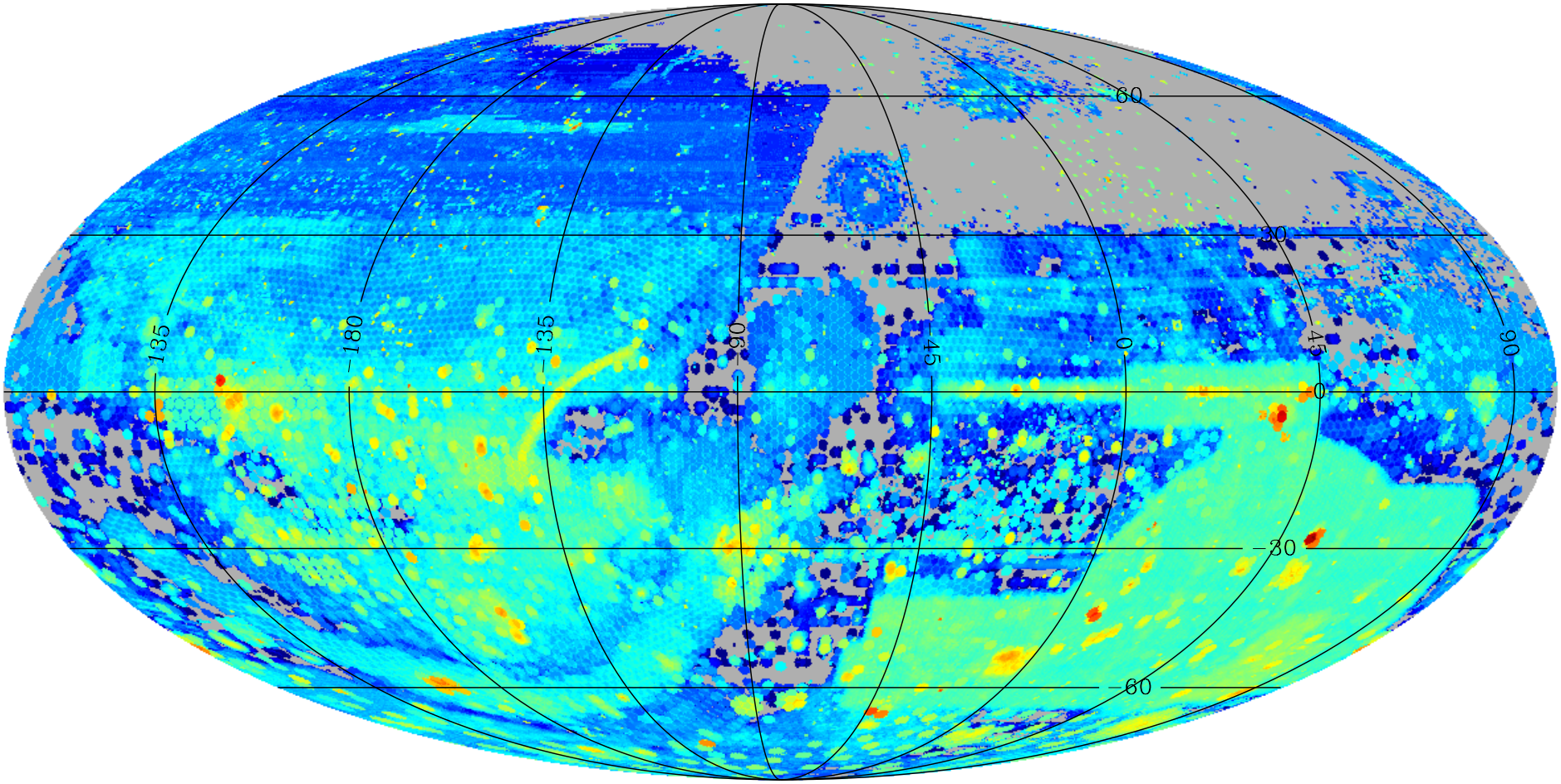
The NOAO Data Lab Project
Introduction

Data Lab Team

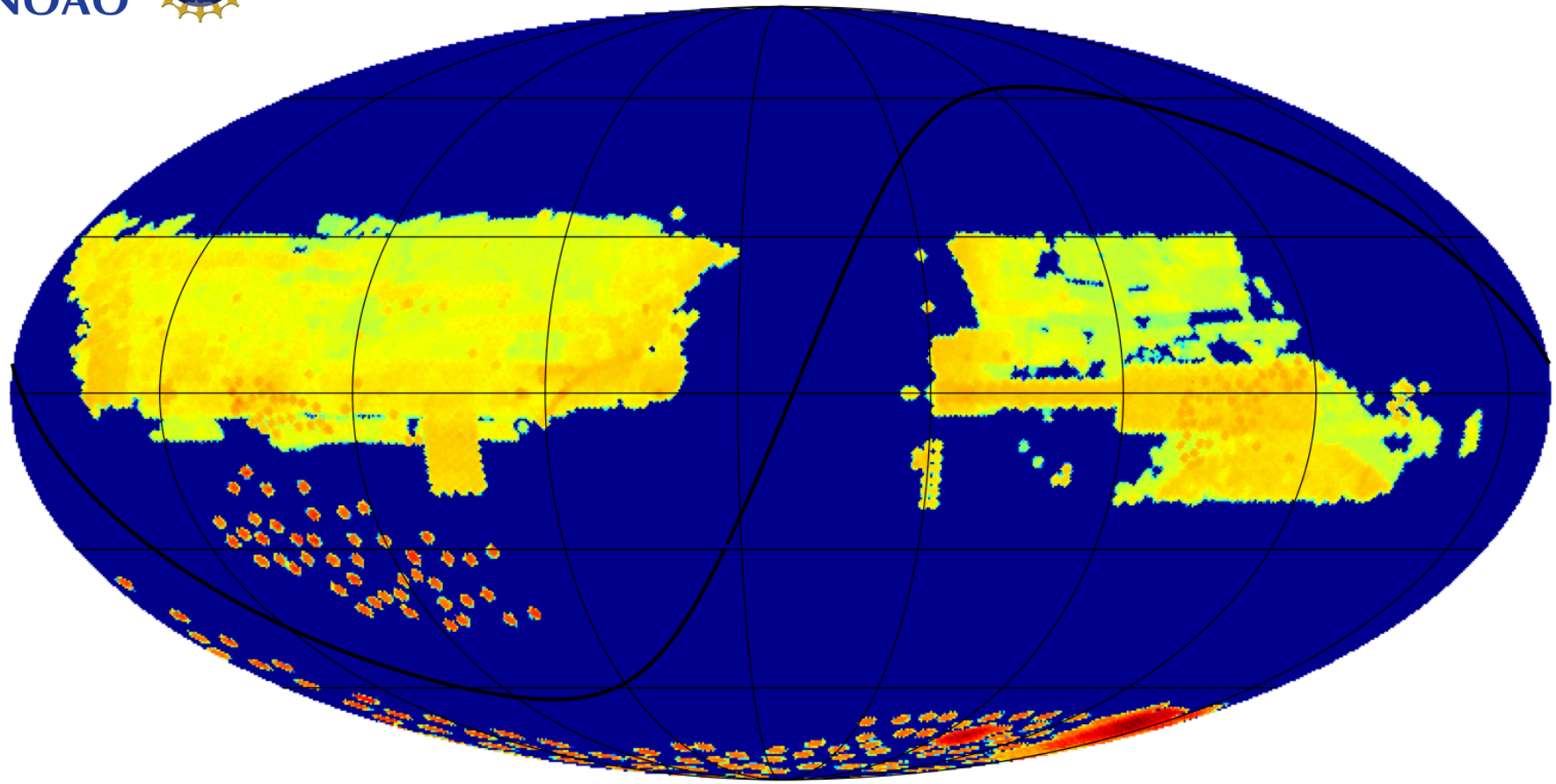
Current team:

- Mike Fitzpatrick, Lead Developer
- Matthew Graham, Scientist/Developer
- Wendy Huang, Software Engineer
- Stephanie Juneau, Data Scientist
- David Nidever, Data Scientist
- Robert Nikutta, Data Scientist
- Pat Norris, Test Engineer
- Knut Olsen, Project Scientist
- Steve Ridgway, Scientist
- Adam Scott, Database Architect
- Pete Wargo, System Administrator

DECam and Mosaic data in June 2017

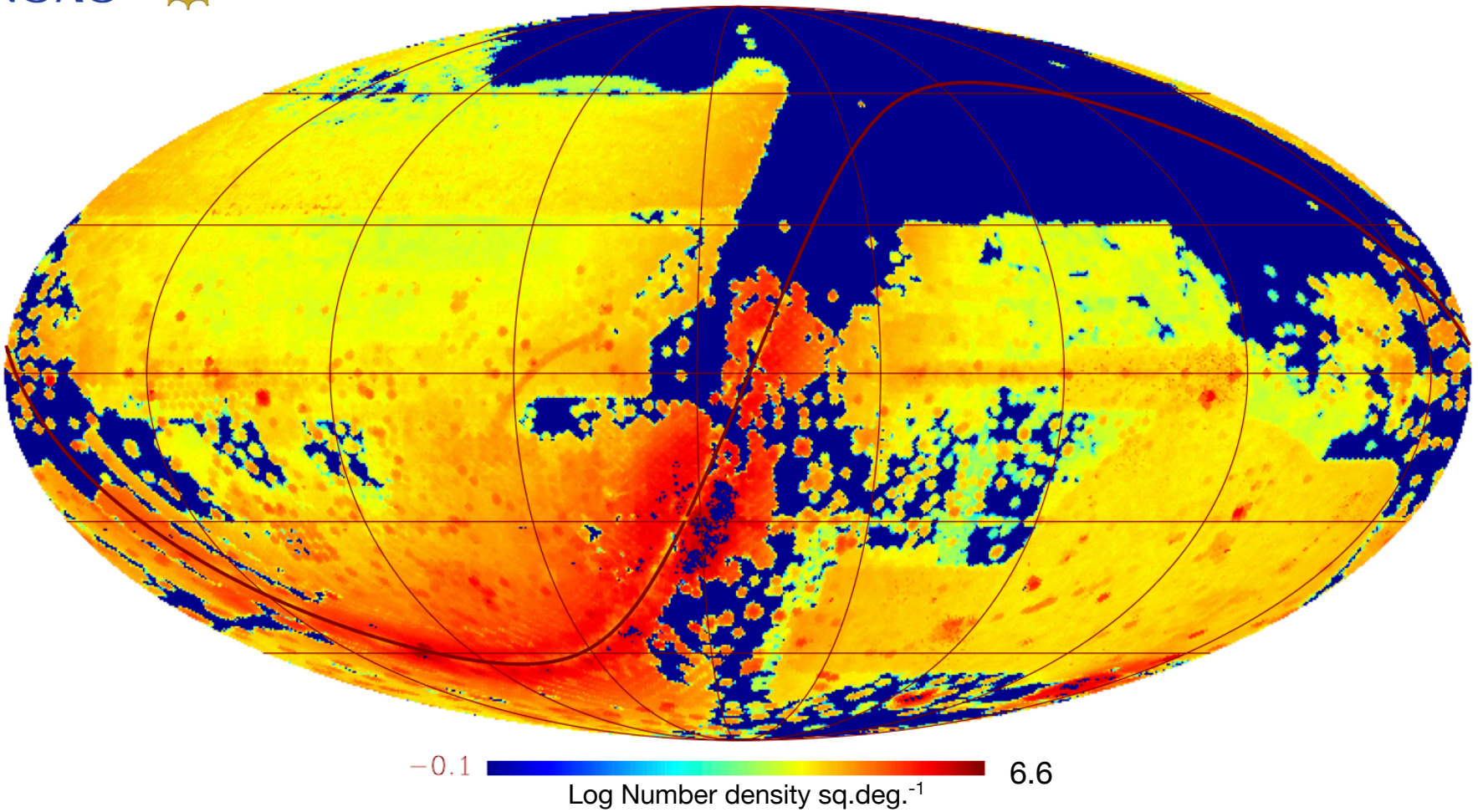


DECaLS DR3 and SMASH Catalogs



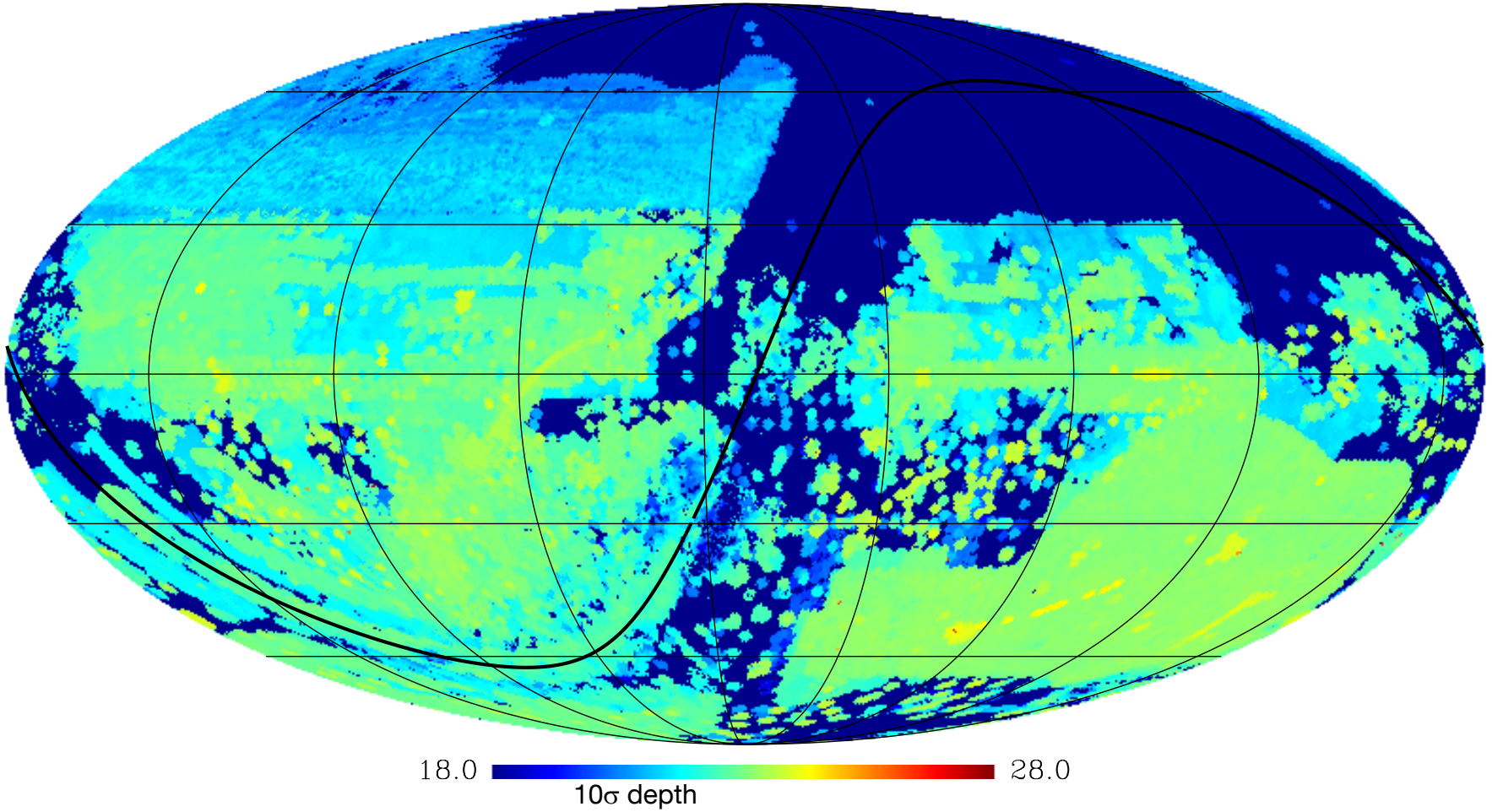
- 900 million objects available now through Data Lab database from these catalogs
- Also available: select tables from SDSS DR13, GAIA DR1, DES SVA1, the Allen NEO catalog, and USNO-A2/B

NOAO All Sky Catalog



- 2.5 billion objects, 20 billion measurements; aperture-based photometry
- Availability planned for September 2017

NOAO All Sky Catalog



- 2.5 billion objects, 20 billion measurements; aperture-based photometry
- Availability planned for September 2017

Data Volume and Complexity

~500 TB (February 2017) of on-target imaging data (t_{exp}>30s) currently from:

- Dark Energy Survey
- Legacy Surveys for DESI Targeting
- Community DECam and Mosaic programs and surveys

Hundreds of TB more coming

Total holdings at PB scale

Large catalogs coming:

- Dark Energy Survey – 45 TB
- Complete DESI Targeting Survey – ~5 TB
- Community programs and surveys – up to several TB each

NOAO Data Lab

Goal:

- Efficient exploration and analysis of large datasets with an emphasis on NOAO wide-field 4-m telescopes

Approach:

- High-value catalogs from NOAO and external sources (e.g. SDSS, GAIA) and NOAO-based images linked to catalog objects
- Data discovery
- Developing intuition through interaction with selected catalog and image set of known objects
- Automation of analysis to aid discovery of unknown objects

Data Lab in a Nutshell

Large Catalogs – Data Lab will serve TB-scale databases and provide personal database storage

Pixel Data – Data Lab will connect users to images and spectra in NOAO Science Archive

Virtual Storage – ~1 TB per user to minimize data transfer

Visualization – Data Lab will enable data exploration

Compute Processing* – Data Lab will allow workflows to run close to the data

Additional features* – Access to published datasets and external data services, data publication, exportable workflows, distributable software

*Some limitations in first release

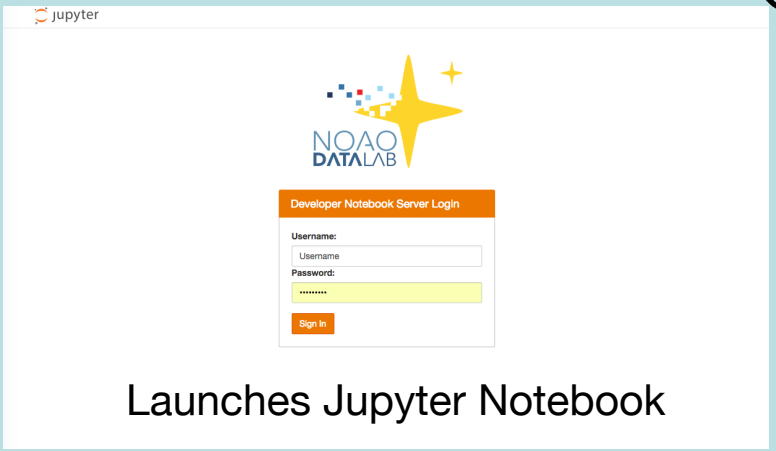
Summary of Current Functions

Function	Method
Sky exploration	Image discovery tool Catalog overlay tool Catalog visualization tool (prototype)
Authentication	Web interface datalab command Python authClient, DL interface
Catalog query	Web interface datalab command line (CLI) Python queryClient, DL interface TOPCAT
Image query	Simple Image Access (SIA) service
Query result storage	myDB Virtual storage space
File transfer	datalab command and Virtual storage space
Analysis	Jupyter notebook server

Example: Star/galaxy/QSO separation

User logs in to Data Lab

1



Launches Jupyter Notebook

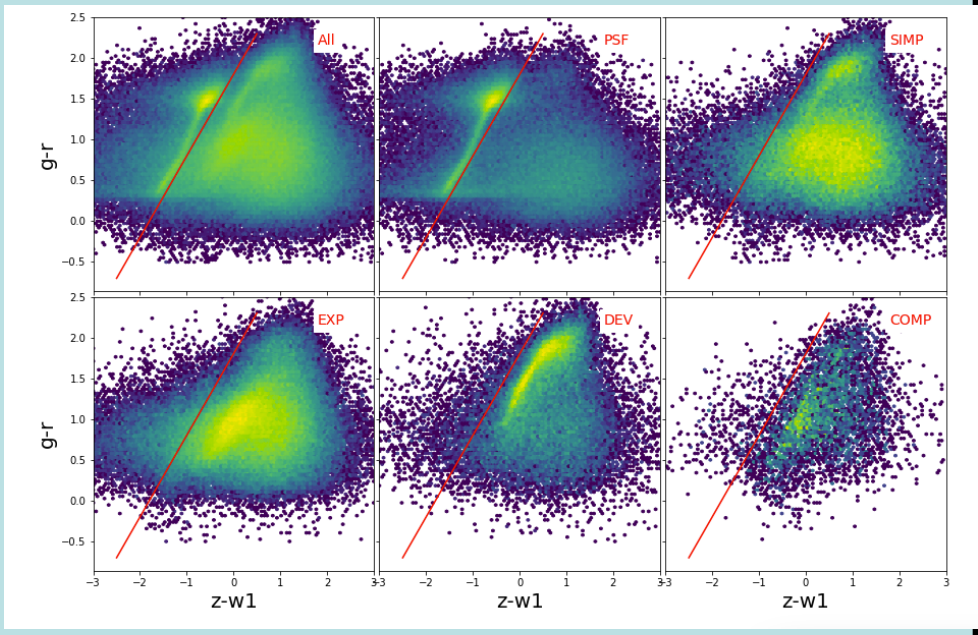
Queries database for DECaLS Tractor and AllWISE photometry

2

```
In [58]: # Write query statement (adql)
         query = """
                 SELECT g as gmag, r as rmag, z as zmag, w1 as w1mag, w2 as w2mag, type,
                        decam_flux_2*sqrt(decam_flux_ivar_2) as snr_g,
                        decam_flux_3*sqrt(decam_flux_ivar_3) as snr_r,
                        decam_flux_5*sqrt(decam_flux_ivar_5) as snr_z
                 FROM ls_dr3.tractor_primary
                 WHERE (decam_flux_2*sqrt(decam_flux_ivar_2)>3 and
                        decam_flux_3*sqrt(decam_flux_ivar_3)>3 and
                        decam_flux_5*sqrt(decam_flux_ivar_5)>3)
                 LIMIT 800000"""

         #         WHERE (decam_nobs_2>1 and decam_nobs_3>1 and decam_nobs_5>1)

         # type          = object type (PSF, SIMP, EXP, DEV, COMP)
         # g,r,z         = AB magnitudes in DECam g,r,z bands
         # w1,w2         = AB magnitudes in WISE bands W1 & W2
         # decam_nobs_X = number of observations in bands g (2), r (3) & z (5)
         #
         # WHERE: requirement that there are more than 1 (i.e., at least 2) observation in each DECaLS band

         print query

                 SELECT g as gmag, r as rmag, z as zmag, w1 as w1mag, w2 as w2mag, type,
                        decam_flux_2*sqrt(decam_flux_ivar_2) as snr_g,
                        decam_flux_3*sqrt(decam_flux_ivar_3) as snr_r,
                        decam_flux_5*sqrt(decam_flux_ivar_5) as snr_z
                 FROM ls_dr3.tractor_primary
                 WHERE (decam_flux_2*sqrt(decam_flux_ivar_2)>3 and
                        decam_flux_3*sqrt(decam_flux_ivar_3)>3 and
                        decam_flux_5*sqrt(decam_flux_ivar_5)>3)
                 LIMIT 800000
```

3

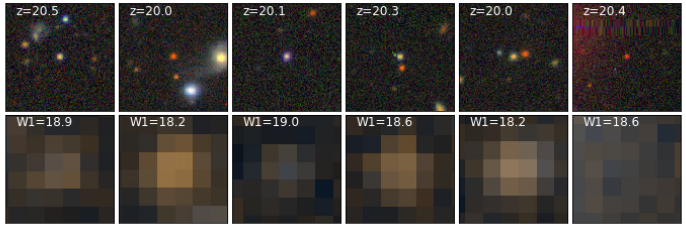
Visualizes color-color diagrams by morphology type



...

4

Checks image cutouts of selected objects



Joins objects to SDSS DR13 spectroscopic catalog

5

Below is the result of a hack by Bela Abolfathi (UCI) put together during the LSSTC Data Science Fellowship program (April 2017).

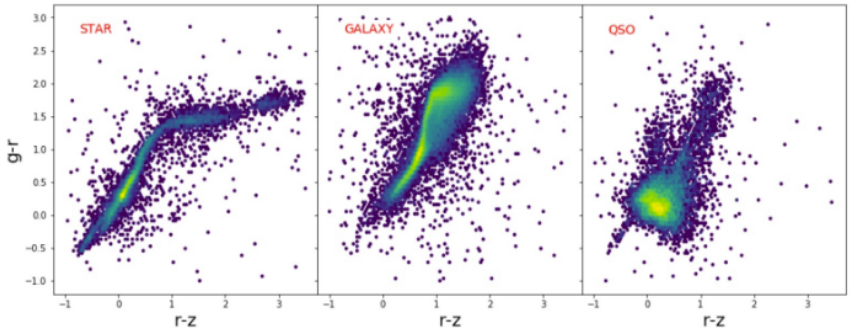


Figure: Joint query with SDSS SpecObj table and split by CLASS (Star, Galaxy, QSO).

Uses as training set for machine learning classification

Below is the result of a hack by Jan-Torge Schindler (UofA) put together during the NOAO Data Lab Tutorial (May 2017).

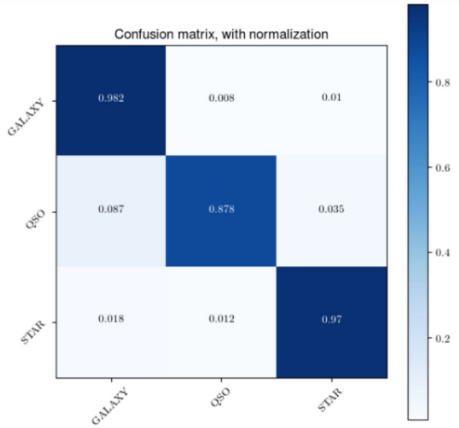


Figure: Confusion matrix, normalized by number in each category. Done from joint query with SDSS SpecObj to build training classify validation set with Machine Learning (Random Forest).

6

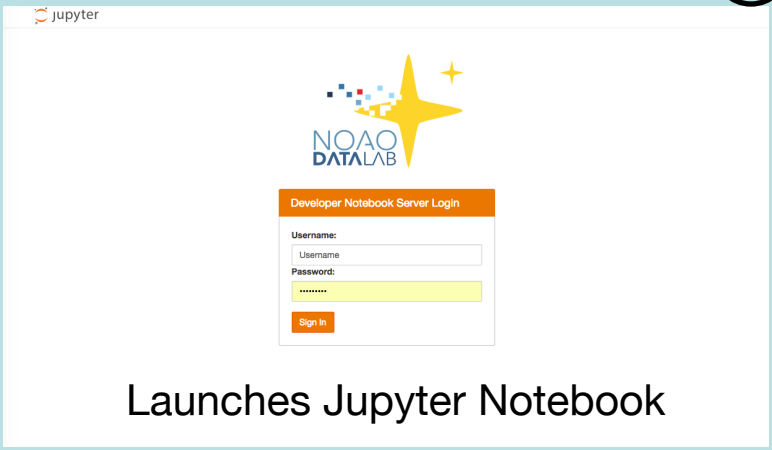
Applies classification techniques to find new objects of interest!

7

Example: Detecting a faint dwarf galaxy

User logs in to Data Lab

1



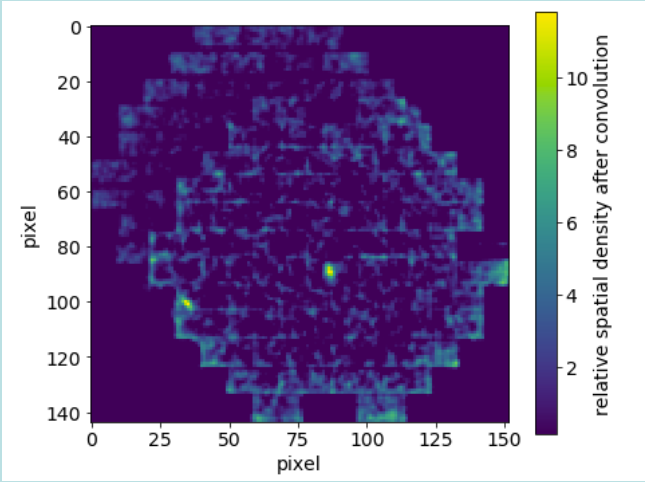
Launches Jupyter Notebook

Queries database for blue stellar objects in SMASH DR1 Field

2

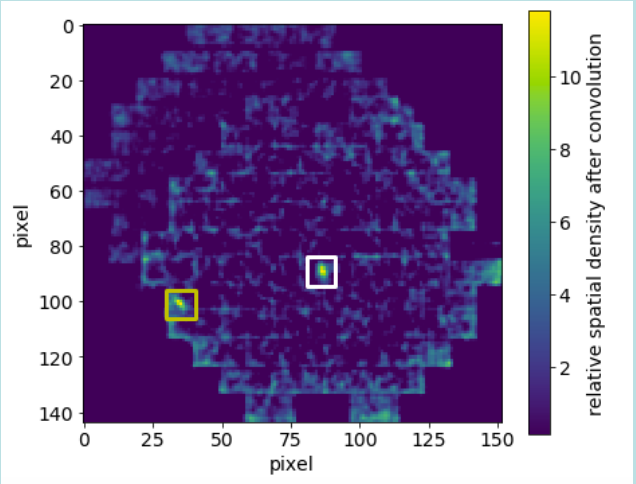
```
field = 169 # SMASH field number to query
depth = 1   # depth (=no short exposures please)

# Create the query string; SQL keyword capitalized for clarity
query_template =\
"""SELECT ra,dec,gmag,rmag,imag FROM smash_dr1.object
   WHERE fieldid = '%d' AND
         depthflag > %d AND
         abs(sharp) < 0.5 AND
         gmag BETWEEN 9 AND 25 AND
         (gmag-rmag) BETWEEN -0.4 AND 0.4"""

query = query_template % (field, depth)
```

Applies filter to spatial distribution



3

Runs automatic peak detection

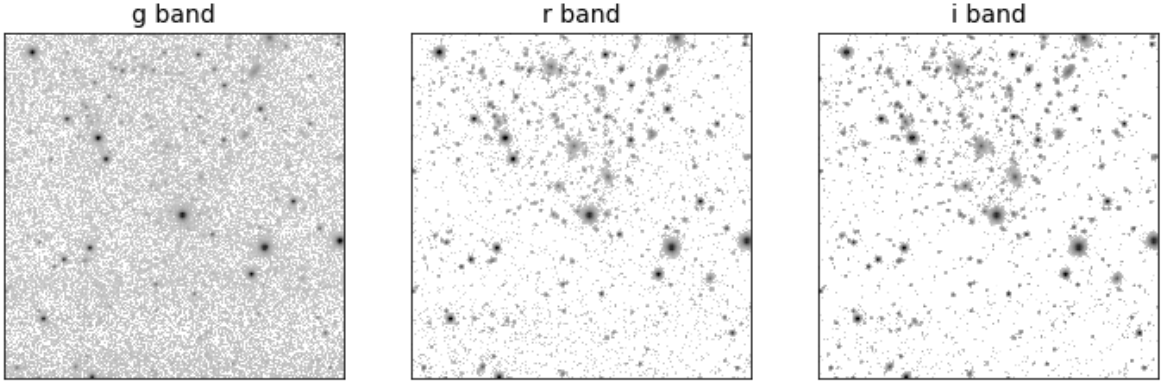


4

...

Queries peak locations for image cutouts

5

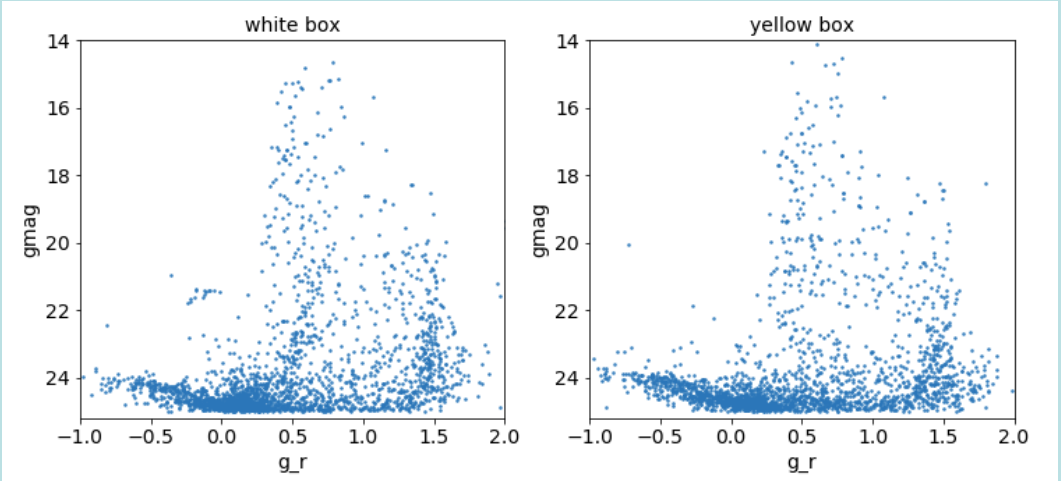


Stores all results in virtual storage…

…and repeat!

7

Queries peak locations for full photometry

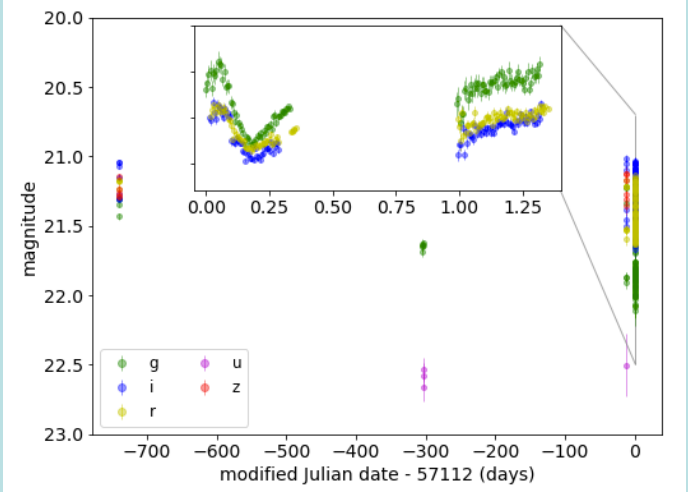


6

Example: Detecting variables

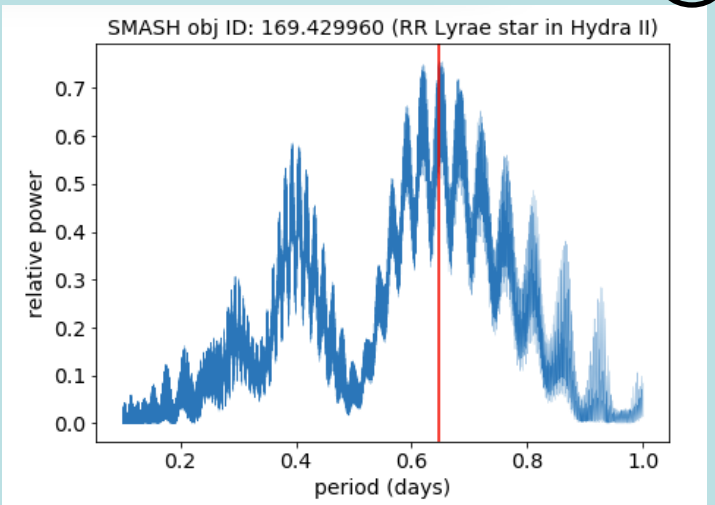
Retrieve light curve of candidate Hydra II variable

1



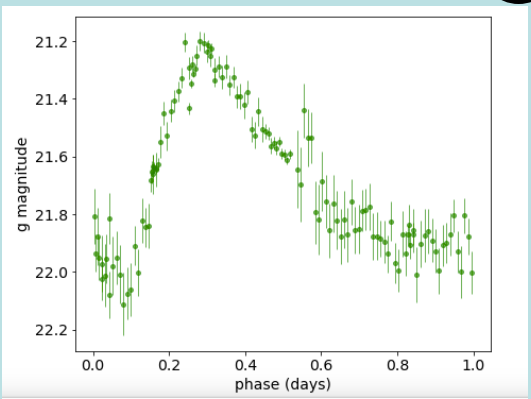
Apply Lomb-Scargle

2



Fold light curve

3

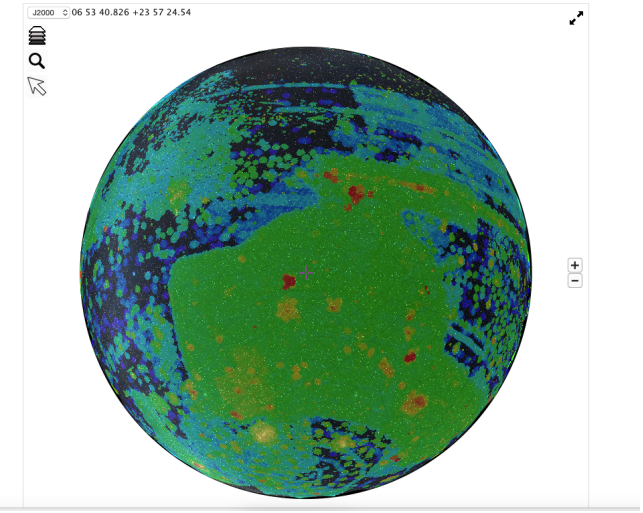


Identify more variables through statistical techniques!

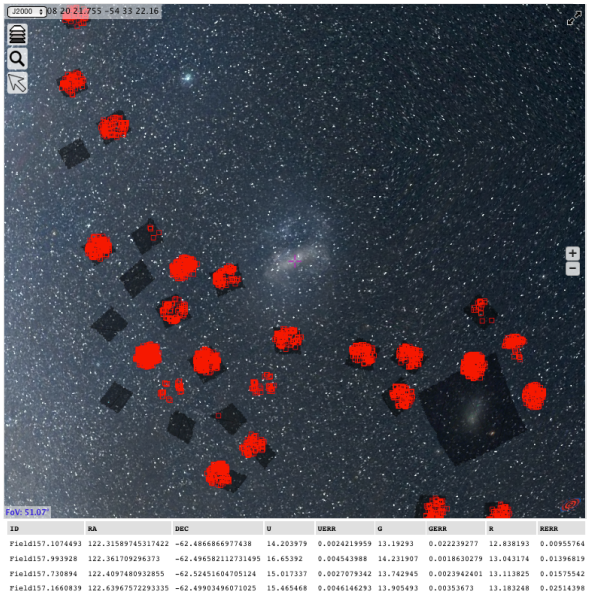
4

Exploring the sky

Images

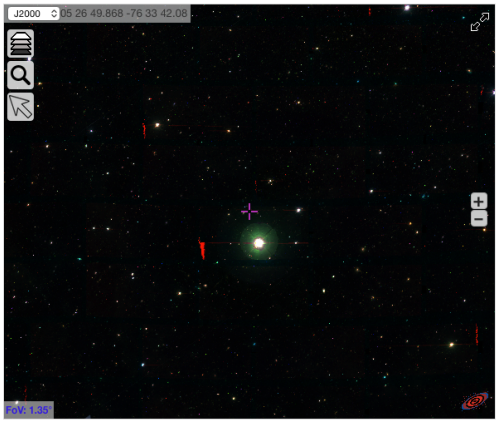


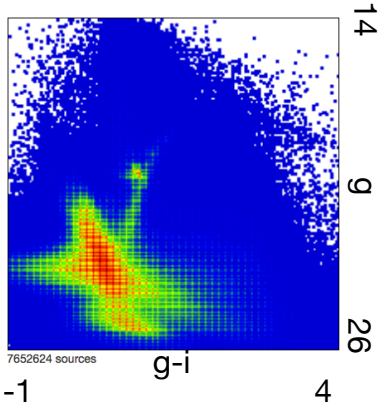
Catalogs



ID	RA	DEC	U	UERR	G	GERR	R	RERR
Field157.1074493	122.31589745317422	-62.4866866977438	14.203979	0.0024219959	13.19293	0.022239277	12.838193	0.00955764
Field157.993928	122.36170296373	-62.4965821127314 95	16.65392	0.004543988	14.231907	0.0018630279	13.043174	0.01396819
Field157.730894	122.40974809328 55	-62.52451604705124	15.017337	0.0027079342	13.742945	0.0023942401	13.113825	0.01575542
Field157.1660839	122.63967572293335	-62.49903496071025	15.465468	0.0046146293	13.905493	0.00353673	13.183248	0.02514398

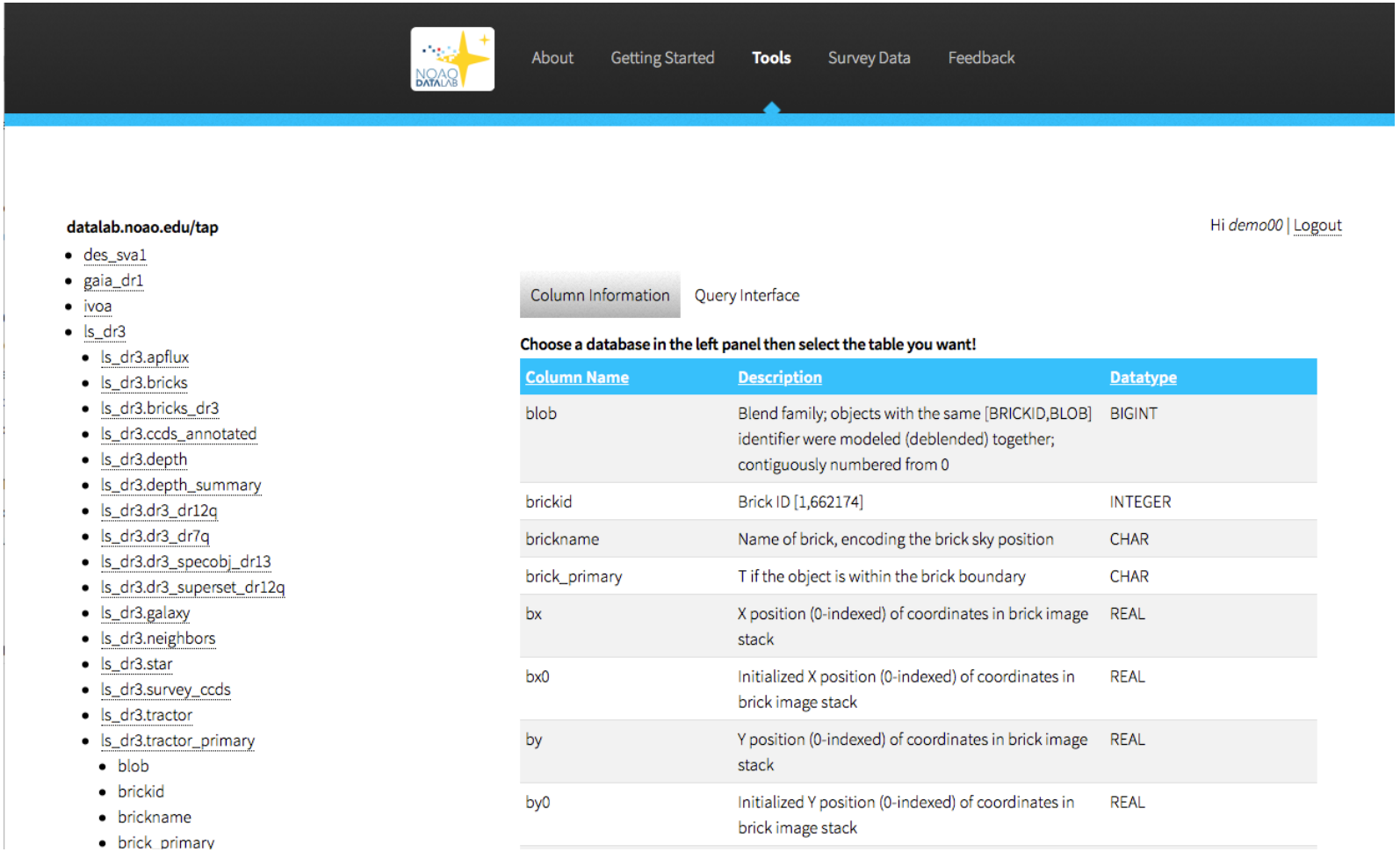
Catalog visualization (prototype)





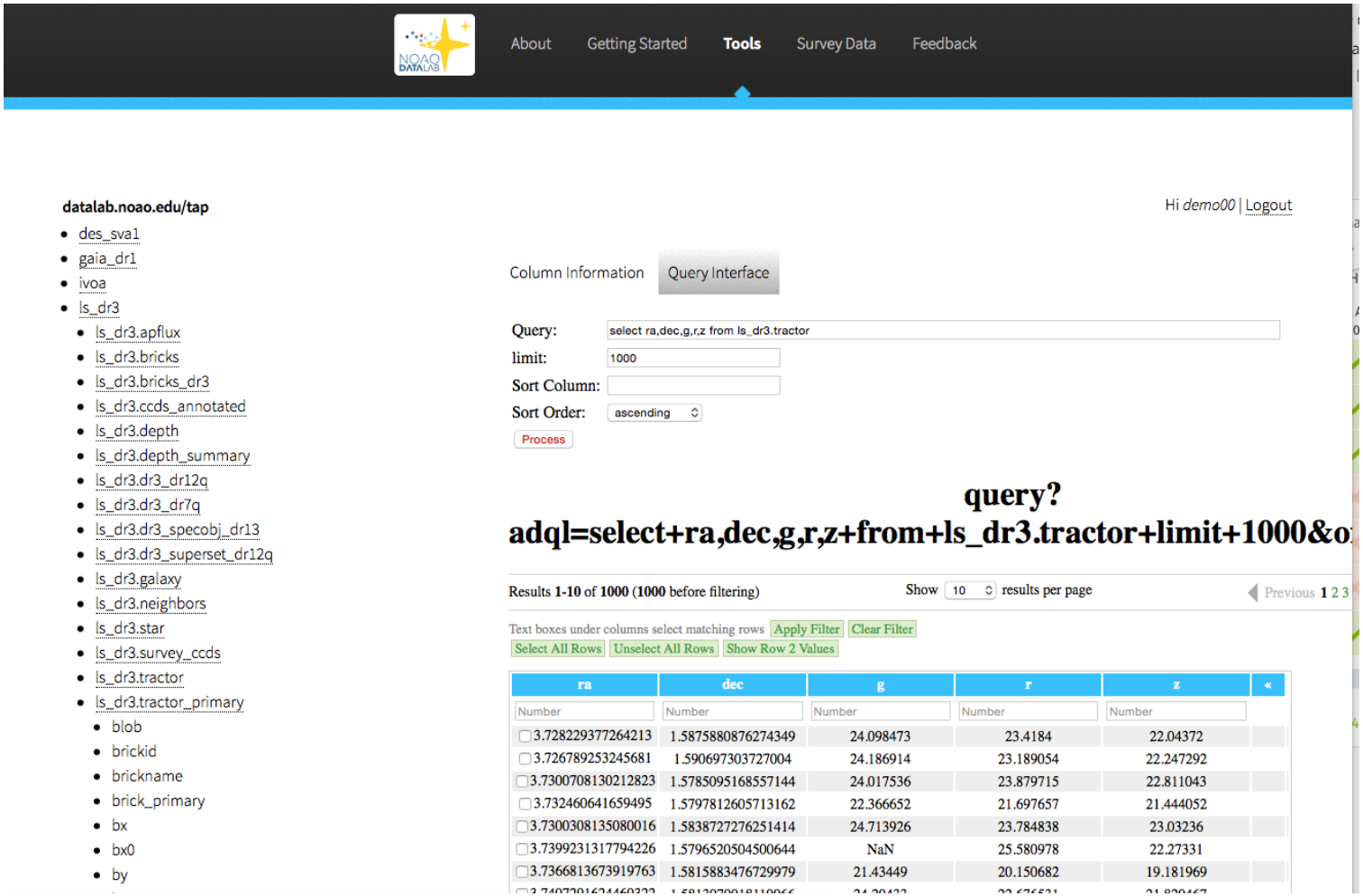
Querying the catalogs

- Through the Data Lab website:



Querying the catalogs

- Through the Data Lab website:



Querying the catalogs

- Through the datalab command:

```
[kolsen@gp02 ~]$ datalab login user=demo00 password=[redacted]
Welcome to the Data Lab, demo00
[kolsen@gp02 ~]$ datalab query sql="select * from usno.a2 limit 10"
id,raj2000_,dej2000_,actflag,mflag,bmag,rmag,epoch,raj2000,dej2000
0150-00069690,00:14:47.196,-68:49:48.92, , ,19.6,17.9,1981.81,3.696648,-68.830256
0150-00070481,00:14:54.972,-68:49:58.22, , ,19.8,18,1981.81,3.72905,-68.832839
0150-00069562,00:14:45.900,-68:49:37.66, , ,18,17.8,1981.81,3.69125,-68.827128
0150-00069750,00:14:47.844,-68:49:29.41, , ,19.4,18,1981.81,3.699348,-68.824837
0150-00070904,00:14:59.041,-68:49:25.26, , ,20.2,18,1981.81,3.746003,-68.823684
0150-00072260,00:15:12.458,-68:54:06.12, , ,18.9,17.1,1981.81,3.801909,-68.9017
0150-00072812,00:15:17.694,-68:54:09.03, , ,16.4,15.2,1981.81,3.823725,-68.902509
0150-00072863,00:15:18.280,-68:53:21.92, , ,17.7,16.5,1981.81,3.826164,-68.889423
0150-00073055,00:15:20.016,-68:53:23.36, , ,18.7,17.5,1981.81,3.8334,-68.889823
0150-00074055,00:15:29.570,-68:54:38.01, , ,19.3,18,1981.81,3.873206,-68.910559

[kolsen@gp02 ~]$ datalab query sql="select * from usno.a2 limit 10" out="mydb://usno_test2"
[kolsen@gp02 ~]$ datalab query sql="select * from usno.a2 limit 10" out="vos://foo2.csv"
[kolsen@gp02 ~]$
```

Querying the catalogs

- Through the Python queryClient module:

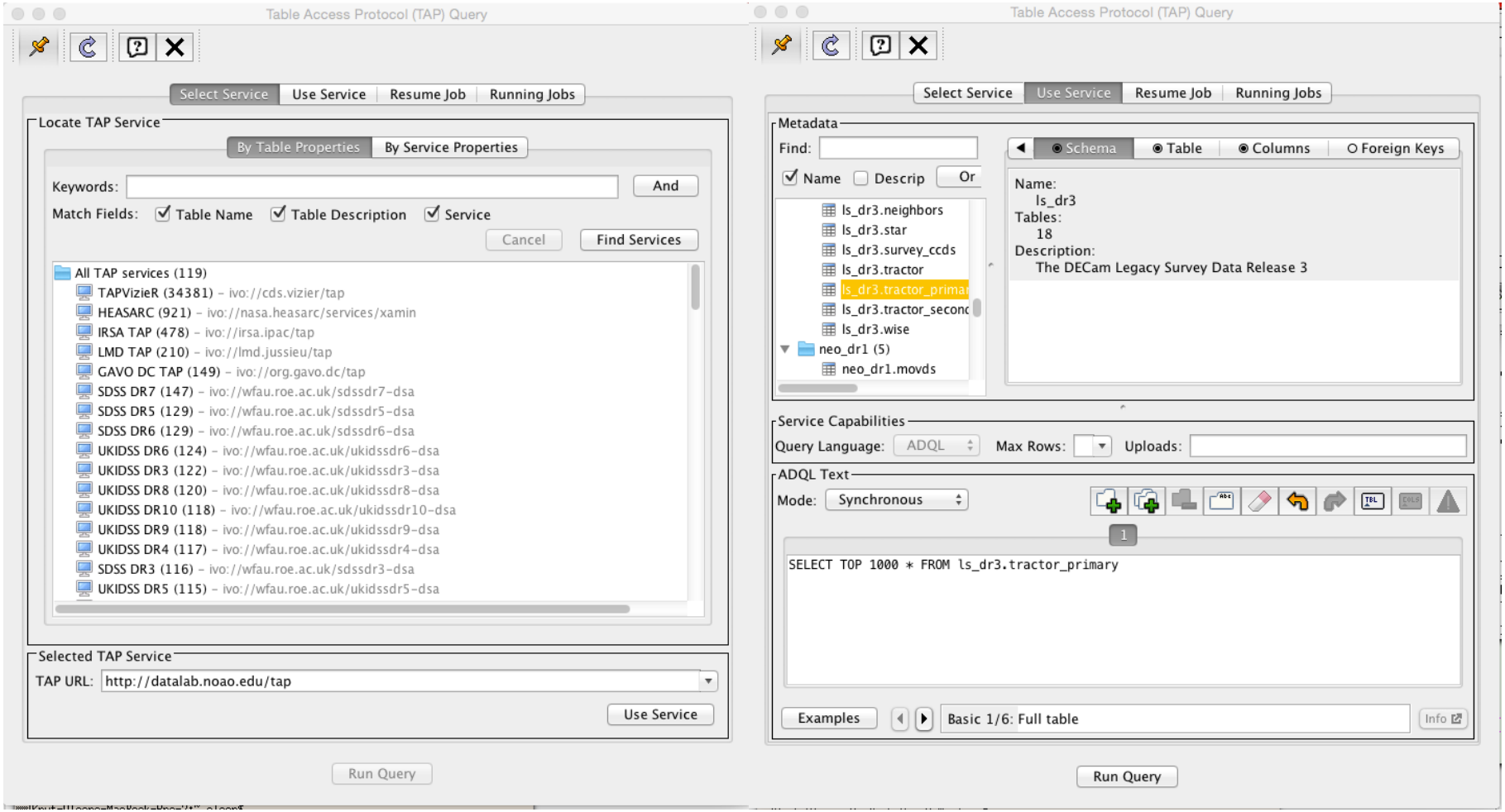
In [4]:
```
from dl import authClient, queryClient
from getpass import getpass
token = authClient.login(raw_input('Enter username: '),getpass('Enter password: '))
```

In [29]:
```
%%time
query="SELECT id,ra,dec,gmag,rmag FROM smash_dr1.object WHERE fieldid=169 LIMIT 100"
try:
  response = queryClient.query(token, sql = query, fmt = 'csv')
except Exception as e:
  print e.message
  raise
print response[:205]
```

```
id,ra,dec,gmag,rmag
169.458572,185.342365895208,-32.1201617232873,24.8856,24.6991
169.460663,185.348188180985,-32.1200524648251,24.665,24.5361
169.1065651,185.353177442806,-32.1208638198927,25.0639,24.6239
CPU times: user 7.4 ms, sys: 956 µs, total: 8.36 ms
Wall time: 53 ms
```

Querying the catalogs

- Through TOPCAT:



Querying the images

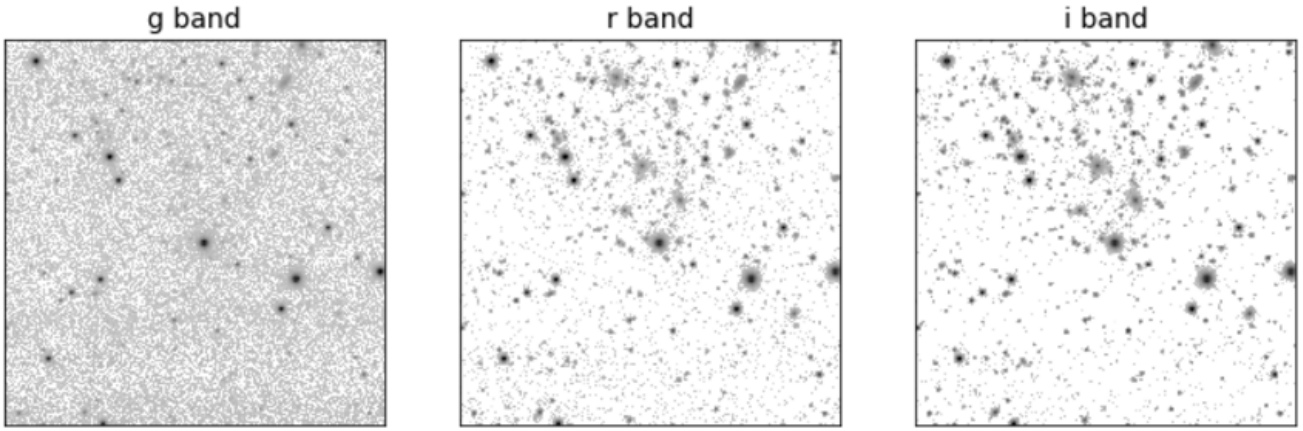
In [14]: Slide Type -

```
bands = list('gri')
images = download_deepest_images(tbl['ra'][1], tbl['dec'][1], fov=0.07, bands=bands) # FOV in de
```

```
The full image list contains 2514 entries
Band g: downloading deepest stacked image...
Band r: downloading deepest stacked image...
Band i: downloading deepest stacked image...
Downloaded 3 images.
```

In [15]: Slide Type -

```
plot_images(images,bands=bands)
```



Saving the results

- myDB:

```
In [29]: query = "select * from usno.b1 limit 1000"
         try:
             response = queryClient.query (token, adql=query, fmt='csv',
                                           out='mydb://mags3')
             #queryClient.list (token, table='mydb://mags3')
         except Exception as e:
             # Handle any errors in the query.  By running this cell multiple times with the same
             # output file, or by using a bogus SQL statement, you can view various error messages.
             print (e.message)
         else:
             if response is not None:
                 print (response)            # print the response
             else:
                 print ("OK")

         http://dlsvcs.datalab.noao.edu/query/list?table=mydb://mags3
         OK
```

```
datalab query sql="select * from usno.a2 limit 10" out="mydb://usno_test2"
```

Saving the results

- Virtual storage:

```
try:
    response = queryClient.query (token, adql=query, fmt='csv',
                                  out='vos://mags.csv')
except Exception as e:
    # Handle any errors in the query.  By running this cell multiple times with the same
    # output file, or by using a bogus SQL statement, you can view various error messages.
    print (e.message)
else:
    if response is not None:
        print (response)           # print the response
    else:
        print ("OK")

# Remove the file we just created, but list it first to show it exists
storeClient.ls (token, name='vos://mags.csv')
storeClient.rm (token, name='vos://mags.csv')
```

```
datalab query sql="select * from usno.a2 limit 10" out="vos://foo2.csv"
```

Virtual storage

- File transfer:

```
[kolsen@gp02 ~]$ datalab put fr=/etc/hosts to=vos://
(1 / 1) /etc/hosts -> vos://hosts
[kolsen@gp02 ~]$ datalab get fr=vos://hosts to=/tmp/myhosts
(1/1) [====================] [   281B] hosts
```

Try it out and get in touch!

- Web: datalab.noao.edu
- Email: datalab@noao.edu
- GitHub: https://github.com/noao-datalab
- Twitter: @NOAODataLab