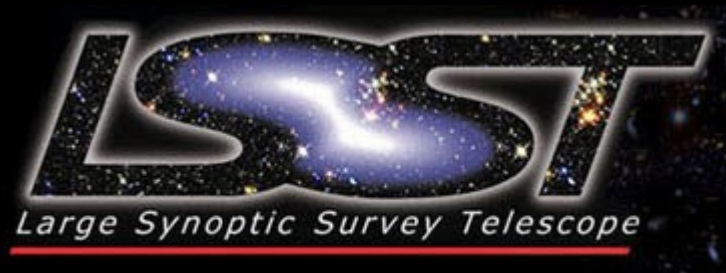


AstroML: Machine Learning for Astronomy

CIDU: 24 Oct 2012

Jake Vanderplas
Andrew Connolly
Zeljko Ivezić
Alex Gray

Python is becoming a new standard tool in Astronomy, and will remain important for the foreseeable future



Machine Learning / Statistical Data Analysis tasks in Astronomy:

- Photometric Redshifts (Regression)
- Source Classification
- Dimensionality Reduction / Visualization
- Clustering
- N-point Statistics
- Period Finding
- Transient & Outlier Detection
- Density Estimation
- Matched Filtering
- Source Extraction
- Cross-matching
- ...

Machine Learning / Statistical Data Analysis tasks in Astronomy:

- Photometric Redshifts (Regression)
- Source Classification
- Dimensionality Reduction / Visualization
- Clustering
- N-point Statistics
- Period Finding
- Transient & Outlier Detection
- Density Estimation
- Matched Filtering
- Source Extraction
- Cross-matching
- ...

Every astronomer needs these sorts of tools, and existing Python packages provide an easy interface to many powerful algorithms.

Statistics, Data Mining and Machine Learning in Astronomy

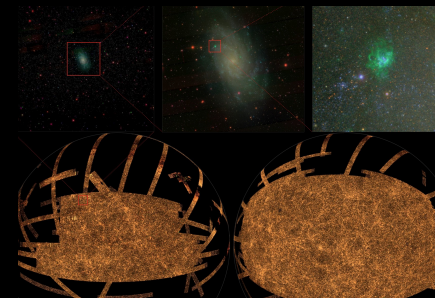
Zeljko Ivezic, Andrew Connolly,
Jacob Vanderplas, Alex Gray

Princeton University Press, 2013

- Complete *Practical* guide to statistical analysis, data exploration, and machine learning
- Example-driven approach, using real data (SDSS, LIGO, LINEAR, WMAP, and others)
- All book figures and examples generated in python (matplotlib), with code available online – for free!
- Supporting python package: *astroML*
- Makes use of *numpy*, *scipy*, *matplotlib*, *scikit-learn*, *pymc*, *healpy*, and others where possible: limited code duplication

Statistics, Data Mining,
and Machine Learning
in Astronomy

Zeljko Ivezic, Andrew Connolly,
Jacob Vanderplas, Alex Gray



AstroML: Python Machine Learning for Astronomy

astroML: Python Datamining for Astronomy — astroML 0.1 documentation - Mozilla Firefox

File Edit View History Bookmarks Tools Help

astroML: Python Datamining for... +

astroml.github.com

Most Visited News ADS OneBusAway 120 TWC Seattle Weather NASA ADS

astroML Home User Guide Book Figures Examples Plots

Google™ Custom Search

AstroML: Machine Learning and Data Mining for Astronomy

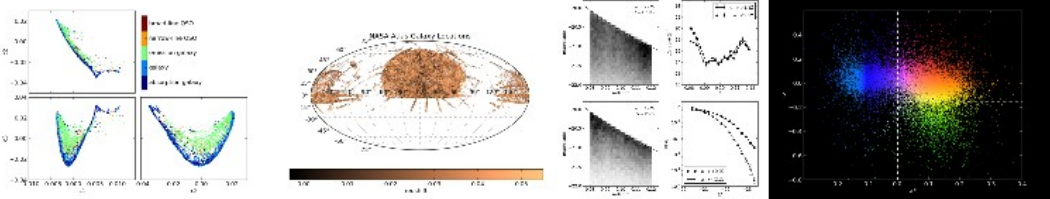
Headline

October 2012: astroML 0.1 has been released! Get the source on Github

Links

astroML Mailing List

GitHub Issue Tracker.



AstroML is a Python module for machine learning and data mining built on numpy, scipy, scikit-learn, and matplotlib, and distributed under the 3-clause BSD license. It contains a growing library of statistical and machine learning routines for analyzing astronomical data in python, loaders for several open astronomical datasets, and a large suite of examples of analyzing and visualizing astronomical datasets.

The goal of astroML is to provide a community repository for fast Python implementations of common tools and routines used for statistical data analysis in astronomy and astrophysics, to provide a uniform and easy-to-use interface to freely available astronomical datasets. We hope this package will be useful to researchers and students of astronomy. The project was started in 2012 to accompany the book **Statistics**,

Download

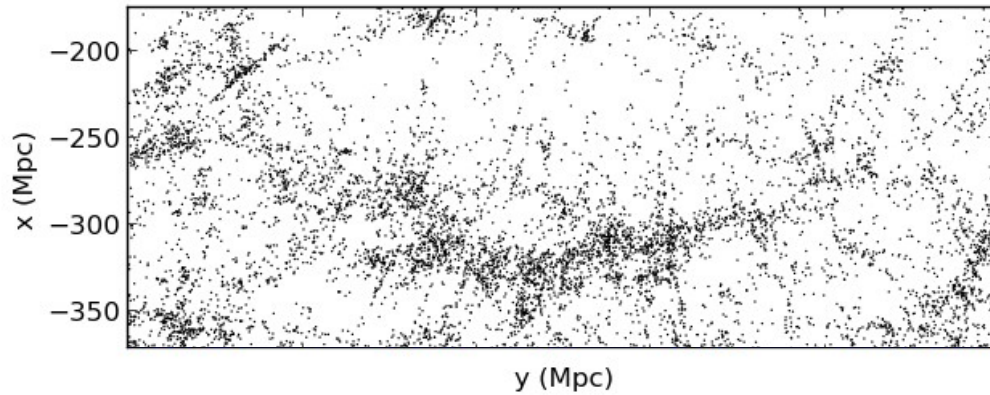
- Source code: [github](#)
- Source tarball: [astroML_0.1.tgz](#)

Statistics, Data Mining, and Machine Learning in Astronomy

Click to view month calendar

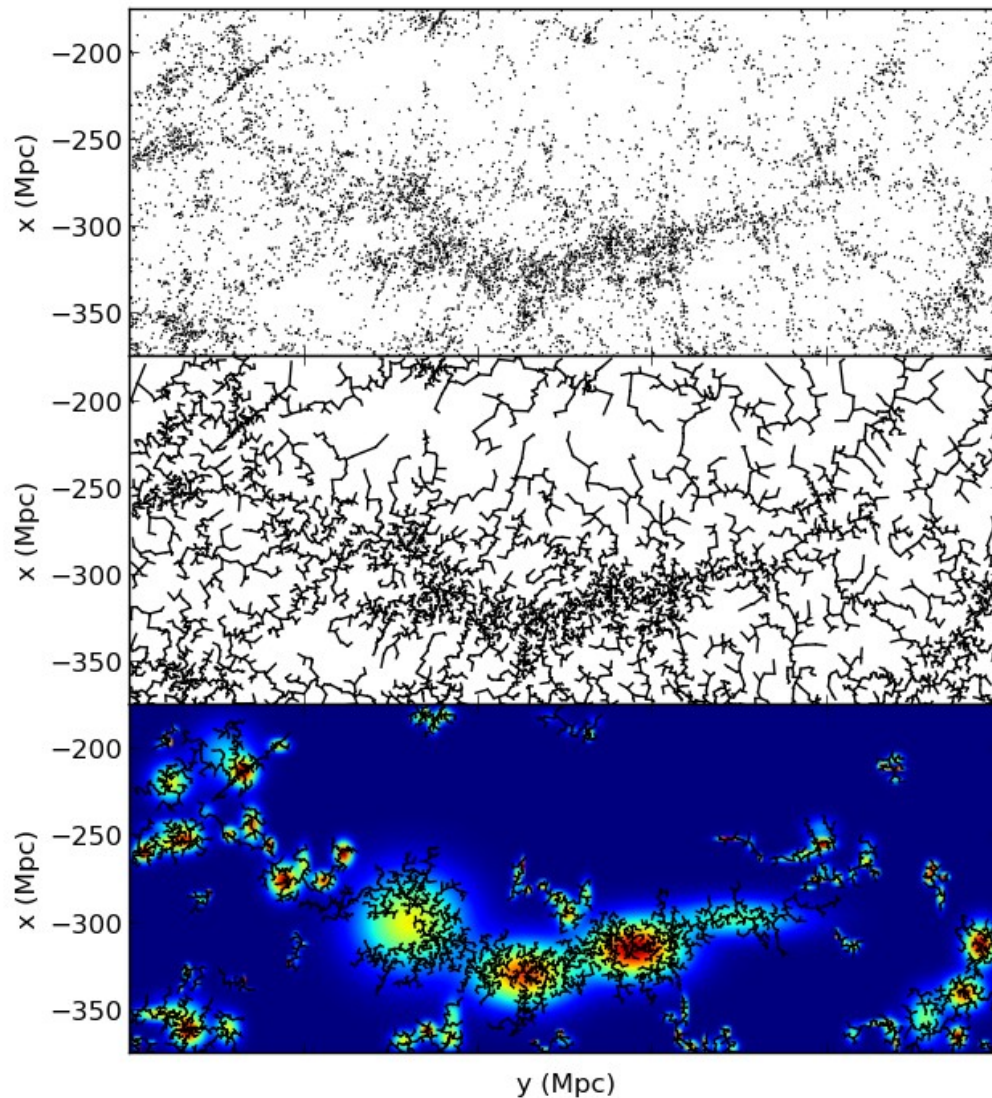
<http://astroML.github.com>

Clustering and Density Estimation: SDSS Great Wall



Projected Galaxy Locations

Clustering and Density Estimation: SDSS Great Wall

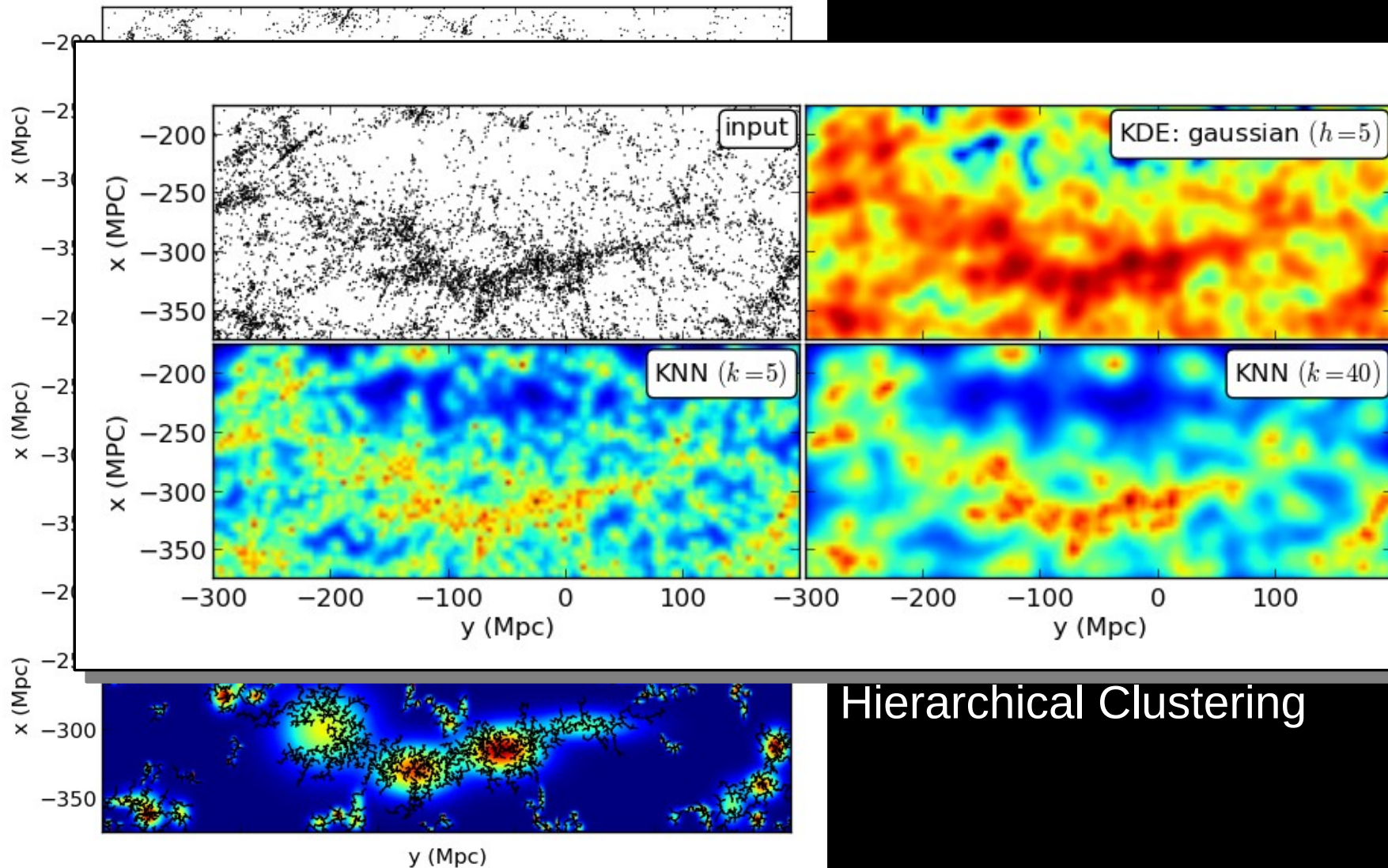


Projected Galaxy Locations

Minimum Spanning Tree

Hierarchical Clustering

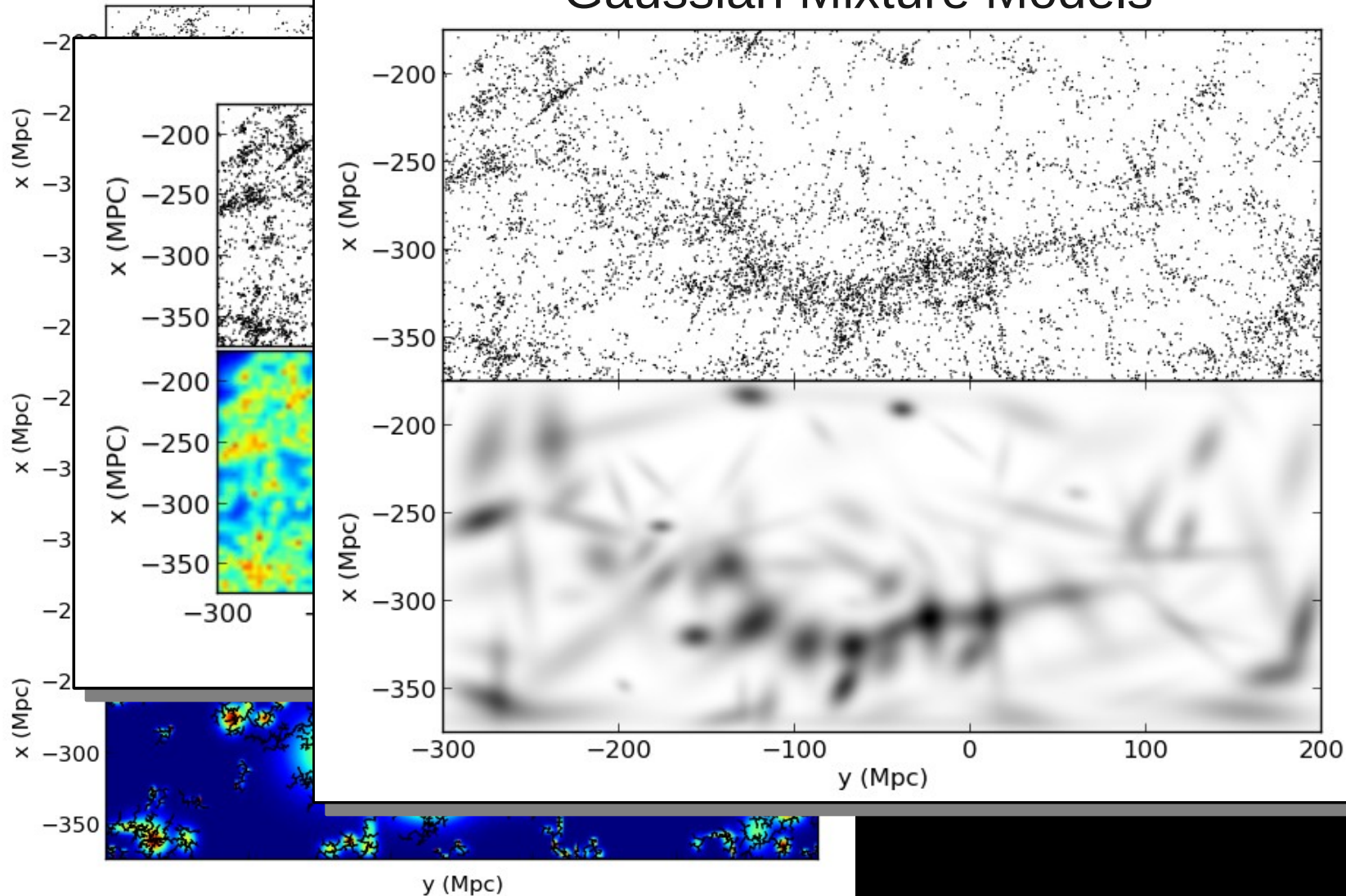
Clustering and Density Estimation: SDSS Great Wall



Hierarchical Clustering

Clustering and Density Estimation: SDSS Great Wall

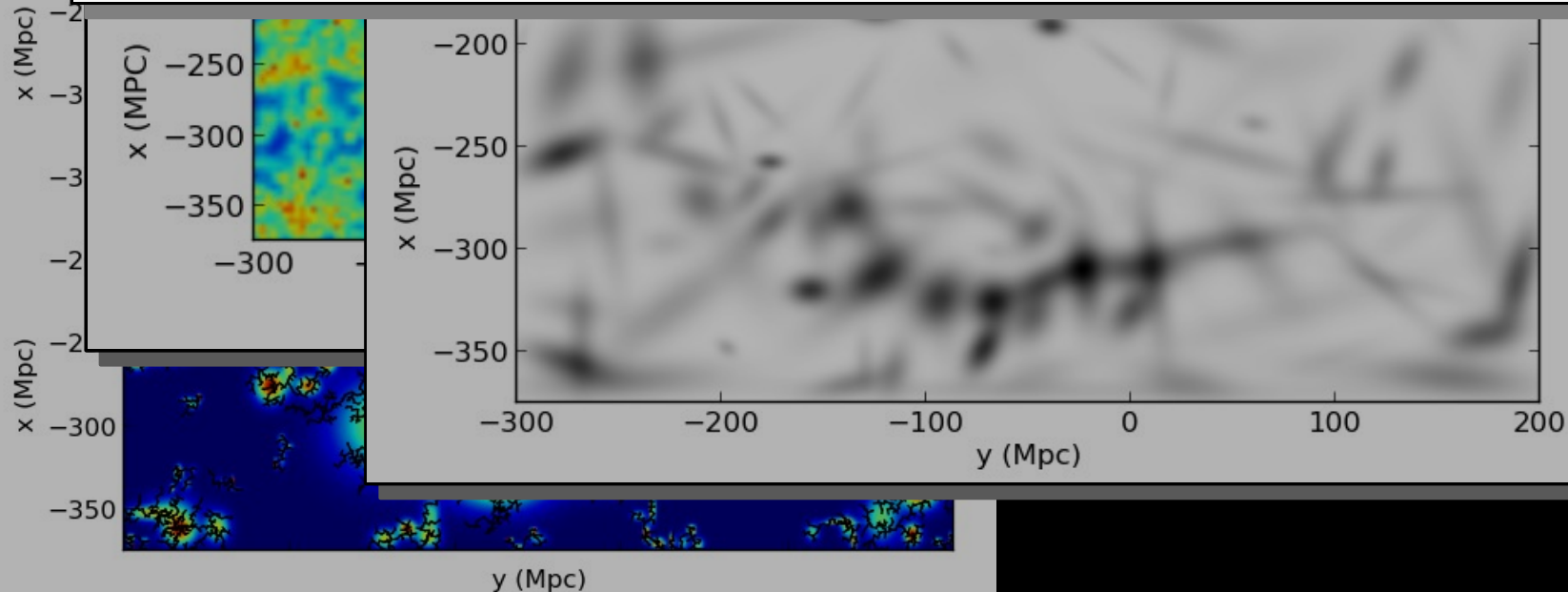
Gaussian Mixture Models



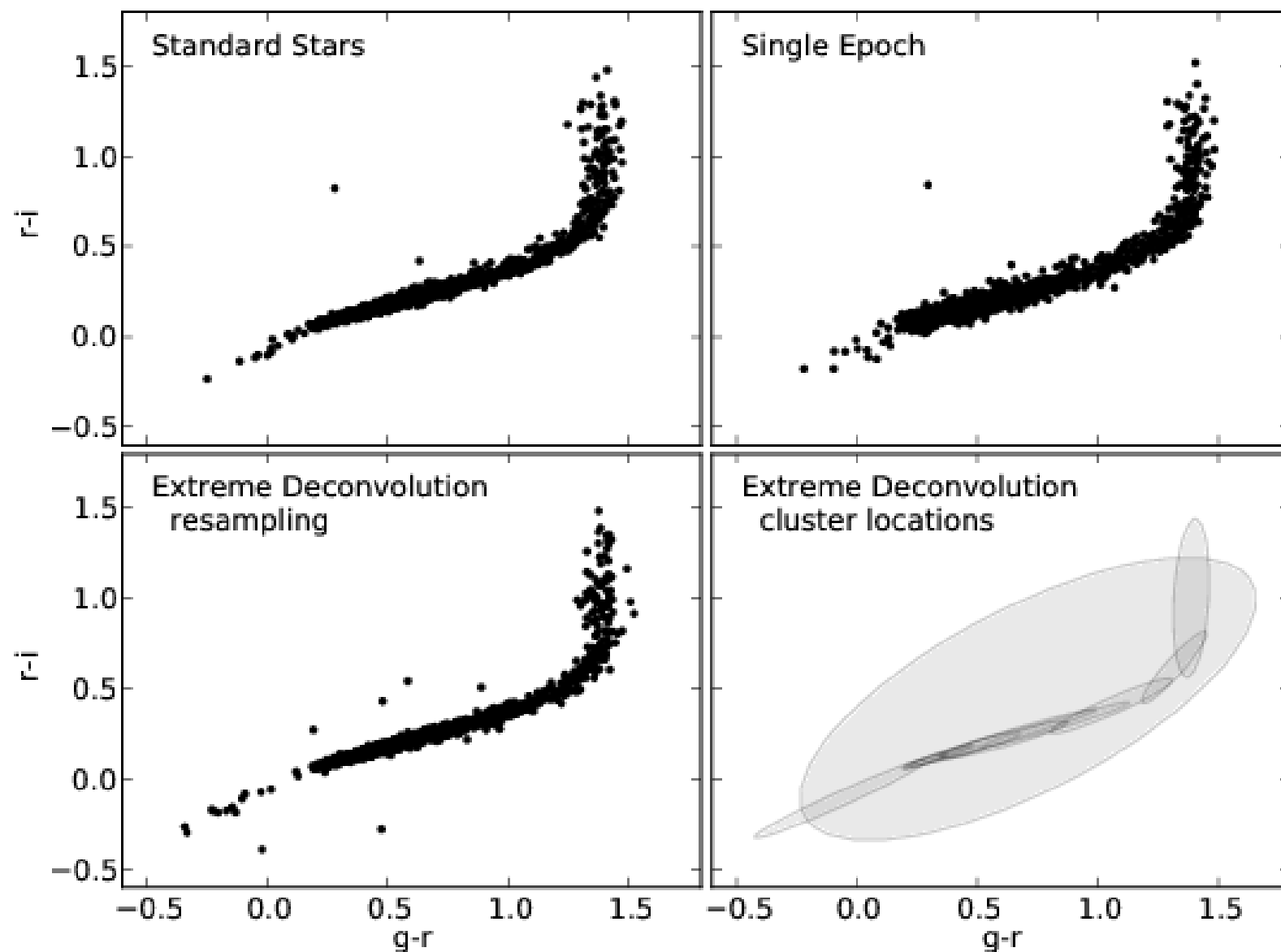
Clustering and Density Estimation: SDSS Great Wall

Gaussian Mixture Models

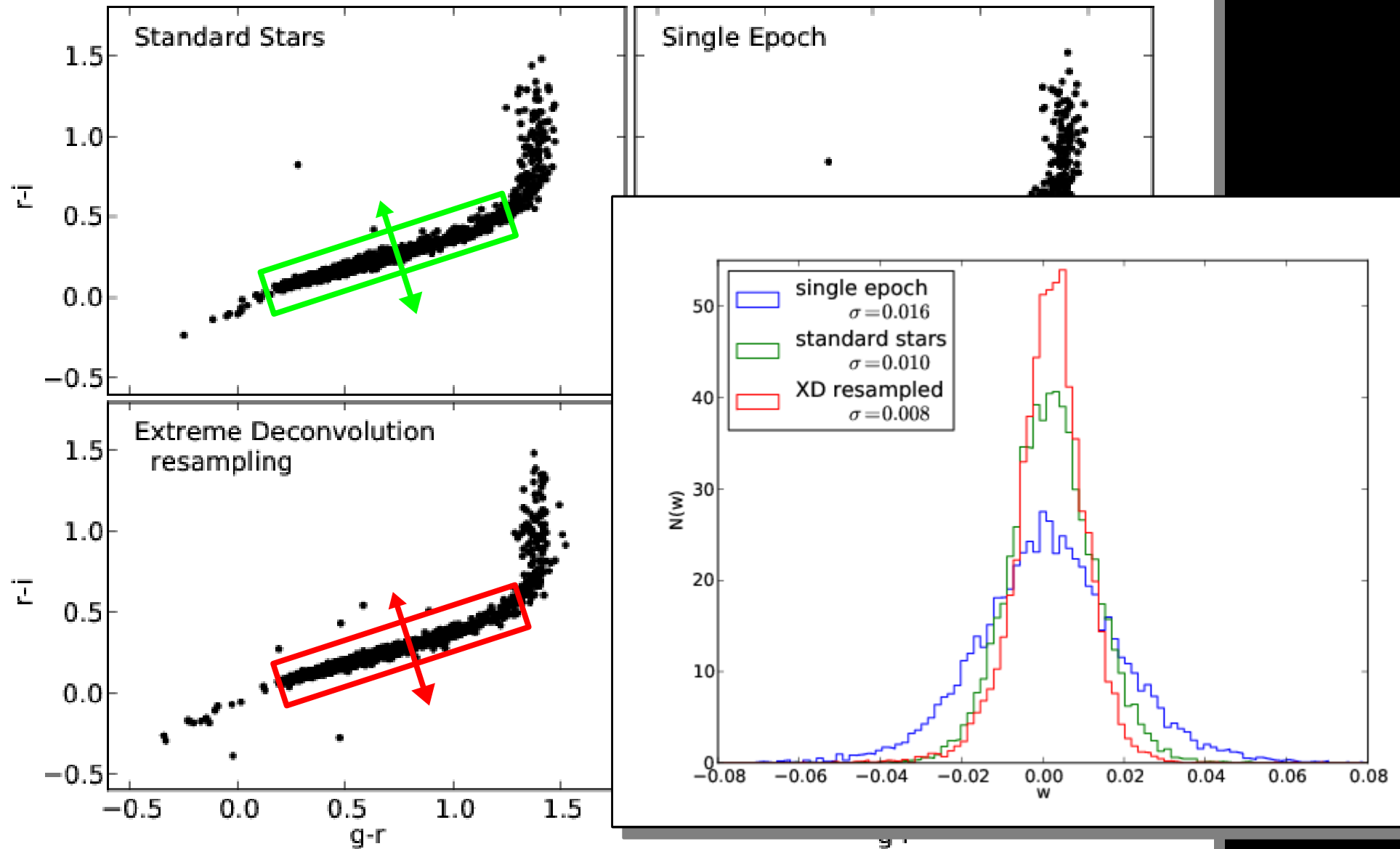
```
from astroML.datasets import fetch_great_wall  
  
from astroML.density_estimation import KDE, KNeighborsDensity  
  
X = fetch_great_wall() # data downloaded and cached automatically  
  
density = KNeighborsDensity(n_neighbors=10).fit(X) # 10 nearest neighbors  
  
density = KDE(metric='gaussian').fit(X) # KDE with gaussian kernel
```



“Extreme Deconvolution” (GMM + errors): SDSS main sequence



“Extreme Deconvolution” (GMM + errors): SDSS main sequence



“Extreme Deconvolution” (GMM + errors): SDSS main sequence



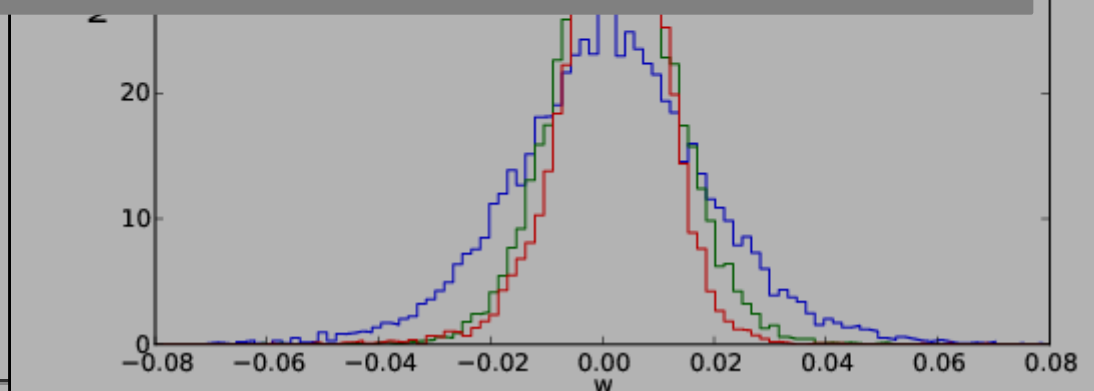
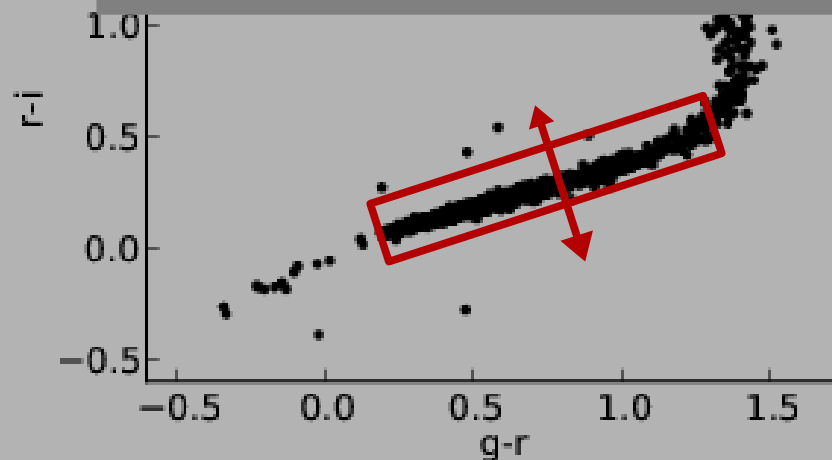
```
from astroML.datasets import fetch_sdss_S82standards
from astroML.density_estimation import XDGMM

data = fetch_sdss_S82standards() # SDSS stripe 82 standard stars

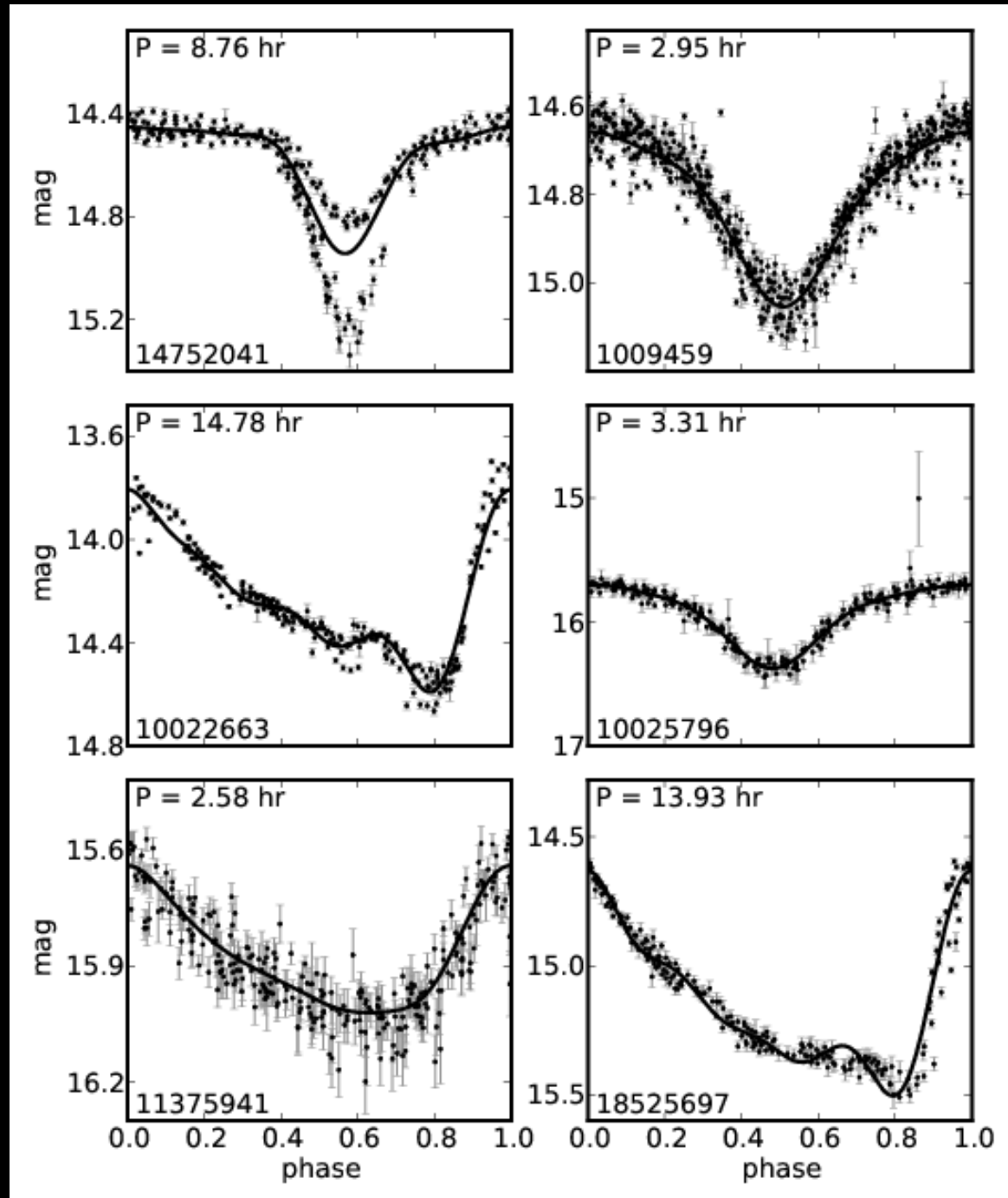
X = np.vstack([data['mmed_u'], data['mmed_g']]).T # u and g magnitudes
Xerr = np.vstack([data['mrms_u'], data['mrms_g']]).T # u and g errors

model = XDGMM(n_components=10).fit(X, Xerr) # fit the XD model

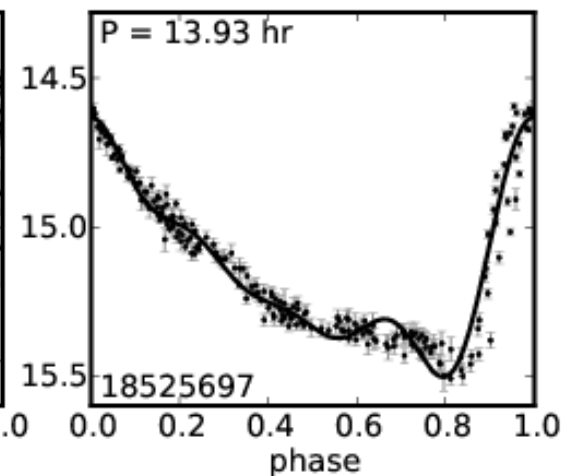
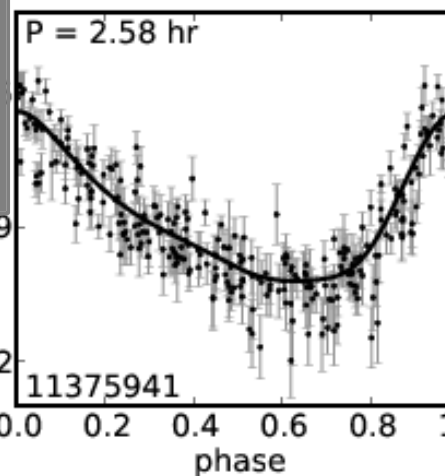
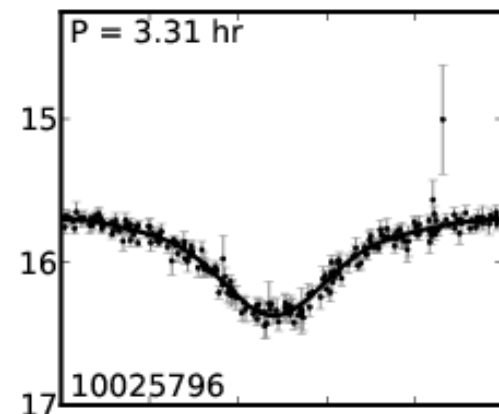
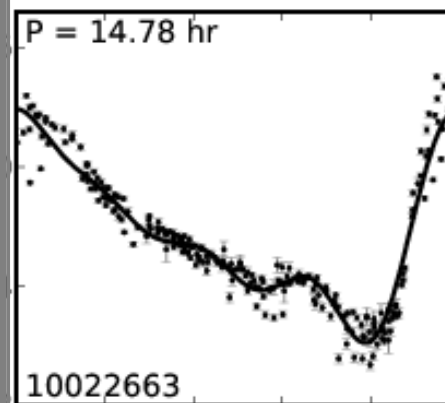
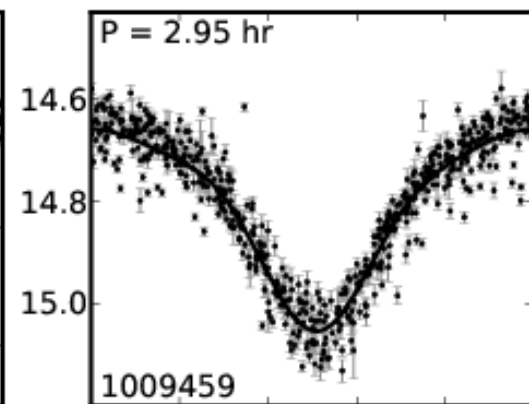
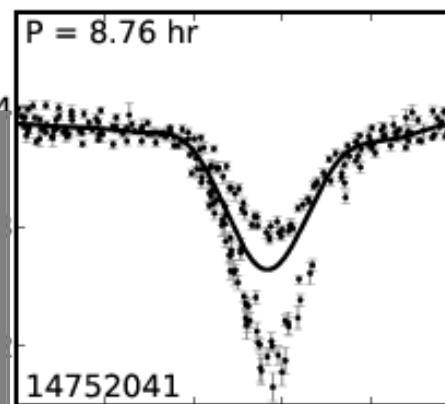
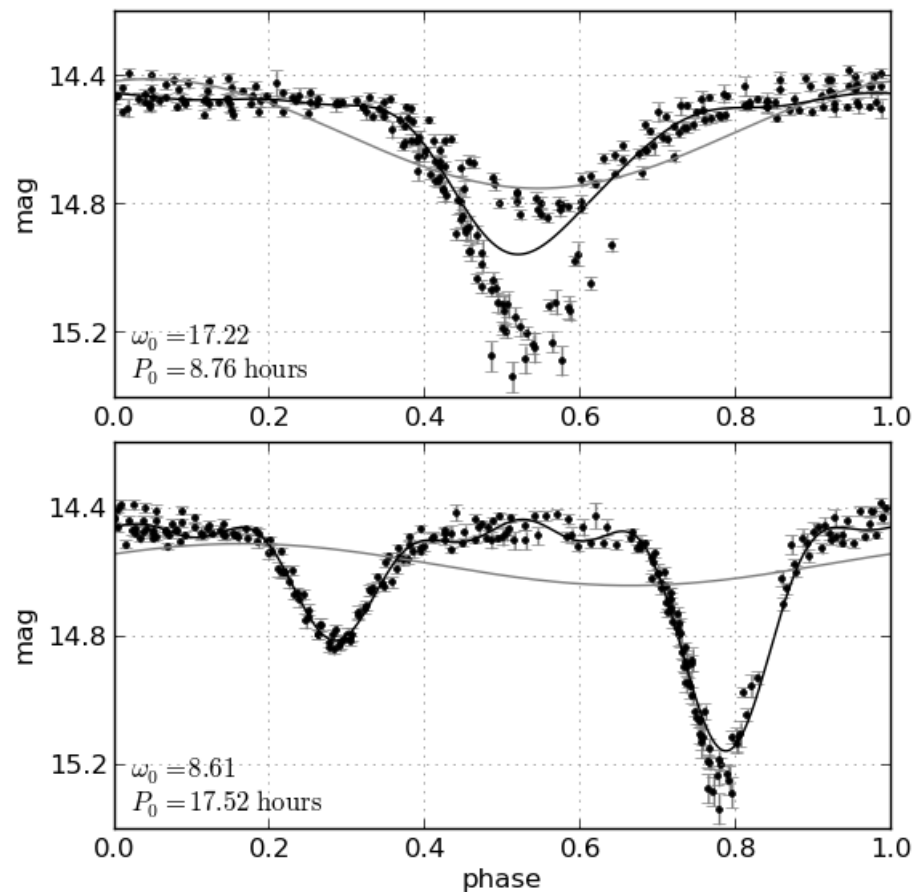
density = model.sample(10000) # Sample 10000 deconvolved points
```



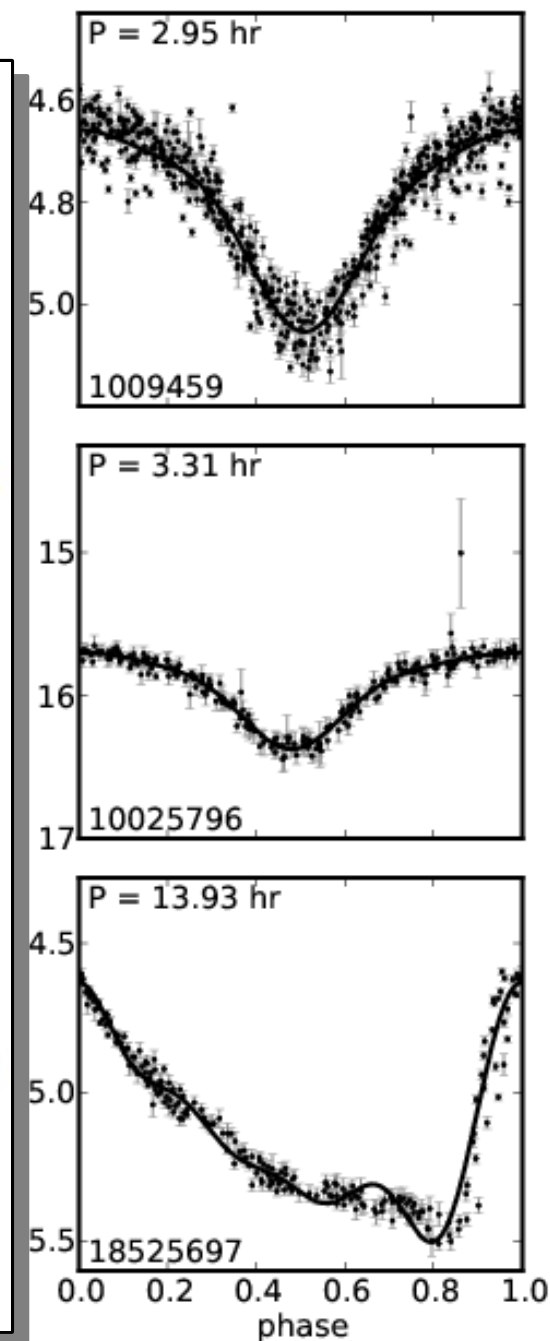
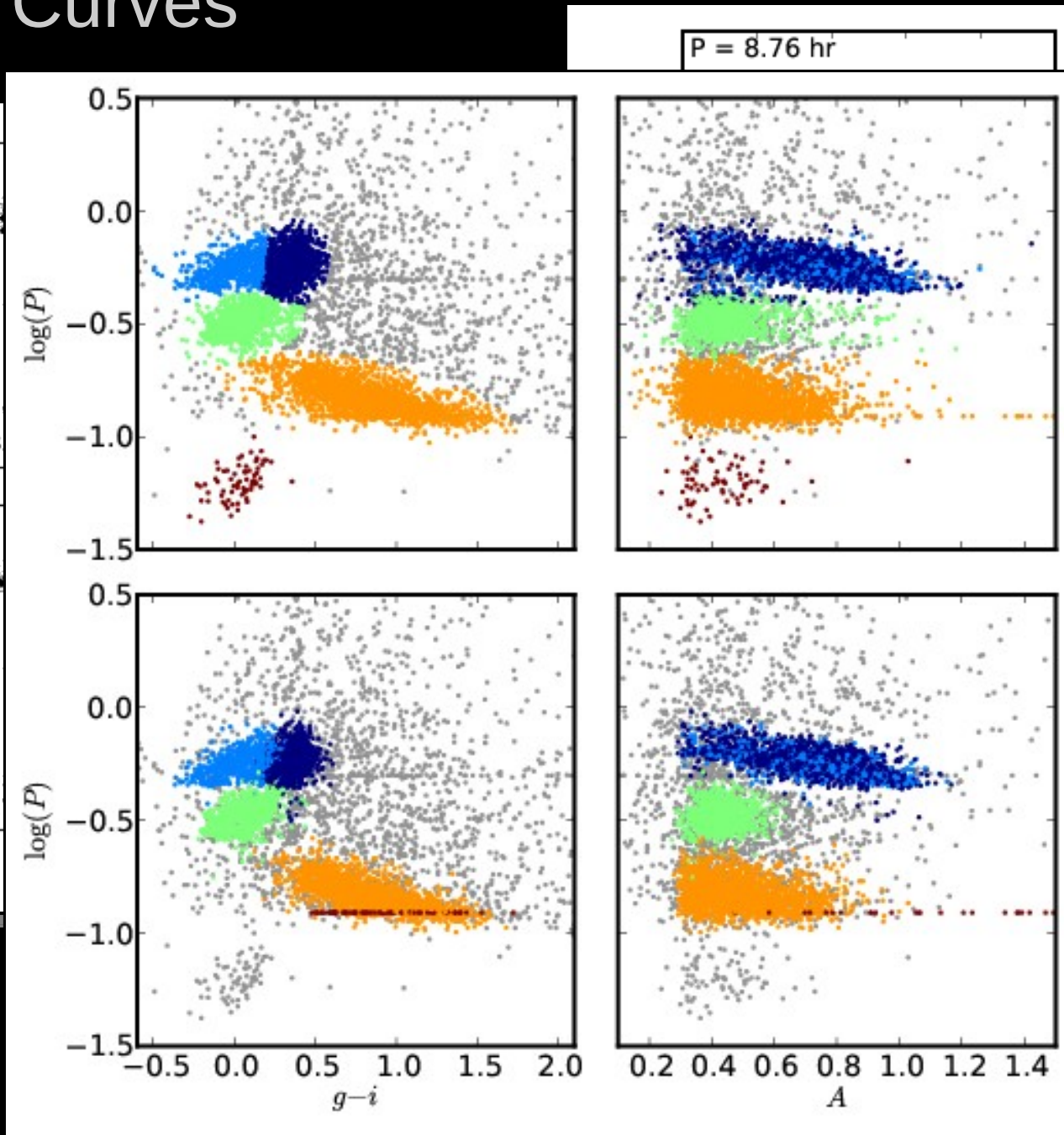
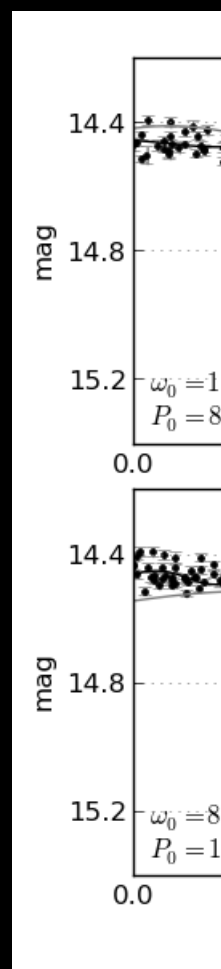
Lomb-Scargle Periodograms: Light Curves



Lomb-Scargle Periodograms: Light Curves



Lomb-Scargle Periodograms: Light Curves

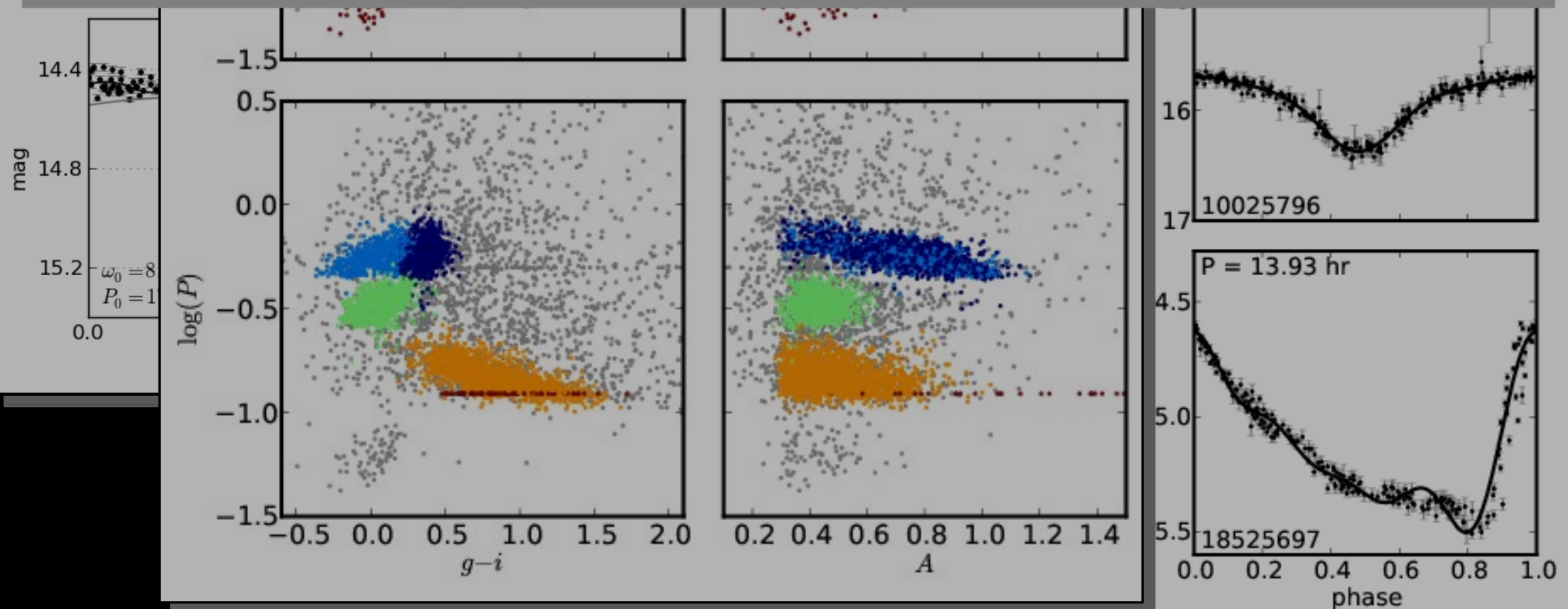


Lomb-Scargle Periodograms: Light Curves

```
from astroML.datasets import fetch_LINEAR_sample
from astroML.time_series import lomb_scargle

data = fetch_LINEAR_sample() # LINEAR is a database of time-domain observations
t, y, dy = data[14752041].T

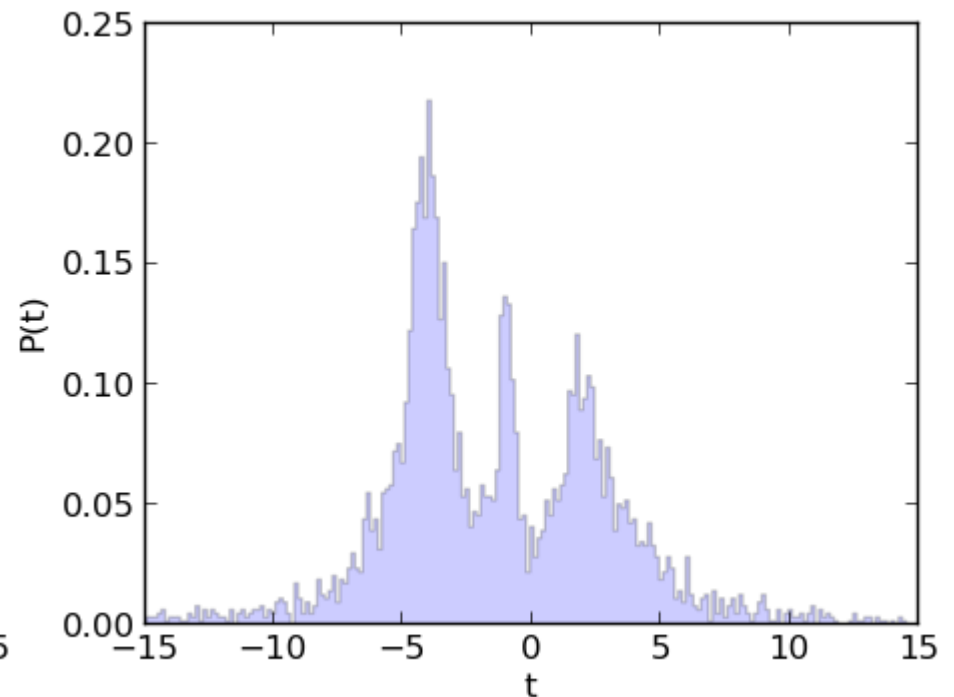
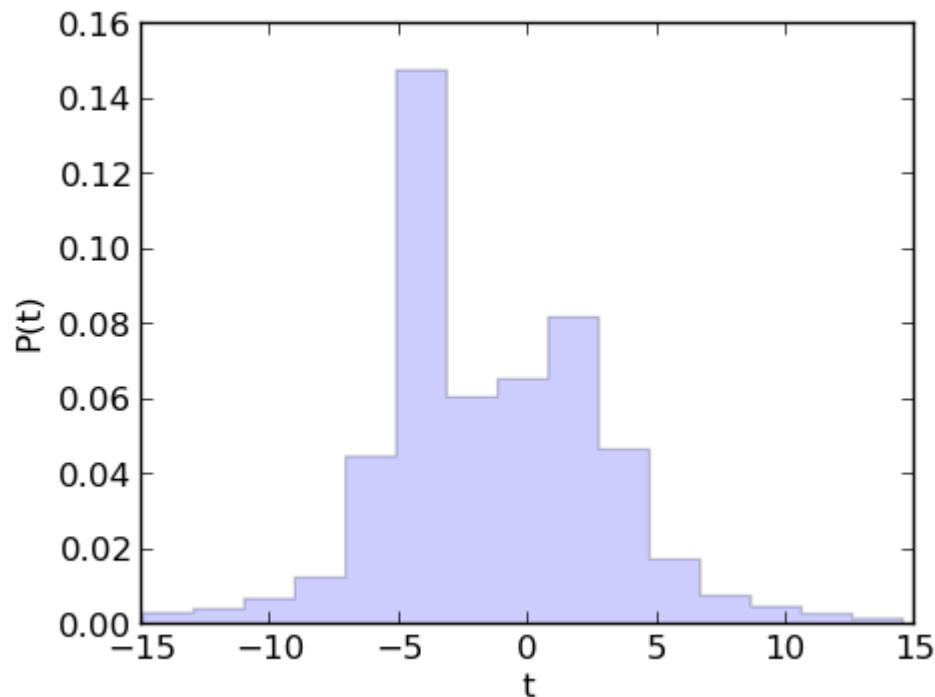
P_LS = lomb_scargle(t, y, dy, omega)
```



Histograms the Right Way:

The problem with histograms: choosing the bin width.

These show the same data set: how do we choose the right binning?

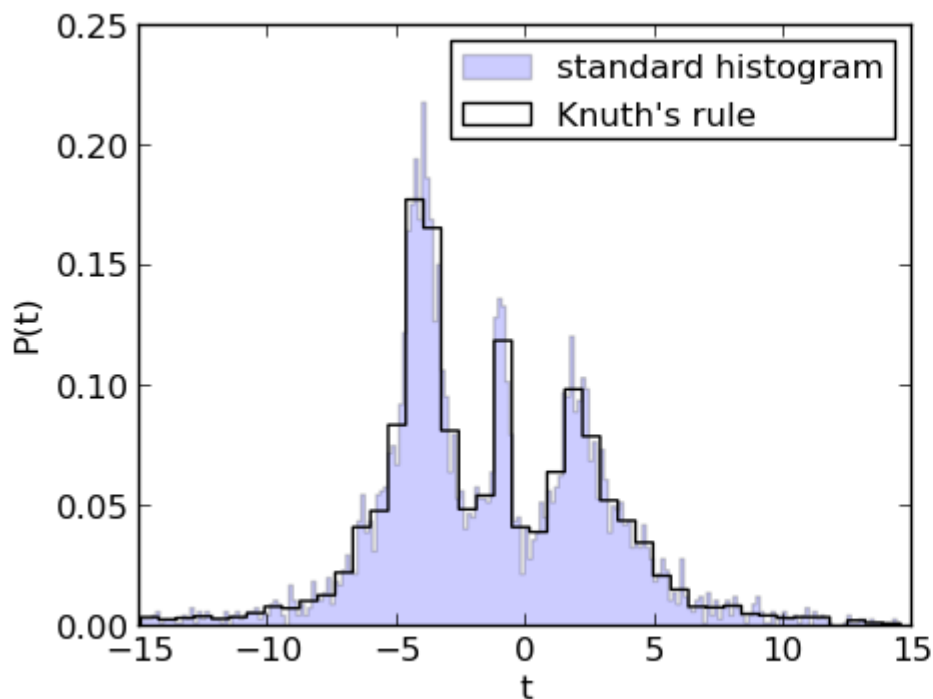


Histograms the Right Way:

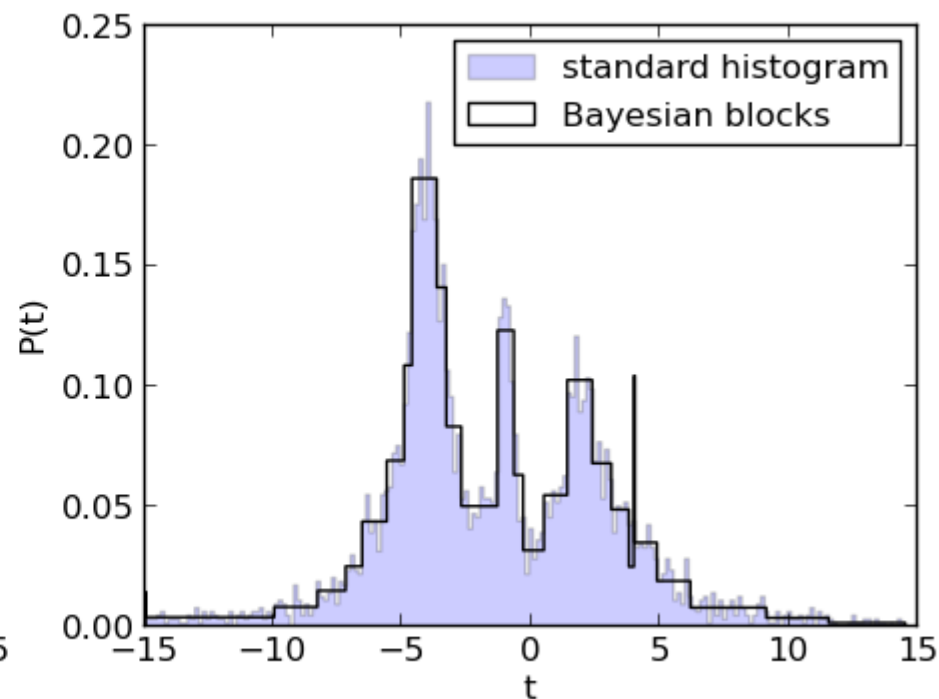
Knuth's Method: optimization over fixed-width bins

Bayesian blocks: optimization over variable-width bins

Knuth's Method



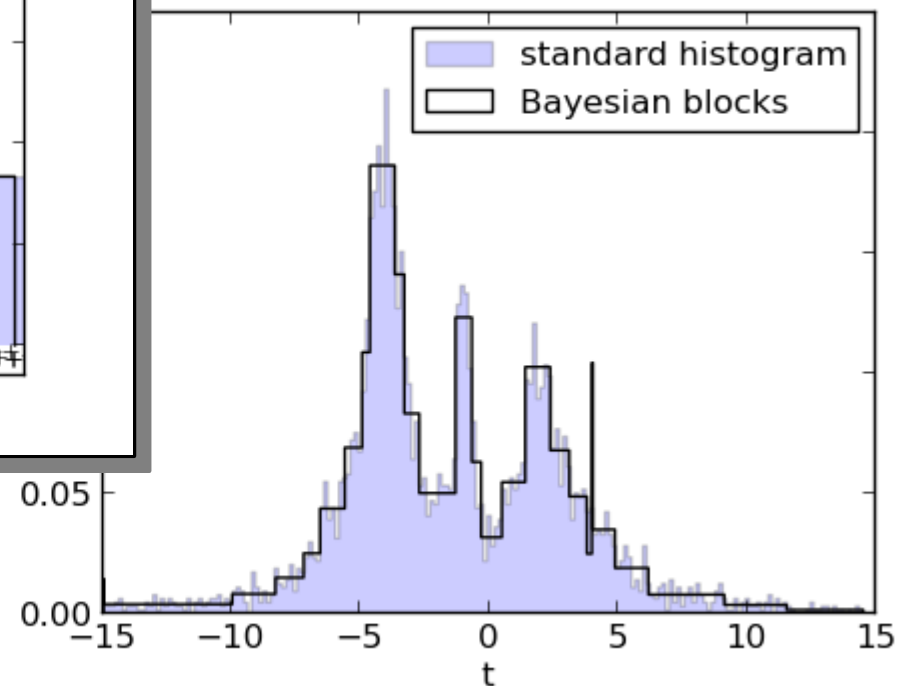
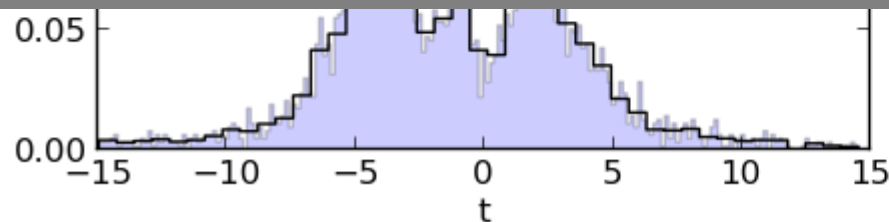
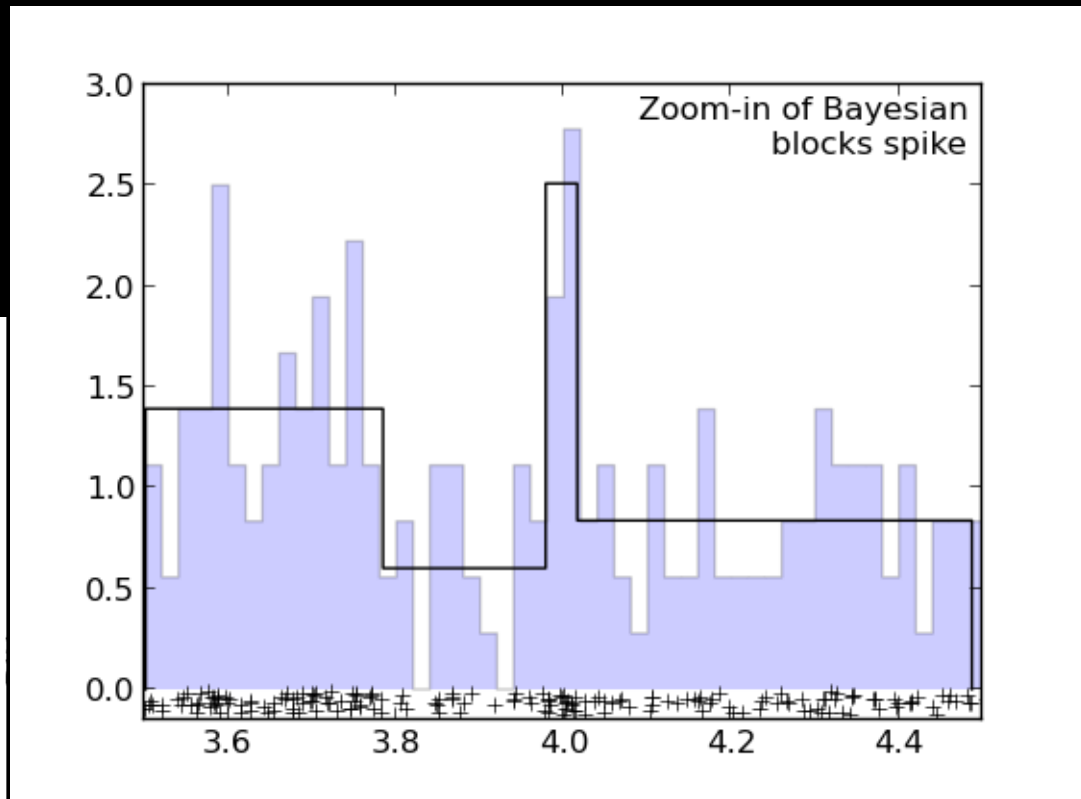
Bayesian Blocks



Histograms the Right Way:

fixed-width bins
variable-width bins

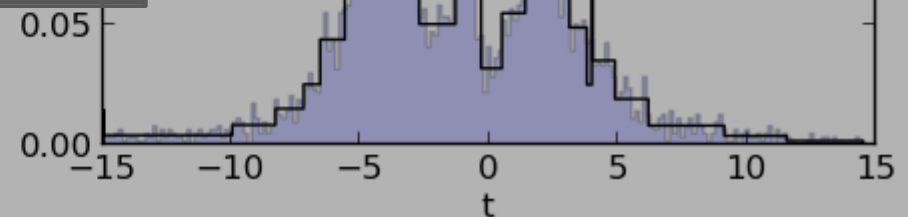
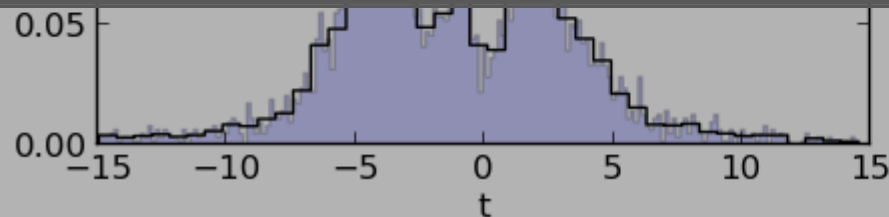
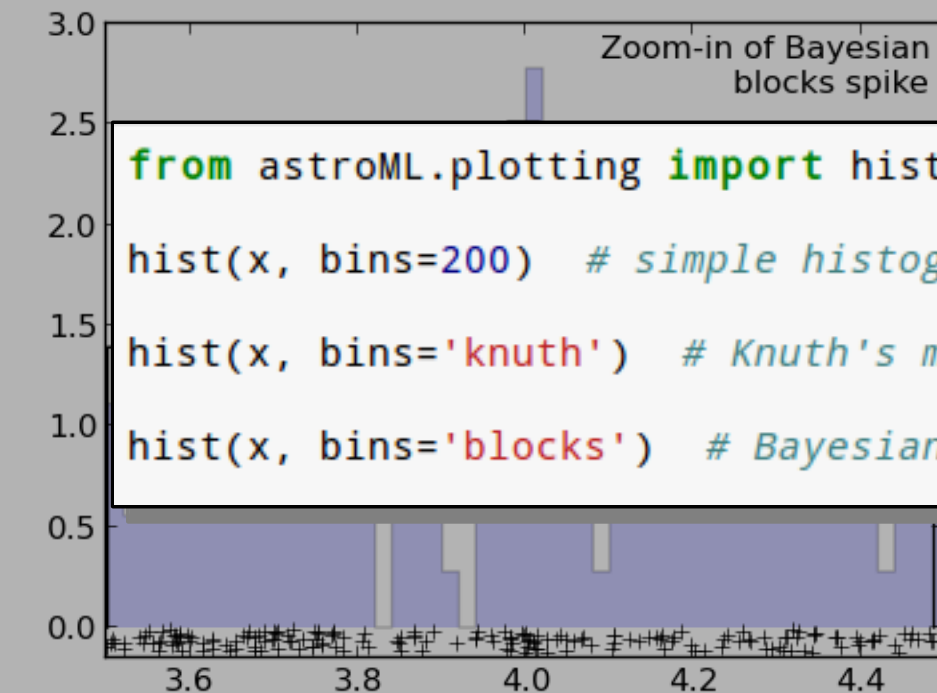
Bayesian Blocks



Histograms the Right Way:

fixed-width bins
variable-width bins

```
from astroML.plotting import hist  
hist(x, bins=200) # simple histogram, equivalent to matplotlib  
hist(x, bins='knuth') # Knuth's method: fixed-width bins  
hist(x, bins='blocks') # Bayesian blocks: variable-width bins
```



The Vision:

- Reproducible Research: provide a standard repository for sharing well-tested algorithms
- Coherent & well-written examples *using real data*: useful for both education and research
- Speed-up data exploration: examples require a minimal amount of code (typically ~10 lines)
- Move tested, useful code upstream for use in other fields.
A few examples:
 - Ball Tree & two-point statistics (scikit-learn 0.10)
 - Minimum Spanning Tree (scipy 0.11)
 - binned_statistics (scipy 0.11)
 - Bayesian blocks in matplotlib?
 - Extreme Deconvolution in scikit-learn?

Thank You

<http://astroML.github.com>