Intro to SciServer

Roman Science Team Community Briefings

2021-11-16

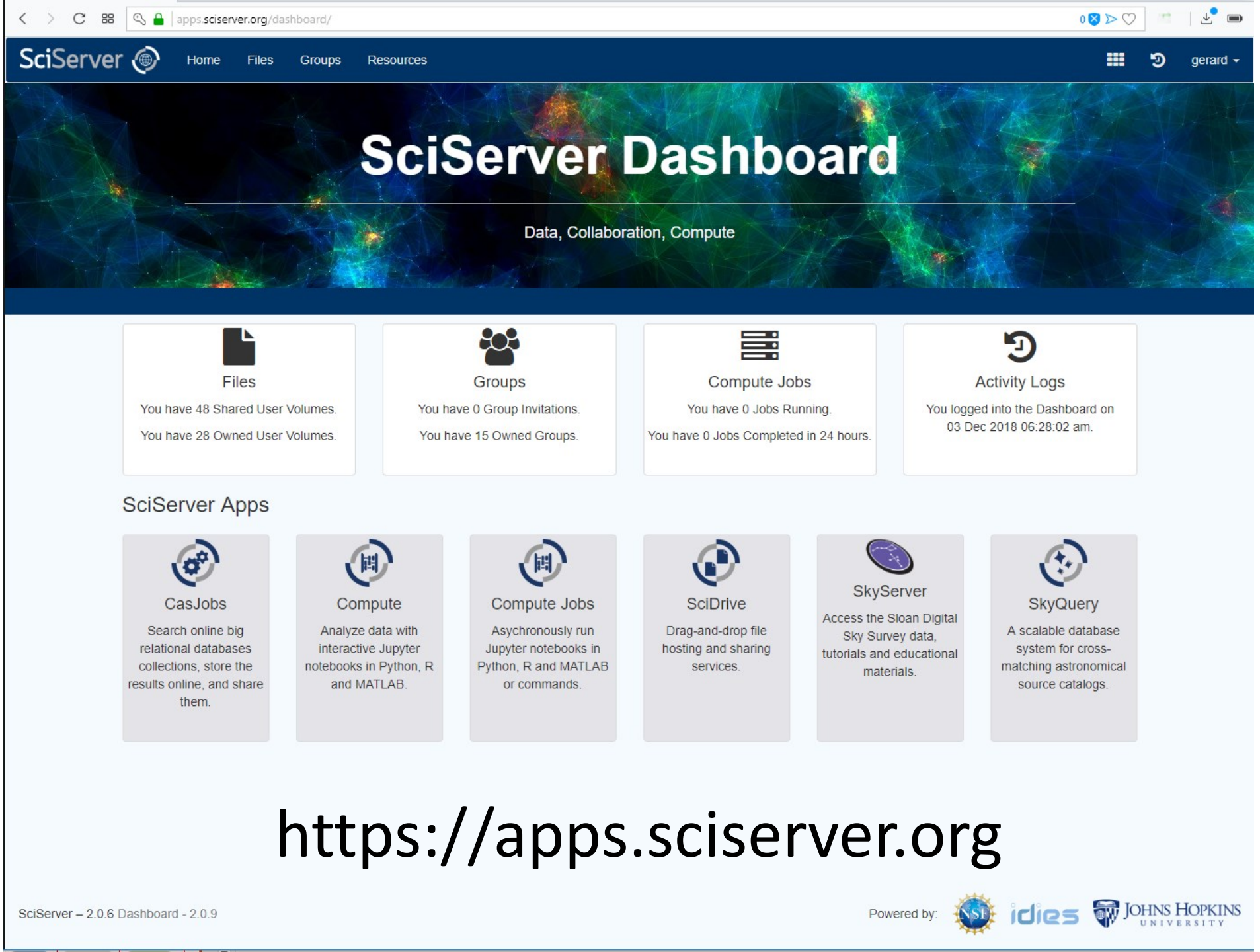
Gerard Lemson (JHU)

From Marc's slides

Goals for Designing Roman Archive Science Tools

- Study and evaluate the best archival practices from across all of astronomy.
- Design and build simple end-to-end simulations. Incorporate these simulations into a scalable, queryable database.
- Establish a common platform for data query interfaces and data exploration.
- Design and prototype a unified object catalog using a simulated catalog, with cross-matches to external surveys.
- Investigate and develop capabilities needed for a GO+GI hybrid archive.
- Identify forward-looking technologies not in production today anywhere (e.g., scripting, fast parallel analysis tools). — Arik
- Develop and build prototypes of highly scalable parallel tools (e.g. cross-correlations inside the archive database). — Bridget
- Implement novel object classification codes based on machine learning techniques.

- Establish a common platform for data query interfaces and data exploration.



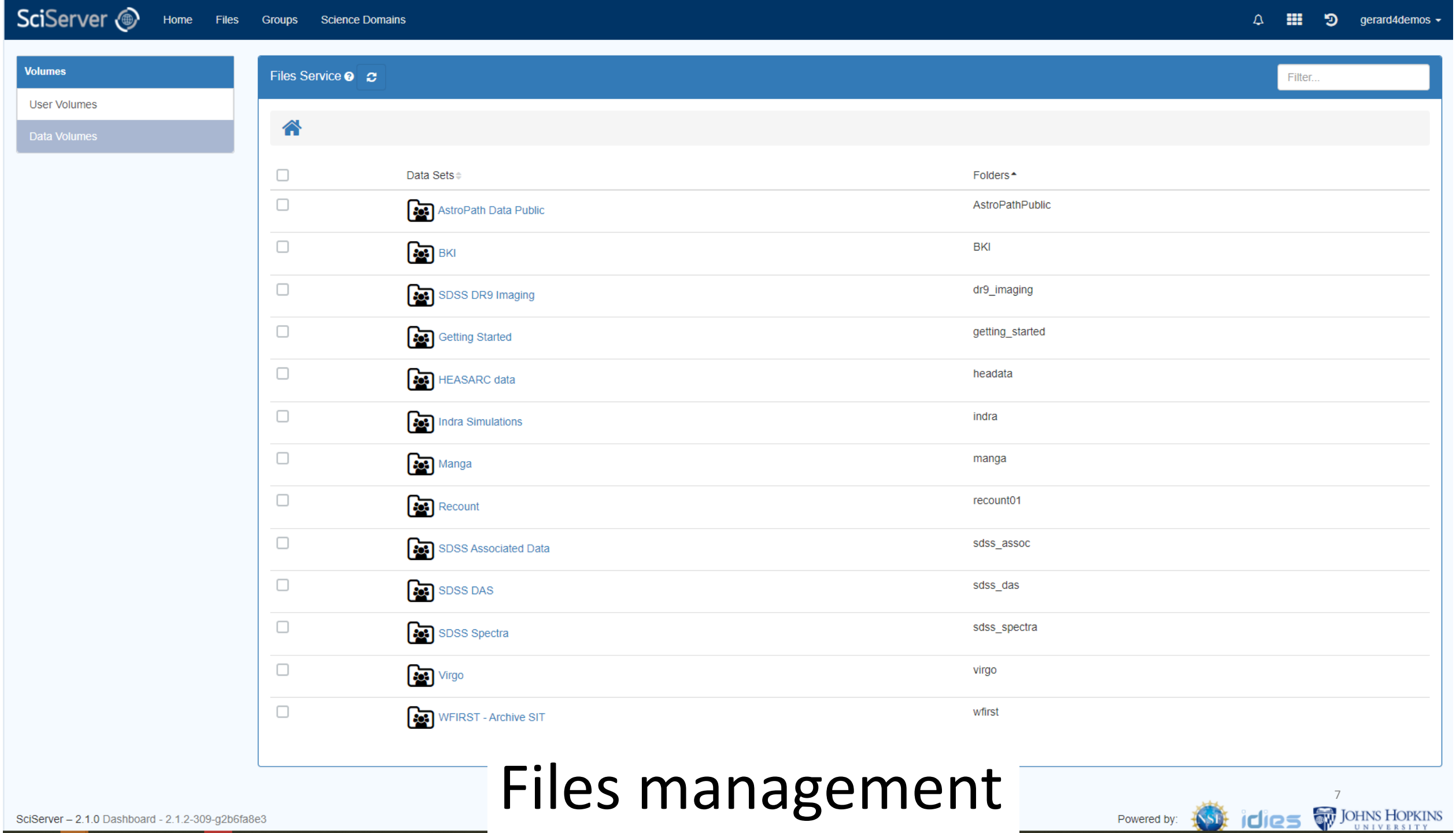
https://apps.sciserver.org

SciServer: a collaborative science platform

- Science Data Hosting (Files and Databases)
  - Query of hosted databases (CasJobs)
  - Data Integration across hosted data sources
- Server-side computational analysis on hosted data:
  - Interactive Jupyter notebooks
  - Batch queues
  - Custom compute environments via Docker containers.
    Incl GPU, Spark, Dask
  - Custom RDB functionality
- Personal Storage (Files and Database)
- Sharing in collaborative user groups
  - Storage volumes, databases, compute images, custom resources
- API Integration

For WFIRST

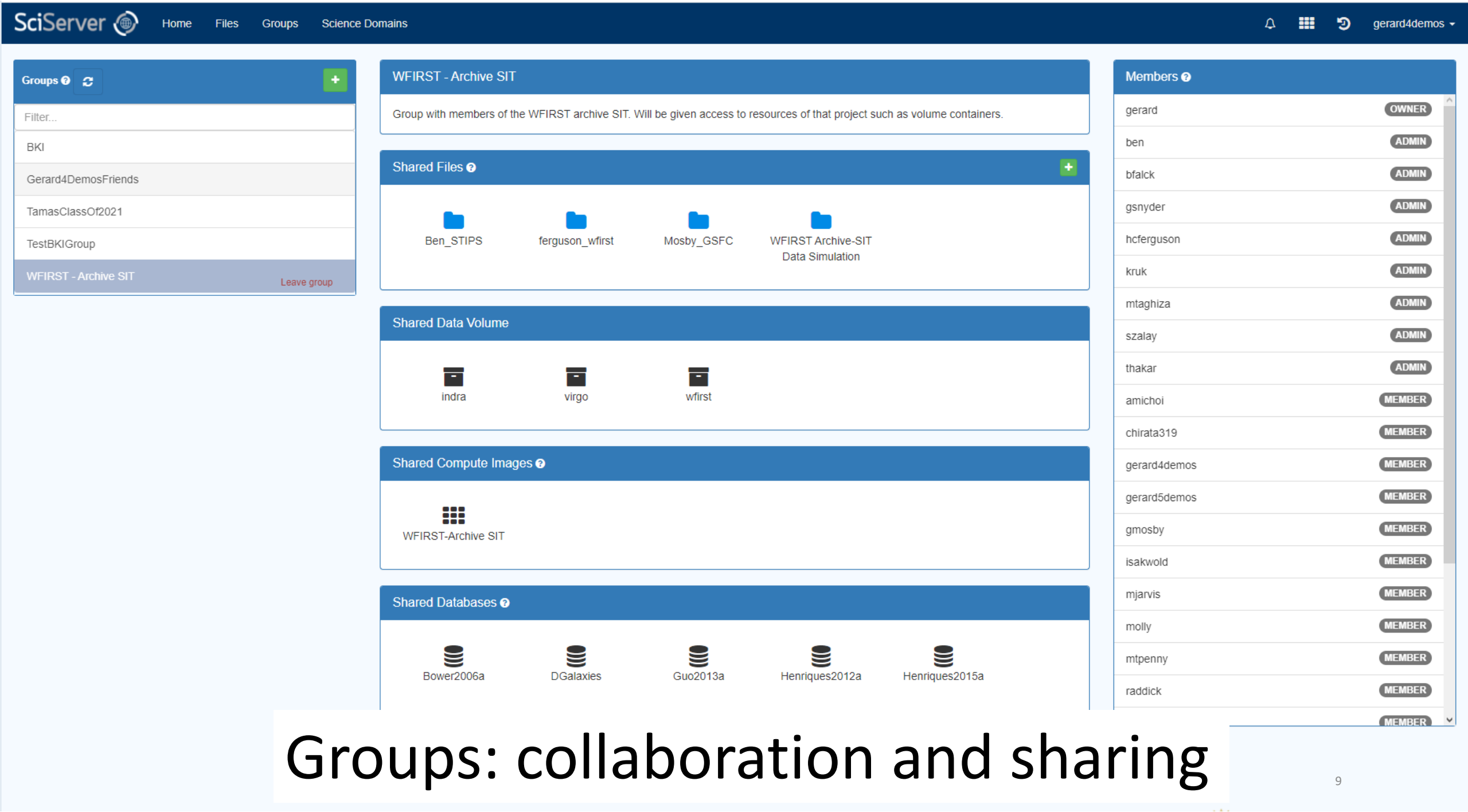
- **WFIRST – Archive SIT data volume**
  - MRObs data sets (pre..post observation images)
  - Illustris light cones
  - Cosmos-3dHST
  - Data used in STIPS
  - Room for more
- **WFIRST Archive SIT** user group
  - Access to WFIRST data volume and compute image
  - Access to Millennium, Indra, Illustris, Eagle databases and raw data
- **WFIRST Archive SIT** compute docker image
  - Millennium Run Observatory
  - Illustris light cone creation (and soon more)
  - Since hackathon 2019: STIPS, GalSim



Files management

For WFIRST

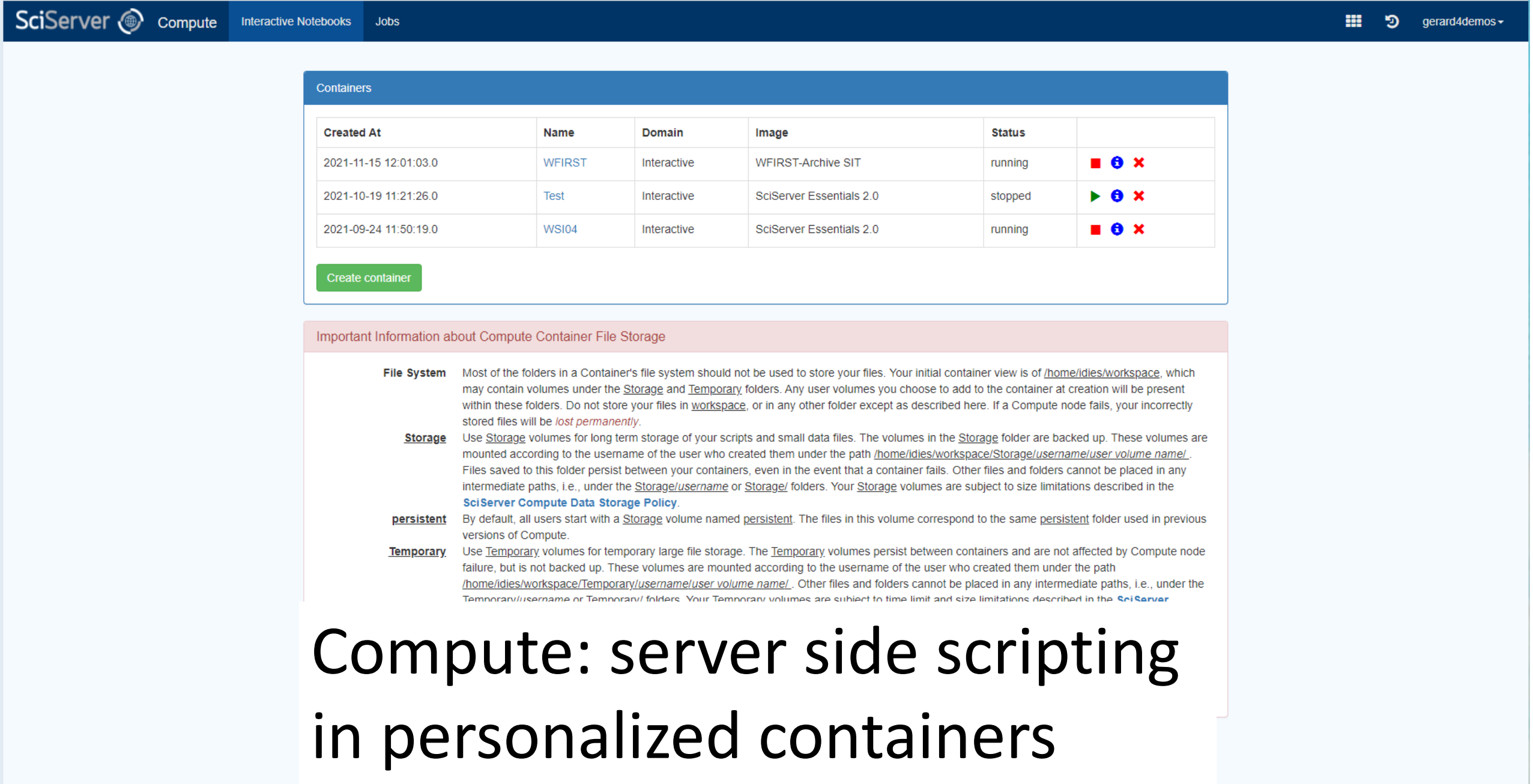
- **WFIRST – Archive SIT** data volume
  - MRObs data sets (pre..post observation images)
  - Illustris light cones
  - Cosmos-3dHST
  - Data used in STIPS
  - Room for more
- **WFIRST Archive SIT user group**
  - Access to WFIRST data volume and compute image
  - Access to Millennium, Indra, Illustris, Eagle databases and raw data
- **WFIRST Archive SIT** compute docker image
  - Millennium Run Observatory
  - Illustris light cone creation (and soon more)
  - Since hackathon 2019: STIPS, GalSim



Groups: collaboration and sharing

For WFIRST

- **WFIRST – Archive SIT** data volume
  - MRObs data sets (pre..post observation images)
  - Illustris light cones
  - Cosmos-3dHST
  - Data used in STIPS
  - Room for more
- **WFIRST Archive SIT** user group
  - Access to WFIRST data volume and compute image
  - Access to Millennium, Indra, Illustris, Eagle databases and raw data
- **WFIRST Archive SIT compute docker image**
  - Millennium Run Observatory
  - Illustris light cone creation (and soon more)
  - Since hackathon 2019: STIPS, GalSim



Compute: server side scripting in personalized containers

Powered by: idies JOHNS HOPKINS UNIVERSITY

Compute in Jupyter notebooks: combining SDSS CAS (source catalogue) with DAS (images)

jupyter 2. Stripe82-coadd-Copy1 Last Checkpoint: 3 minutes ago (unsaved changes)

File Edit View Insert Cell Kernel Help

Code CellToolbar

```
import SciServer.LoginPortal as Login
token=Login.getToken()
import pandas
import tables
import numpy as np
import astropy
from astropy.io import fits
from astropy import wcs
import skimage.io
import urllib
import os
import matplotlib
import matplotlib.pyplot as plt
```

In [18]:

```
sql="""
SELECT a.objid as head, c.objid2 as match, b.matchcount,
       p.fieldid as head_field, d.fieldid as match_field,
       dbo.fGetUrlFitsCFrame(d.fieldid, 'g') as fits_g,
       dbo.fGetUrlFitsCFrame(d.fieldid, 'r') as fits_r,
       dbo.fGetUrlFitsCFrame(d.fieldid, 'z') as fits_z,
       p.ra, d.ra as match_ra, p.dec, d.dec as match_dec
       , p.petror90_r
  from (select top 1 * from galaxy where objId=8658194378960928809) a
   join matchhead b on a.objid=b.objid        -- join with matchhead
   join photoobj p on a.objid=p.objid         -- get matchhead photoobj
   join match c on c.objid1=b.objid           -- join with all the matches
   join photoobjall d on c.objid2=d.objid     -- get match photoobj
order by d.fieldid                              -- order by matchhead objid
"""
queryResponse = SciServer.CasJobs.executeQuery(sql, "Stripe82",token=token)
obss = pandas.read_csv(queryResponse,index_col=None)
obss[:10]
```

Out[18]:

	head	match	matchcount	head_field	match_field	fits_g
0	8658194378960928809	8658194430499553320	57	8658194378960928768	8658194430499553280	http://das.sdss.org/imaging/5622/
1	8658194378960928809	8658194477742948377	57	8658194378960928768	8658194477742948352	http://das.sdss.org/imaging/5633/
2	8658194378960928809	8658194516375371821	57	8658194378960928768	8658194516375371776	http://das.sdss.org/imaging/5642/
3	8658194378960928809	8658194585083510793	57	8658194378960928768	8658194585083510784	http://das.sdss.org/imaging/5658/
4	8658194378960928809	8658194804163018771	57	8658194378960928768	8658194804163018752	http://das.sdss.org/imaging/5709/
5	8658194378960928809	8658194954470752297	57	8658194378960928768	8658194954470752256	http://das.sdss.org/imaging/5744/
6	8658194378960928809	8658195018907910161	57	8658194378960928768	8658195018907910144	http://das.sdss.org/imaging/5759/40/corr/
7	8658194378960928809	8658195044651040803	57	8658194378960928768	8658195044651040768	http://das.sdss.org/imaging/5765/40/corr/
8	8658194378960928809	8658195066151239724	57	8658194378960928768	8658195066151239680	http://das.sdss.org/imaging/5770/40/corr/

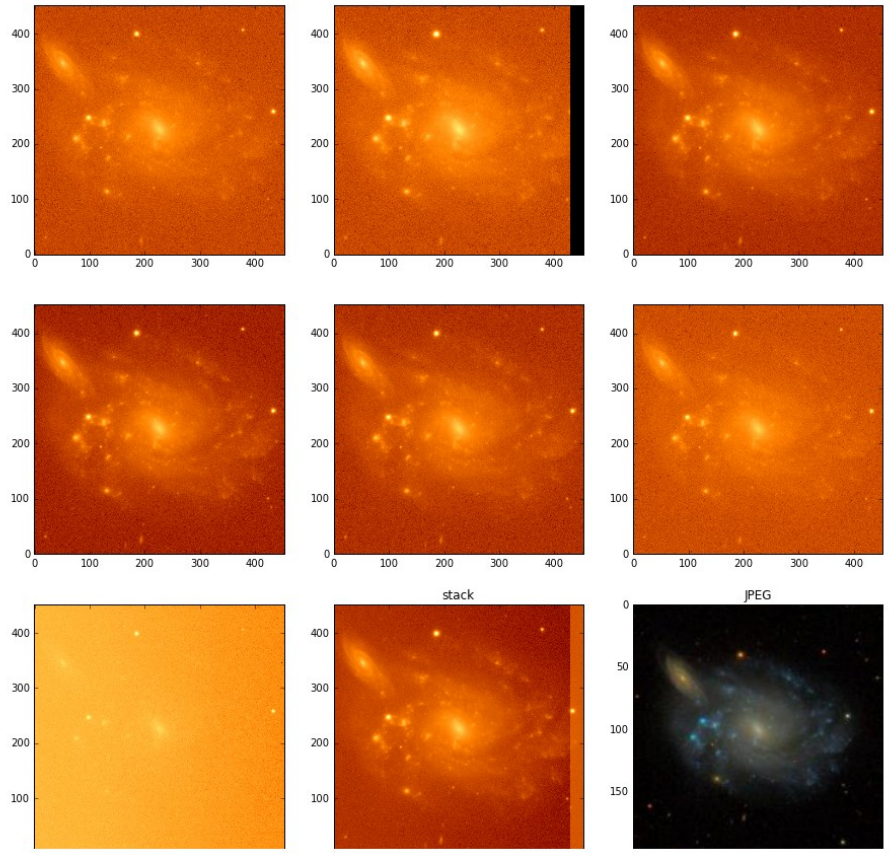
Python 3

jupyter 2. Stripe82-coadd-Copy1 Last Checkpoint: a minute ago (autosaved)

File Edit View Insert Cell Kernel Help Python 3

Code CellToolbar

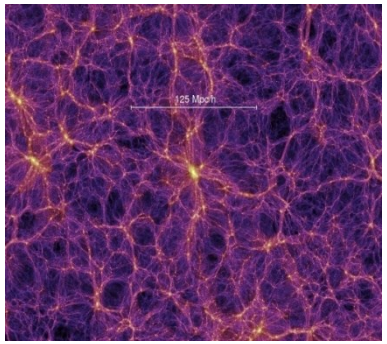
Out[17]: <matplotlib.text.Text at 0x7f2aa174a6d8>



- Design and build simple end-to-end simulations. Incorporate these simulations into a scalable, queryable database.

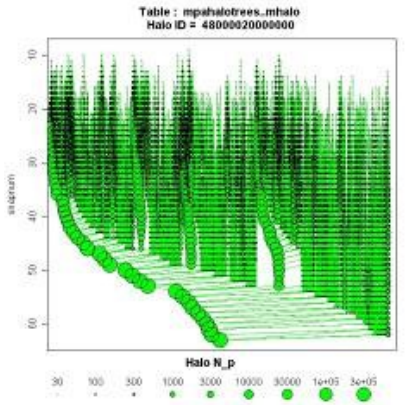
Example: Millennium Run Observatory in SciServer

Particles

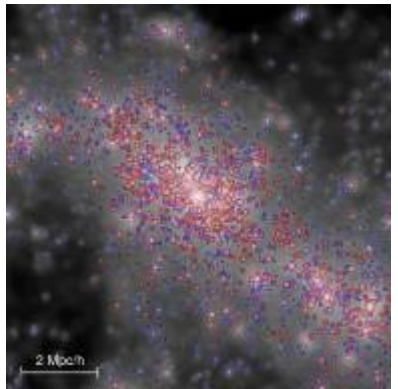


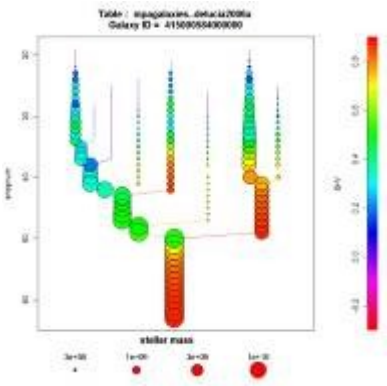
Clusters/halos

Halo merger trees

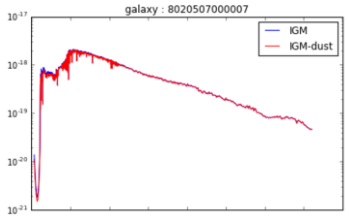


Synthetic galaxies (SAM)



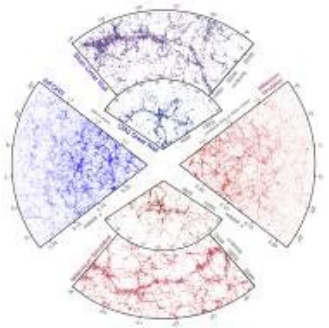


Synthetic spectra



Synthetic images

Mock catalogues



Compute in Jupyter notebooks: analyzing DM halo density profiles in Millennium Run

jupyter CosmoUC_5_PlotHalo Last Checkpoint: 11 minutes ago (autosaved)

File Edit View Insert Cell Kernel Help Notebook saved Python 3

Code CellToolbar

Dark-matter Halos in a Cosmological Simulation

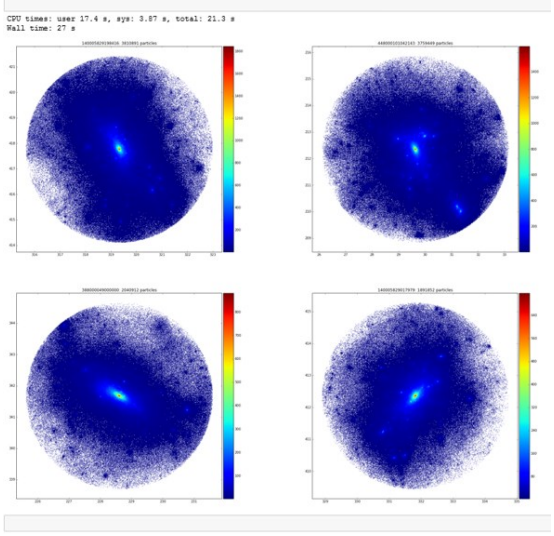
In [3]:
```
import SciServer.LoginPortal as Login
token = Login.getToken()
import SciServer.CasJobs
import pandas
import tables
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

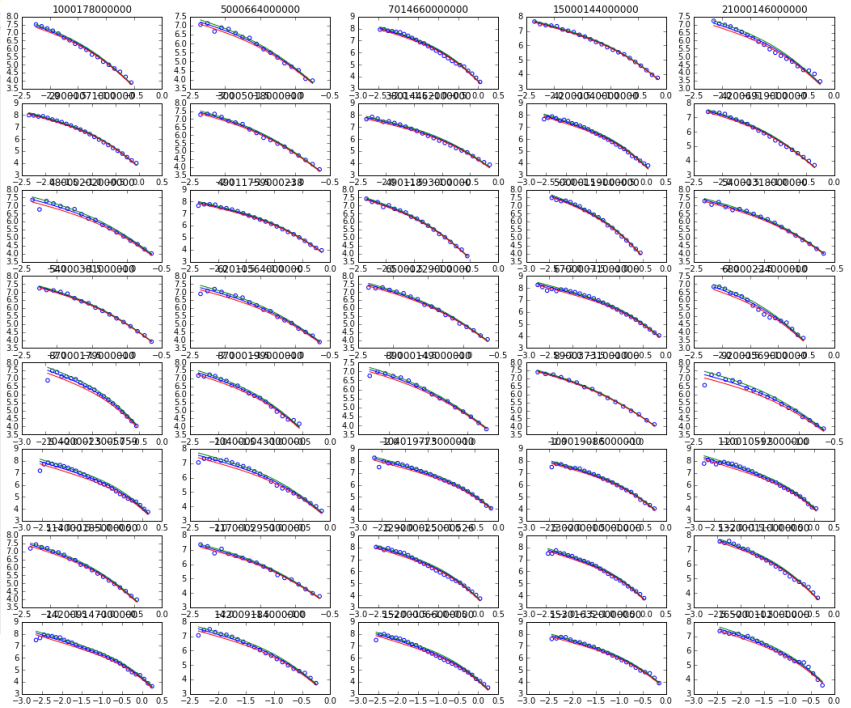
In [10]:
```
%%time
queryString = """
select top 100000 p.x-hh.x as x,p.y-hh.y as y,p.z-hh.z as z
from mpahalotrees.mr hh
   cross apply dbo.MillenniumParticles(hh.snapnum,
     dbo.Sphere::New(hh.x,hh.y,hh.z,3*hh.halfmassradius).ToString()) p
  where hh.haloid=84000007000000 order by newid()
    """
responseStream = SciServer.CasJobs.executeQuery(queryString, token=token,context="SimulationDB")
df = pandas.read_csv(responseStream, index_col=None)
```

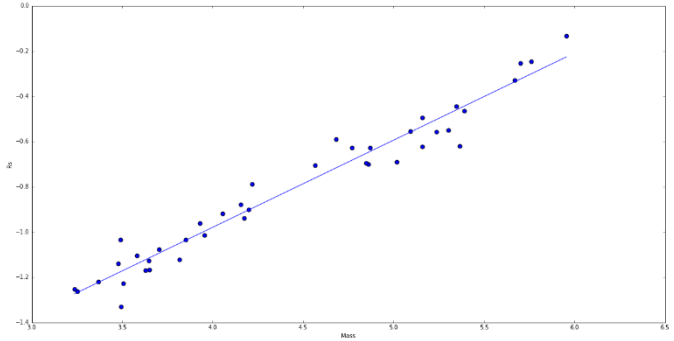
```
CPU times: user 351 ms, sys: 184 ms, total: 535 ms
Wall time: 5.27 s
```

In [13]:
```
fig = plt.figure(figsize=(15, 15))
ax = fig.add_subplot(111, projection='3d')
ax.scatter(df.x,df.y, df.z,s=0.001)
```

Out[13]: <mpl_toolkits.mplot3d.art3d.Path3DCollection at 0x7fe86428b9e8>







Light cones: aiming for most massive cluster at z

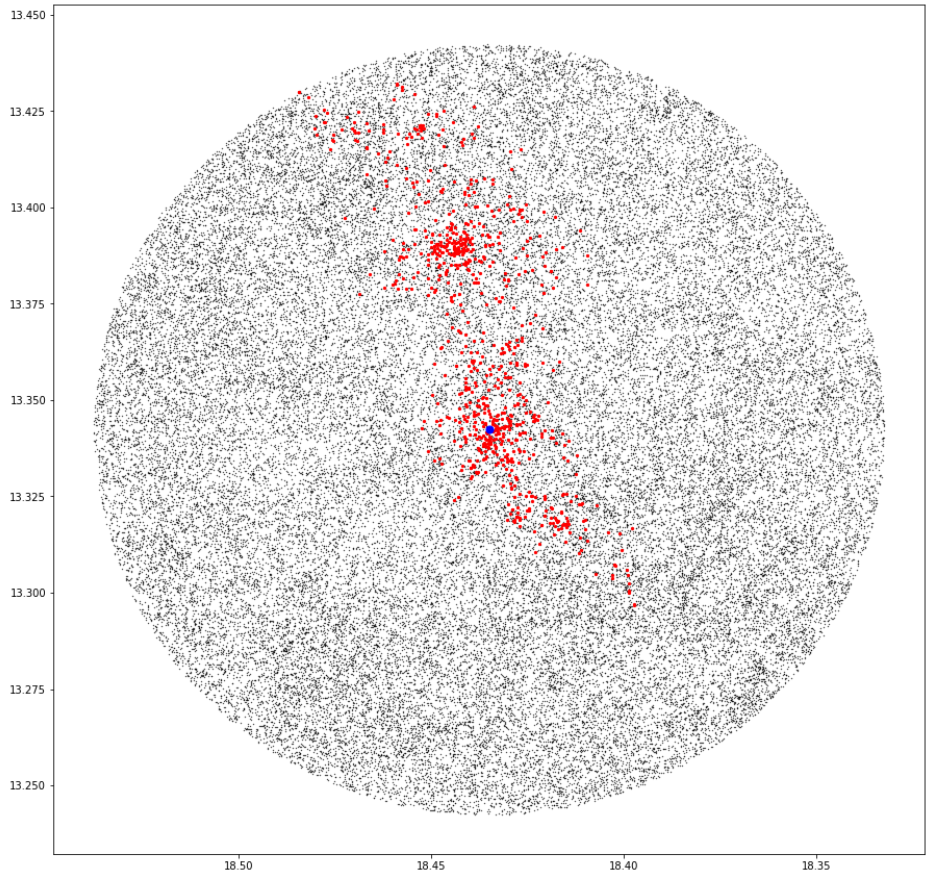
```
select top 1 h.redshift, h.haloid
,      h.x,h.y,h.z,h.m_crit200
,      g.fofCentralId
from mrscplanck1 h
  join Henriques2015a.MRscPlanck1 g
    on g.haloid=h.haloid and g.type=0
 where h.snapnum=32
   and h.firsthaloinfofgroupid=h.haloid
 order by h.m_crit200 desc
```

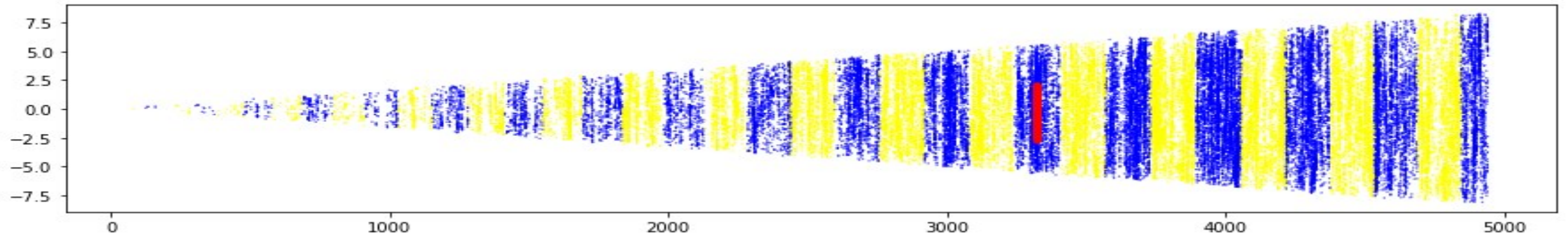
```
In [107]: lc=LGalaxies_lightcone(sim=sim,O=0,D=D,angle=angle,lo_z=0,
                                 hi_z=hi_z,table="Henriques2015a.MRscPlanck1",DEBUG=False)
```

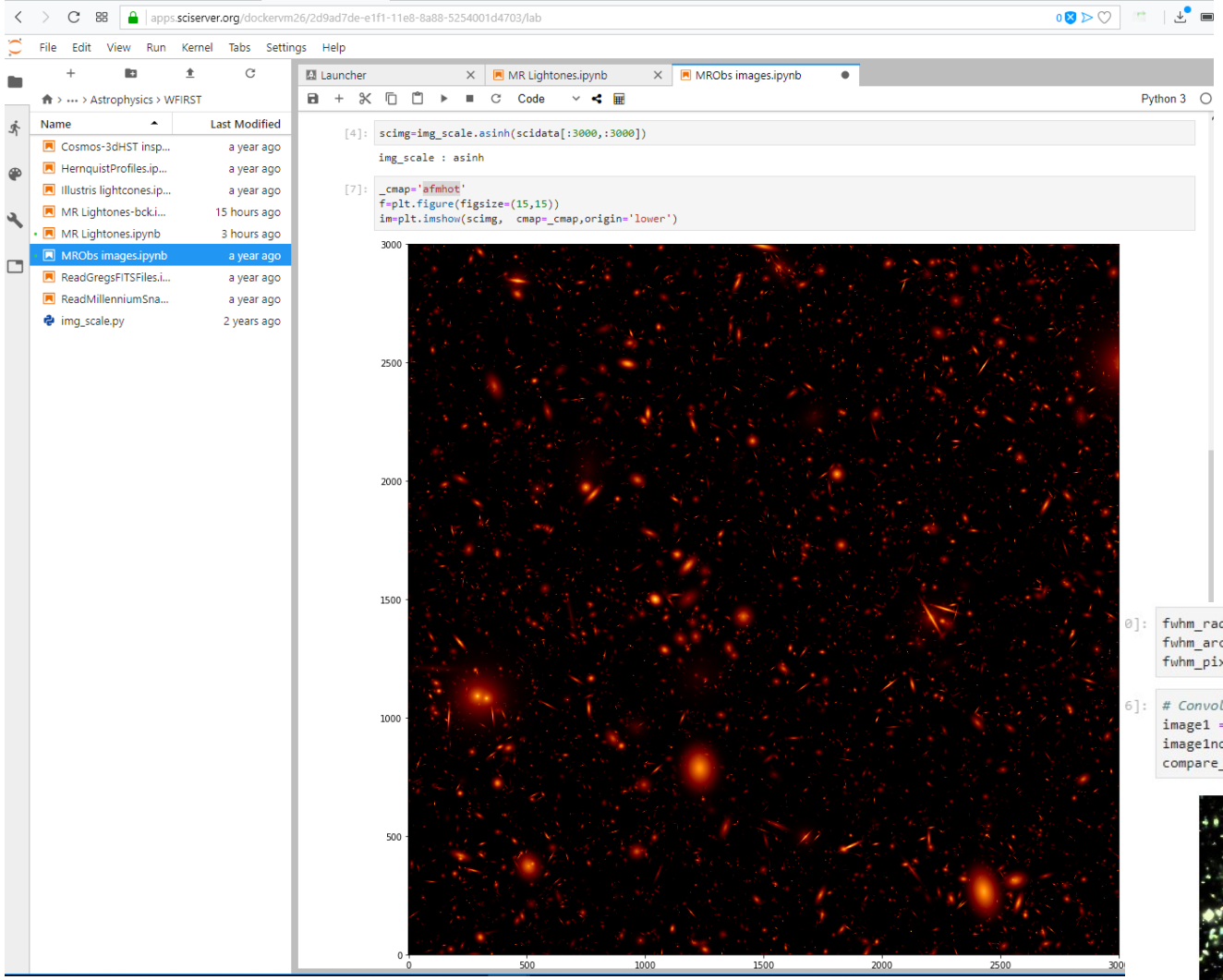
```
In [122]: lc
```

Out[122]:

	galaxyid	fofCentralId	snapnum	sdssg_dust	sdssg_bulgedust	stellarspinx	stellarspiny	stellarspinz	stellarr	
0	9005715000032	9005715000032	31	-16.50759	99.00000	0.091221	-0.078054	0.091848	0.00	
1	9003029000032	9003029000032	31	-15.33060	99.00000	0.035986	-0.110764	-0.098178	0.003478	0.000000 ...

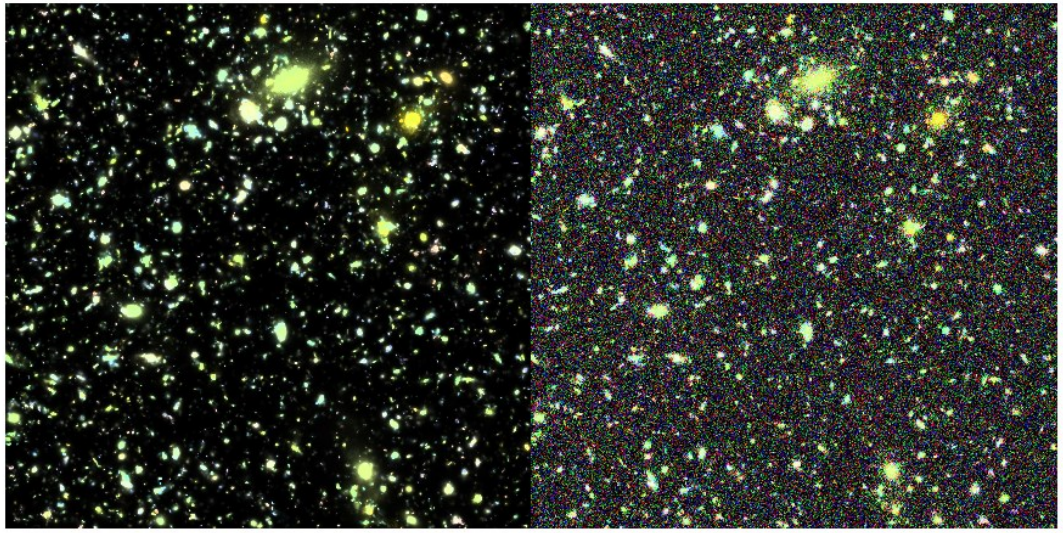







```
0]: fwhm_rad = 1.22*np.asarray([.98,1.44,1.84])*1.e-6/2.4
fwhm_arcsec = fwhm_rad*(180/np.pi)*3600
fwhm_pix = fwhm_arcsec/.11 # .11"/pixel
```

```
6]: # Convolve with a Gaussian PSF and add Gaussian noise
image1 = make_interactive(ba,ga,ra,2,.5)
image1noise = make_interactive(ba,ga,ra,2,.5,fwhm_pixels=fwhm_pix,sigma_tuple=[.2,.2,.2])
compare_images(image1,image1noise)
```



MRObs and Illustris Images on SciServer

See talks by Bridget, Arik, Swara for more use cases

External deployments (Kubernetes)

- NAOJ: PFS
- MPE: eROSITA, HETDEX
- NIST: Hurricane Maria investigation, Additive Manufacturing Benchmark
- JH ICTR: Precision Medicine Analytics Platform