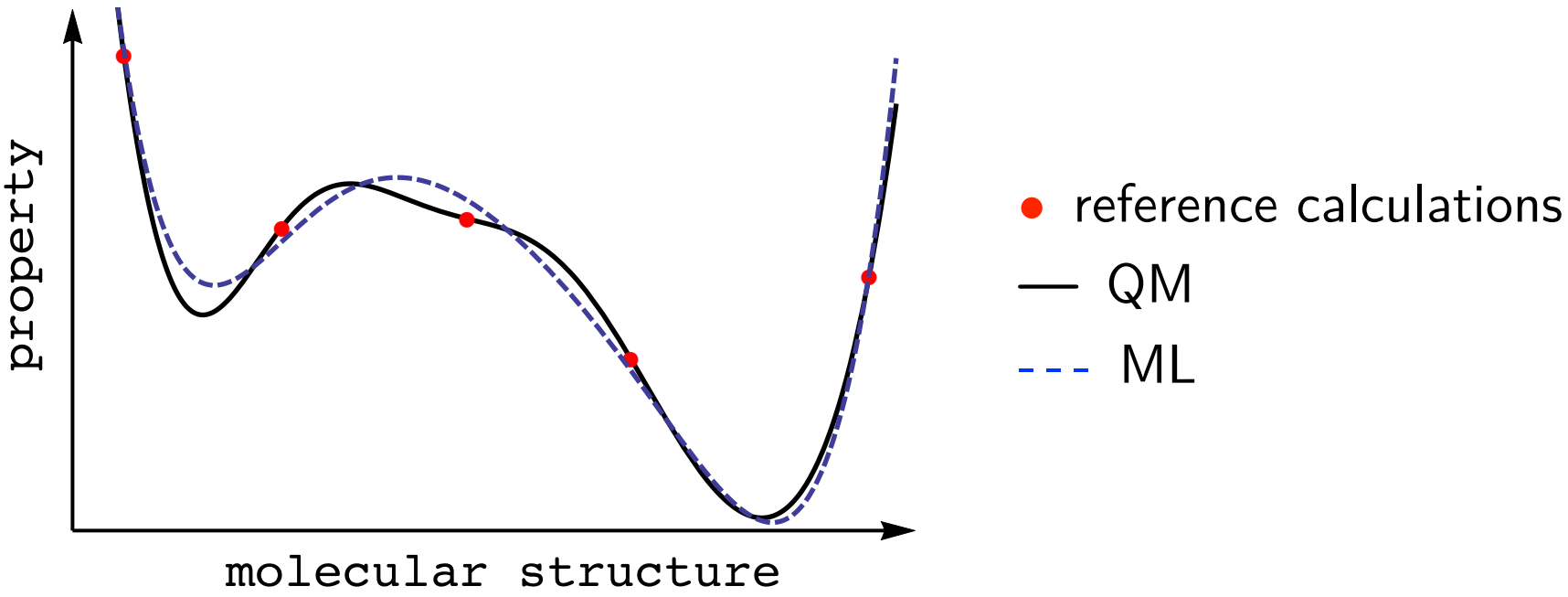
Abrv.	Method	Runtime
FCI	Full Configuration Interaction (CISDTQ)	$O(N^{10})$
CC	Coupled Cluster (CCSD(T))	$O(N^7)$
FCI	Full Configuration Interaction (CISD)	$O(N^6)$
MP2	Møller-Plesset second order perturbation theory	$O(N^5)$
HF	Hartree-Fock	$O(N^4)$
DFT	Density Functional Theory (Kohn-Sham)	$O(N^{3-4})$
TB	Tight Binding	$O(N^3)$
MM	Molecular Mechanics	$O(N^2)$

$N =$ system size

Is it possible to be both accurate and fast?

The key idea

- Exploit redundancy in related QM calculations
- **Interpolate** between QM calculations using ML
- Smoothness assumption (regularization)



Relationship to other models

quantum chemistry	force fields	machine learning
generally applicable	limited domain	generally applicable
no or little fitting	fitting to one class	refitted to any dataset
form from physics	form from physics	form from statistics
deductive	mostly deductive	inductive
few or no parameters	some parameters	many parameters
slow	fast	in between
small systems	large systems	large systems

Machine learning

Machine learning (ML) studies algorithms whose performance **improves with data** ("learning from experience"). Mitchell, McGraw Hill, 1997

Data $\boldsymbol{X}$ $\rightarrow$  $\rightarrow$ Model $\hat{f}$

- widely applied, many problem types and algorithms
- systematic identification of regularity in data for prediction & analysis
- interpolation in high-dimensional spaces
- inductive, data-driven; empirical in a principled way
- connections to statistics, mathematics, computer science, physics, ...
  example: information theory

Problem types

Unsupervised learning: Data do not have labels
Given $\{x_i\}_{i=1}^n$, find structure

- dimensionality reduction

Burges, now Publishers, 2010

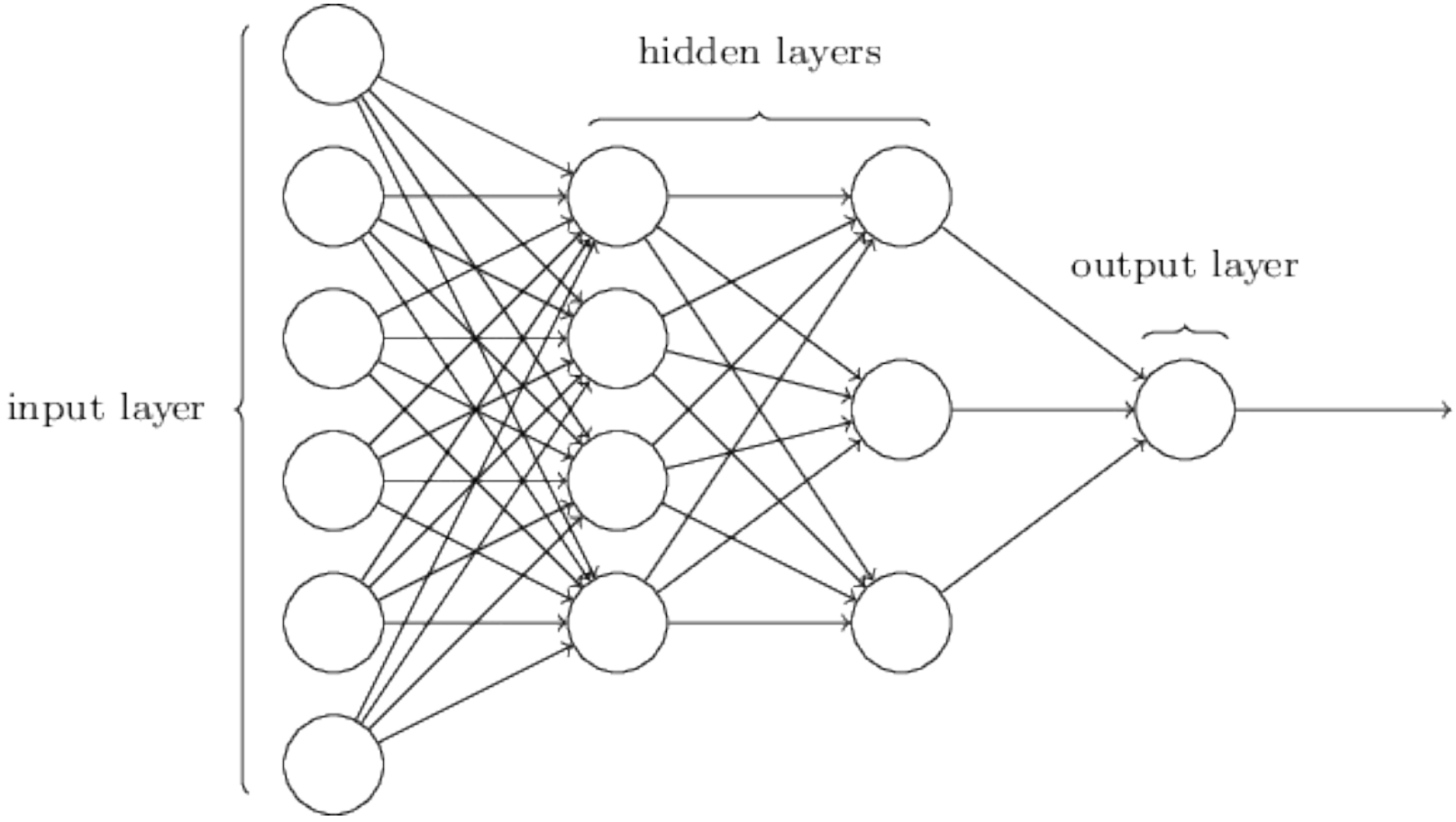
Supervised learning: Data have labels
Given $\{(x_i, y_i)\}_{i=1}^n$, predict $\tilde{y}$ for new $\tilde{x}$

- novelty detection
- classification
- regression
- structured output learning

Semi-supervised learning: Some data have labels
Given $\{(x_i, y_i)\}_{i=1}^n$ and $\{x_i\}_{i=1}^m$, $m \gg n$, predict $\tilde{y}$ for new $\tilde{x}$

Active learning: Algorithm chooses data to label
Choose n data $\{x_i\}_{i=1}^n$ to predict $\tilde{y}$ for new $\tilde{x}$

Artificial neural networks



$$f(x_{i,j}) = h\Big(\sum_{k=1}^{n_i} w_{i-1,k} f(x_{i-1,k})\Big)$$

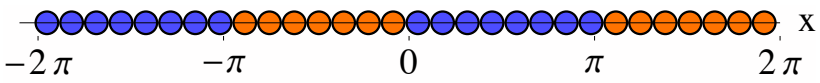
- parametric model
- universal function approximator
- training via non-convex optimization

Kernel learning

The kernel trick

Idea:

- **Transform** samples into higher-dimensional space
- **Implicitly** compute inner products there
- Rewrite linear algorithm to use only inner products



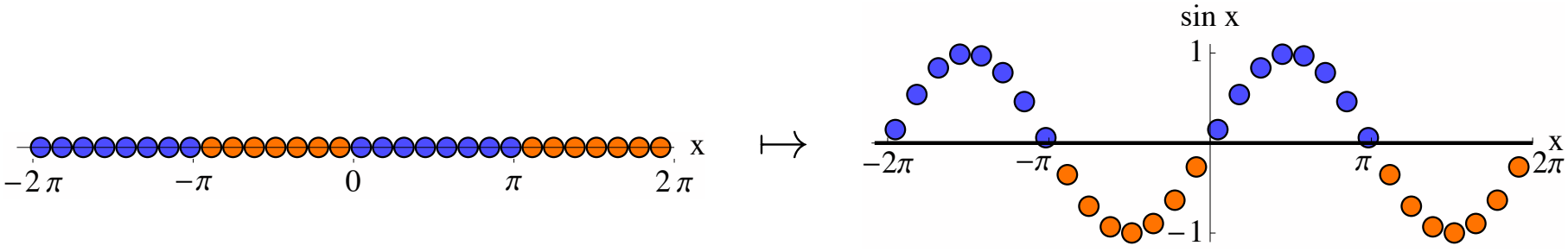
Input space $\mathcal{X}$

Schölkopf, Smola: Learning with Kernels, 2002; Hofmann et al.: *Ann. Stat.* 36, 1171, 2008.

The kernel trick

Idea:

- **Transform** samples into higher-dimensional space
- **Implicitly** compute inner products there
- Rewrite linear algorithm to use only inner products



Input space $\mathcal{X}$ $\xrightarrow{\phi}$ Feature space $\mathcal{H}$

$$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}, \qquad k(x, z) = \langle \phi(x), \phi(z) \rangle$$

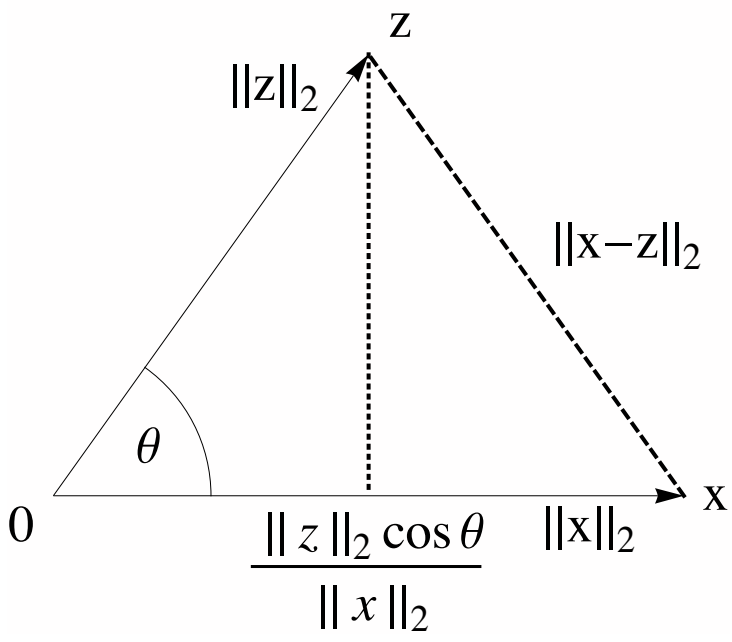
Schölkopf, Smola: Learning with Kernels, 2002; Hofmann et al.: *Ann. Stat.* 36, 1171, 2008.

Kernel functions

Kernels correspond to **inner products**.

If $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is symmetric positive semi-definite,
then $k(x, z) = \langle \phi(x), \phi(z) \rangle$ for some $\phi : \mathcal{X} \to \mathcal{H}$.

Inner products encode information about lengths and angles:
$||x - z||^2 = \langle x, x \rangle - 2 \langle x, z \rangle + \langle z, z \rangle$, $\quad \cos\theta = \frac{\langle x,z \rangle}{||x||\ ||z||}$.



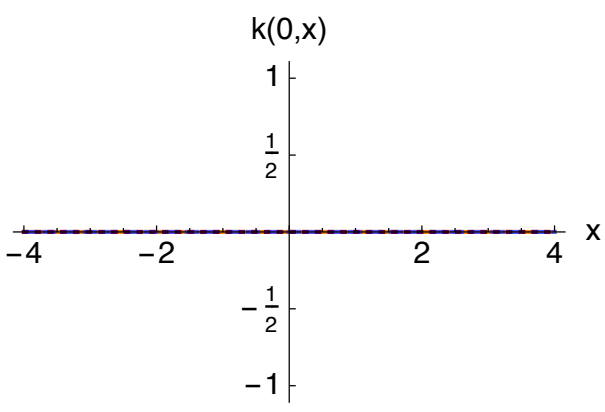
- well characterized function class
- closure properties
- access data only by $\boldsymbol{K}_{ij} = k(x_i, x_j)$
- $\mathcal{X}$ can be any non-empty set

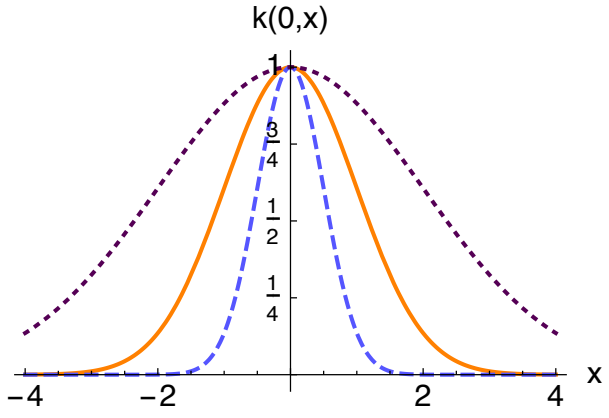
Examples of kernel functions

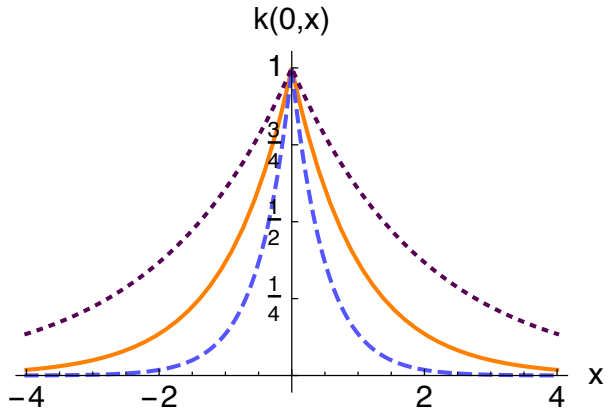
Linear kernel
$k(\boldsymbol{x}, \boldsymbol{z}) = \langle \boldsymbol{x}, \boldsymbol{z} \rangle$

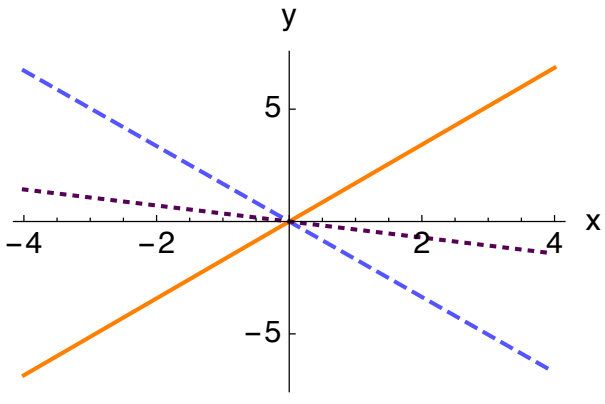
Gaussian kernel
$\exp(-\|\boldsymbol{x} - \boldsymbol{z}\|^2 / 2\sigma^2)$

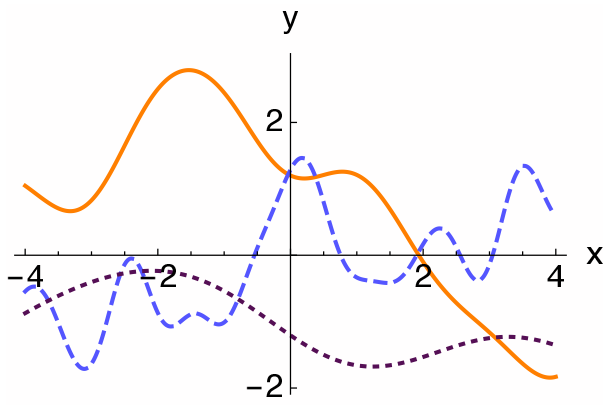
Laplacian kernel
$\exp(-\|\boldsymbol{x} - \boldsymbol{z}\|_1 / \sigma)$

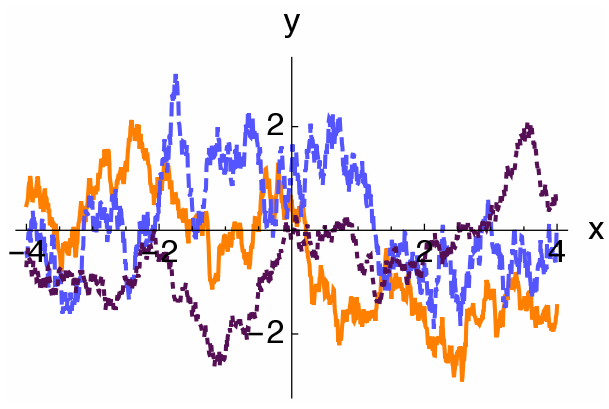












Comparison of linear and kernel ridge regression

Ridge regression

Minimizing

$$\min_{\beta \in \mathbb{R}^d} \sum_{i=1}^{n} \left(f(\boldsymbol{x_i}) - y_i\right)^2 + \lambda ||\beta||^2$$

yields

$$\beta = \left(\boldsymbol{X}^T \boldsymbol{X} + \lambda \boldsymbol{I}\right)^{-1} \boldsymbol{X}^T \boldsymbol{y}$$

for models

$$f(\boldsymbol{x}) = \sum_{i=1}^{d} \beta_i \boldsymbol{x}_i$$

Kernel ridge regression

Minimizing

$$\min_{\alpha \in \mathbb{R}^n} \sum_{i=1}^{n} \left(f(\boldsymbol{x_i}) - y_i\right)^2 + \lambda ||f||_{\mathcal{H}}^2$$

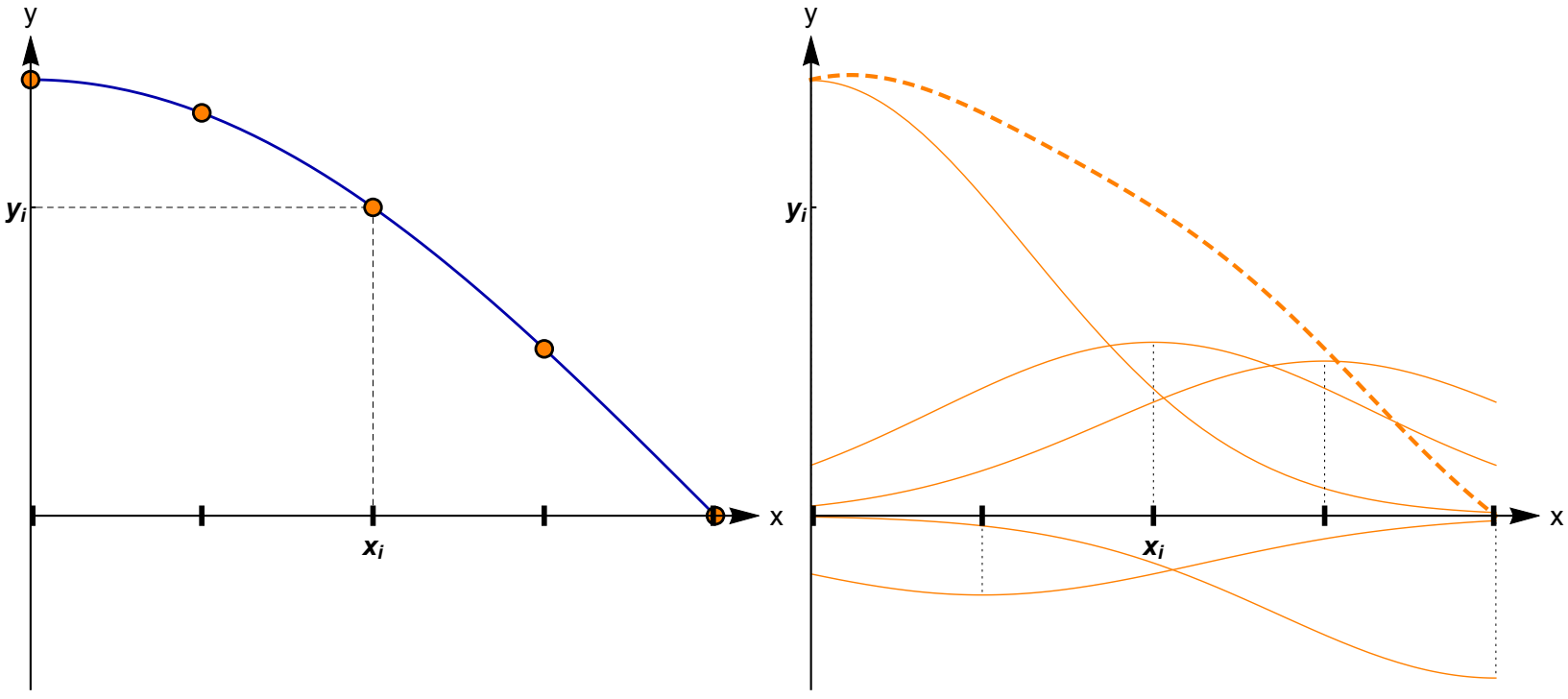
yields

$$\alpha = \left(\boldsymbol{K} + \lambda \boldsymbol{I}\right)^{-1} \boldsymbol{y}.$$

for models

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i k(\boldsymbol{x_i}, \boldsymbol{x})$$

The basis function picture



— learned $y = \cos(x)$

• training samples (x_i, y_i)

— basis functions

- - - prediction f

Vu et al, *Int J Quant Chem* 115: 1115, 2015

Representer theorem

Kernel models have form

$$f(\boldsymbol{z}) = \sum_{i=1}^{n} \alpha_i k(\boldsymbol{x_i}, \boldsymbol{z})$$

due to the **representer theorem**:

Any function minimizing a regularized risk functional

$$\ell\Big((\boldsymbol{x_i}, y_i, f(\boldsymbol{x_i}))_{i=1}^{n}\Big) + g(\|f\|)$$

admits to above representation.

Intuition:

- model lives in space spanned by training data
- weighted sum of basis functions

Schölkopf, Herbrich & Smola, COLT 2001

Centering in kernel feature space

Centering $\boldsymbol{X}$ and $\boldsymbol{y}$ is equivalent to having a bias term b.

For kernel models, center in kernel feature space:

$$\tilde{k}(\boldsymbol{x}, \boldsymbol{z}) = \left\langle \phi(\boldsymbol{x}) - \frac{1}{n}\sum_{i=1}^{n}\phi(\boldsymbol{x_i}), \phi(\boldsymbol{z}) - \frac{1}{n}\sum_{i=1}^{n}\phi(\boldsymbol{x_i}) \right\rangle$$

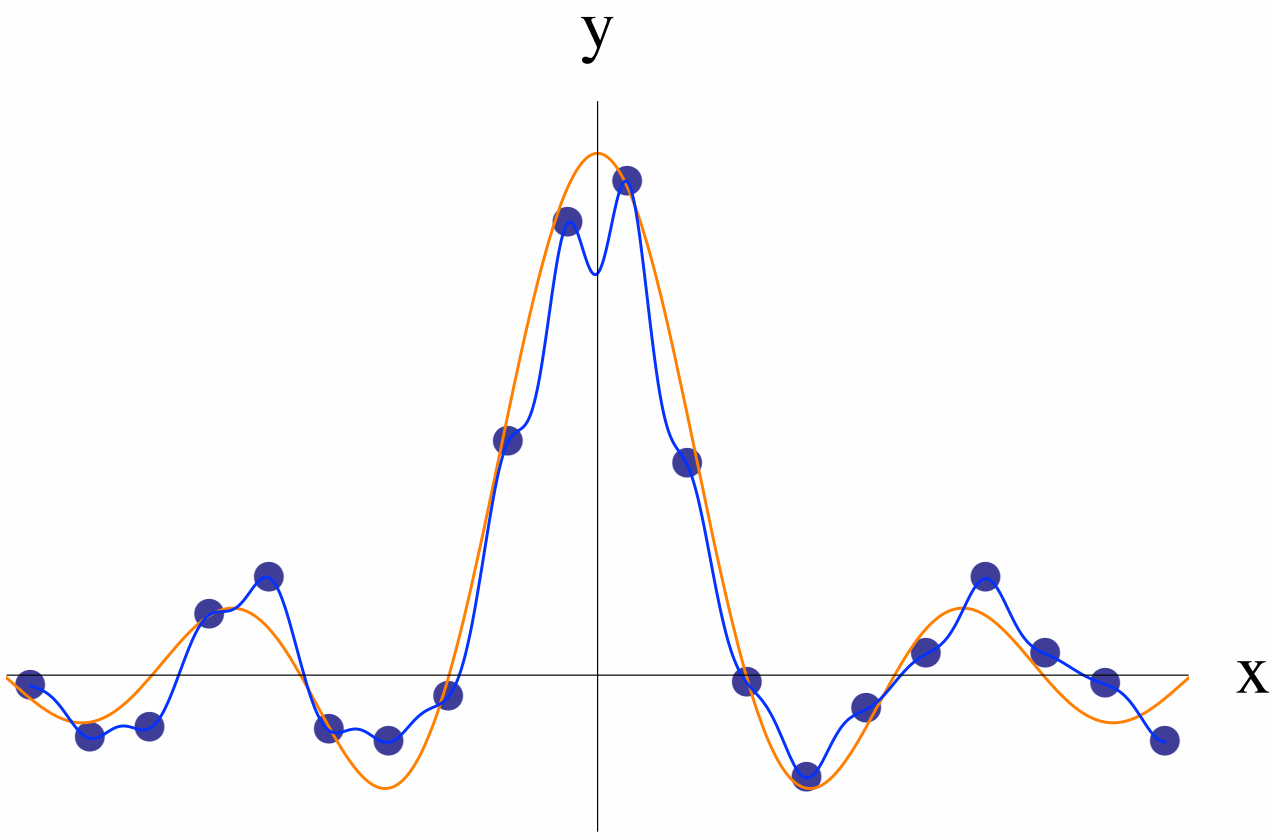
$$\Rightarrow \tilde{\boldsymbol{K}} = \left(\boldsymbol{I} - \frac{1}{n}\mathbf{1}\right)\boldsymbol{K}\left(\boldsymbol{I} - \frac{1}{n}\mathbf{1}\right)$$

Some kernels like Gaussian and Laplacian kernels do not need centering

Poggio *et al.*, Tech. Rep., 2001

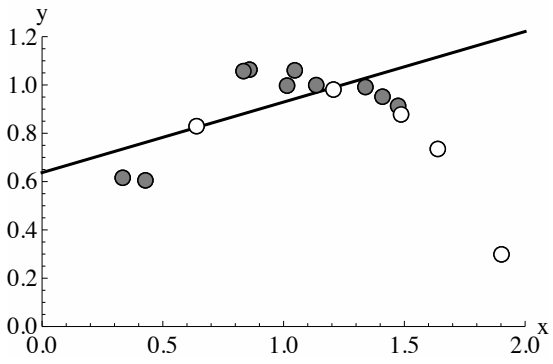
Model building

How regularization helps against overfitting

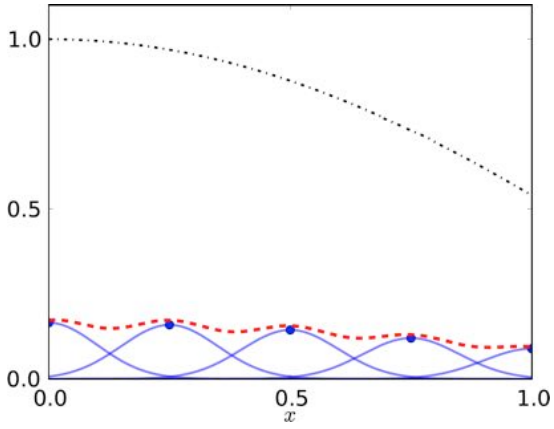


Effect of regularization

Underfitting

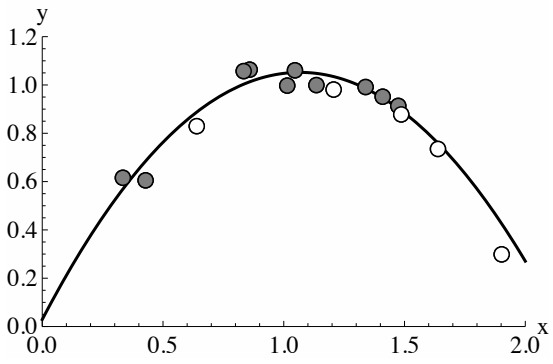


0.123 / 0.443

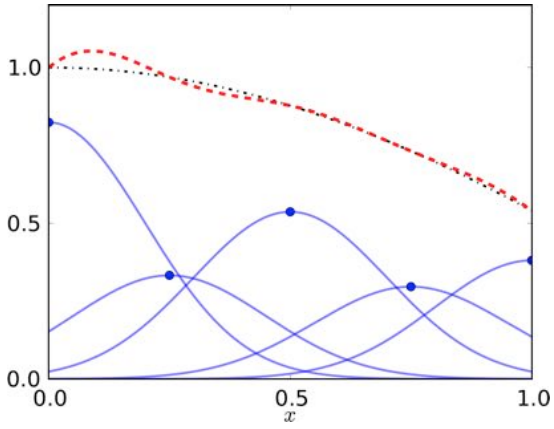


λ too large

Fitting

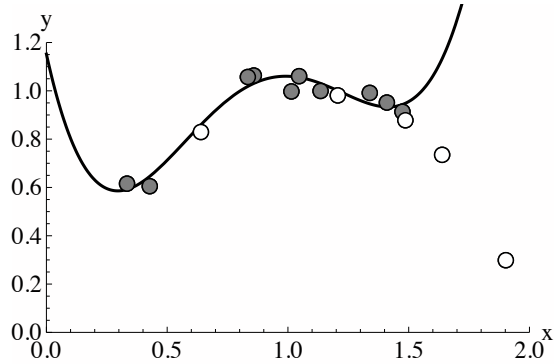


0.044 / 0.068

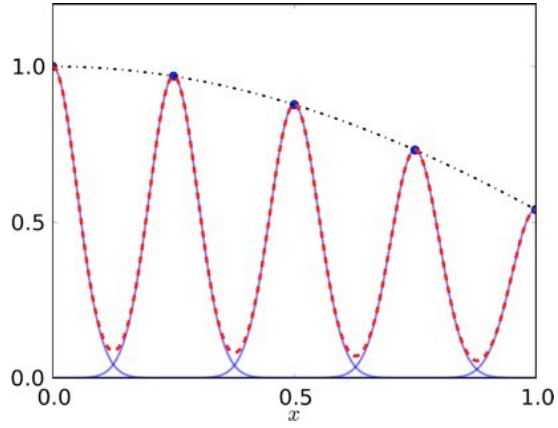


λ right

Overfitting



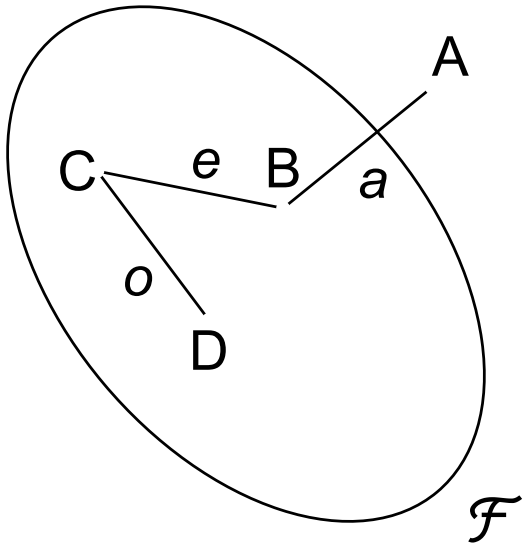
0.036 / 0.939



λ too small

Rupp, PhD thesis, 2009; Vu et al, *Int. J. Quant. Chem.*, 1115, 2015

Learning theory



prediction error =

approximation error a

$+$ estimation error e

$+$ optimization error o

$\mathcal{F}$ = model class, A = true model, B = best model in class, C = best identifiable model (data), D = best identifiable model (optimization)

Changes in size of $\mathcal{F} \Leftrightarrow a$ vs. $e \Leftrightarrow$ **bias-variance trade-off**

Bottou & Bousquet, *NIPS* 2007

Validation

Why?

- assess model performance
- optimize free parameters (hyperparameters)

Which statistics?

- root mean squared error (RMSE)
- mean absolute error (MAE)
- maximum error
- squared correlation coefficient (R^2)

What else can we learn from validation?

- distribution of errors, not only summary statistics
- convergence of error with number of samples

Validation

Golden rule:
Never use training data for validation

Violation of this rule leads to overfitting
by measuring flexibility in fitting instead of generalization ability
rote learner example

If there is sufficient data:

- divide data into two subsets, training and validation
- build model on training subset
- estimate error of trained model on validation subset

Sometimes an external validation set is used in addition.

Statistical validation

If too few data, statistical re-sampling methods can be used, such as cross-validation, bagging, bootstrapping, jackknifing

***k*-fold cross-validation:**

- divide data into k evenly sized subsets
- for $i = 1, \ldots, k$,
  build model on union of subsets $\{1, \ldots, k\} \setminus \{i\}$
  and validate on subset i

All model building steps must be repeated for data splits:

- all pre-processing such as feature selection and centering
- optimization of hyperparameters

Hansen et al, *J. Chem. Theor. Comput.*, 3404, 2103

Hyperparameters: physically motivated choices

Length scale σ:

$\sigma \approx \|\boldsymbol{x} - \boldsymbol{z}\|_1$

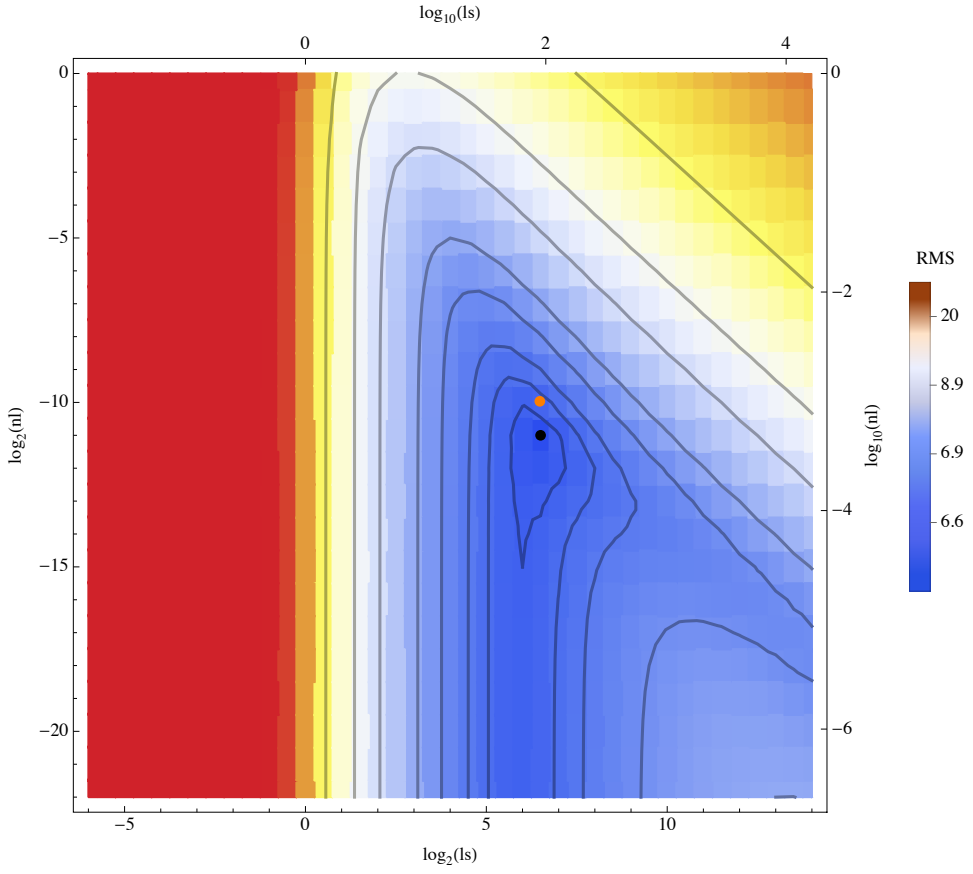
median nearest neighbor distance

Regularization strength λ:

$\hat{=}$ noise variance (Bayesian)

$\hat{=}$ leeway around y_i for fitting

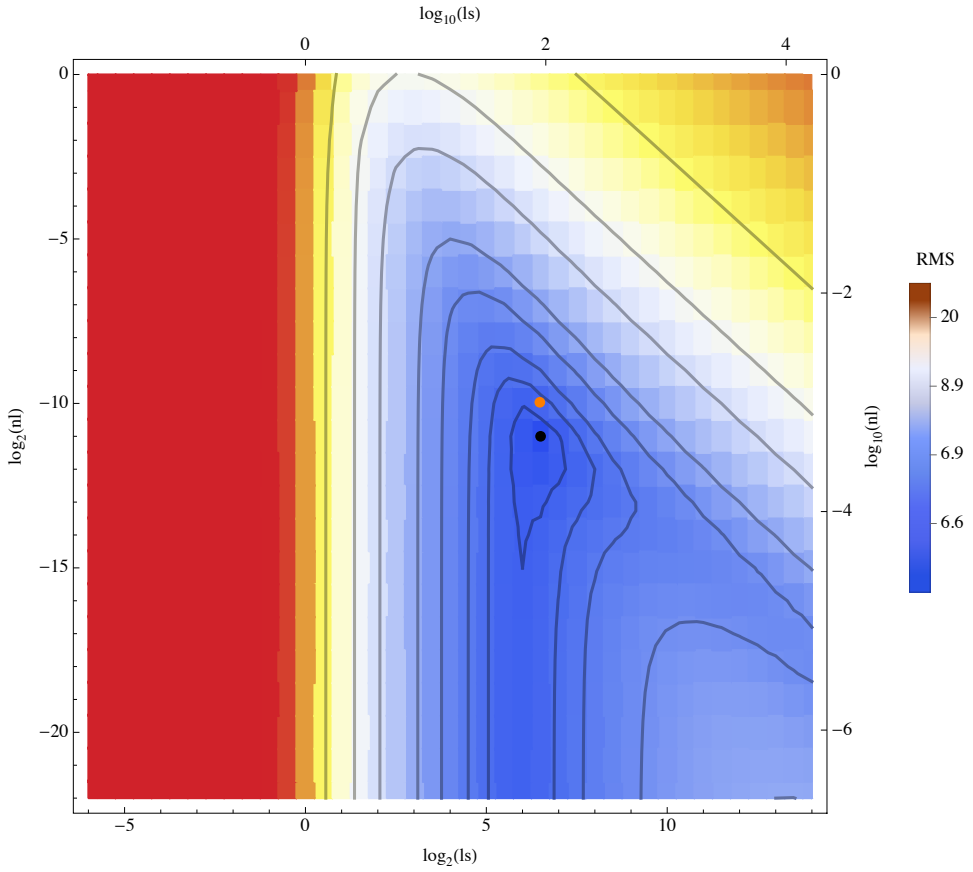
$\Rightarrow$ target accuracy



Rupp: *Int. J. Quant. Chem.*, 1058, 2015

Hyperparameters: statistically motivated choices

- data-driven method for choosing hyperparameters
- optimize using grid search or gradient descent
- use statistical validation to estimate error
- for validation and hyperparameter optimization, use nested data splits



Rupp: *Int. J. Quant. Chem.*, 1058, 2015

Nested data splits

- **never use data from training in validation**
- for performance assessment **and** hyperparameter optimization, use nested cross-validation or nested hold-out sets
- beware of overfitting

Example 1: plain overfitting

× train on all data, predict all data

✓ split data, train, predict

Example 2: centering

× center data, split data, train & predict

✓ split data, center training set, train, center test set, predict

Example 3: cross-validation with feature selection

× feature selection, cross-validation

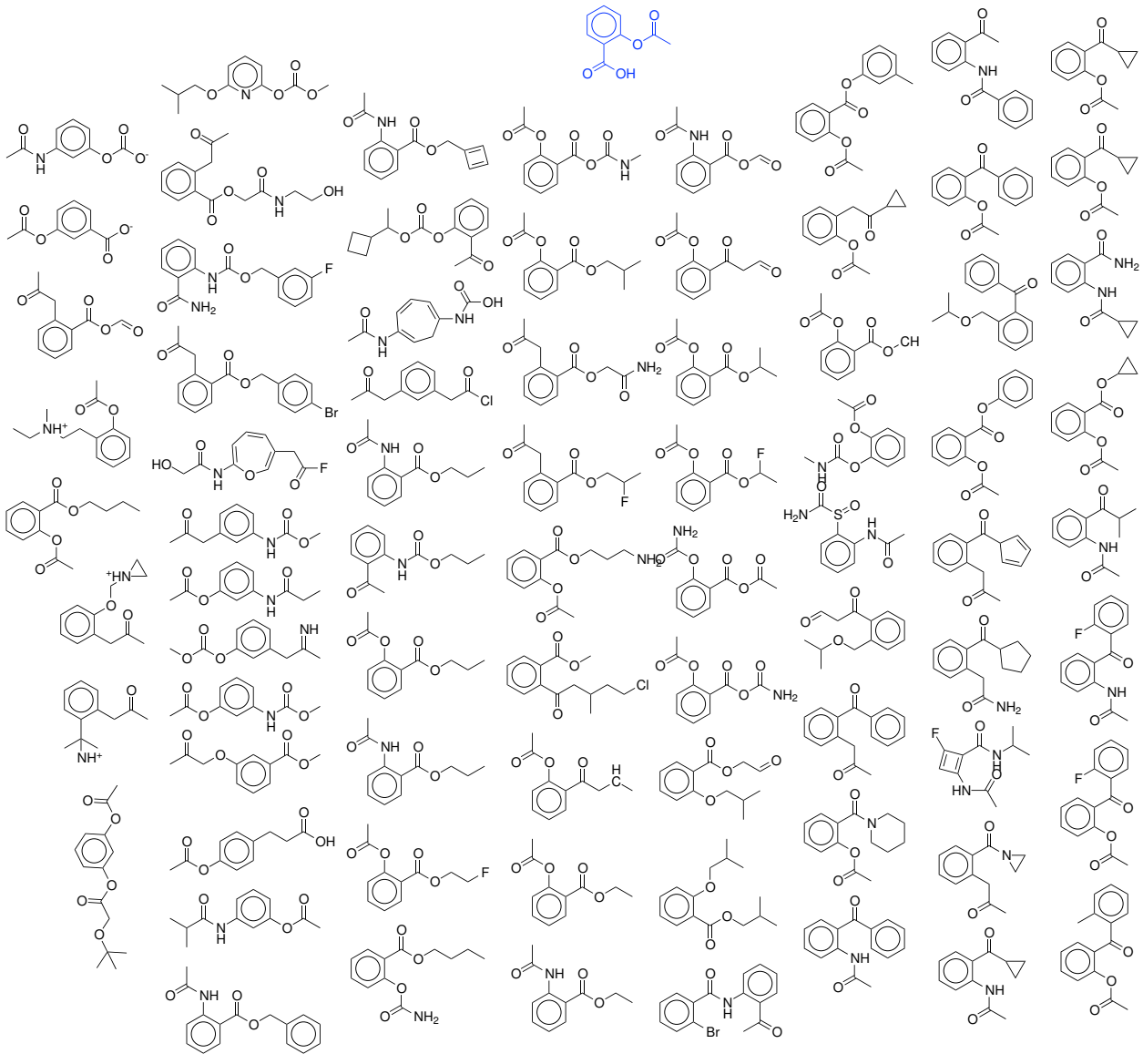
✓ feature selection for each split of cross-validation

Property prediction

Examples

- screening: **chemical interpolation**
  Rupp *et al.*, Phys. Rev. Lett. 108(5): 058301, 2012
- molecular dynamics: potential energy surfaces
  Behler, Phys. Chem. Chem. Phys. 13(40): 17930, 2011
- dynamics simulations: crack propagation in silicon
  Li et al, *Phys Rev Lett* 114: 096405, 2015.
- crystal structure prediction: (meta)stable states
  Ghiringhelli *et al.*, Phys. Rev. Lett. 114(10): 105503, 2015
- density functional theory: kinetic energies
  Snyder *et al.*, Phys. Rev. Lett. 108(25): 253002, 2012
- transition state theory: dividing surfaces
  Pozun *et al.*, J. Chem. Phys. 136(17): 174101, 2012
- amorphous systems: relaxation in glassy liquids
  Schoenholz, Cubuk et al, *Nat. Phys.* 12(5): 469, 2016
- design: stable interface search
  Kiyohara, Oda, Tsuda, Mizoguchi, *Jpn. J. Appl. Phys.* 55(4): 045502, 2016

The combinatorial nature of chemical/materials space

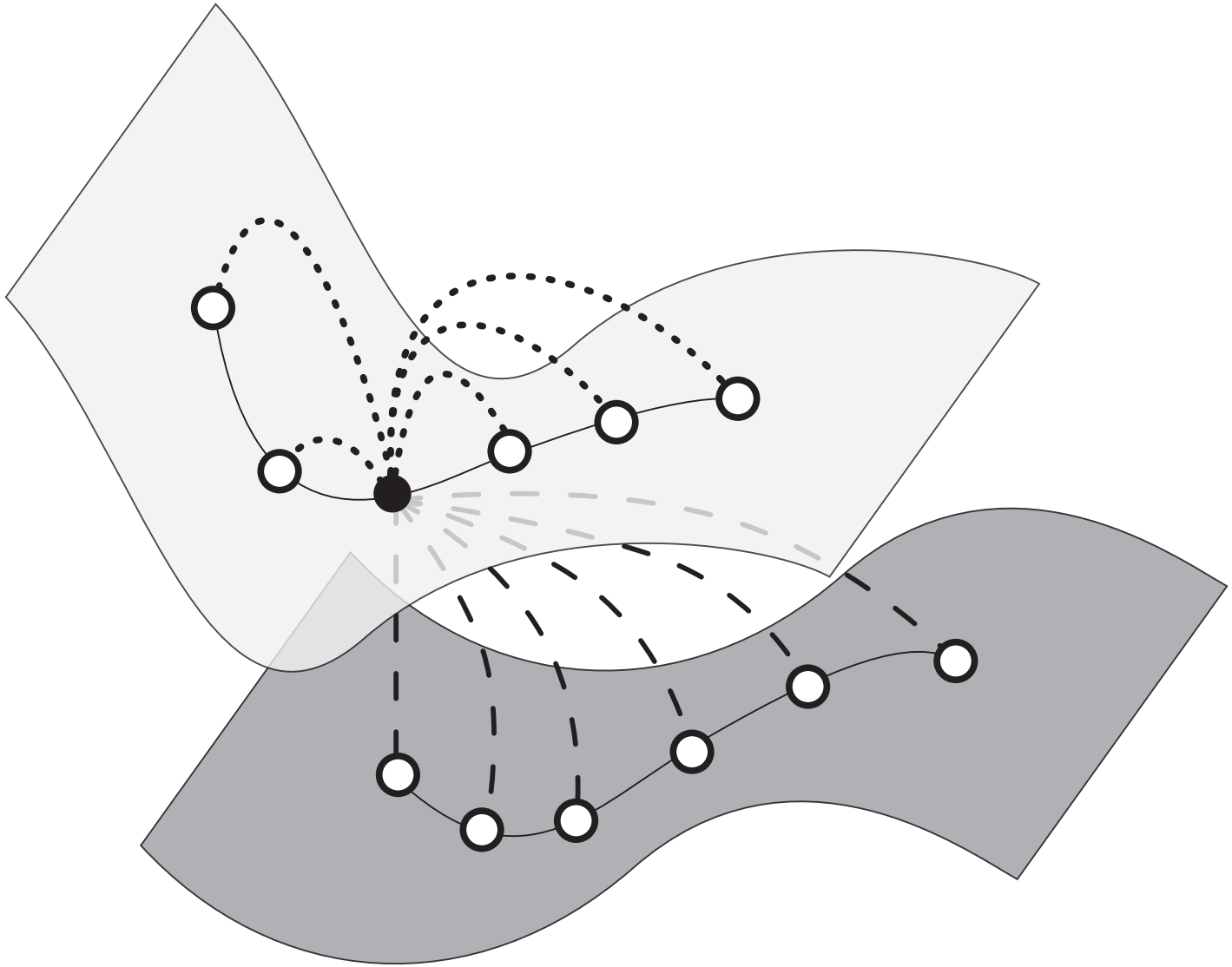


- molecule space: graph theory
- materials space: group theory
- combinatorial explosion

aspirin derivatives

Gómez-Bombarelli et al, arXiv, 2016

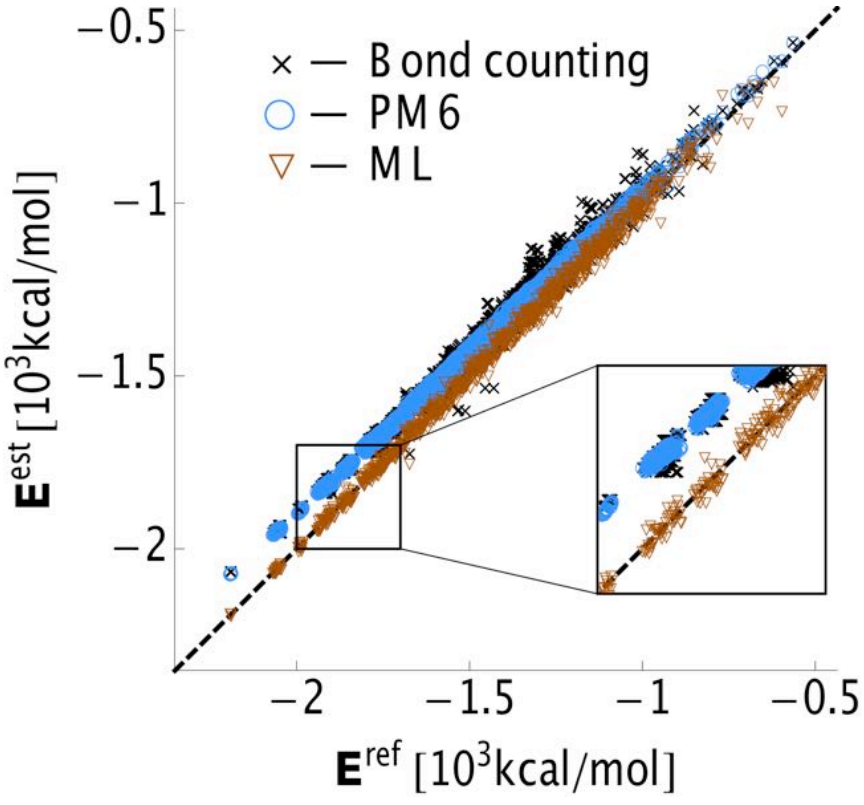
Learning potential energy surfaces

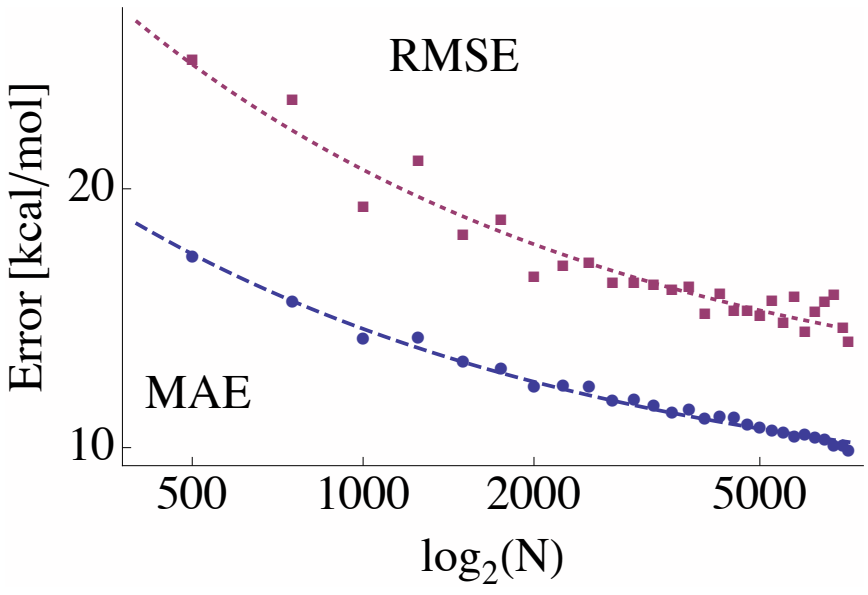


Chang, von Lilienfeld: *CHIMIA* 68, 602, 2014
von Lilienfeld, *Int. J. Quant. Chem.* 113, 1676, 2013.

Predicting atomization energies

- 7 165 small organic molecules (H,C,N,O,S; 1–7 non-H atoms)
- DFT PBE0 atomization energies
- kernel ridge regression, Gaussian kernel $k(\mathbf{M}, \mathbf{M}') = \exp\left(-\frac{d^2(\mathbf{M},\mathbf{M}')}{2\sigma^2}\right)$



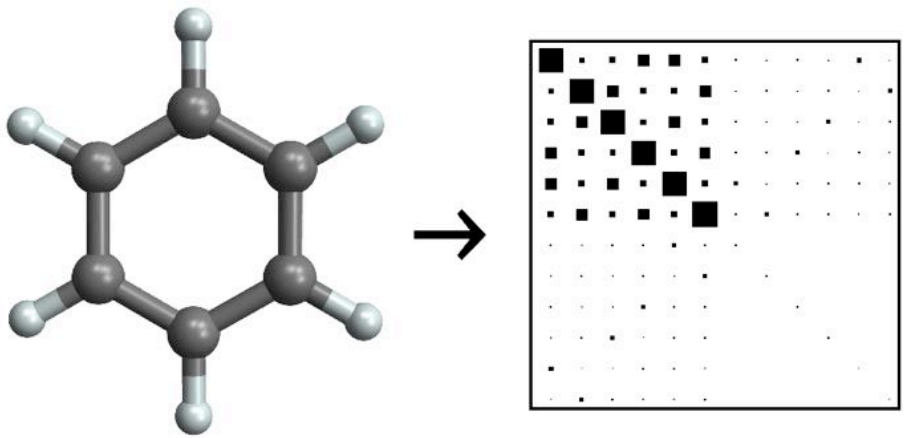


Rupp, Tkatchenko, Müller, von Lilienfeld: *Phys. Rev. Lett.*, 2012

Coulomb matrix

$$\mathbf{M}_{ij} = \begin{cases} \frac{1}{2} Z_i^{2.4} & \text{for } i = j \\ \frac{Z_i Z_j}{|\mathbf{R}_i - \mathbf{R}_j|} & \text{for } i \neq j \end{cases}$$

- atom positions $\boldsymbol{R_i}$ and proton numbers Z_i
- sort by simultaneously permuting rows and columns
- Frobenius norm, pad with zeros to allow different sizes

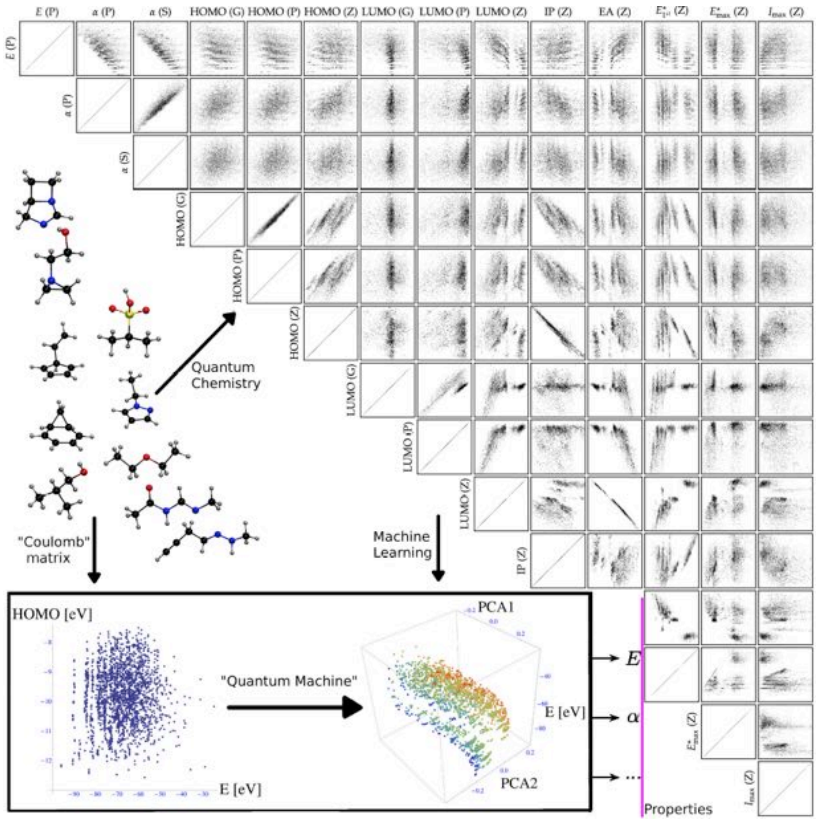


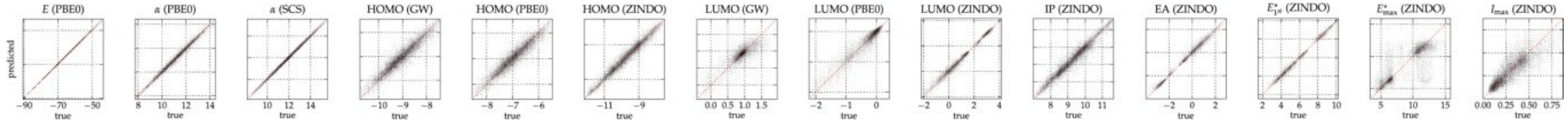
Rupp, Tkatchenko, Müller & von Lilienfeld; *Phys. Rev. Lett.* 2012

Extension to other properties

Learning the map from molecular structure to molecular properties

- various properties
- various levels of theory
- small organic molecules
- Coulomb matrix representations
- kernel learning,
  deep neural networks
- for 5k training molecules, errors
  are comparable to the reference

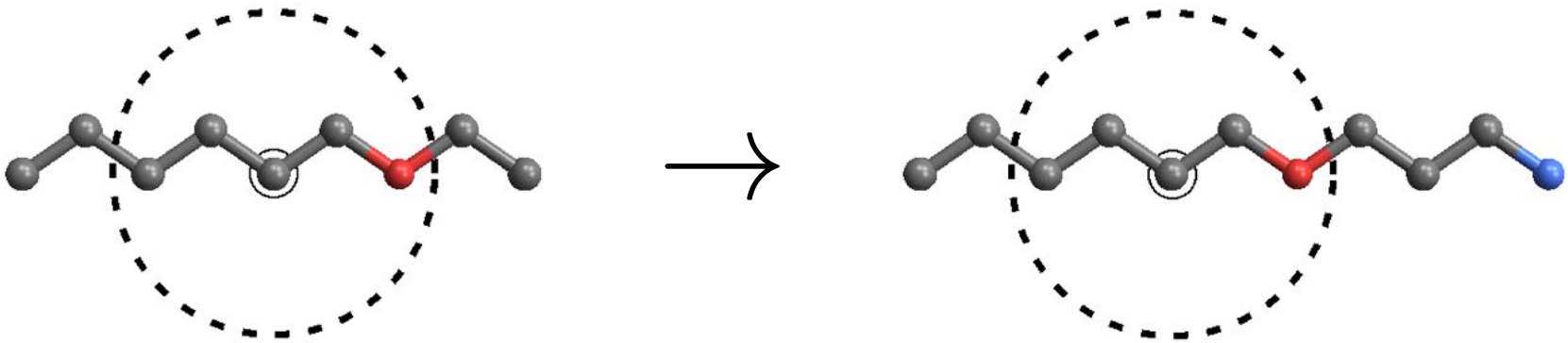




Montavon et al, *New. J. Phys.*, 2013; Hansen et al, *J. Chem. Theor. Comput.*, 2013.

Local properties

Local interpolation is global extrapolation.



- **linear scaling** of computational effort with system size
- size consistent in the limit
- requires **partitioning** for global properties

Bartók et al, *Phys Rev Lett* 104, 2010
Rupp et al, *J Phys Chem Lett* 6, 2015
Behler, *J Phys Condens Matter* 26, 2014

Summary

1. machine learning finds regularity in data for analysis or prediction, improving with more data
2. kernel trick: implicit transformation to high-dimensional spaces, with kernel ridge regression as example
3. for validation, avoid over-fitting by following the golden rule
4. interpolation of electronic structure calculations;
   example: atomization energies of organic molecules

Tutorial

Matthias Rupp:
Machine Learning for Quantum Mechanics in a Nutshell
International Journal of Quantum Chemistry 15(16): 1058–1073, 2015
`http://doi.org/10.1002/qua.24954`

Links

`http://mrupp.info` (Publications)
`http://qmml.org` (Datasets)