

# Practical Pipelining using Python





# Data flood

- More projects than ever conducted **survey mode** – even PhD projects

- Wide area or dispersed: **many pointings**
- **Multi-epoch**: years of ongoing observations and many observers
- Multi-wavelength: **linked with sister surveys** on other telescopes



- Huge amount of **meta-data** necessary to manage successful science:

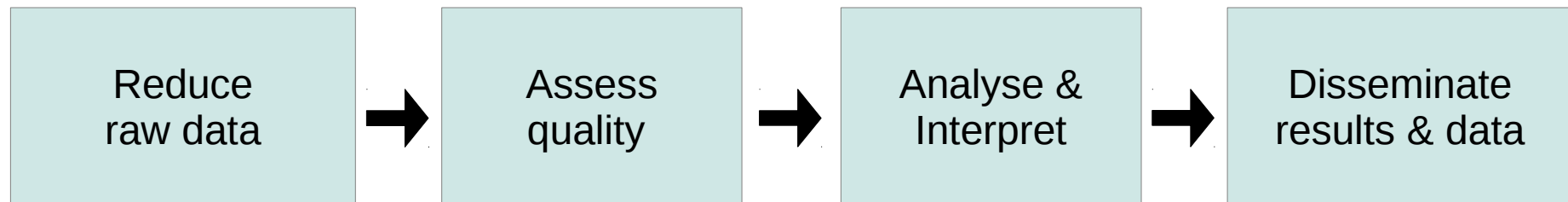
- Coordinates, sensitivity, outages, coverage, imaging quality, source-finding, cross-matching, quality control, publishing data-products

- Culture of **transparency**, consistency and **reproducibility**

- Transparent processing from final data-product to raw observations
- Consistent processing across dataset using well tested software
- Version control so that results are reproducible after later releases

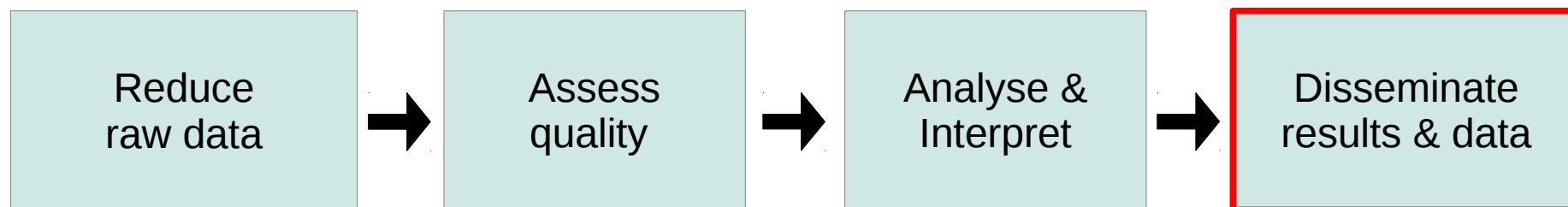
- **Worth building a software pipeline!**

# Typical pipeline flow



- Data reduction (ingestion, calibration, imaging)
- Quality control (re-observe bad data, weighting, flagging)
- Basic Analysis (source-finding, clump/spectral-fitting, catalogues)
- Advanced analysis (multi-wavelength cross-matching, time-series, object identification)
- Data access and citizen science (web-publication, virtual observatory)
- **Must be a meta-data trail from the start – requires a database**

# Python – a sticky language



- Image and table data accessible using **astropy** module (FITS & ASCII)
- Manage **metadata** via simple built-in database or external relational database
- Easily manipulate data in memory to create diagnostic metrics (numpy)
- Flexible plotting ability via **matplotlib** and **APLpy**
- Well-tested suite of **analysis tools**, e.g., source-finding (Aegean, Blobcat), model-fitting (MPFIT, scipy.optimize, EMCEE).
- Python code is **easy to read** and should be published alongside papers
- Web-servers (e.g. Apache2, bottle) understand python. Build dynamic web-interfaces to your scripts and share with your co-authors and community:
  - Catalogue & image servers, 'Zooniverse' style analysis (user-based)
  - Automatically generate plots, tables and reports for quality assessment

# Python – astropy and friends

- Until ~2012 python astronomy modules were scattered around the web
- Core modules now unified under **astropy** banner & **affiliated packages**
  - Astropy.io.fits (formerly PyFITS)
  - Astropy.io.ascii
  - Astropy.wcs (formerly PyWCS)
  - Astropy.coordinates
  - Astropy.convolution
- Useful affiliated packages:
  - APLpy - Image plotting
  - PyVO - query the virtual observatory
  - Astroquery – access web services
- Useful non-astronomy packages
  - MySQL DB
  - Matplotlib
  - MPFIT
  - SQLite3
  - SciPy
  - EMCEE





# Python – beware the unknown

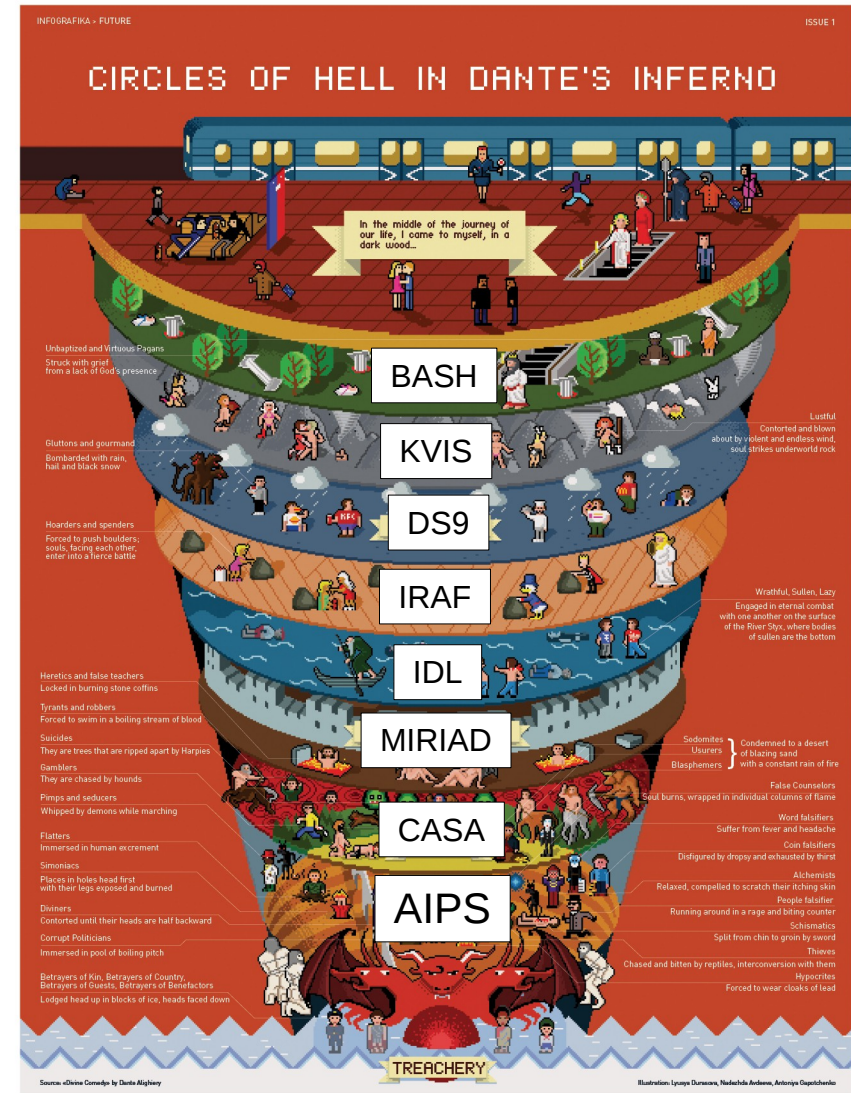


- **Astropy** is still under development
- **Core** packages **mature** & well tested
- **Newer packages** incomplete & in flux
- Newer packages often expose less well-known python functionality and add useful features
- Sometimes necessary to peer behind the curtain and **get your hands dirty**
  - `Astropy.io.ascii` ~ `numpy.loadtxt`
  - `Astropy.table` ~ `numpy.recarray`
- APLpy easily augmented
  - Matplotlib under the hood
  - Combine with other figures
- No substitute for ***really understanding what is being done to your data!***



# Python – accessing external software

- **Good:** Often easiest to make **system calls**
  - Tell python where the package lives and set up the \$ENVIRONMENT
  - Pipe into STDOUT to view output of task in real time
- **Bad(ish):** Some external packages have dedicated python wrappers
  - MIRIAD → **MIRpy**
  - AIPS → **parseltongue** and **obittalk**
  - CASA → **casapy** \*
  - Increases *complexity* sometimes (bad)
- **Super-Ugly:** Can interface with *ancient* user-driven software using **pyexpect**
  - Pretends to be the user when dealing with Q&A command line interfaces



\* **casapy** comes with its own version of Python, which means you may need to manage this environment in parallel, i.e., duplicate installed modules

# Python – accessing external software

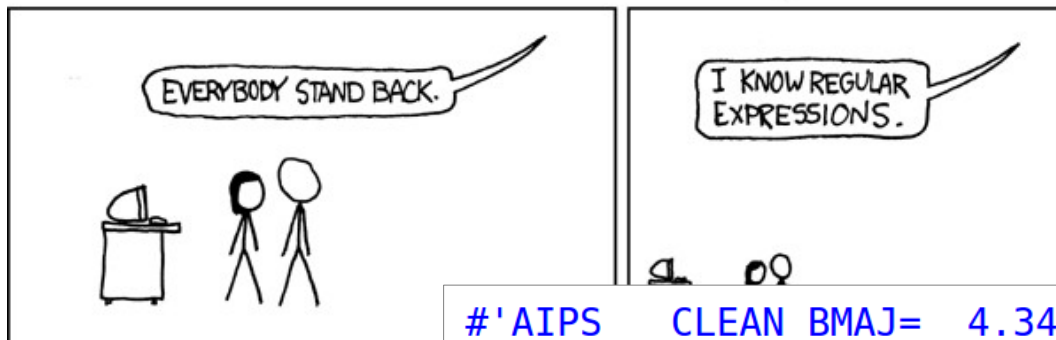


- Regular expressions (**RegEx**)
- Match patterns in any ASCII string
- Incredibly useful for **parsing** the output of programs and tasks, e.g., **MIRIAD uindex**

`\S+` = 1 or more non-space chars

`\s*` = 0 or more spaces

`()` = groups of text to extract



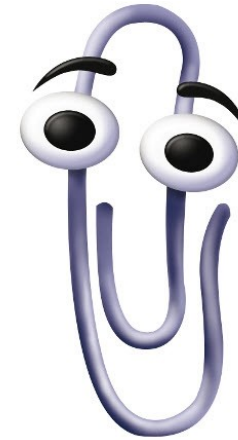
```
#'AIPS    CLEAN BMAJ= 4.3403E-04 BMIN= 3.1039E-04 BPA= -11.55'
beamRe = 'AIPS\s+CLEAN\sBMAJ=\s+(\S+)\s+BMIN=\s+(\S+)\s+BPA=\s+(\S+)'
beamPat = re.compile(beamHistStr)
```

```
for i in range(len(history)):
    mch = bmHistPat.match(history[i])
    if mch:
        bmaj = float(mch.group(1))
        bmin = float(mch.group(2))
        bpa = float(mch.group(3))
        break
```



# Databases – Why?

- Replaces **Excel** (or Topcat, ASCII) in your workflow
- Advantages: **scriptable**, **multi-user** access, *fast*
- Large projects: run on a **dedicated server** (MySQL)
- Small projects: database **built-into python** (SQLite3)
- Server-client model with choice of interface client:
  - **Graphical** clients which mimics **spreadsheets**
  - **Queries** through a script or command-line return a **filtered** or **joined table**
- Under the bonnet:
  - Tables connected by **key** columns with **unique entries**



Sometimes I just popup for no reason at all. Like now.



	ID	Name	RA deg	Dec deg
1				
2				
3				
4				
5				
6				
7				

	ID	Flux jy	dFlux jy	rms jy
1				
2				
3				
4				
5				
6				
7				

# SQLite3 – Python's built-in database

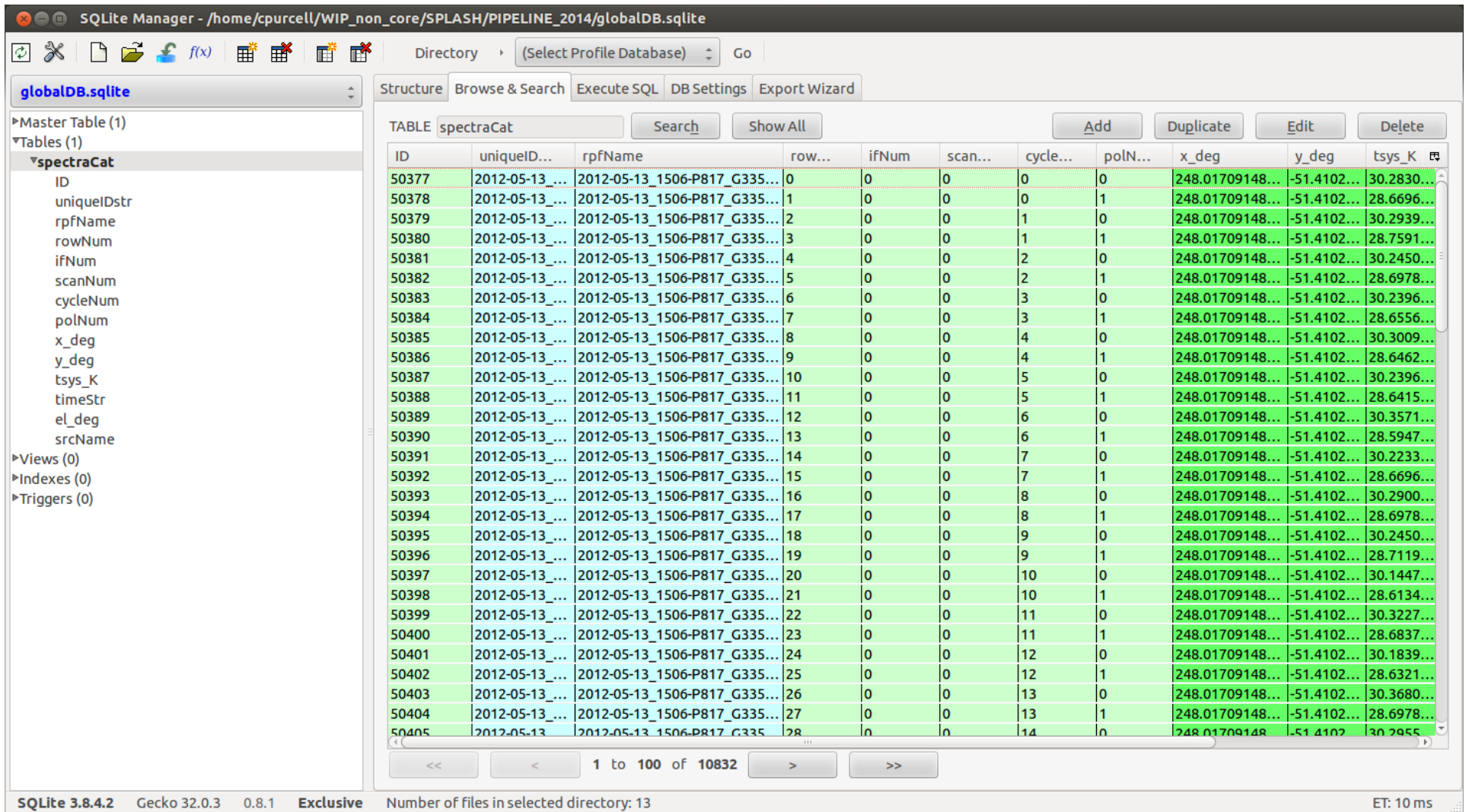
- Distributed with python since V2.5
- **Serverless:** creates a database file
- File can be moved, copied and read on any other system (OS-agnostic)
- Can also operate in memory
- **No** configuration required
- Open source
- Crash safe (prevents data corruption)
- Already used by many well known programs to store data on your computer:



<http://www.sqlite.org>

The Google logo, consisting of the word "Google" in its characteristic multi-colored font.

# SQLite3 – access through Firefox



The screenshot shows the SQLite Manager Firefox plug-in interface. The main window displays a table named 'spectraCat' with the following columns: ID, uniqueID..., rpfName, row..., ifNum, scan..., cycle..., polN..., x\_deg, y\_deg, and tsys\_K. The table contains 28 rows of data, with the first row having ID 50377 and the last row having ID 50405. The interface includes a sidebar with a tree view showing the database structure, a top menu bar with options like 'Structure', 'Browse & Search', 'Execute SQL', 'DB Settings', and 'Export Wizard', and a bottom status bar showing 'SQLite 3.8.4.2', 'Gecko 32.0.3', '0.8.1', 'Exclusive', and 'Number of files in selected directory: 13'.

ID	uniqueID...	rpfName	row...	ifNum	scan...	cycle...	polN...	x_deg	y_deg	tsys_K
50377	2012-05-13_...	2012-05-13_1506-P817_G335...	0	0	0	0	0	248.01709148...	-51.4102...	30.2830...
50378	2012-05-13_...	2012-05-13_1506-P817_G335...	1	0	0	0	1	248.01709148...	-51.4102...	28.6696...
50379	2012-05-13_...	2012-05-13_1506-P817_G335...	2	0	0	1	0	248.01709148...	-51.4102...	30.2939...
50380	2012-05-13_...	2012-05-13_1506-P817_G335...	3	0	0	1	1	248.01709148...	-51.4102...	28.7591...
50381	2012-05-13_...	2012-05-13_1506-P817_G335...	4	0	0	2	0	248.01709148...	-51.4102...	30.2450...
50382	2012-05-13_...	2012-05-13_1506-P817_G335...	5	0	0	2	1	248.01709148...	-51.4102...	28.6978...
50383	2012-05-13_...	2012-05-13_1506-P817_G335...	6	0	0	3	0	248.01709148...	-51.4102...	30.2396...
50384	2012-05-13_...	2012-05-13_1506-P817_G335...	7	0	0	3	1	248.01709148...	-51.4102...	28.6556...
50385	2012-05-13_...	2012-05-13_1506-P817_G335...	8	0	0	4	0	248.01709148...	-51.4102...	30.3009...
50386	2012-05-13_...	2012-05-13_1506-P817_G335...	9	0	0	4	1	248.01709148...	-51.4102...	28.6462...
50387	2012-05-13_...	2012-05-13_1506-P817_G335...	10	0	0	5	0	248.01709148...	-51.4102...	30.2396...
50388	2012-05-13_...	2012-05-13_1506-P817_G335...	11	0	0	5	1	248.01709148...	-51.4102...	28.6415...
50389	2012-05-13_...	2012-05-13_1506-P817_G335...	12	0	0	6	0	248.01709148...	-51.4102...	30.3571...
50390	2012-05-13_...	2012-05-13_1506-P817_G335...	13	0	0	6	1	248.01709148...	-51.4102...	28.5947...
50391	2012-05-13_...	2012-05-13_1506-P817_G335...	14	0	0	7	0	248.01709148...	-51.4102...	30.2233...
50392	2012-05-13_...	2012-05-13_1506-P817_G335...	15	0	0	7	1	248.01709148...	-51.4102...	28.6696...
50393	2012-05-13_...	2012-05-13_1506-P817_G335...	16	0	0	8	0	248.01709148...	-51.4102...	30.2900...
50394	2012-05-13_...	2012-05-13_1506-P817_G335...	17	0	0	8	1	248.01709148...	-51.4102...	28.6978...
50395	2012-05-13_...	2012-05-13_1506-P817_G335...	18	0	0	9	0	248.01709148...	-51.4102...	30.2450...
50396	2012-05-13_...	2012-05-13_1506-P817_G335...	19	0	0	9	1	248.01709148...	-51.4102...	28.7119...
50397	2012-05-13_...	2012-05-13_1506-P817_G335...	20	0	0	10	0	248.01709148...	-51.4102...	30.1447...
50398	2012-05-13_...	2012-05-13_1506-P817_G335...	21	0	0	10	1	248.01709148...	-51.4102...	28.6134...
50399	2012-05-13_...	2012-05-13_1506-P817_G335...	22	0	0	11	0	248.01709148...	-51.4102...	30.3227...
50400	2012-05-13_...	2012-05-13_1506-P817_G335...	23	0	0	11	1	248.01709148...	-51.4102...	28.6837...
50401	2012-05-13_...	2012-05-13_1506-P817_G335...	24	0	0	12	0	248.01709148...	-51.4102...	30.1839...
50402	2012-05-13_...	2012-05-13_1506-P817_G335...	25	0	0	12	1	248.01709148...	-51.4102...	28.6321...
50403	2012-05-13_...	2012-05-13_1506-P817_G335...	26	0	0	13	0	248.01709148...	-51.4102...	30.3680...
50404	2012-05-13_...	2012-05-13_1506-P817_G335...	27	0	0	13	1	248.01709148...	-51.4102...	28.6978...
50405	2012-05-13_...	2012-05-13_1506-P817_G335...	28	0	0	14	0	248.01709148...	-51.4102...	30.2955...

- SQLite Manager Firefox plug-in: edit, query & export table data

<https://addons.mozilla.org/en-US/firefox/addon/sqlite-manager/>

# SQLite3 – access through Python



- Python SQLite3 database access takes place via **cursor object**
- Queries are designed in a special statements which are used to **delete, insert** or **select** information from the database tables.
- Interface is almost identical when accessing heavy-duty databases (MySQL, PostgreSQL).

```
# Connect to the database file
dbFile = 'myDatabase.sqlite'
conn = sqlite3.connect(dbFile)
cursor = conn.cursor()

# Fetch all of 'myTable'
statement = "SELECT * FROM myTable;"
cursor.execute(sql)
rows = cursor.fetchall()

# Disconnect
cursor.close()
conn.close()
```

GREETINGS PROFESSOR FALKEN

HELLO

A STRANGE GAME.  
THE ONLY WINNING MOVE IS  
NOT TO PLAY.

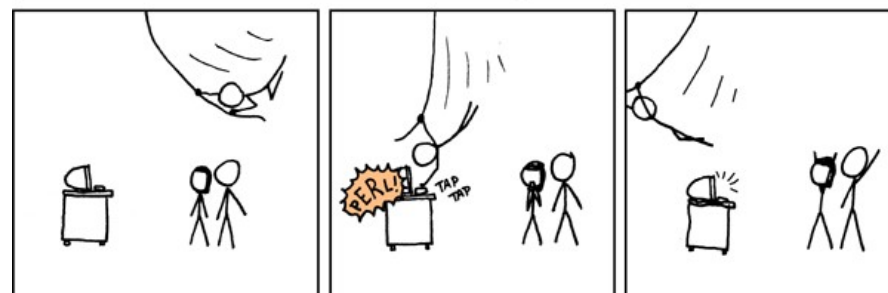
HOW ABOUT A NICE GAME OF CHESS?



# Databases – Sequel SQL



- 'Structured Query Language' (SQL)
  - Close to English in syntax
  - Basic trigonometric and mathematical operations
  - Powerful regular expressions



# SQL – creating a table

## Raw SQL

```
CREATE TABLE spectraCat (  
ID INTEGER PRIMARY KEY,  
uniqueIDstr VARCHAR(100),  
rpfName VARCHAR(50),  
rowNum INTEGER,  
ifNum INTEGER,  
scanNum INTEGER,  
cycleNum INTEGER,  
polNum INTEGER,  
x_deg DOUBLE,  
y_deg DOUBLE,  
tsys_K DOUBLE,  
timeStr VARCHAR(50),  
el_deg DOUBLE,  
srcName VARCHAR(50));
```

## Python

```
# Create a table  
sql = """  
CREATE TABLE spectraCat (  
ID INTEGER PRIMARY KEY,  
uniqueIDstr VARCHAR(100),  
rpfName VARCHAR(50),  
rowNum INTEGER,  
ifNum INTEGER,  
scanNum INTEGER,  
cycleNum INTEGER,  
polNum INTEGER,  
x_deg DOUBLE,  
y_deg DOUBLE,  
tsys_K DOUBLE,  
timeStr VARCHAR(50),  
el_deg DOUBLE,  
srcName VARCHAR(50));  
"""  
cursor.execute(sql)
```

# SQL – populating a database

## Raw SQL

```
INSERT OR REPLACE  
INTO tableName  
  (column1,  
   column2,...  
   columnN)  
VALUES (value1,  
         value2,...  
         valueN);
```

## Python

```
# Set some values for columns|  
colName1 = 'column1'  
val1 = 2.0  
colName2 = 'column2'  
val2 = 3.0  
  
# Form the SQL statement in parts  
sql = "INSERT OR REPLACE INTO %s" % tableName  
sql += ' (%s, %s)' % (colName1, colName2)  
sql += ' VALUES (?, ?)'  
  
# Note: we put '?' in place of values  
# and then use the 2nd argument of the  
# execute statement to specify the values  
cursor.execute(sql, (val1, val2))  
  
# Call to write the changes to disk  
conn.commit()
```

# SQL – querying a database

## Raw SQL

```
SELECT column1, column5|
FROM tableName
WHERE
column1 = <condition1>
AND column2 = <condition2>
AND (column3 > value1 OR
      column3 < value2)
ORDER BY column5;
```

## Python

```
# Coordinate limits
xLimUp = 0.5; xLimDn = -0.5

# SQL
sql = '''SELECT
uniqueName,
x_deg,
y_deg,
flux_Jybm
FROM spectraTab
WHERE coordTab.x_deg<?
AND coordTab.x_deg>?
ORDER BY x_deg
'''

# Execute using limits
cursor.execute(sql, (xLimUp, xLimDn))
rows = cursor.fetchall()
```



# SQL – joining two tables

## Raw SQL

```
SELECT
table1.column1,
table1.column2,
table2.column5
FROM table1 INNER JOIN table2
ON table1.column1 = table2.column1
WHERE
table1.column1 = <condition1>
AND table1.column2 = <condition2>
AND (table2.column3 > value1 OR
      table2.column3 < value2)
ORDER BY table2.column5;
```

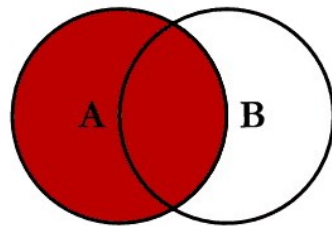
## Python

```
# Coordinate limits
xLimUp = 0.5; xLimDn = -0.5

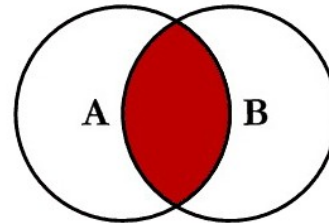
# SQL
sql = '''SELECT
coordTab.uniqueName,
coordTab.x_deg,
coordTab.y_deg,
spectraTab.flux_Jybm
FROM coordTab LEFT JOIN spectraTab
ON coordTab.uniqueName =
|spectraTab.uniqueName
WHERE coordTab.x_deg<?
AND coordTab.x_deg>?
ORDER BY x_deg
'''

# Execute using limits
cursor.execute(sql, (xLimUp, xLimDn))
rows = cursor.fetchall()
```

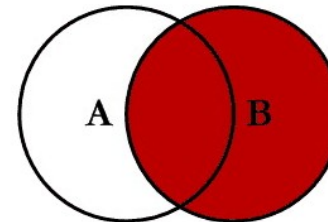
# SQL – joining two tables



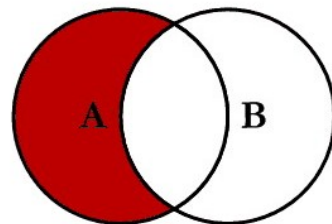
Left Join



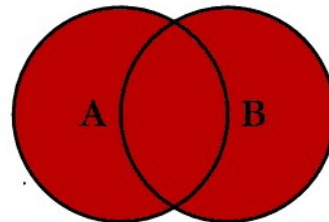
Inner Join



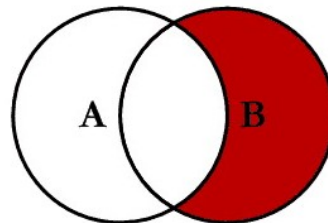
Right Join



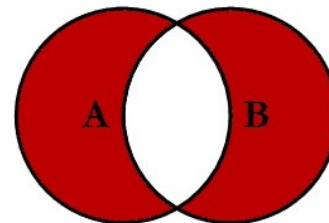
Left Join  
WHERE B  
IS NULL



Outer Join



Right Join  
WHERE A  
IS NULL



Outer Join  
WHERE A IS NULL  
OR B IS NULL

# Python – Numpy record arrays

- **Numpy** is the array-processing and vector algebra module for python
- Arrays commonly defined as a **single data-type** (e.g., 32-bit float)
- A more rich definition exists which allows mixing of types - **recarrays**
- Columns can be named and assigned different variable types (dtypes)
- Natural match for tabular data stored in a database
- Some database-like functionality using logical statements and filters
- Augmented version used in the **astropy.table** module



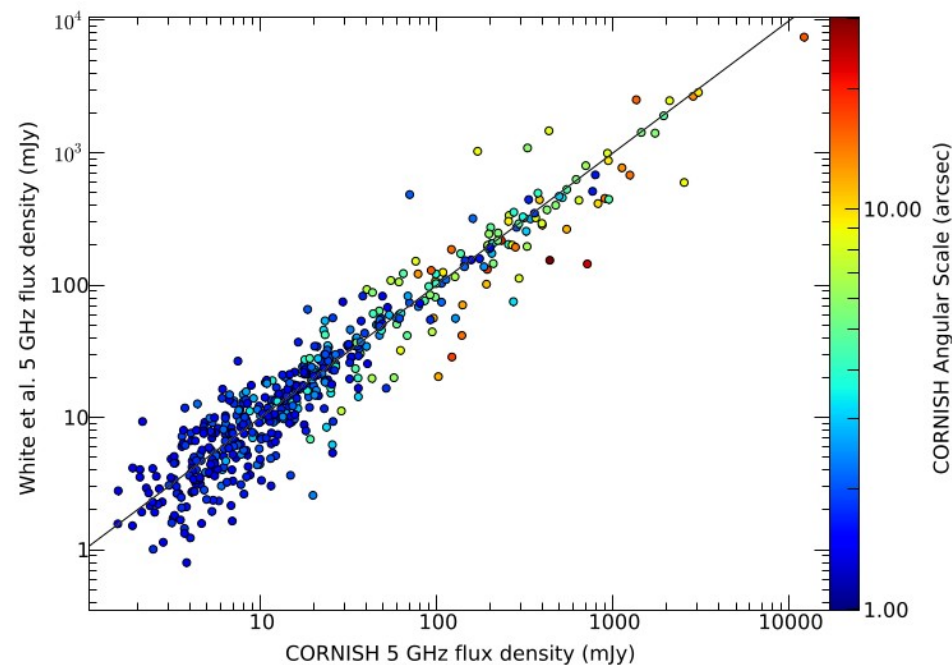
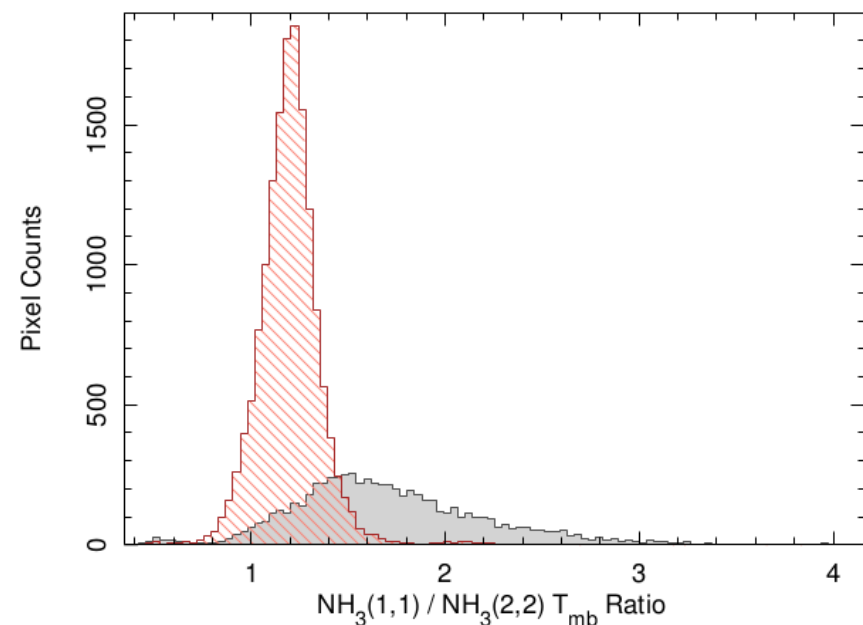
```
# Initialising a numpy array
myArr = np.zeros(shape=(3), dtype='f4')

# Initialising a record array:
dtLst = [('name', 'a10'),
         ('maxFlux', 'f4'),
         ('RMS', 'f4')]
myRec = np.zeros(shape=(3), dtype=dtLst)

# Accessing a record array:
myRec['name']      # by column
myRec[0]          # by row
```

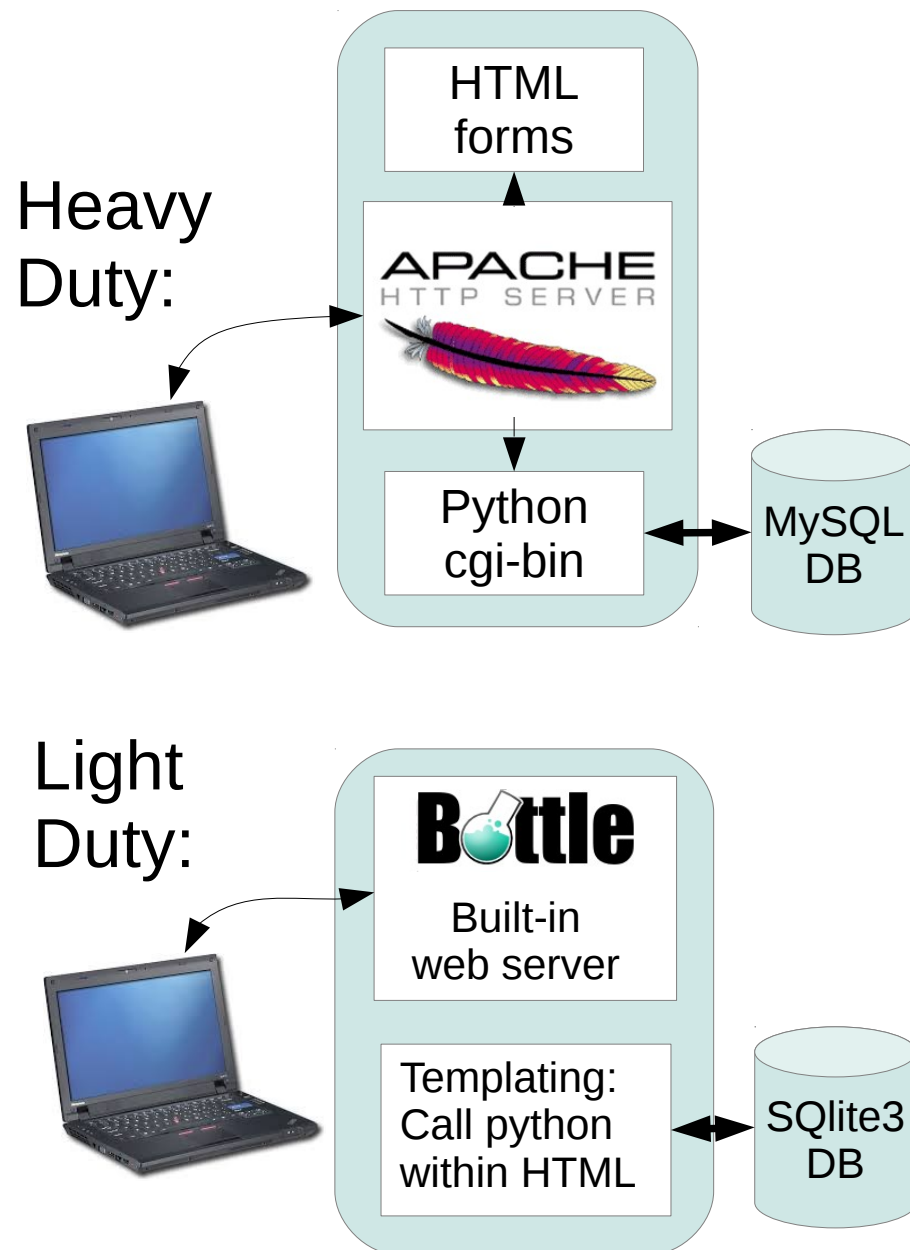
# Plotting from a database

- **Query driven plotting** is a very **powerful** tool for exploring data
- The **query** defines the input **data**
- Two most useful plots in science:
  - Histogram
  - Scatter plot
- Build simple python script to read one or more queries for comparison
- Control flags allow manipulation of plot parameters



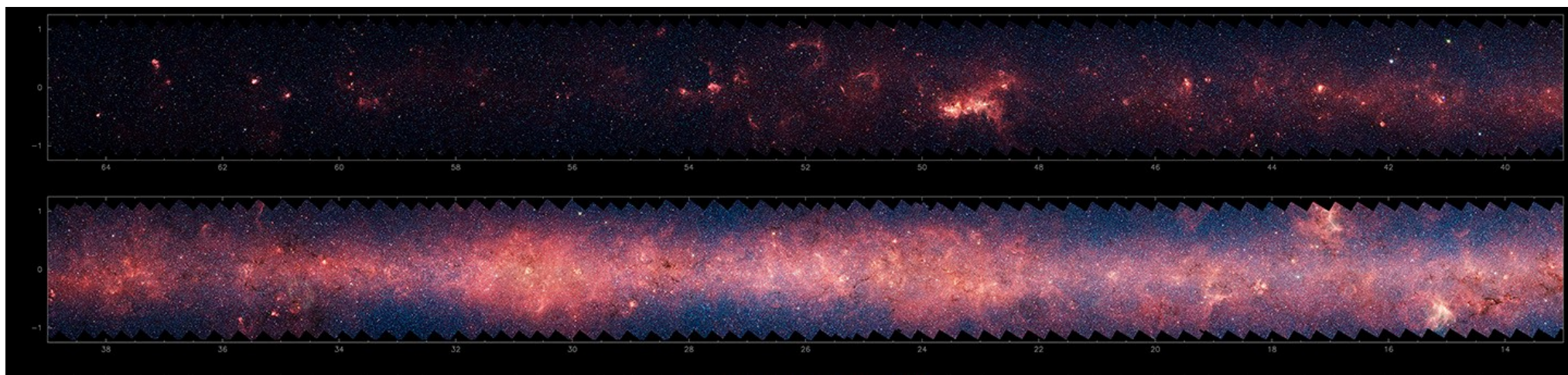
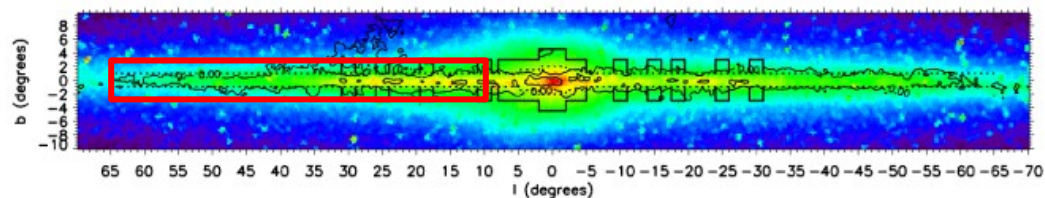
# Sharing your science – web access

- Interface to your project at **all stages**
  - Managing observations (e.g., MALT90)
  - Raw data quality control (daily reports)
  - Data-reduction control (selection)
  - User driven tasks: **eyeball carnage**
- Two main choices for web
  - Heavy duty server e.g., **Apache2**
  - Light web 'framework' e.g., **Bottle**
- **Apache**: Common Gateway Interface
  - Python scripts in the cgi-bin directory
  - Matching forms in separate HTML
- **Bottle**: Templating language
  - HTML template defines forms and calls python in one place
  - **Stand-alone** or use within Apache



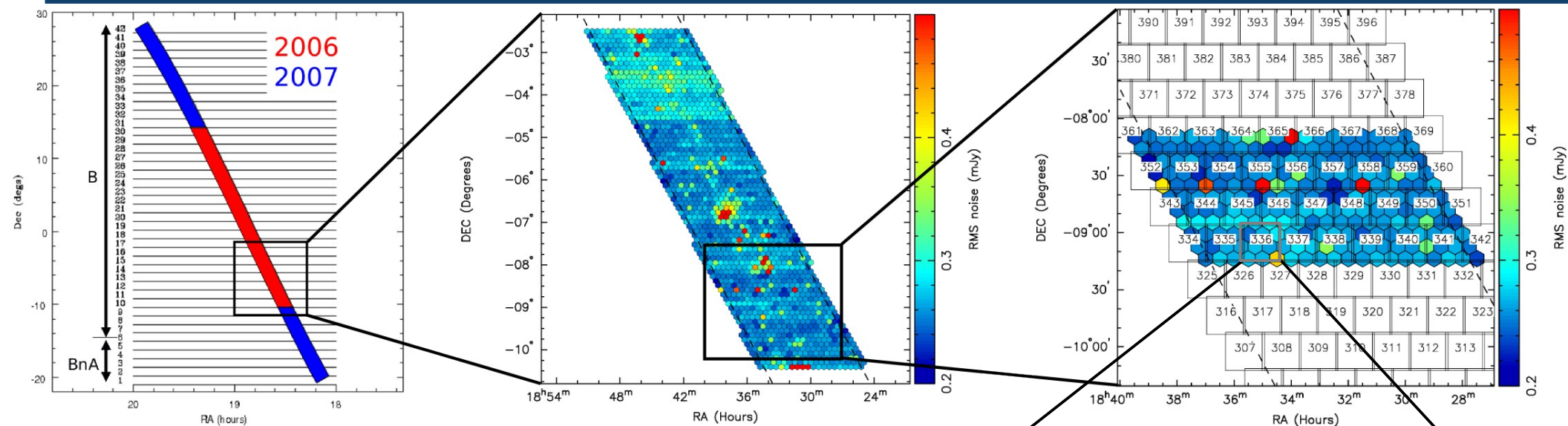
# Pipeline case study – CORNISH

- Northern GLIMPSE I region
- $10^{\circ} < l < 65^{\circ}$        $|b| < 1^{\circ}$
- 100 square degrees
- 8.5' primary beam, 1.5" resolution
- $< 0.4$  mJy/bm rms noise level
- **9351 fields**, hexagonal pattern
  - 2x 40s cuts  $\rightarrow$  **18702** data

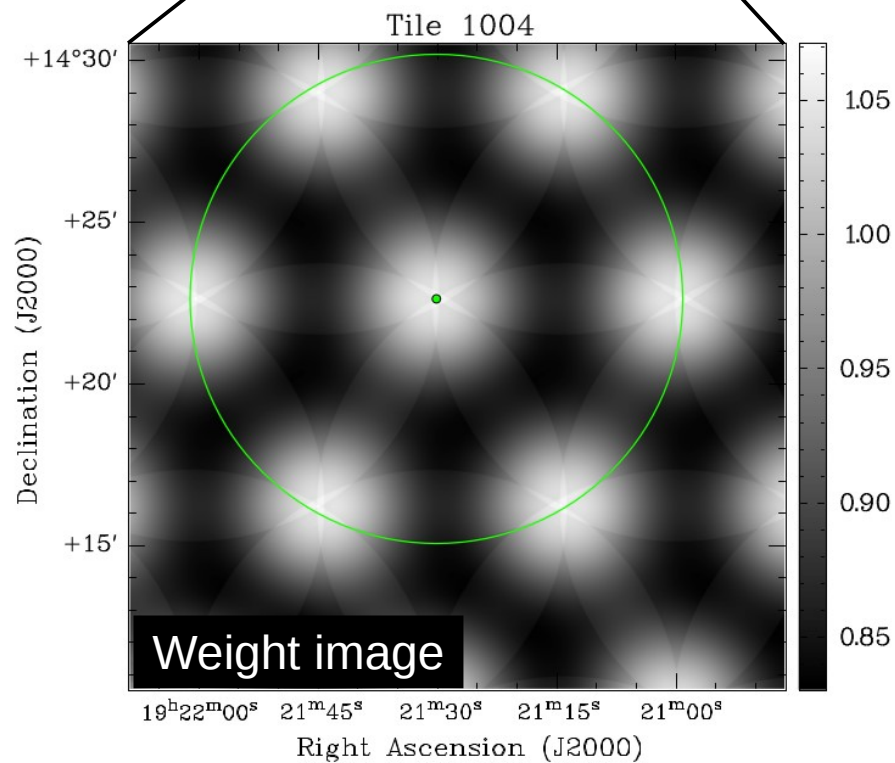




# Pipeline case study – CORNISH

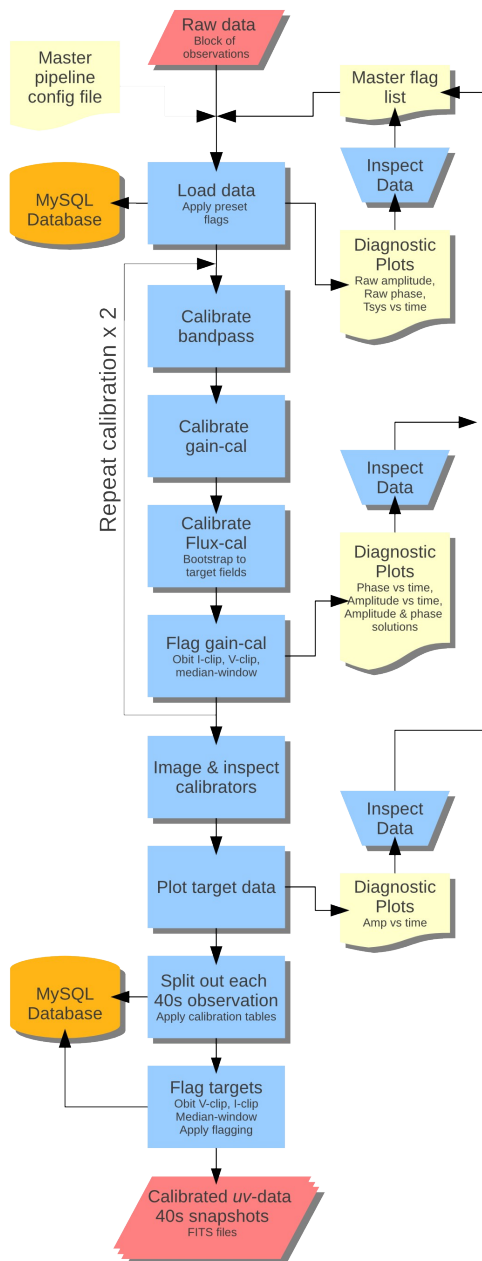


- Primary data product is a mosaiced image 'tile'
- Observations conducted over 3 years
- Important to quickly reduce data and track data-quality so bad fields could be re-observed

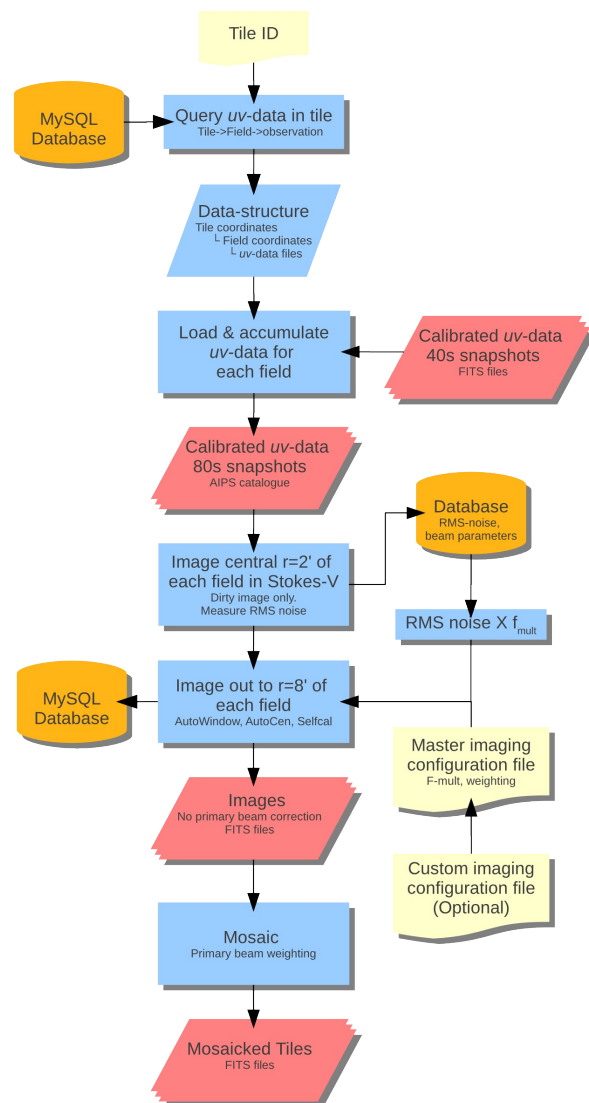


# Pipeline case study – CORNISH

## Calibration:



## Imaging:

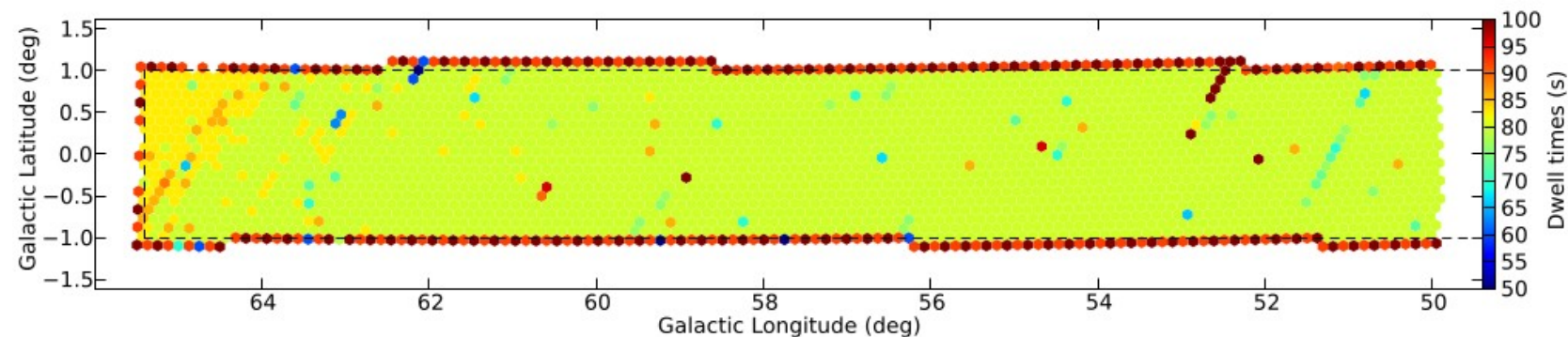
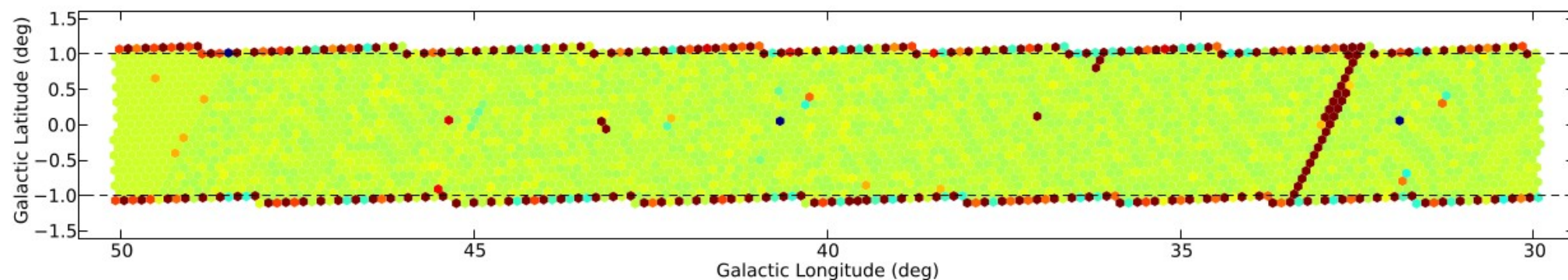
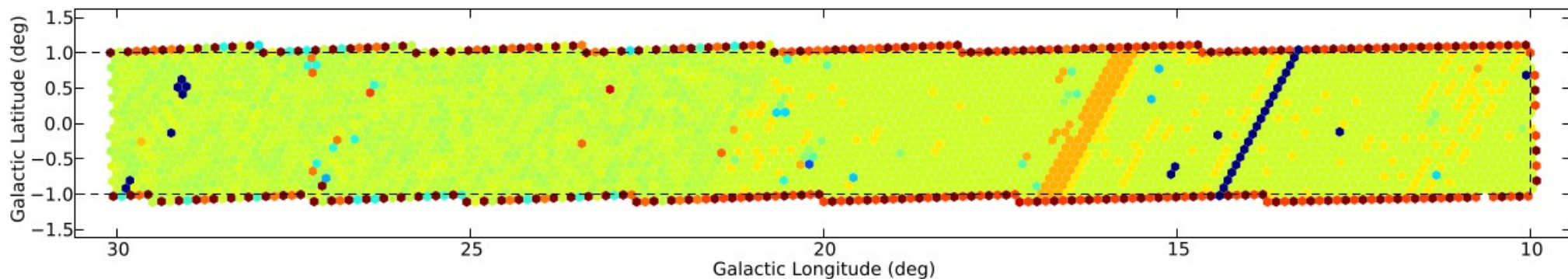


- Fairly standard data-reduction
- However, observations took place during a **rolling antenna upgrade**, control **software upgrade** and peak **electrical storm season**
- Python based pipeline was driven by ASCII configuration files and stored all metadata in a MySQL database
- Raw and intermediate data were stored on disk as *uv*- or image-format FITS files
- Database was instrumental for managing data producing **quality control diagnostic plots**



# Pipeline case study – CORNISH

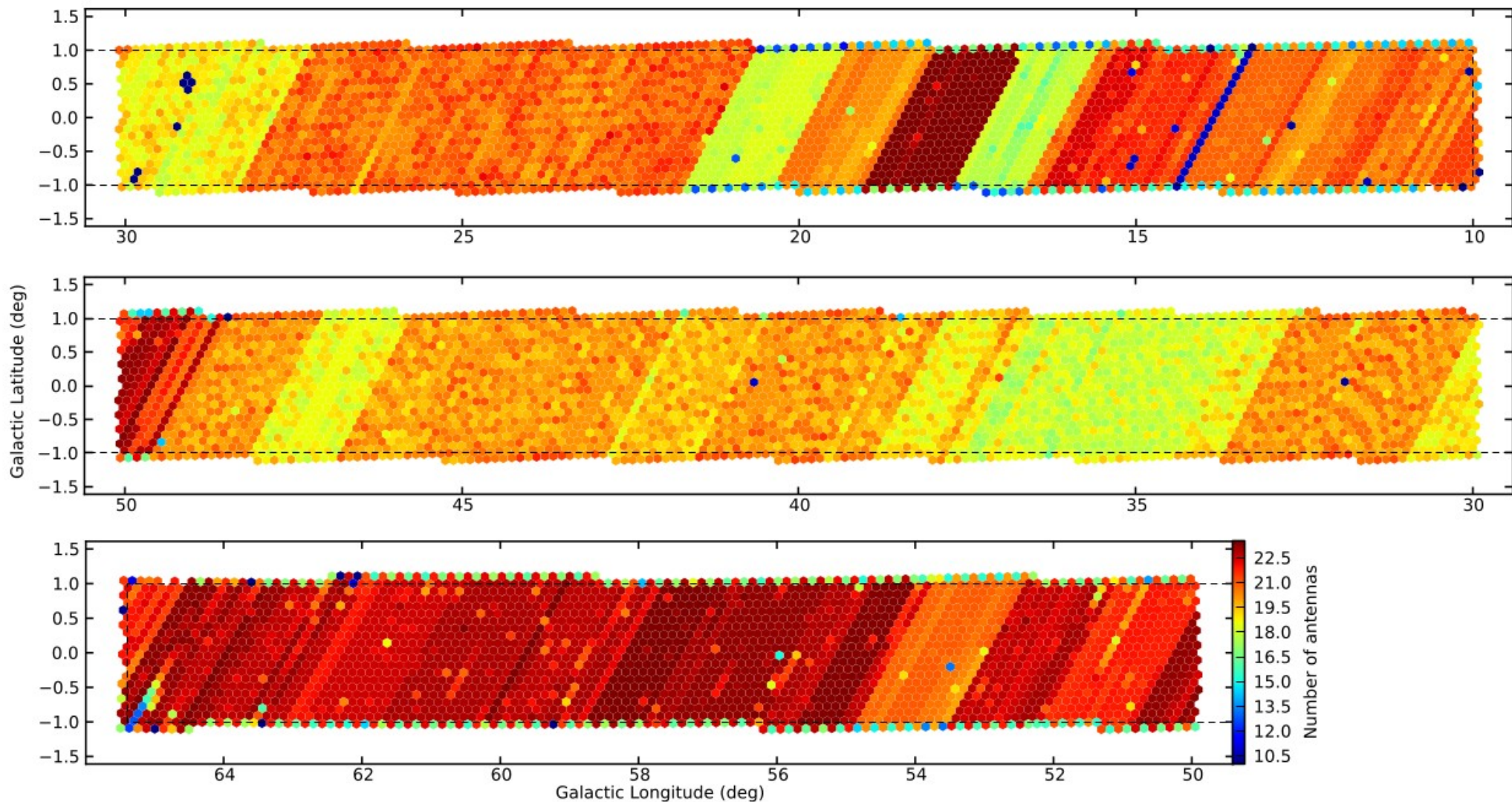
Dwell Times:



- Actual dwell times 77 – 80 seconds for most fields

# Pipeline case study – CORNISH

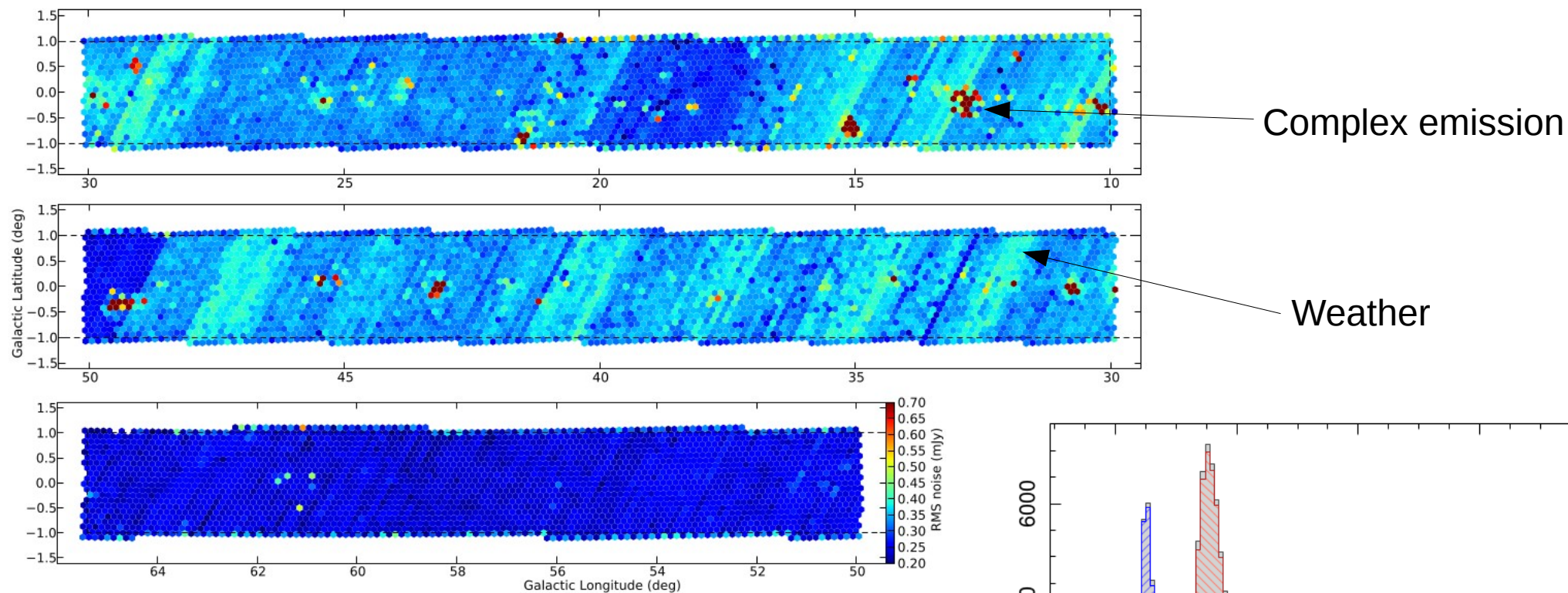
Effective number of antennas:



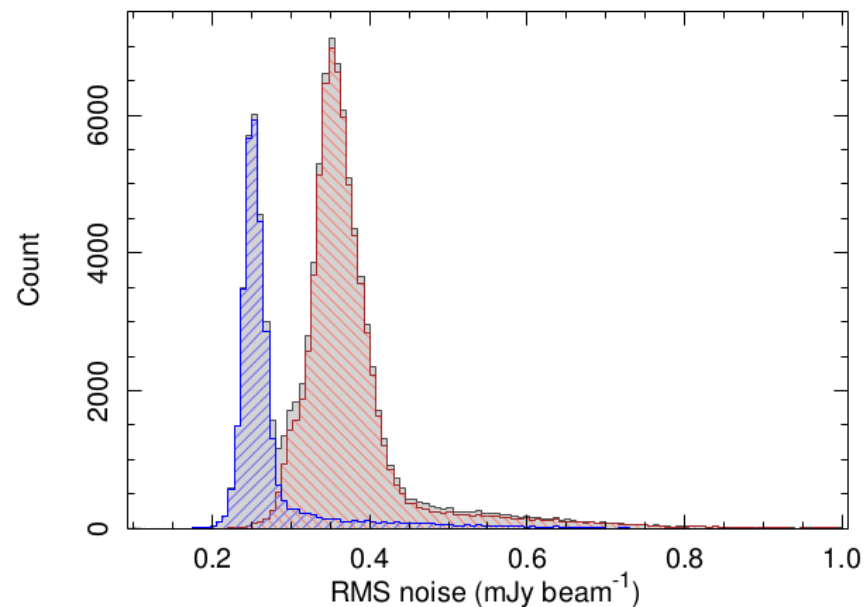
- After flagging  $> 18$  effective antennas for most fields.

# Pipeline case study – CORNISH

RMS noise per field (scale: 0.2 -0.7 mJy/beam)

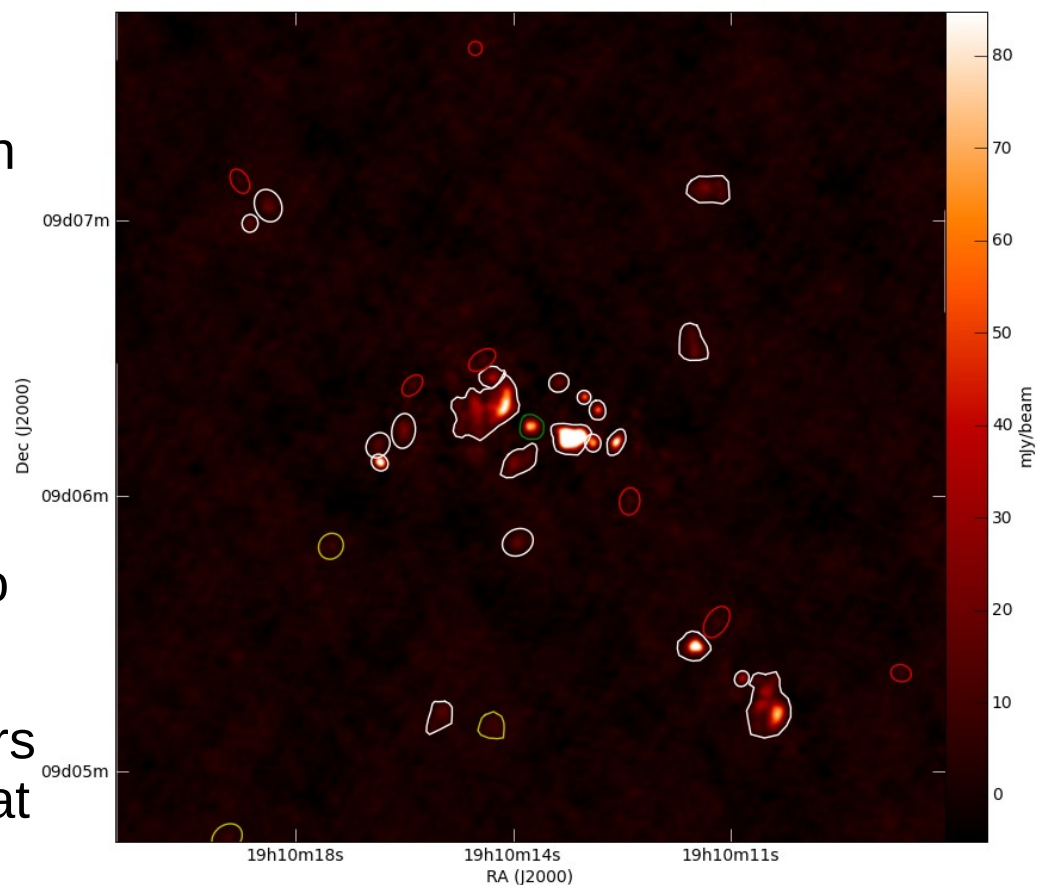


- Sensitivity split between two Epochs
  - 0.25 mJy/beam (EVLA, high-dec)
  - 0.35 mJy/beam (mixed VLA/EVLA)



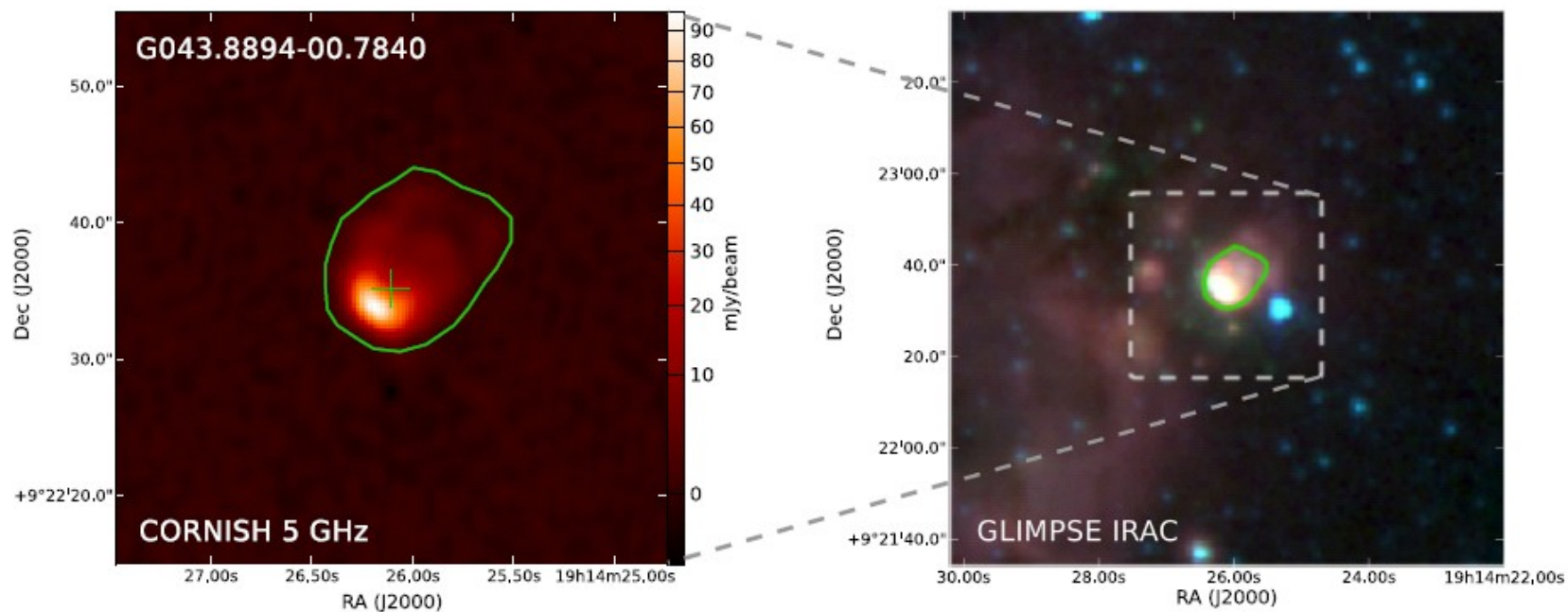
# Pipeline case study – CORNISH

- Once the data was imaged progressed onto source-finding to create a catalogue
- Characterised unresolved emission using Gaussians (*Obit Soufnd*)
- Measured extended sources using polygonal aperture photometry
- Source finders wrote directly into the database and external python scripts were used to merge the two types of catalogue
- Note 1: More modern source finders available now - Aegean and Blobcat
- Note 2: Python Shapely module very useful for manipulating polygons <http://toblerity.org/shapely/>



<https://github.com/PaulHancock/Aegean>

<http://blobcat.sourceforge.net/>

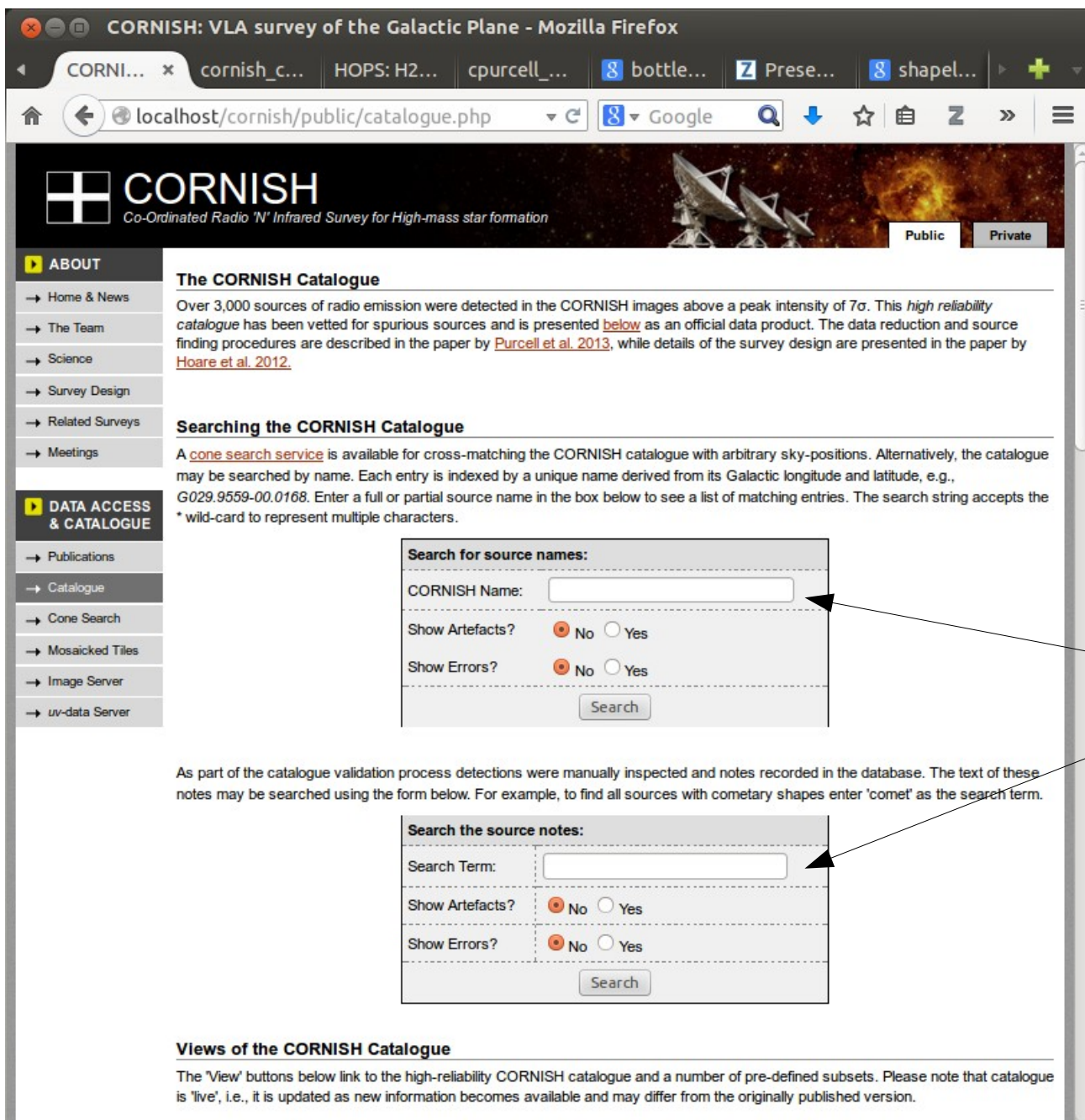


- **Co-Ordinated Radio 'N' Infrared Survey for High-mass star formation**
- Key science requires reference to the mid-IR GLIMPSE data
- However, most objects of interest are **extended** – difficult to automate
- Solution: **Eyeball carnage!**



# Pipeline case study – CORNISH

- Web interface to the image and catalogue data using Apache and Python CGI



CORNISH: VLA survey of the Galactic Plane - Mozilla Firefox

localhost/cornish/public/catalogue.php

**CORNISH**  
Co-Ordinated Radio 'N' Infrared Survey for High-mass star formation

**ABOUT**

- Home & News
- The Team
- Science
- Survey Design
- Related Surveys
- Meetings

**DATA ACCESS & CATALOGUE**

- Publications
- Catalogue
- Cone Search
- Mosaicked Tiles
- Image Server
- uv-data Server

**The CORNISH Catalogue**

Over 3,000 sources of radio emission were detected in the CORNISH images above a peak intensity of  $7\sigma$ . This *high reliability catalogue* has been vetted for spurious sources and is presented [below](#) as an official data product. The data reduction and source finding procedures are described in the paper by [Purcell et al. 2013](#), while details of the survey design are presented in the paper by [Hoare et al. 2012](#).

**Searching the CORNISH Catalogue**

A [cone search service](#) is available for cross-matching the CORNISH catalogue with arbitrary sky-positions. Alternatively, the catalogue may be searched by name. Each entry is indexed by a unique name derived from its Galactic longitude and latitude, e.g., *G029.9559-00.0168*. Enter a full or partial source name in the box below to see a list of matching entries. The search string accepts the \* wild-card to represent multiple characters.

**Search for source names:**

CORNISH Name:

Show Artefacts?  No  Yes

Show Errors?  No  Yes

As part of the catalogue validation process detections were manually inspected and notes recorded in the database. The text of these notes may be searched using the form below. For example, to find all sources with cometary shapes enter 'comet' as the search term.

**Search the source notes:**

Search Term:

Show Artefacts?  No  Yes

Show Errors?  No  Yes

**Views of the CORNISH Catalogue**

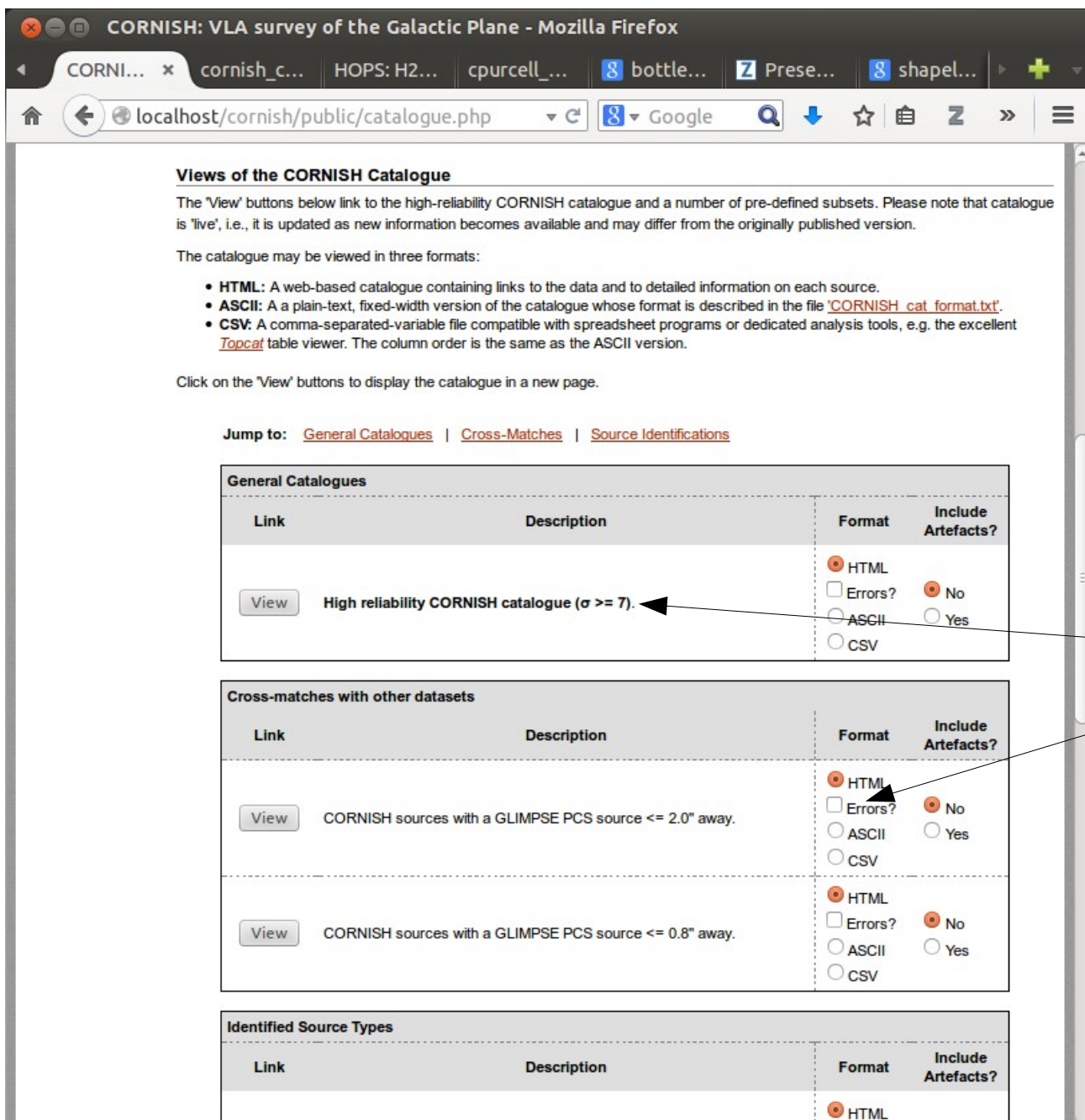
The 'View' buttons below link to the high-reliability CORNISH catalogue and a number of pre-defined subsets. Please note that catalogue is 'live', i.e., it is updated as new information becomes available and may differ from the originally published version.

Search tools:

- By partial name
- By user notes

# Pipeline case study – CORNISH

- Web interface to the image and catalogue data using Apache and Python CGI



**Views of the CORNISH Catalogue**

The 'View' buttons below link to the high-reliability CORNISH catalogue and a number of pre-defined subsets. Please note that catalogue is 'live', i.e., it is updated as new information becomes available and may differ from the originally published version.

The catalogue may be viewed in three formats:

- **HTML:** A web-based catalogue containing links to the data and to detailed information on each source.
- **ASCII:** A plain-text, fixed-width version of the catalogue whose format is described in the file '[CORNISH\\_cat\\_format.txt](#)'.
- **CSV:** A comma-separated-variable file compatible with spreadsheet programs or dedicated analysis tools, e.g. the excellent [Topcat](#) table viewer. The column order is the same as the ASCII version.

Click on the 'View' buttons to display the catalogue in a new page.

Jump to: [General Catalogues](#) | [Cross-Matches](#) | [Source Identifications](#)

Link	Description	Format	Include Artefacts?
<a href="#">View</a>	High reliability CORNISH catalogue ( $\sigma \geq 7$ ).	<input checked="" type="radio"/> HTML <input type="checkbox"/> Errors? <input type="radio"/> ASCII <input type="radio"/> CSV	<input checked="" type="radio"/> No <input type="radio"/> Yes

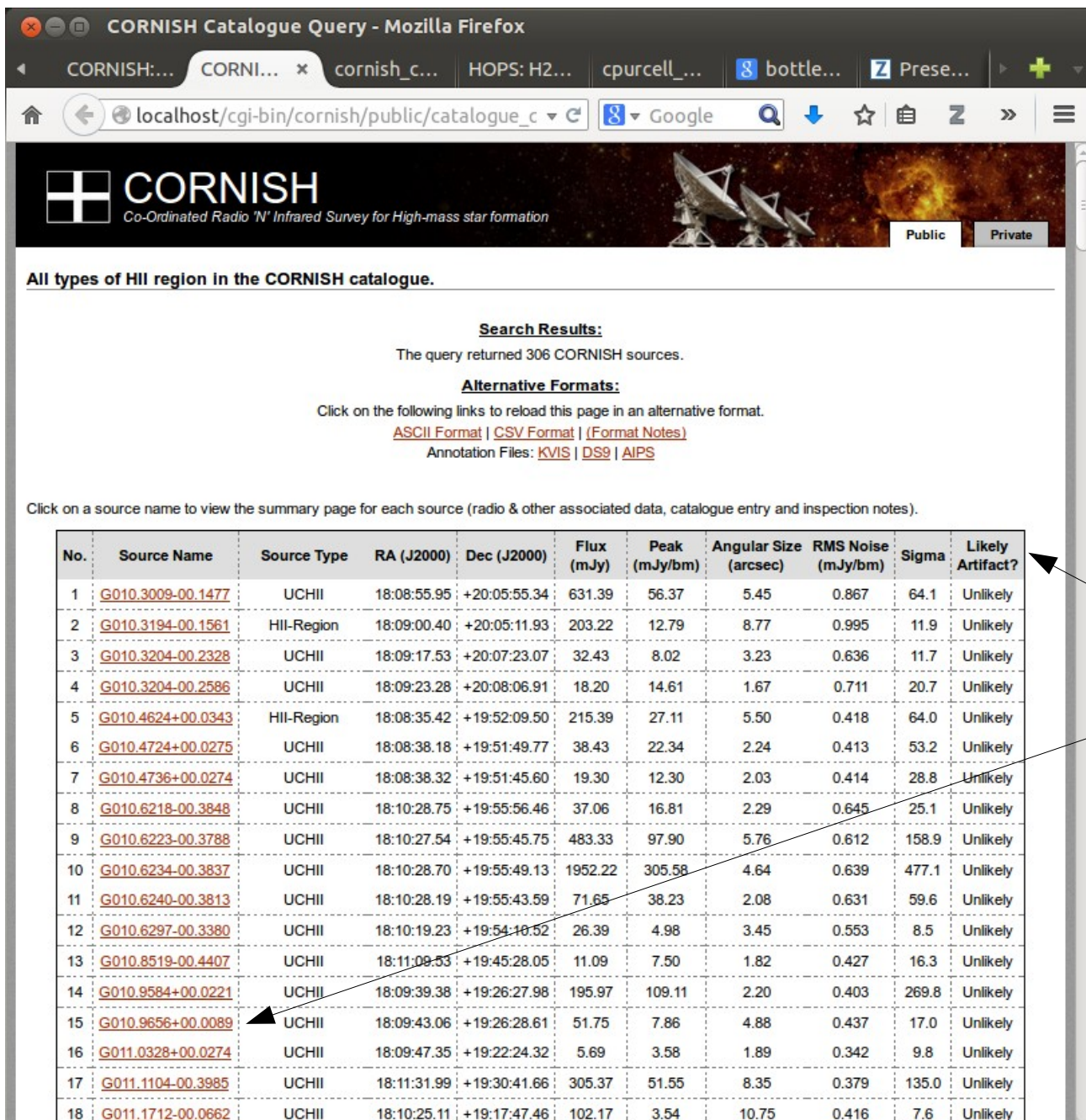
Link	Description	Format	Include Artefacts?
<a href="#">View</a>	CORNISH sources with a GLIMPSE PCS source $\leq 2.0''$ away.	<input checked="" type="radio"/> HTML <input type="checkbox"/> Errors? <input type="radio"/> ASCII <input type="radio"/> CSV	<input checked="" type="radio"/> No <input type="radio"/> Yes
<a href="#">View</a>	CORNISH sources with a GLIMPSE PCS source $\leq 0.8''$ away.	<input checked="" type="radio"/> HTML <input type="checkbox"/> Errors? <input type="radio"/> ASCII <input type="radio"/> CSV	<input checked="" type="radio"/> No <input type="radio"/> Yes

Link	Description	Format	Include Artefacts?
		<input checked="" type="radio"/> HTML	

Catalogue access

- Pre-defined SQL queries
- Links to catalogue and image server

# Pipeline case study – CORNISH



CORNISH Catalogue Query - Mozilla Firefox

localhost/cgi-bin/cornish/public/catalogue\_c

**CORNISH**  
Co-Ordinated Radio 'N' Infrared Survey for High-mass star formation

Public Private

All types of HII region in the CORNISH catalogue.

**Search Results:**  
The query returned 306 CORNISH sources.

**Alternative Formats:**  
Click on the following links to reload this page in an alternative format.  
[ASCII Format](#) | [CSV Format](#) | [\(Format Notes\)](#)  
Annotation Files: [KVIS](#) | [DSS](#) | [AIPS](#)

Click on a source name to view the summary page for each source (radio & other associated data, catalogue entry and inspection notes).

No.	Source Name	Source Type	RA (J2000)	Dec (J2000)	Flux (mJy)	Peak (mJy/bm)	Angular Size (arcsec)	RMS Noise (mJy/bm)	Sigma	Likely Artifact?
1	<a href="#">G010.3009-00.1477</a>	UCHII	18:08:55.95	+20:05:55.34	631.39	56.37	5.45	0.867	64.1	Unlikely
2	<a href="#">G010.3194-00.1561</a>	HII-Region	18:09:00.40	+20:05:11.93	203.22	12.79	8.77	0.995	11.9	Unlikely
3	<a href="#">G010.3204-00.2328</a>	UCHII	18:09:17.53	+20:07:23.07	32.43	8.02	3.23	0.636	11.7	Unlikely
4	<a href="#">G010.3204-00.2586</a>	UCHII	18:09:23.28	+20:08:06.91	18.20	14.61	1.67	0.711	20.7	Unlikely
5	<a href="#">G010.4624+00.0343</a>	HII-Region	18:08:35.42	+19:52:09.50	215.39	27.11	5.50	0.418	64.0	Unlikely
6	<a href="#">G010.4724+00.0275</a>	UCHII	18:08:38.18	+19:51:49.77	38.43	22.34	2.24	0.413	53.2	Unlikely
7	<a href="#">G010.4736+00.0274</a>	UCHII	18:08:38.32	+19:51:45.60	19.30	12.30	2.03	0.414	28.8	Unlikely
8	<a href="#">G010.6218-00.3848</a>	UCHII	18:10:28.75	+19:55:56.46	37.06	16.81	2.29	0.645	25.1	Unlikely
9	<a href="#">G010.6223-00.3788</a>	UCHII	18:10:27.54	+19:55:45.75	483.33	97.90	5.76	0.612	158.9	Unlikely
10	<a href="#">G010.6234-00.3837</a>	UCHII	18:10:28.70	+19:55:49.13	1952.22	305.58	4.64	0.639	477.1	Unlikely
11	<a href="#">G010.6240-00.3813</a>	UCHII	18:10:28.19	+19:55:43.59	71.65	38.23	2.08	0.631	59.6	Unlikely
12	<a href="#">G010.6297-00.3380</a>	UCHII	18:10:19.23	+19:54:10.52	26.39	4.98	3.45	0.553	8.5	Unlikely
13	<a href="#">G010.8519-00.4407</a>	UCHII	18:11:09.53	+19:45:28.05	11.09	7.50	1.82	0.427	16.3	Unlikely
14	<a href="#">G010.9584+00.0221</a>	UCHII	18:09:39.38	+19:26:27.98	195.97	109.11	2.20	0.403	269.8	Unlikely
15	<a href="#">G010.9656+00.0089</a>	UCHII	18:09:43.06	+19:26:28.61	51.75	7.86	4.88	0.437	17.0	Unlikely
16	<a href="#">G011.0328+00.0274</a>	UCHII	18:09:47.35	+19:22:24.32	5.69	3.58	1.89	0.342	9.8	Unlikely
17	<a href="#">G011.1104-00.3985</a>	UCHII	18:11:31.99	+19:30:41.66	305.37	51.55	8.35	0.379	135.0	Unlikely
18	<a href="#">G011.1712-00.0662</a>	UCHII	18:10:25.11	+19:17:47.46	102.17	3.54	10.75	0.416	7.6	Unlikely

- Web interface to the image and catalogue data using Apache and Python CGI

• Catalogue table filtered by SourceType HII region

• Links to source-summary and cross-match page



# Pipeline case study – CORNISH

CORNISH Source Summary for G010.6234-00.3837 - Mozilla Firefox

localhost/cgi-bin/cornish/public/summary\_si

18h10m34s 18h10m30s 18h10m27s 18h10m24s  
RA (J2000)

5 GHz radio image (FITS FILE)

**Annotations:** Green = current source, White = high-reliability sources ( $\geq 7\sigma$ ), Yellow = 6 to  $7\sigma$  sources, Red = 5 to  $6\sigma$  sources.  
Click within 10" of any source with  $\sigma > 6$  to open its data summary page in a new window.

**CORNISH Catalogue Entry**

Source Name	Source Type	RA (J2000)	Dec (J2000)	Flux (mJy)	Peak (mJy/bm)	RMS Noise (mJy/bm)	Angular Size (arcsec)	Sigma	Likely Artefact?
G010.6234-00.3837	UCHII	18:10:28.70	+19:55:49.13	1952.22	305.58	0.64	4.64	477.1	Unlikely
<b>Uncertainties:</b>	<b>Confidence:</b> Good	0.104"	0.104"	176.18	27.20		0.002		

Click on the "likely artefact" edit the source artefact status or the "source type" to classify the source type.

**CORNISH Catalogue Flags:**

In cluster? <sup>a</sup>	Yes	Near survey edge? <sup>b</sup>	No	High-noise region? <sup>c</sup>	Yes
High 5 $\sigma$ density? <sup>d</sup>	Yes	Near bright source? <sup>e</sup>	No	Smooth Weighting? <sup>f</sup>	Yes
Overlap 5 $\sigma$ ? <sup>g</sup>	No	Overlap 7 $\sigma$ ? <sup>h</sup>	Yes	Nearest 5 $\sigma$ source	G010.6324-00.3765 (d = 39.9")
Nearest 7 $\sigma$ source	G010.6218-00.3848 (d = 8.5")				

**Notes:**

<sup>a</sup> Source is within 12" of another source.  
<sup>b</sup> Source is within 2' of the survey edge.  
<sup>c</sup> Source is in a region with a RMS-noise greater than 0.45 mJy/beam  
<sup>d</sup> Region contains greater than 7 detections of 5 to 7-sigma sources.  
<sup>e</sup> Source is within 3' of another source which has a peak flux density greater than 500 mJy/beam.  
<sup>f</sup> Source is within the half-power primary beam-width of a field which was imaged using the smoothed weighting scheme.  
<sup>g</sup> One or more 5 $\sigma$  to 7 $\sigma$  source overlaps with the current source.  
<sup>h</sup> One or more 7 $\sigma$ + sources overlaps with the current source.

The large-scale CORNISH environment

Density map of sources above 5 $\sigma$ .

- Web interface to the image and catalogue data using Apache and Python CGI

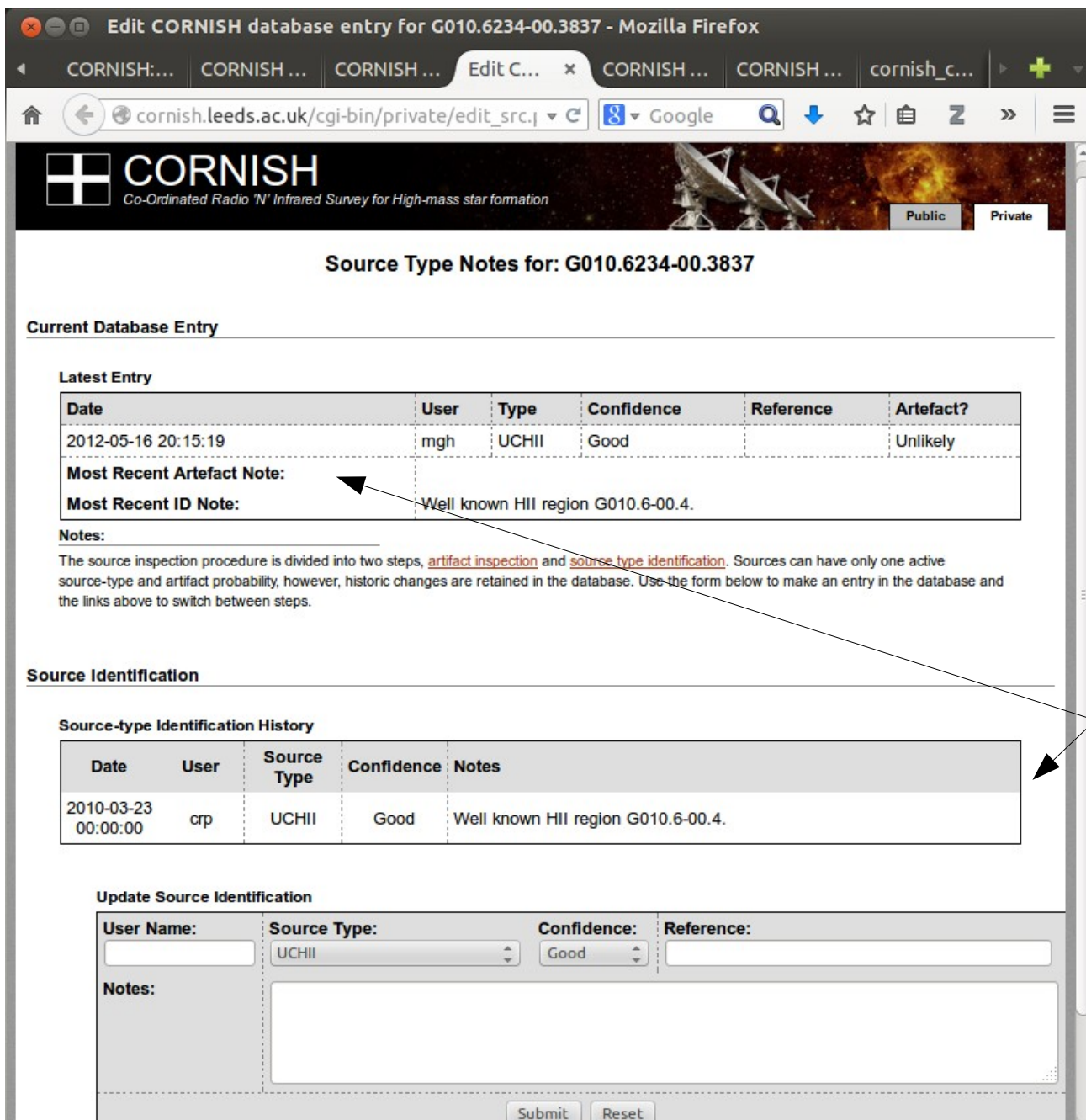
- Source summary page
- Details of source pulled from the database
- Links to edit pages



Made using APLpy

# Pipeline case study – CORNISH

- Web interface to the image and catalogue data using Apache and Python CGI



Browser: Mozilla Firefox  
 URL: cornish.leeds.ac.uk/cgi-bin/private/edit\_src.i

**CORNISH**  
 Co-Ordinated Radio 'N' Infrared Survey for High-mass star formation

Public Private

**Source Type Notes for: G010.6234-00.3837**

**Current Database Entry**

**Latest Entry**

Date	User	Type	Confidence	Reference	Artefact?
2012-05-16 20:15:19	mgh	UCHII	Good		Unlikely

**Most Recent Artefact Note:**

**Most Recent ID Note:** Well known HII region G010.6-00.4.

**Notes:**  
 The source inspection procedure is divided into two steps, [artifact inspection](#) and [source type identification](#). Sources can have only one active source-type and artifact probability, however, historic changes are retained in the database. Use the form below to make an entry in the database and the links above to switch between steps.

**Source Identification**

**Source-type Identification History**

Date	User	Source Type	Confidence	Notes
2010-03-23 00:00:00	crp	UCHII	Good	Well known HII region G010.6-00.4.

**Update Source Identification**

User Name:

Source Type: UCHII

Confidence: Good

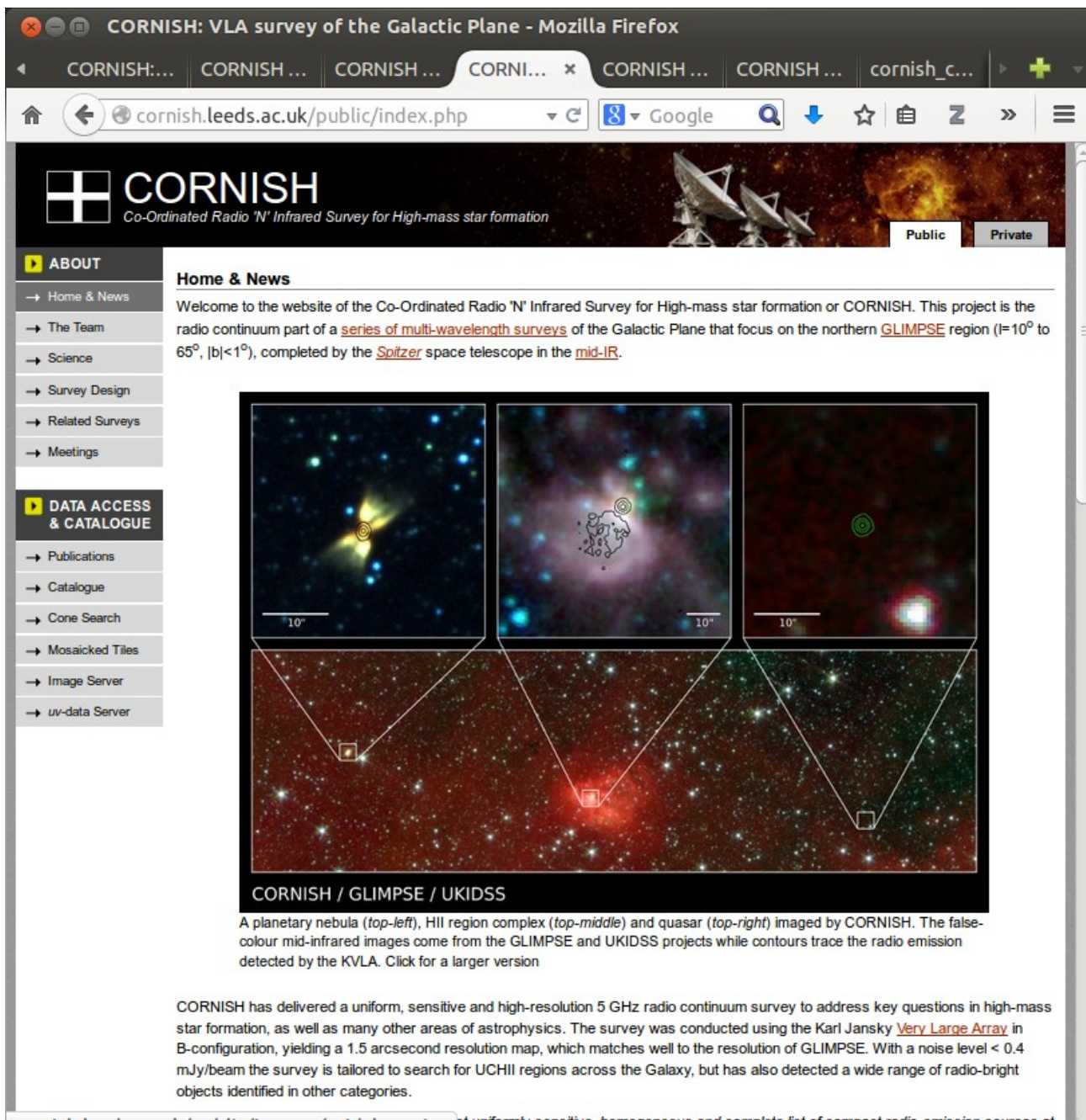
Reference:

Notes:

Submit Reset

- Form allowing editing of identified source type and notes in the DB.
- Tracks history and user

# Pipeline case study – CORNISH



CORNISH: VLA survey of the Galactic Plane - Mozilla Firefox

cornish.leeds.ac.uk/public/index.php

**CORNISH**  
Co-Ordinated Radio 'N' Infrared Survey for High-mass star formation

Public Private

**ABOUT**

- Home & News
- The Team
- Science
- Survey Design
- Related Surveys
- Meetings

**DATA ACCESS & CATALOGUE**

- Publications
- Catalogue
- Cone Search
- Mosaicked Tiles
- Image Server
- uv-data Server

**Home & News**

Welcome to the website of the Co-Ordinated Radio 'N' Infrared Survey for High-mass star formation or CORNISH. This project is the radio continuum part of a [series of multi-wavelength surveys](#) of the Galactic Plane that focus on the northern [GLIMPSE](#) region ( $l=10^\circ$  to  $65^\circ$ ,  $|b|<1^\circ$ ), completed by the [Spitzer](#) space telescope in the [mid-IR](#).

CORNISH / GLIMPSE / UKIDSS

A planetary nebula (top-left), HII region complex (top-middle) and quasar (top-right) imaged by CORNISH. The false-colour mid-infrared images come from the GLIMPSE and UKIDSS projects while contours trace the radio emission detected by the VLBA. Click for a larger version

CORNISH has delivered a uniform, sensitive and high-resolution 5 GHz radio continuum survey to address key questions in high-mass star formation, as well as many other areas of astrophysics. The survey was conducted using the Karl Jansky [Very Large Array](#) in B-configuration, yielding a 1.5 arcsecond resolution map, which matches well to the resolution of GLIMPSE. With a noise level < 0.4 mJy/beam the survey is tailored to search for UCHII regions across the Galaxy, but has also detected a wide range of radio-bright objects identified in other categories.

- Live web page at:  
<http://cornish.leeds.ac.uk>
- Catalogue server
- uv-data server
- Image cutout server and clickable maps
- Batch Image server using uploaded catalogue
- Cross-matching service
- All done in Python

# Summary and notes on the future

- Take-away message:
  - Python can stitch practically *everything* together and **make your life easier**
  - Huge range of well-tested libraries (although don't treat as a black-box)
  - Lends itself well collaborating on the web and publishing data
  - If you are short on time and just want to learn **one tool – learn Python**
- Caveats:
  - Not always the best tool for the job
  - For very large web projects javascript browser based plotting will be superior

Thanks for listening!

**Links on next slide** 

- NumPy RecArrays:
  - <http://docs.scipy.org/doc/numpy-1.8.1/reference/generated/numpy.recarray.html>
  - <http://docs.scipy.org/doc/numpy/user/basics.rec.html>
- SQL & databases:
  - <https://docs.python.org/2/library/sqlite3.html>
  - <http://zetcode.com/db/sqlitepythontutorial/>
  - <https://www.sqlite.org/lang.html>
  - <http://mysql-python.sourceforge.net/MySQLdb.html>
  - <http://zetcode.com/db/mysqlpython/>
- Regular expressions:
  - <https://docs.python.org/2/howto/regex.html>
  - [http://www.tutorialspoint.com/python/python\\_reg\\_expressions.htm](http://www.tutorialspoint.com/python/python_reg_expressions.htm)
- Plotting:
  - <http://aplpy.github.io/>
  - <http://matplotlib.org/> And also <http://plplot.sourceforge.net/> (for PGPLOT users)