Python for Variable Star Astronomy

AAVSO Fall Meeting 2017

Matt Craig

Department of Physics and Astronomy

Slides/links at: https://github.com/mwcraig/aavso-talk
or: https://goo.gl/Lq1B3D

Acknowledgements

MINNESOTA STATE UNIVERSITY MOORHEAD

- Current students
  - Erin Aadland
  - Andy Block
  - Jane Glanzer
  - Elias Holte
  - Laura Maixner
  - Stefan Nelson
  - Elizabeth Dougherty
- Former students
  - Nathan Walker
  - Laura Herzog
  - Michael Meraz
  - Connor Stotts
  - Nathan Heidt
- Colleagues (MSUM)
  - Juan Cabanela
  - Linda Winkler

Outline

- Motivation
- Python and Jupyter
  - The five-minute overview
- The Astropy project
  - Core package: astropy
  - Relevant affiliated packages
- Graphical interfaces
  - Data reduction
  - Photometry
  - Image viewing
- Science application: per-image magnitude transforms

AstroImageJ

- Use AstroImageJ to:
    - Choose sources
        - Click on them once
        - Save as list
    - Perform aperture photometry with local background subtraction
        - Reject outlying pixels in annulus
- Result
    - instrumental magnitudes for night of data
- AIJ: Collins et al, AJ, 153 no. 2 (2017) https://doi.org/10.3847/1538-3881/153/2/77

Motivation: context

- undergraduate-only program
- 5 ± 3 new astro emphasis students/year
  - 0.5/year go to graduate school
- prepare students for a (non-astronomy) career
- at least one observational astro project while an undergraduate

Motivation: contraints

- No programming experience
- Need record of student work
- Use existing, well-supported packages
  - Preferably developed by someone else

Challenges

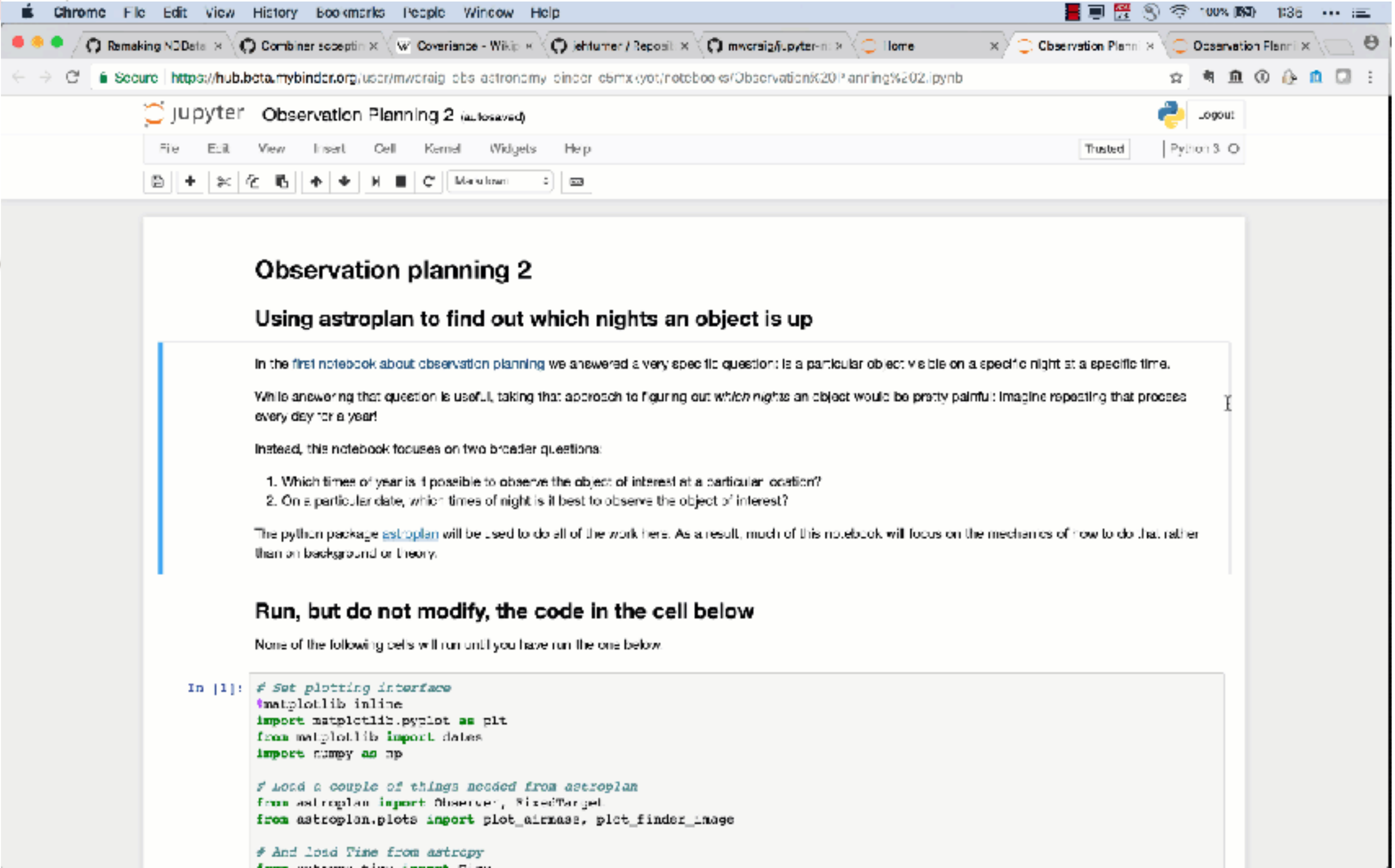
- Installation
  - Easiest: Use the Anaconda Python distribution
- Launching notebook
  - Open terminal
  - Change to correct directory
  - type command
  - wait for browser…

Python

- Rapid adoption in astronomy
- Widely used outside astronomy
- Fast numerical libraries
- Rich set of plotting packages
- Accessible (used in 1st-year physics)
- Powerful
  - analysis environment for LSST
  - pipeline for JWST

Jupyter notebooks

- Combine text, code, output in single document


Chrome File Edit View History Bookmarks People Window Help
Remaking NDData
Combiner acceptin
Covariance - Wiki
jehturner / Reposit
mwcraig/jupyter-n
Home
Observation Plann
Observation Plann
Secure https://hub.beta.mybinder.org/user/mwcraig-obs-astronomy-binder-c6mxyot/notebooks/Observation%20Planning%202.ipynb
jupyter Observation Planning 2 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Markdown

Observation planning 2

Using astroplan to find out which nights an object is up

In the first notebook about observation planning we answered a very specific question: Is a particular object visible on a specific night at a specific time.

While answering that question is useful, taking that approach to figuring out *which nights* an object would be pretty painful: Imagine repeating that process every day for a year!

Instead, this notebook focuses on two broader questions:

1. Which times of year is it possible to observe the object of interest at a particular location?
2. On a particular date, which times of night is it best to observe the object of interest?

The python package astroplan will be used to do all of the work here. As a result, much of this notebook will focus on the mechanics of how to do that rather than on background or theory.

Run, but do not modify, the code in the cell below

None of the following cells will run until you have run the one below

In [1]:

```
# Set plotting interface
%matplotlib inline
import matplotlib.pyplot as plt
from matplotlib import dates
import numpy as np

# Load a couple of things needed from astroplan
from astroplan import Observer, FixedTarget
from astroplan.plots import plot_airmass, plot_finder_image

# And load Time from astropy
```

Jupyter notebook

- Flexible architecture can run on:
  - completely on your laptop, or
  - secure server with authenticated accounts
  - public server without authentication
  - cloud servers
    - To try the notebook you just saw go to:

      https://goo.gl/RG4uLg

      then “Observation Planning 2.ipynb”
    - Server dies after ~1 hour of idle time

The Astropy project

- Community effort to coordinate development of Python for Astronomy
- All code
  - open source, permissive license
  - automated tests run on every change
  - extensively documentation
- astropy core
  - components common to most astronomy
- affiliated packages
  - more specialized tools
  - Roughly 35 packages (and growing)

astropy core

- `astropy` package includes:
  - times (UTC, TT, TAI, TDB..)
  - coordinates (ICRS, FK4, FK5, Galactic, …)
  - units
  - tables
  - WCS (translates pixels ↔ sky)
  - modeling and fitting
  - Lomb-Scargle periodogram

affiliated packages

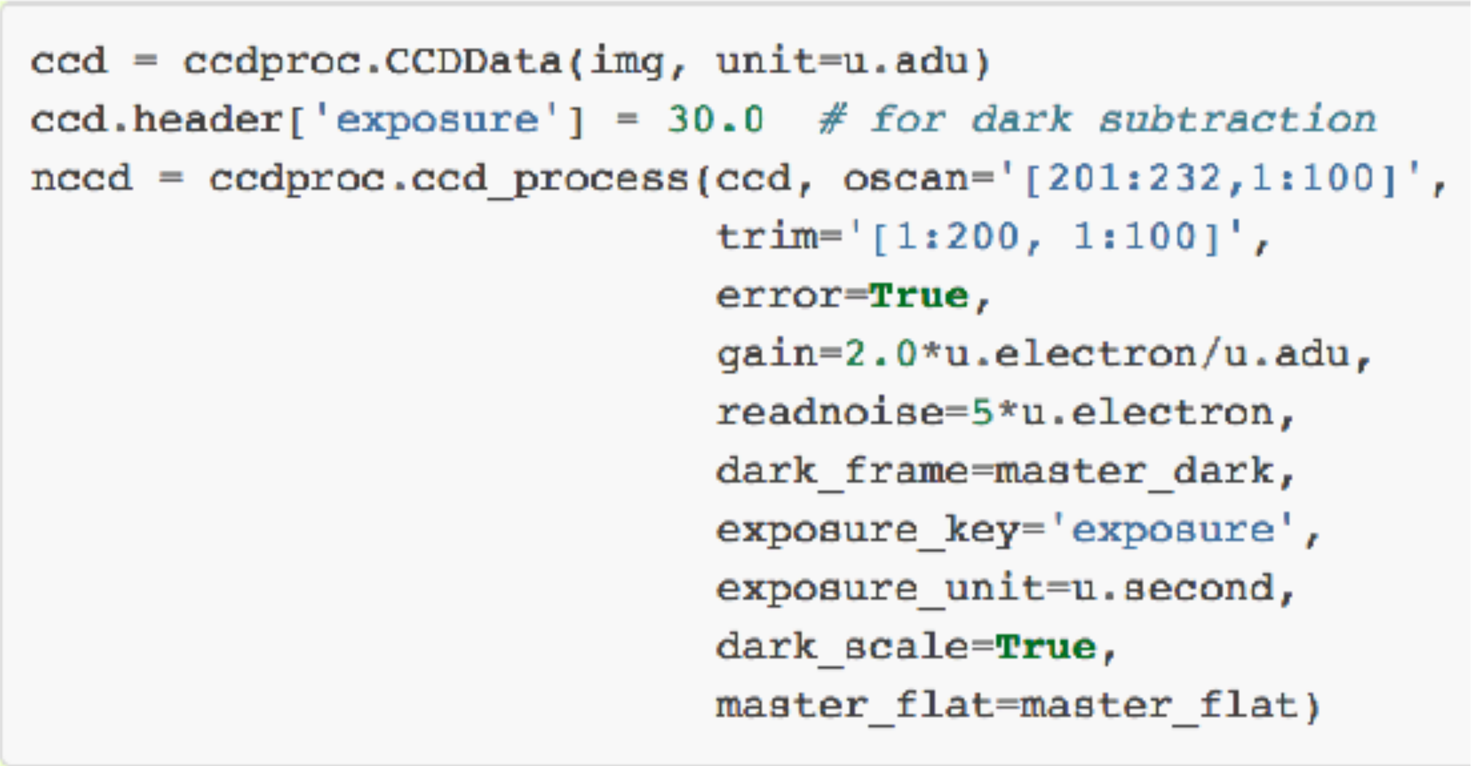
- image reduction
  - ccdproc
    - make masters
    - apply calibration masters to data
    - align and combine based on WCS (reproject)
    - cosmic ray removal (astroscrappy)
- photometry
  - photutils
    - background removal
    - aperture photometry
    - PSF photometry
  - sep
    - [technically, not affiliated]
    - Internals of SExtractor, wrapped in Python

affiliated packages

- catalog query/data retrieval
  - astroquery provides Python interfaces to
    - Simbad
    - Vizier (VSX, APASS, …)
    - Gaia
    - NIST
    - IRSA dust extinction
    - …
- Planning
  - astroplan
    - airmass/visiblity charts
    - basic finding charts
    - observation scheduler/optimizer
- Visualization
  - ginga
    - framework for writing visualization tools

Data reduction

- Snippet for reducing one image



```
ccd = ccdproc.CCDData(img, unit=u.adu)
ccd.header['exposure'] = 30.0  # for dark subtraction
nccd = ccdproc.ccd_process(ccd, oscan='[201:232,1:100]',
                           trim='[1:200, 1:100]',
                           error=True,
                           gain=2.0*u.electron/u.adu,
                           readnoise=5*u.electron,
                           dark_frame=master_dark,
                           exposure_key='exposure',
                           exposure_unit=u.second,
                           dark_scale=True,
                           master_flat=master_flat)
```

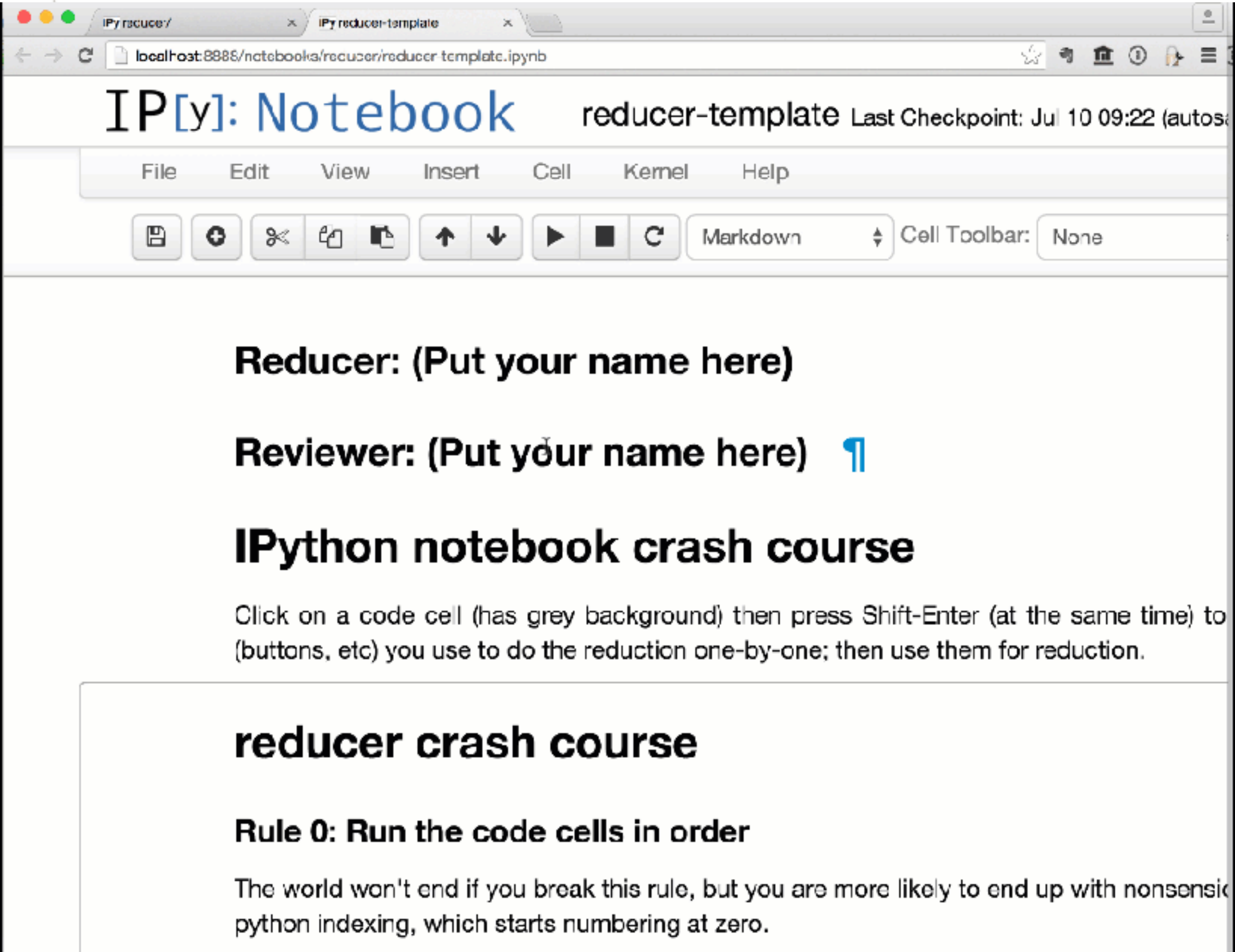
Data reduction

Coordinators:
Steve Crawford (@crawfordsm)
Michael Seifert (@MSeifert04)
Matt Craig (@mwcraig)

Yoonsoo P. Bach (@ysBach)
Kyle Barbary (@kbarbary)
Javier Blasco (@javierblasco)
Christoph Deil (@cdeil)
Carlos Gomez (@carlgogo)
Hans Moritz Günther (@hamogu)
Forrest Gasdia (@EP-Guy)
Nathan Heidt (@heidtha)
Elias Holte (@Sondanaa)
Anthony Horton (@AnthonyHorton)
Jennifer Karr (@JenniferKarr)
James McCormac (@jmccormac01)

Stefan Nelson (@stefannelson)
Joe Philip Ninan (@indiajoe)
Punyaslok Pattnaik (@Punyaslok)
Adrian Price-Whelan (@adrn)
Evert Rol (@evertrol)
William Schoenell (@wschoenell)
Sourav Singh (@souravsingh)
Brigitta Sipocz (@bsipocz)
Connor Stotts (@stottsco)
Ole Streicher (@olebole)
JVSN Reddy (@janga1997)
Erik Tollerud (@eteq)
Zè Vinícius (@mirca)
Josh Walawender (@joshwalawender)
Nathan Walker (@walkerna22)
Jiyong Youn (@hletrd)

Data reduction: reducer



IPy reduce/
IPy reducer-template
localhost:8888/notebooks/reducer/reducer-template.ipynb
IP[y]: Notebook
reducer-template Last Checkpoint: Jul 10 09:22 (autos
File
Edit
View
Insert
Cell
Kernel
Help
Markdown
Cell Toolbar:
None

Reducer: (Put your name here)

Reviewer: (Put your name here) ¶

IPython notebook crash course

Click on a code cell (has grey background) then press Shift-Enter (at the same time) to (buttons, etc) you use to do the reduction one-by-one; then use them for reduction.

reducer crash course

Rule 0: Run the code cells in order

The world won't end if you break this rule, but you are more likely to end up with nonsensic python indexing, which starts numbering at zero.

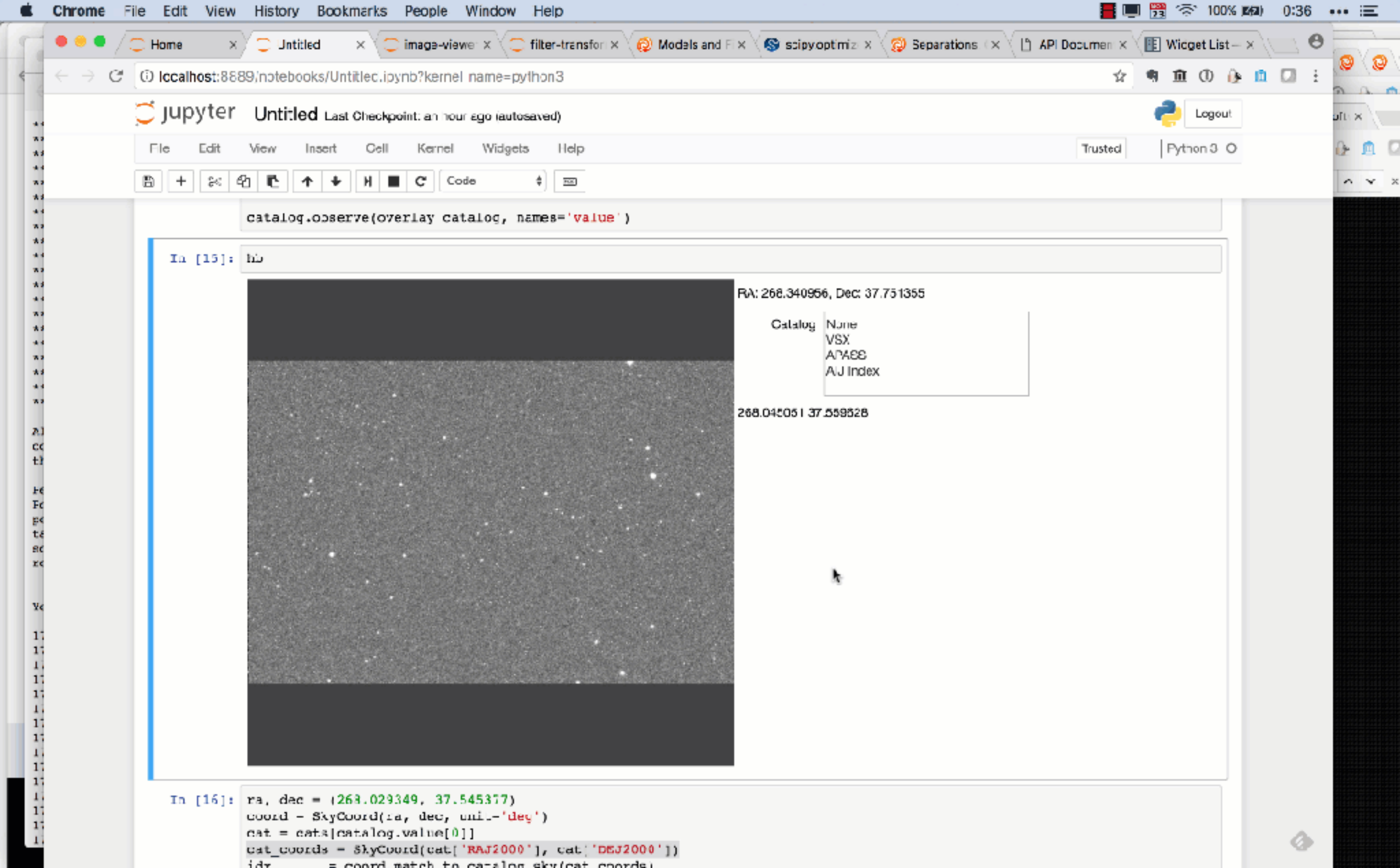
reducer

- jupyter notebook with interactive widgets
- Try it at: https://goo.gl/rgyLf6

Photometry, currently

- Wrapper around photutils to
  - detect stellar sources in an image
  - perform aperture photometry, with
  - rejection of outlying background pixels
- GUI on the way
  - Ideas?
  - Wish list?
- NOTE: all aperture photometry in rest of talk done in AstroImageJ

Image viewer in notebook


Chrome File Edit View History Bookmarks People Window Help
Home
Untitled
image-viewe
filter-transfor
Models and F
scipy.optimiz
Separations
API Documen
Widget List –
localhost:8889/notebooks/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Code

```
catalog.observe(overlay_catalog, names='value')
```

```
In [15]: hb
```

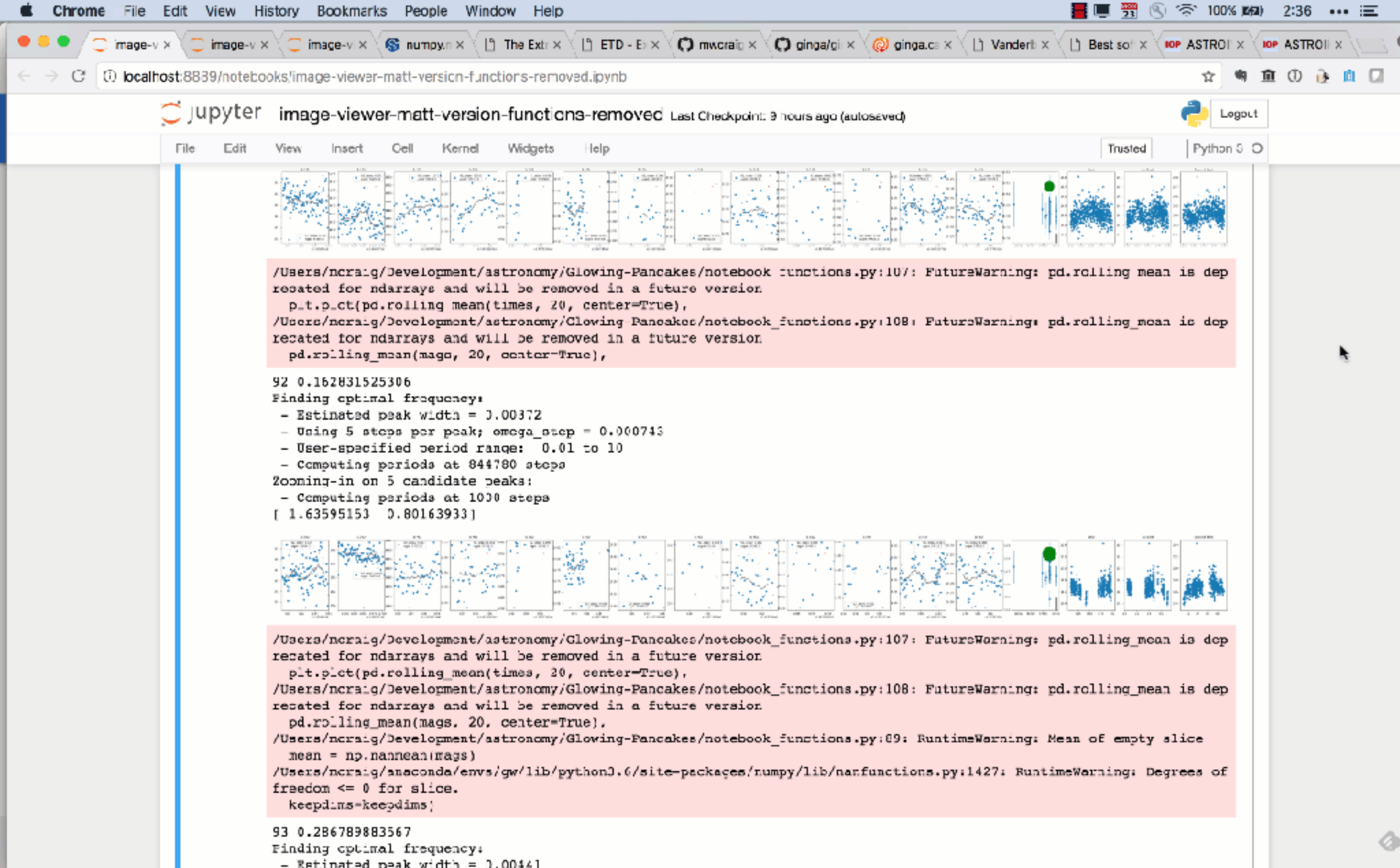
RA: 268.340956, Dec: 37.751355

Catalog
- None
- VSX
- APASS
- AIJ Index

268.045051 37.559528

```
In [16]: ra, dec = (268.029349, 37.545377)
         coord = SkyCoord(ra, dec, unit='deg')
         cat = cats[catalog.value[0]]
         cat_coords = SkyCoord(cat['RAJ2000'], cat['DEJ2000'])
         idx, ... = coord.match_to_catalog_sky(cat_coords)
```

star 92/VSX J175159.5+373058


Chrome File Edit View History Bookmarks People Window Help
100% 2:36
localhost:8839/notebooks/image-viewer-matt-version-functions-removed.ipynb
jupyter image-viewer-matt-version-functions-removed Last Checkpoint: 9 hours ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3

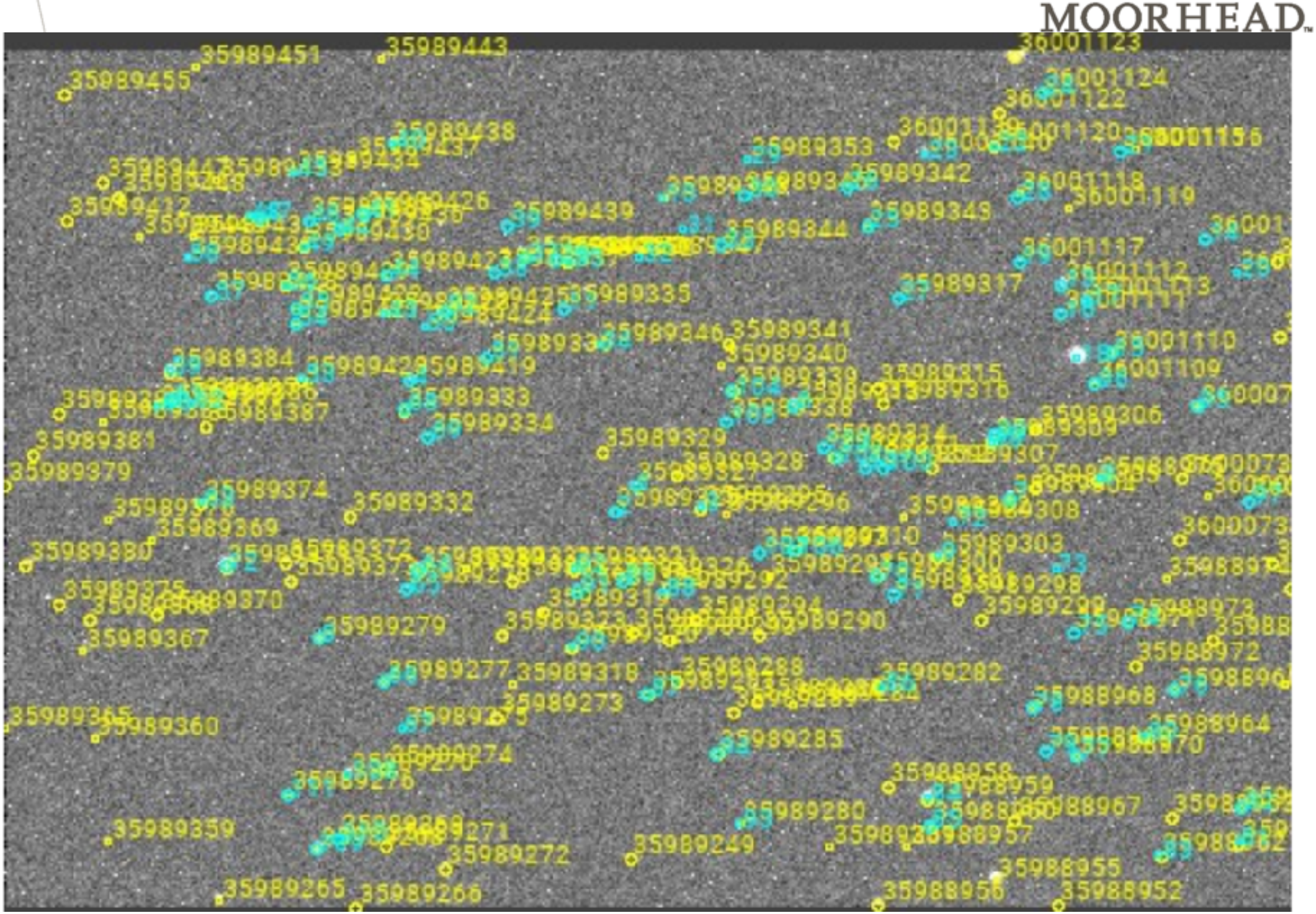
```
/Users/mcraig/Development/astronomy/Glowing-Pancakes/notebook_functions.py:107: FutureWarning: pd.rolling_mean is dep
recated for ndarrays and will be removed in a future version
  plt.plot(pd.rolling_mean(times, 20, center=True),
/Users/mcraig/Development/astronomy/Glowing-Pancakes/notebook_functions.py:108: FutureWarning: pd.rolling_mean is dep
recated for ndarrays and will be removed in a future version
  pd.rolling_mean(mags, 20, center=True),

92 0.162831525306
Finding optimal frequency:
 - Estimated peak width = 0.00372
 - Using 5 steps per peak; omega_step = 0.000743
 - User-specified period range:  0.01 to 10
 - Computing periods at 844780 steps
Zooming-in on 5 candidate peaks:
 - Computing periods at 1000 steps
[ 1.63595153  0.80163933]
```

```
/Users/mcraig/Development/astronomy/Glowing-Pancakes/notebook_functions.py:107: FutureWarning: pd.rolling_mean is dep
recated for ndarrays and will be removed in a future version
  plt.plot(pd.rolling_mean(times, 20, center=True),
/Users/mcraig/Development/astronomy/Glowing-Pancakes/notebook_functions.py:108: FutureWarning: pd.rolling_mean is dep
recated for ndarrays and will be removed in a future version
  pd.rolling_mean(mags, 20, center=True),
/Users/mcraig/Development/astronomy/Glowing-Pancakes/notebook_functions.py:89: RuntimeWarning: Mean of empty slice
  mean = np.nanmean(mags)
/Users/mcraig/anaconda/envs/gw/lib/python3.6/site-packages/numpy/lib/nanfunctions.py:1427: RuntimeWarning: Degrees of
freedom <= 0 for slice.
  keepdims=keepdims)

93 0.286789883567
Finding optimal frequency:
 - Estimated peak width = 0.00441
```

frame-by-frame transform

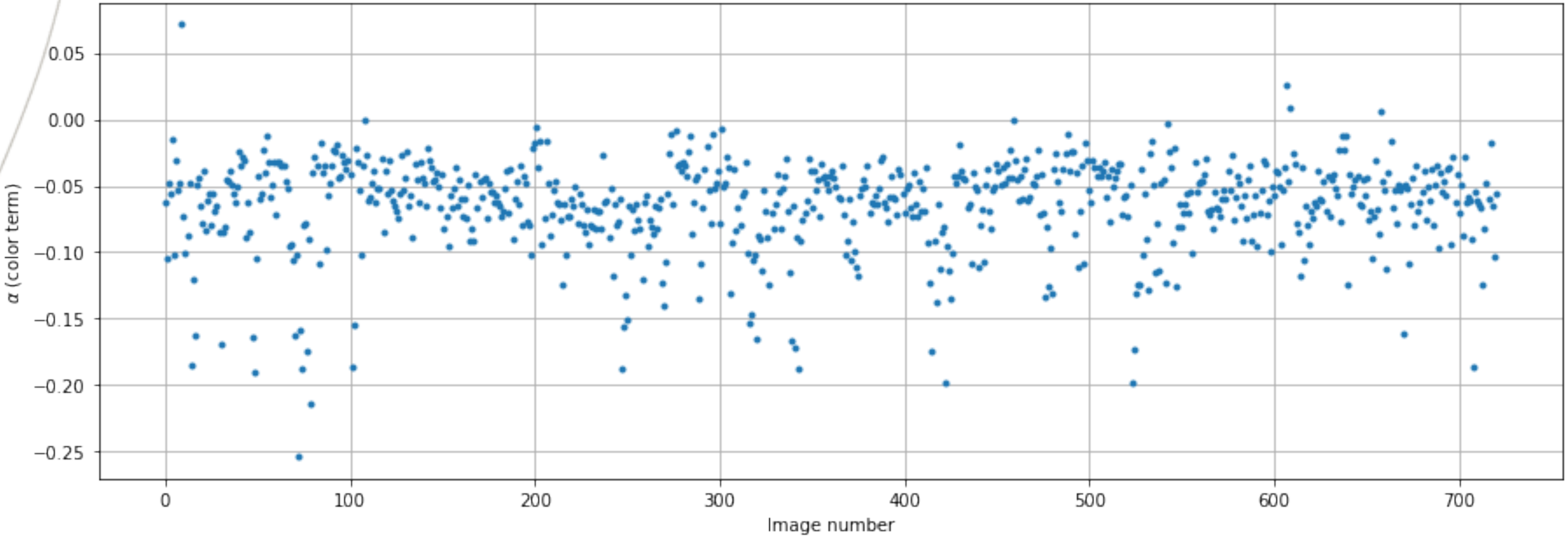


frame-by-frame transform

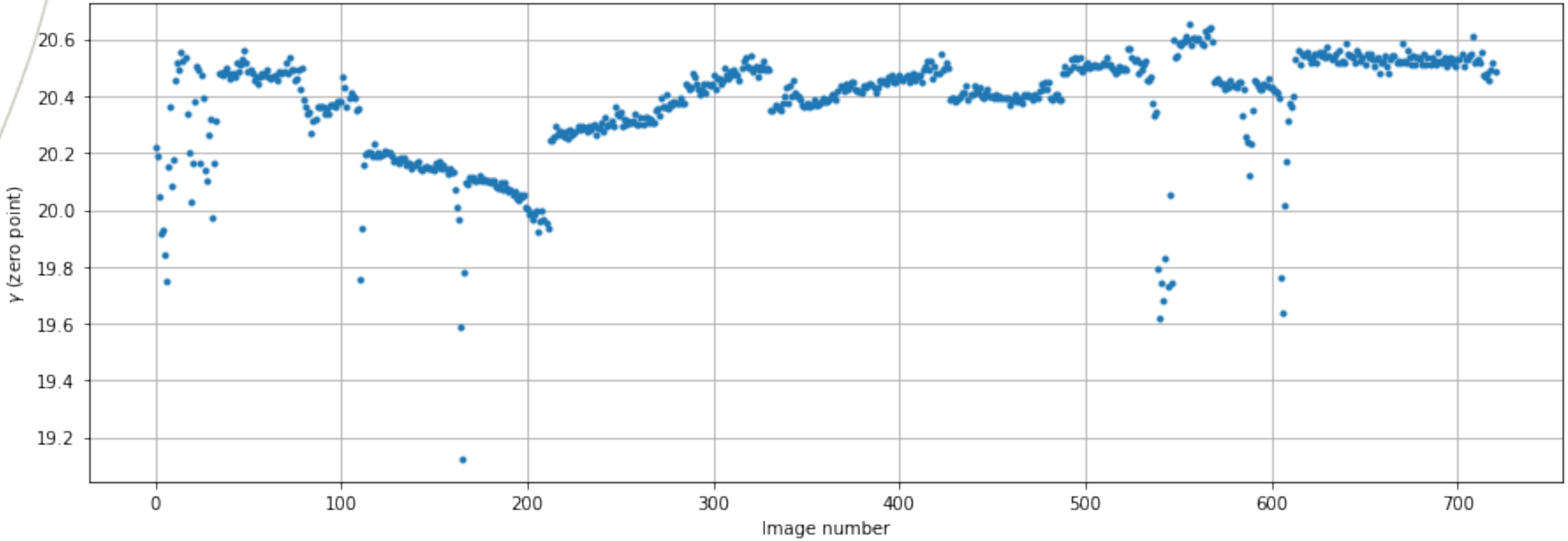
- match photometer star to APASS
  - Filter APASS: $\delta r < 0.05, \qquad \delta(B-V) < 0.1$
  - Transform APASS r to R
- for each frame:
  - calculate transform coefficients including color correction

$$R_{APASS} - R_{inst} = \alpha(B-V)_{APASS} + \gamma$$

Color coefficient



zero point



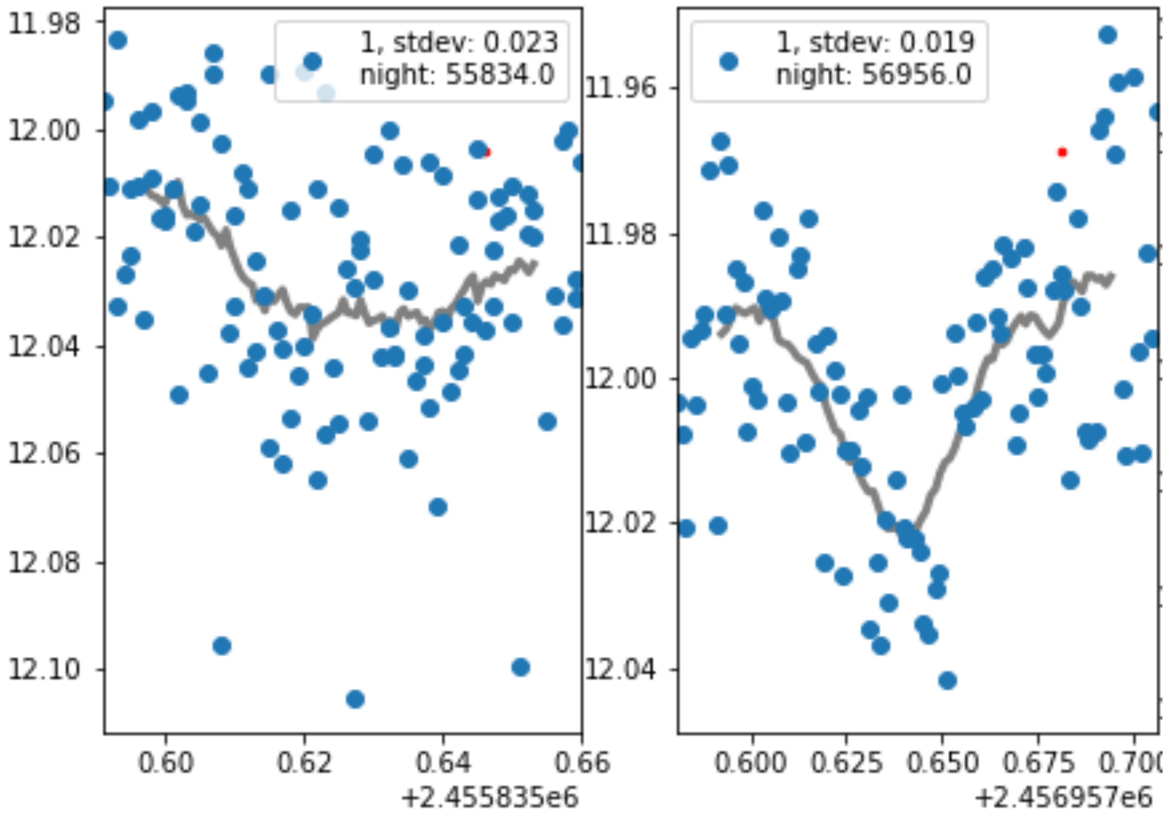
20.6
20.4
20.2
20.0
19.8
19.6
19.4
19.2
γ (zero point)
0
100
200
300
400
500
600
700
Image number

transform all stars

- match each star to APASS
  - No quality restriction on APASS stars
- frame-by-frame
  - Apply transform for that frame to all stars

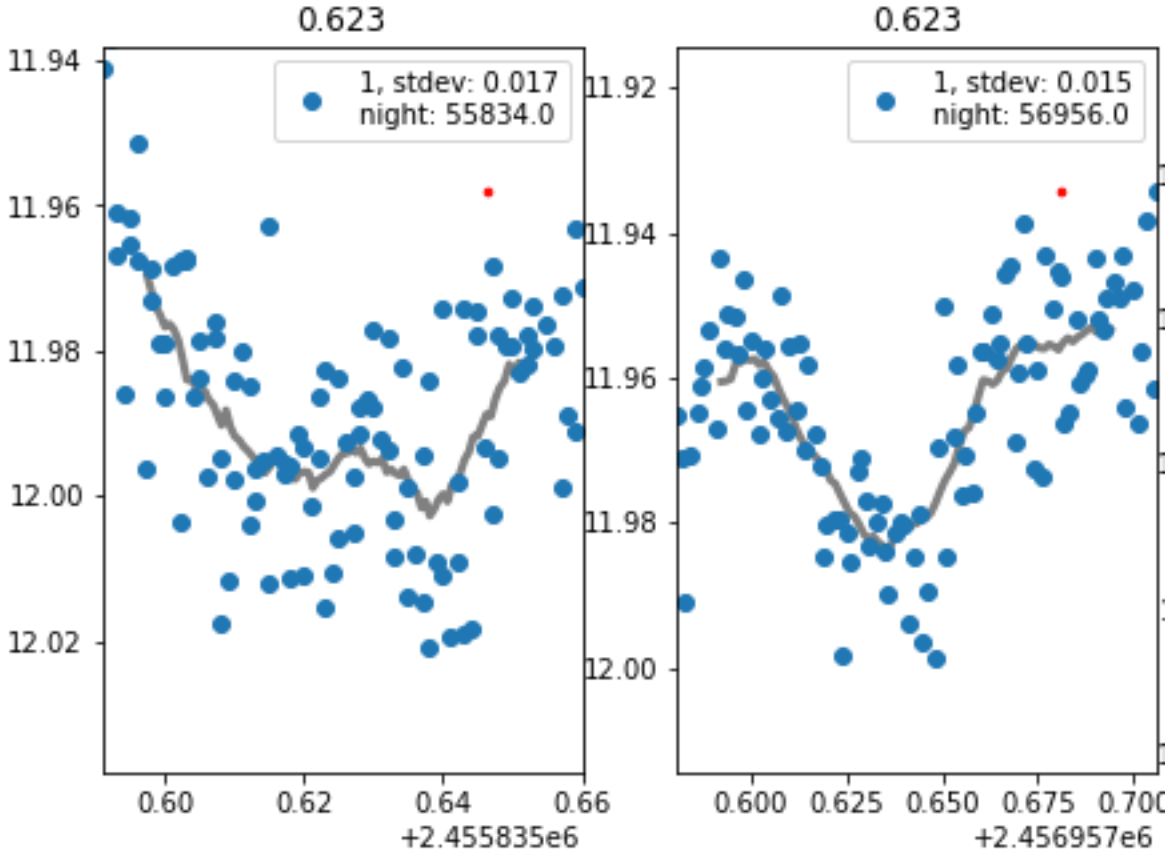
No color term

- TrES-3, two transits
- Corrected for zero point only



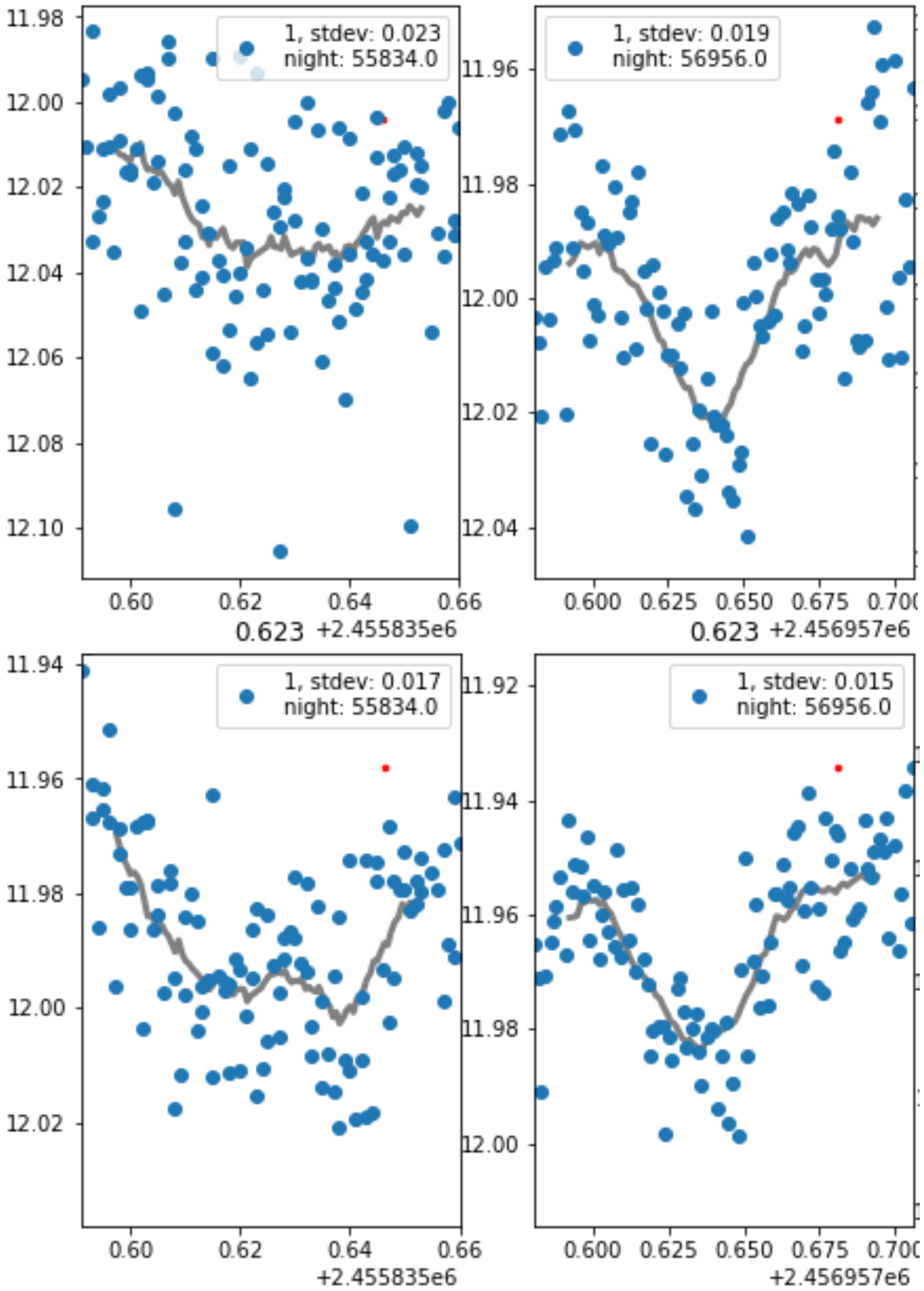
Linear color term

- Same two nights, linear color correction



do color corrections!

MINNESOTA STATE UNIVERSITY
MOORHEAD



Next steps

- Instrumental colors vs APASS colors
- Test in other filters
- More robust period estimation
- Bundle photometry code into one package
    - package name?
- Address installation

Questions?

Slides/links at:

https://github.com/mwcraig/aavso-talk

or

https://goo.gl/Lq1B3D

Links

- lemon: http://lemon.readthedocs.org/en/latest/
    - end-to-end data reduction and photometry
- OSCAAR: http://oscaar.github.io/OSCAAR/
    - Focuses on exoplanet transit measurements
- gatspy: http://www.astroml.org/gatspy/
    - fast Lomb-Scargle implementation
- conda-build-all: https://github.com/SciTools/conda-build-all
    - eases the pain of building packages
- sep: http://sep.readthedocs.org/en/v0.5.x/
    - Photometry (uses internals from SExtractor)
- astroquery: http://astroquery.readthedocs.org/
    - Search a variety of online data sources from python.
- ginga: https://ejeschke.github.io/ginga/
    - Image viewer framework (and a reference viewer)
- ccdproc: http://ccdproc.readthedocs.org/en/latest/
    - Data reduction
- photutils: https://photutils.readthedocs.org/en/latest/
    - Photometry (including, but not limited to, IRAF-equivalents)
- AstroImageJ: http://www.astro.louisville.edu/software/astroimagej/
    - Very nice graphical interface with sophisticated fitting and graphing
- reducer: http://reducer.readthedocs.org/en/latest/
    - Widget-interface to ccdproc reduction
- glowing-waffles: https://github.com/glowing-waffle/glowing-waffles
    - Very much work-in-progress, examples from today will be up there by Tue, 3/22/16
- feder_image_shuffle: https://github.com/mwcraig/feder_image_shuffle
    - Among other things, makes jpeg images and gallery pages, also demonstrates interacting with Github API.
- msumastro: https://github.com/mwcraig/msumastro
    - Infrastructure for adding metadata (largely telescope specific)