



The NOAO Data Lab Project Introduction

Knut Olsen
for the Data Lab team

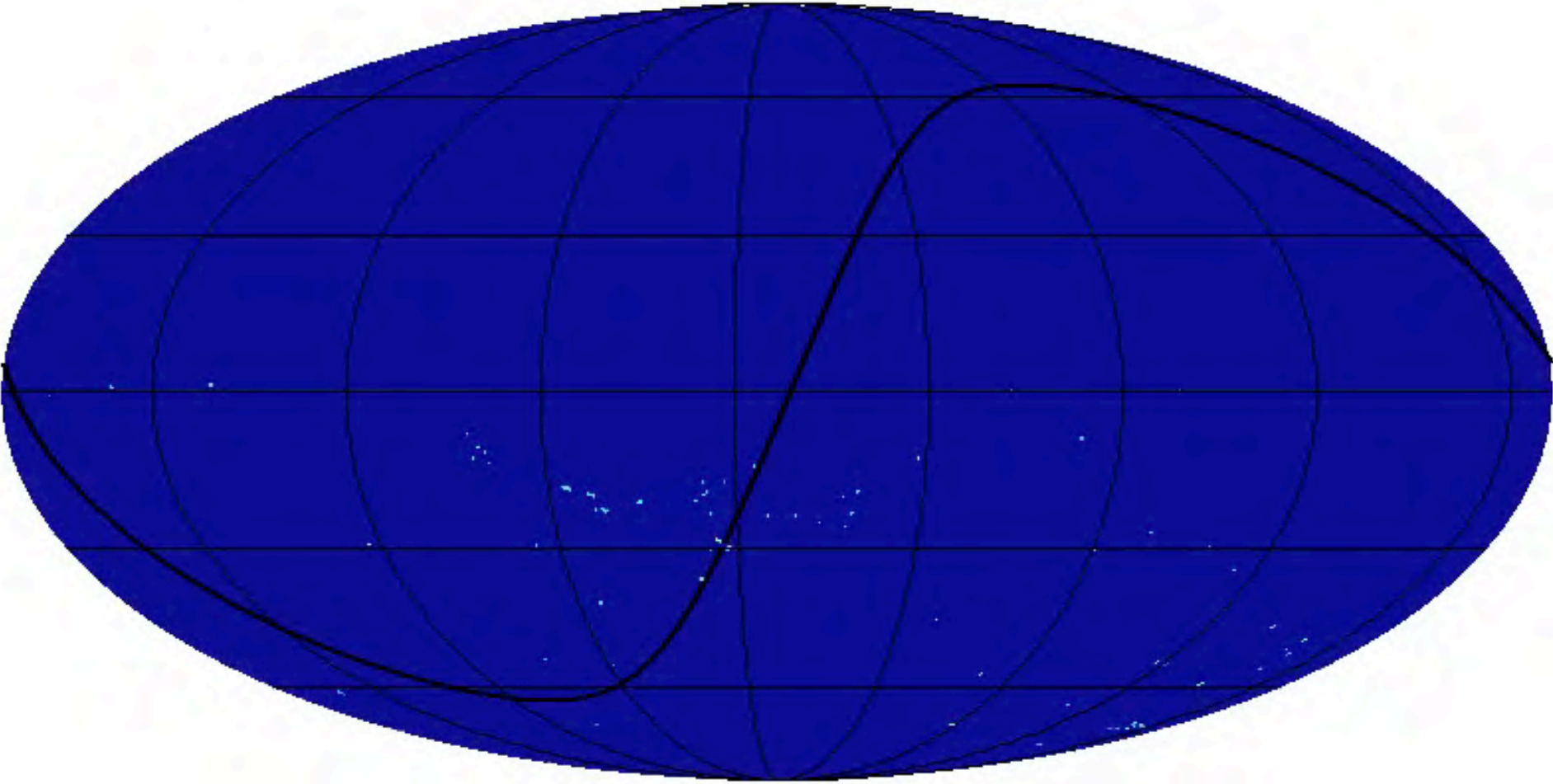


Current team:

- 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

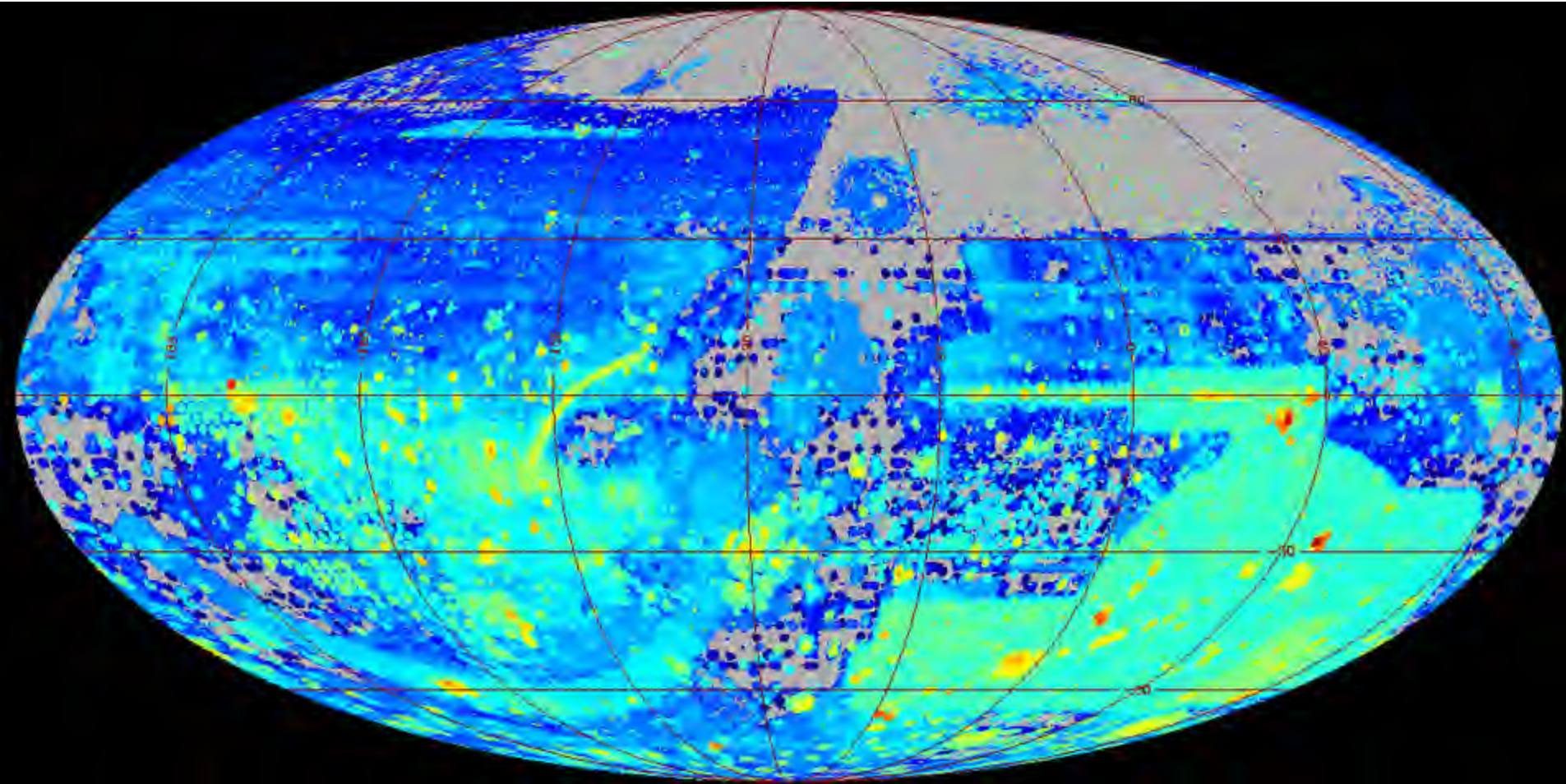


NOAO wide field imaging data over time

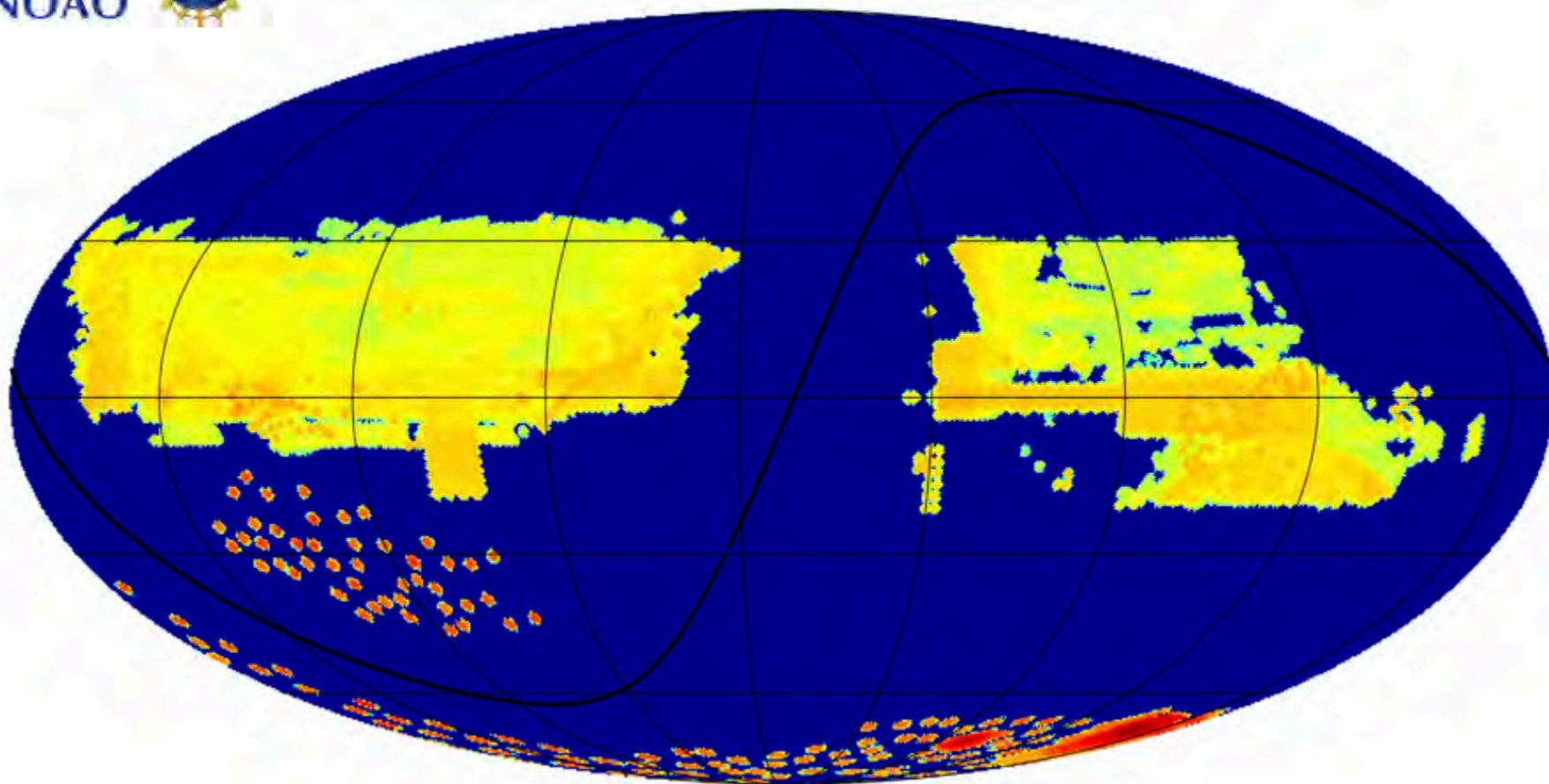




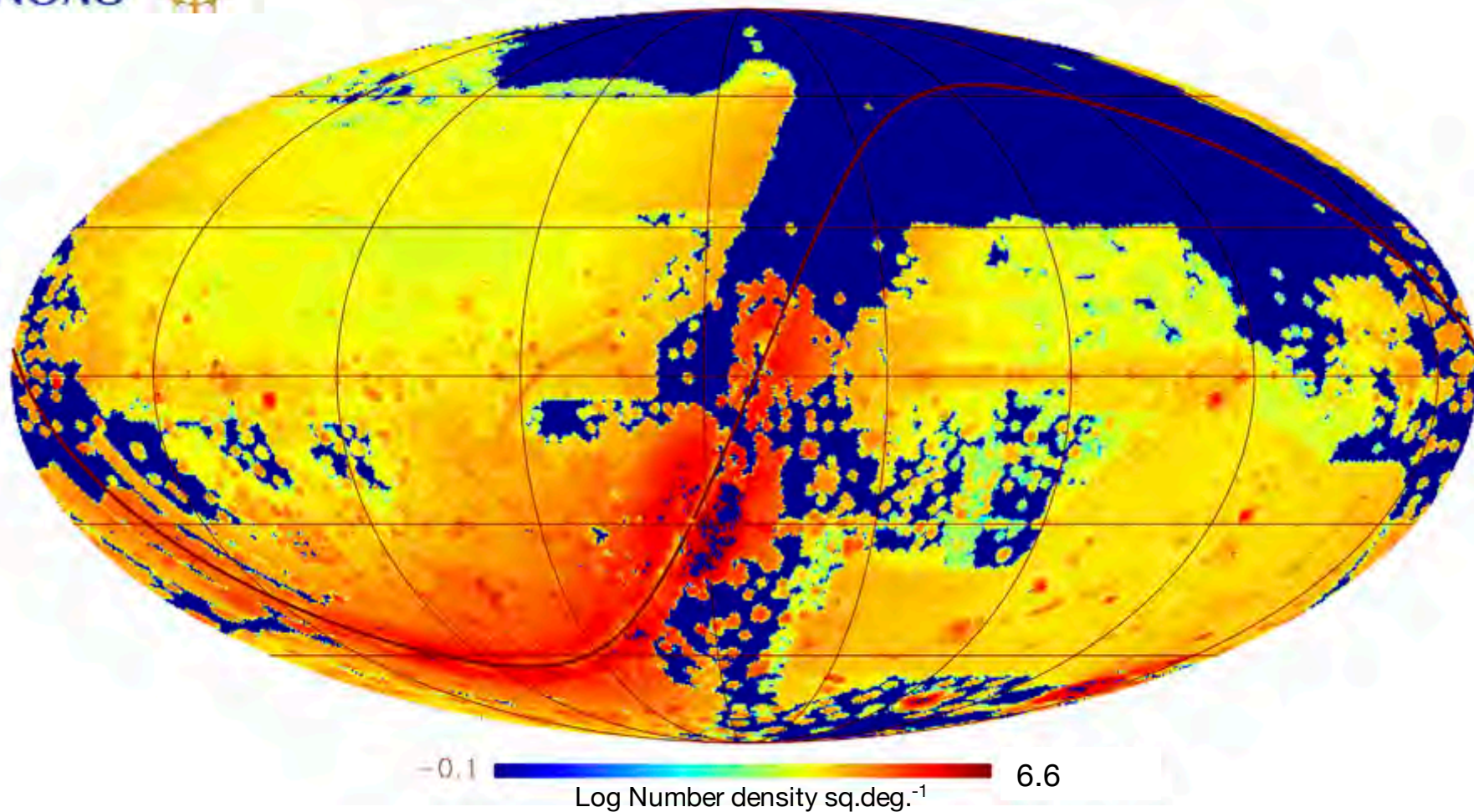
DECam and Mosaic data in February 2017



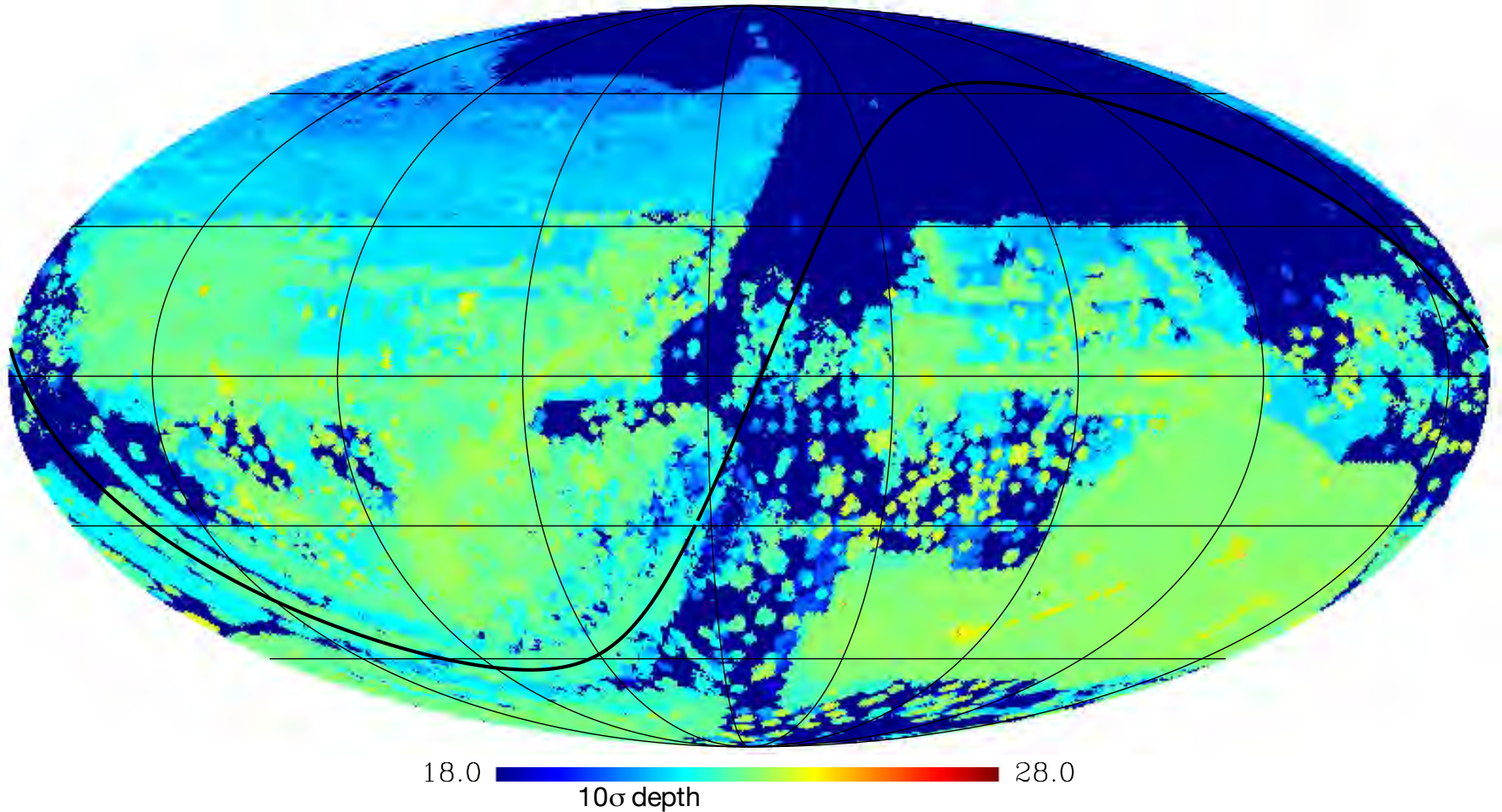
DECaLS DR3 and SMASH Catalogs



- 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



- 2.5 billion objects, 20 billion measurements; aperture-based photometry
- Available soon



- 2.5 billion objects, 20 billion measurements; aperture-based photometry
- Available soon



Data Volume and Complexity

~500 TB (February 2017) of on-target imaging data ($t_{\text{exp}} > 30\text{s}$) 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



- **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



Large Catalogs – Data Lab will serve TB-scale databases

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

Virtual Storage – Minimizes 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, helpers module
Catalog query	Web interface datalab command Python queryClient, helpers module TOPCAT
Image query	Simple Image Access service
Query result storage	myDB Virtual storage space
File transfer	datalab command and Virtual storage space
Analysis	Jupyter notebook server



Major Milestones

- March 2015: Conceptual Design Review
 - Lisa Storrie-Lombardi (Chair), Severin Gaudet, Zeljko Ivezic, Connie Rockosi, Beth Willman reviewed Science Case & Requirements, System Architecture, Operations Concept & Requirements, and Schedule
- Fall 2015 hiring campaign
- June 2016 San Diego AAS Demo
- August 2016 Interim Review
 - Lisa Storrie-Lombardi (Chair), Severin Gaudet, Zeljko Ivezic, Ed Olszewski, Beth Willman, and Dennis Zaritsky reviewed progress and Year 2 plan
- January 2017 AAS SMASH DR1 and DECaLS DR3
- Summer 2017 first public release
- End 2017/Early 2018 NOAO All-Sky Catalog, Legacy Survey DR4/5, DES DR1

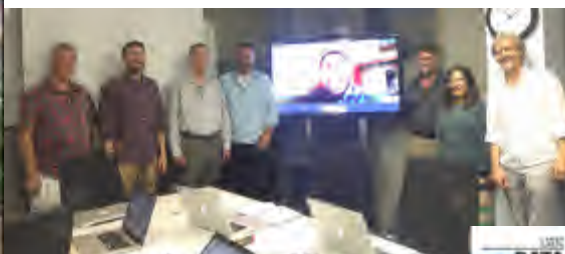
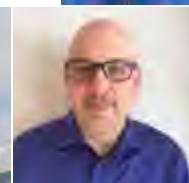
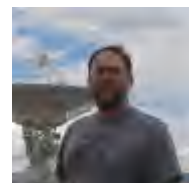
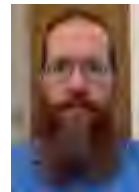
Outreach activities

- San Diego AAS Demo (June 2016)
- Internal early adopter program (Sep 2016 – present)
- Big Data Academy Science Café sessions (February & April 2017)
- Demo and hacking at Detecting the Unexpected workshop (March 2017)
- LSST Data Science Fellowship program presentation and hack session (April 2017)
- Tucson local Tutorial (2 sessions, May 8 2017)
- Presentation and hacking at Time-domain Alert Science workshop (May 23 2017)
- Hack session at DESI Collaboration Meeting (June 2017)
- Detecting Dwarf Galaxies LSSTC Workshop (Fall 2017)



Public release 2017!

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



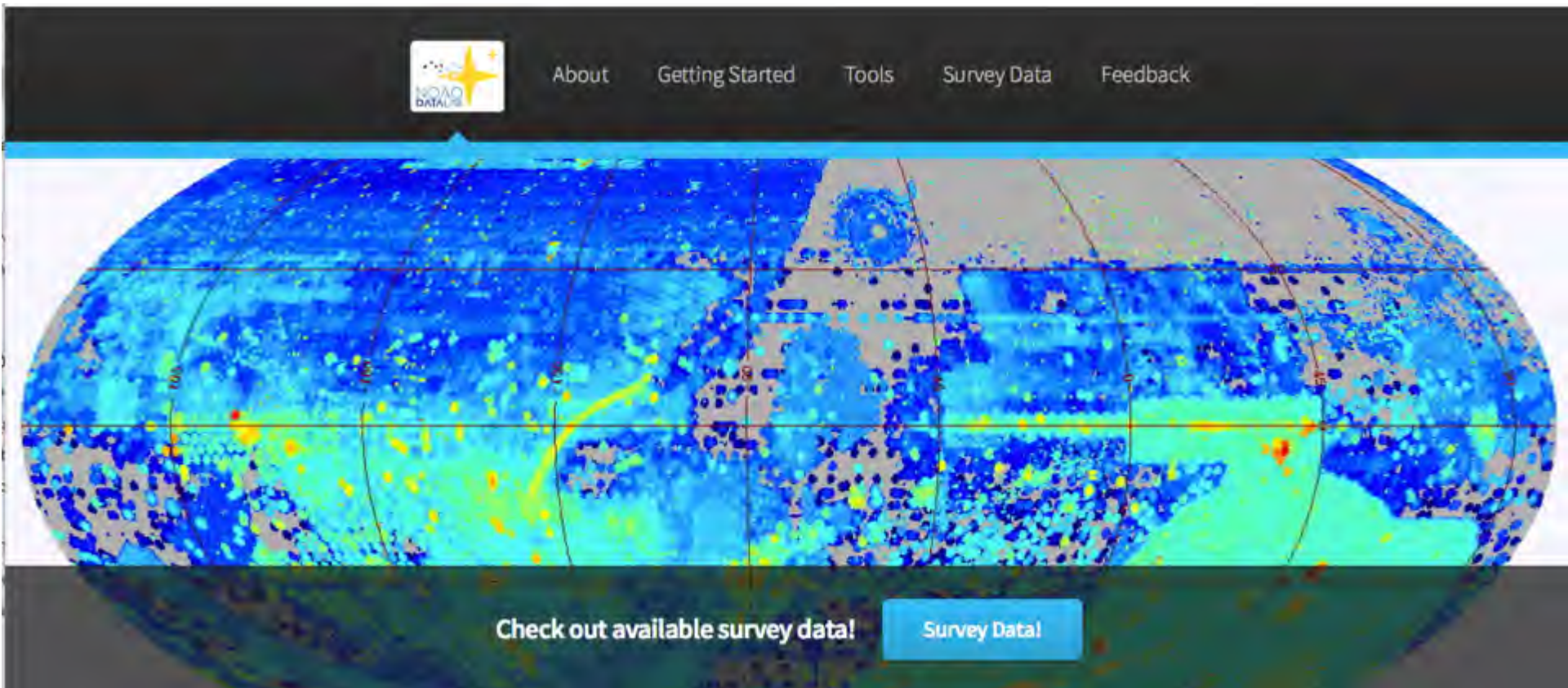
- Star/galaxy/QSO separation (Juneau)
- Hydra II and RR Lyrae star discovery (Nikutta)

Detailed slides

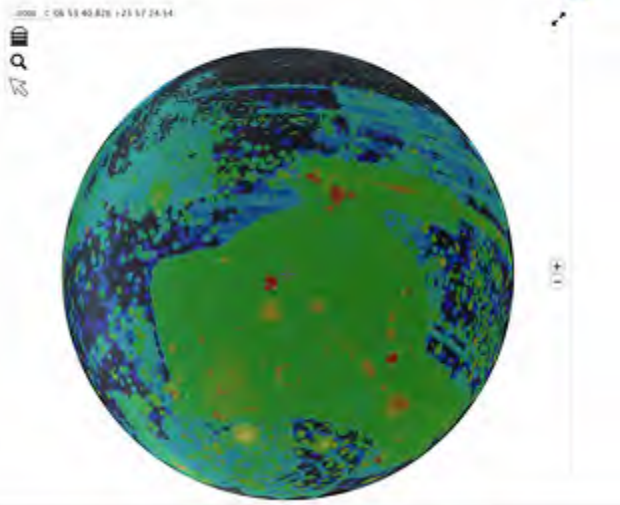


Getting started with the Data Lab

- datalab.noao.edu/tutdev



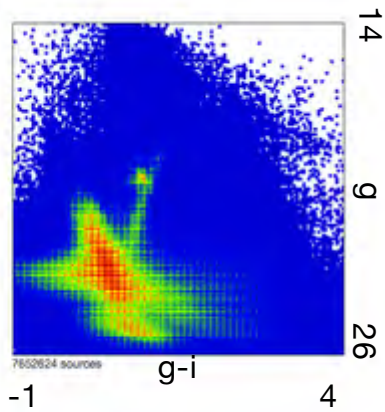
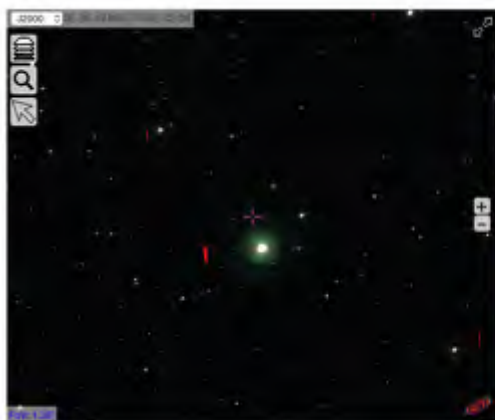
Images



Catalogs



Catalog visualization (prototype)





Querying the catalogs

- Through the Data Lab website:

The screenshot shows the NOAO Data Lab website interface. At the top, there is a navigation bar with links for 'About', 'Getting Started', 'Tools', 'Survey Data', and 'Feedback'. Below the navigation bar, the main content area is divided into two panels. On the left, there is a list of databases under the heading 'datalab.noao.edu/tap'. On the right, there is a 'Column Information' panel with a 'Query Interface' tab. The 'Column Information' panel displays a table of columns with their names, descriptions, and datatypes.

datalab.noao.edu/tap

- [des_sval](#)
- [gaia_dr1](#)
- [ivoa](#)
- [ls_dr3](#)
 - [ls_dr3.apflux](#)
 - [ls_dr3.bricks](#)
 - [ls_dr3.bricks_dr3](#)
 - [ls_dr3.ccds_annotated](#)
 - [ls_dr3.depth](#)
 - [ls_dr3.depth_summary](#)
 - [ls_dr3.dr3_dr12q](#)
 - [ls_dr3.dr3_dr7q](#)
 - [ls_dr3.dr3_specobj_dr13](#)
 - [ls_dr3.dr3_superset_dr12q](#)
 - [ls_dr3.galaxy](#)
 - [ls_dr3.neighbors](#)
 - [ls_dr3.star](#)
 - [ls_dr3.survey_ccds](#)
 - [ls_dr3.tractor](#)
 - [ls_dr3.tractor_primary](#)
 - blob
 - brickid
 - brickname
 - brick_primary

Column Information | Query Interface

Choose a database in the left panel then select the table you want!

Column Name	Description	Datatype
blob	Blend family; objects with the same [BRICKID,BLOB] identifier were modeled (deblended) together; contiguously numbered from 0	BIGINT
brickid	Brick ID [1,662174]	INTEGER
brickname	Name of brick, encoding the brick sky position	CHAR
brick_primary	T if the object is within the brick boundary	CHAR
bx	X position (0-indexed) of coordinates in brick image stack	REAL
bx0	Initialized X position (0-indexed) of coordinates in brick image stack	REAL
by	Y position (0-indexed) of coordinates in brick image stack	REAL
by0	Initialized Y position (0-indexed) of coordinates in brick image stack	REAL



Querying the catalogs

- Through the Data Lab website:

The screenshot shows the Data Lab website interface. On the left, there is a navigation menu with a tree structure under 'datalab.noao.edu/tap'. The main content area has a dark header with the 'NOAO DATALAB' logo and navigation links: 'About', 'Getting Started', 'Tools', 'Survey Data', and 'Feedback'. Below the header, there are two tabs: 'Column Information' and 'Query Interface'. The 'Query Interface' tab is active, showing a query input field with the text 'select ra,dec,g,r,z from ls_dr3.tractor', a 'limit' field set to '1000', and a 'Sort Order' dropdown set to 'ascending'. A 'Process' button is visible below the query fields. Below the query interface, there is a large text overlay that reads 'query?' and 'adql=select+ra,dec,g,r,z+from+ls_dr3.tractor+limit+1000&o'. Below this, the results are displayed as a table with 10 columns: 'ra', 'dec', 'b', 'r', 'z', and a dropdown arrow. The table contains 10 rows of data, each with a checkbox in the 'ra' column. The results are labeled 'Results 1-10 of 1000 (1000 before filtering)'. At the bottom of the results, there are links for 'Apply Filter', 'Clear Filter', 'Select All Rows', 'Unselect All Rows', and 'Show Row 2 Values'.

datalab.noao.edu/tap

- des_sval
- gaia_dr1
- ivoa
- ls_dr3
 - ls_dr3.apflux
 - ls_dr3.bricks
 - ls_dr3.bricks_dr3
 - ls_dr3.ccds_annotated
 - ls_dr3.depth
 - ls_dr3.depth_summary
 - ls_dr3.dr3_dr12q
 - ls_dr3.dr3_dr7q
 - ls_dr3.dr3_specobj_dr13
 - ls_dr3.dr3_superset_dr12q
 - ls_dr3.galaxy
 - ls_dr3.neighbors
 - ls_dr3.star
 - ls_dr3.survey_ccds
 - ls_dr3.tractor
 - ls_dr3.tractor_primary
 - blob
 - brickid
 - brickname
 - brick_primary
 - bx
 - bx0
 - by

Hi demo00 | [Logout](#)

Column Information Query Interface

Query:

limit:

Sort Column:

Sort Order:

[Process](#)

query?

adql=select+ra,dec,g,r,z+from+ls_dr3.tractor+limit+1000&o

Results 1-10 of 1000 (1000 before filtering) Show 10 results per page [Previous](#) 1 2 3

Text boxes under columns select matching rows [Apply Filter](#) [Clear Filter](#)
[Select All Rows](#) [Unselect All Rows](#) [Show Row 2 Values](#)

ra	dec	b	r	z	
Number	Number	Number	Number	Number	
<input type="checkbox"/> 3.728229377264213	1.5875880876274349	24.098473	23.4184	22.04372	
<input type="checkbox"/> 3.726789253245681	1.590697303727004	24.186914	23.189054	22.247292	
<input type="checkbox"/> 3.7300708130212823	1.5785095168557144	24.017536	23.879715	22.811043	
<input type="checkbox"/> 3.732460641659495	1.5797812605713162	22.366652	21.697657	21.444052	
<input type="checkbox"/> 3.7300308135080016	1.5838727276251414	24.713926	23.784838	23.03236	
<input type="checkbox"/> 3.7399231317794226	1.5796520504500644	NaN	25.580978	22.27331	
<input type="checkbox"/> 3.7366813673919763	1.5815883476729979	21.43449	20.150682	19.181969	



Querying the catalogs

- Through the datalab command:

```
[kolsen@gp02 ~]$ datalab login user=demo00 password=  
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 [27]: from dl import authClient, queryClient
```

```
In [28]: token = authClient.login ('demo00', 'XXXXXXX')
```

```
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 the helpers.py module:

```
In [23]: %%time
          from dl import helpers
          Q = helpers.Querist('demo00', ' ')
          tbl = Q(query, outfmt='table') # or:
          df = Q(query, outfmt='pandas')
```

Returning Astropy Table

Returning Pandas dataframe

CPU times: user 18 ms, sys: 13.3 ms, total: 31.3 ms

Wall time: 131 ms



Querying the catalogs

- Through TOPCAT:

The image displays two screenshots of the Table Access Protocol (TAP) Query interface, showing the process of locating and querying a service.

Left Screenshot: Locate TAP Service

- Buttons: Select Service, Use Service, Resume Job, Running Jobs
- Search Method: **By Table Properties** (selected), By Service Properties
- Keywords: [] And
- Match Fields: Table Name Table Description Service
- Buttons: Cancel, Find Services
- List of All TAP services (119):
 - TAPVizieR (34381) - ivo://cds.vizier/tap
 - HEASARC (921) - ivo://nasa.heasarc/services/xamin
 - IRSA TAP (478) - ivo://irsa.ipac/tap
 - LMD TAP (210) - ivo://lmd.jussieu/tap
 - GAVO DC TAP (149) - ivo://org.gavo.de/tap
 - SDSS DR7 (147) - ivo://wfau.roe.ac.uk/sdssdr7-dsa
 - SDSS DR5 (129) - ivo://wfau.roe.ac.uk/sdssdr5-dsa
 - SDSS DR6 (129) - ivo://wfau.roe.ac.uk/sdssdr6-dsa
 - UKIDSS DR6 (124) - ivo://wfau.roe.ac.uk/ukidssdr6-dsa
 - UKIDSS DR3 (122) - ivo://wfau.roe.ac.uk/ukidssdr3-dsa
 - UKIDSS DR8 (120) - ivo://wfau.roe.ac.uk/ukidssdr8-dsa
 - UKIDSS DR10 (118) - ivo://wfau.roe.ac.uk/ukidssdr10-dsa
 - UKIDSS DR9 (118) - ivo://wfau.roe.ac.uk/ukidssdr9-dsa
 - UKIDSS DR4 (117) - ivo://wfau.roe.ac.uk/ukidssdr4-dsa
 - SDSS DR3 (116) - ivo://wfau.roe.ac.uk/sdssdr3-dsa
 - UKIDSS DR5 (115) - ivo://wfau.roe.ac.uk/ukidssdr5-dsa
- Selected TAP Service: <http://datalab.noao.edu/tap>
- Buttons: Use Service, Run Query

Right Screenshot: Metadata

- Buttons: Select Service, Use Service, Resume Job, Running Jobs
- Metadata View: Schema (selected), Table, Columns, Foreign Keys
- Find: []
- View: Name Descrip Or
- Table List:
 - ls_dr3.neighbors
 - ls_dr3.star
 - ls_dr3.survey_ccds
 - ls_dr3.tractor
 - ls_dr3.tractor_primary** (highlighted)
 - ls_dr3.tractor_secon
 - ls_dr3.wise
 - neo_dr1 (5)
 - neo_dr1.movds
- Table Details for **ls_dr3.tractor_primary**:
 - Name: ls_dr3
 - Tables: 18
 - Description: The DECam Legacy Survey Data Release 3
- Service Capabilities:
 - Query Language: ADQL
 - Max Rows: []
 - Uploads: []
- ADQL Text:
 - Mode: Synchronous
 - Query: `SELECT TOP 1000 * FROM ls_dr3.tractor_primary`
- Buttons: Examples, Basic 1/6: Full table, Info, Run Query

Querying the images

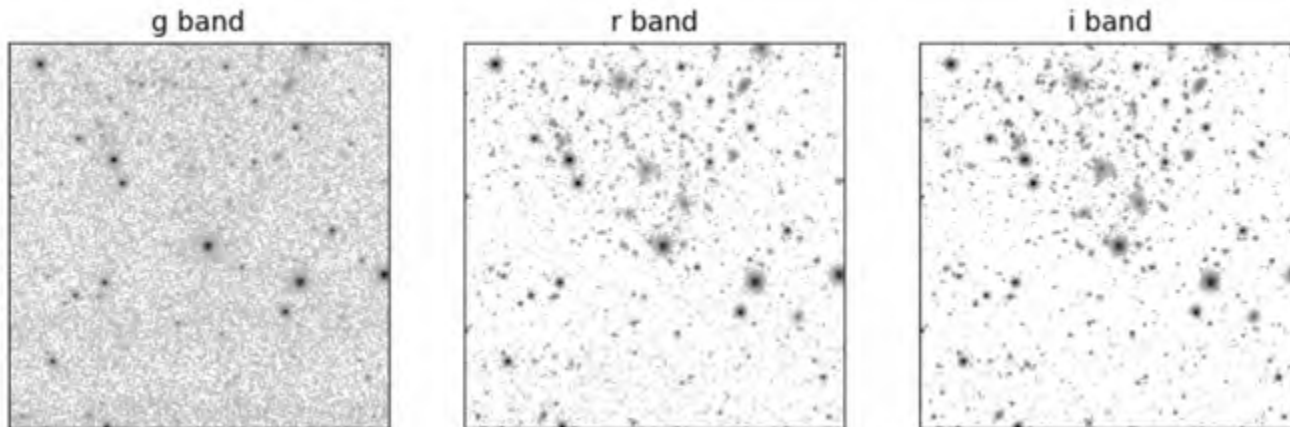
In [14]:

```
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]:

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





Querying the images

```
# set up SIA
from pyvo.dal import sia
#DEF_ACCESS_URL = "http://datalab.noao.edu/sia/smash"
DEF_ACCESS_URL = "http://zeus1.sdm.noao.edu/siapv1"
svc = sia.SIAService(DEF_ACCESS_URL)

# a little func to download the deepest stacked images
def download_deepest_images(ra,dec,fov=0.1,bands=list('gri')):
    imgTable = svc.search((ra,dec), (fov/np.cos(dec*np.pi/180), fov), verbosity=2).votable.to_table
    print "The full image list contains", len(imgTable), "entries"

    sel0 = (imgTable['proctype']=='Stacked') & (imgTable['prodtype']=='image') # basic selection
    images = []
    for band in bands:
        print "Band %s:" % band,
        sel = sel0 & (imgTable['obs_bandpass']==band) # add 'band' to selection
        Table = imgTable[sel] # select
        row = Table[np.argmax(Table['exptime'].data.data.astype('float'))] # pick image with longest exposure
        url = row['access_url'] # get the download URL
        print 'downloading deepest stacked image...'
        img = io.fits.getdata(utils.data.download_file(url,cache=True,show_progress=False,timeout=30))
        images.append(img)

    print "Downloaded %d images." % len(images)
    return images

# multi panel image plotter
def plot_images(images,geo=None,panelsize=4,bands=list('gri'),cmap=matplotlib.cm.gray_r):
    n = len(images)
    if geo is None: geo = (n,1)

    fig = p.figure(figsize=(geo[0]*panelsize,geo[1]*panelsize))
    for j,img in enumerate(images):
        ax = fig.add_subplot(geo[1],geo[0],j+1)
        ax.imshow(img,origin='lower',interpolation='none',cmap=cmap,norm=matplotlib.colors.PowerNorm(1))
```



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" ■
```



- 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
```

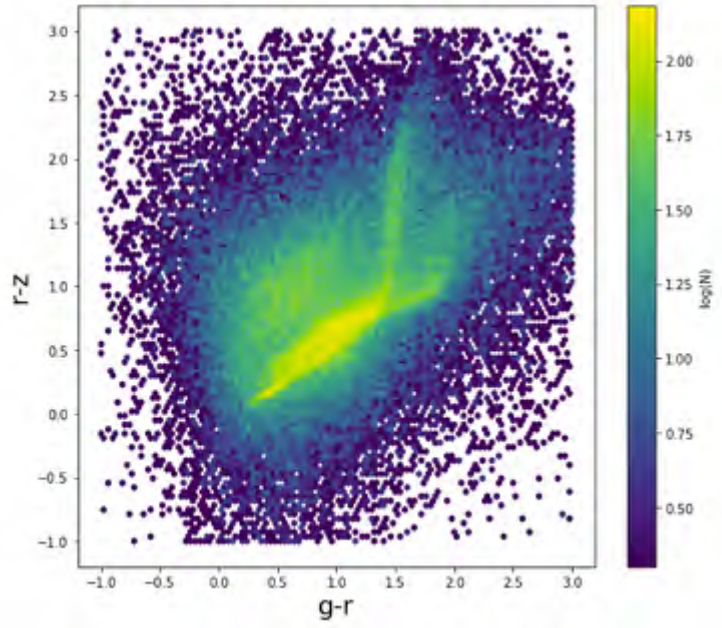
- Jupyter notebook server

Optical Color-Color Diagram

```
In [5]: col0 = trac['g_r'] #g-r color
        coll = trac['r_z'] #r-z color

        # 2D-histogram of objects
        fig, ax1 = plt.subplots(1, 1, figsize=(9, 8))
        im1 = ax1.hexbin(col0, coll, bins='log', cmap=plt.cm.viridis,
                        mincnt=1, extent=(-1., 3, -1., 3))
        ax1.set_ylabel('r-z',fontsize=20)
        ax1.set_xlabel('g-r',fontsize=20)

        #color bar
        cb = plt.colorbar(im1,label='log(N)')
```





Pending: running custom code

- Containerization of tasks
- Job Manager to handle task running, interactions, and completion



Pending: data publication

- Document to provide guidance on schema, metadata for custom imaging, etc.
- Database ingest tools and metadata scraping
- Services to provide e.g. custom image and catalog HiPS generation for sky viewer
- Webpage template