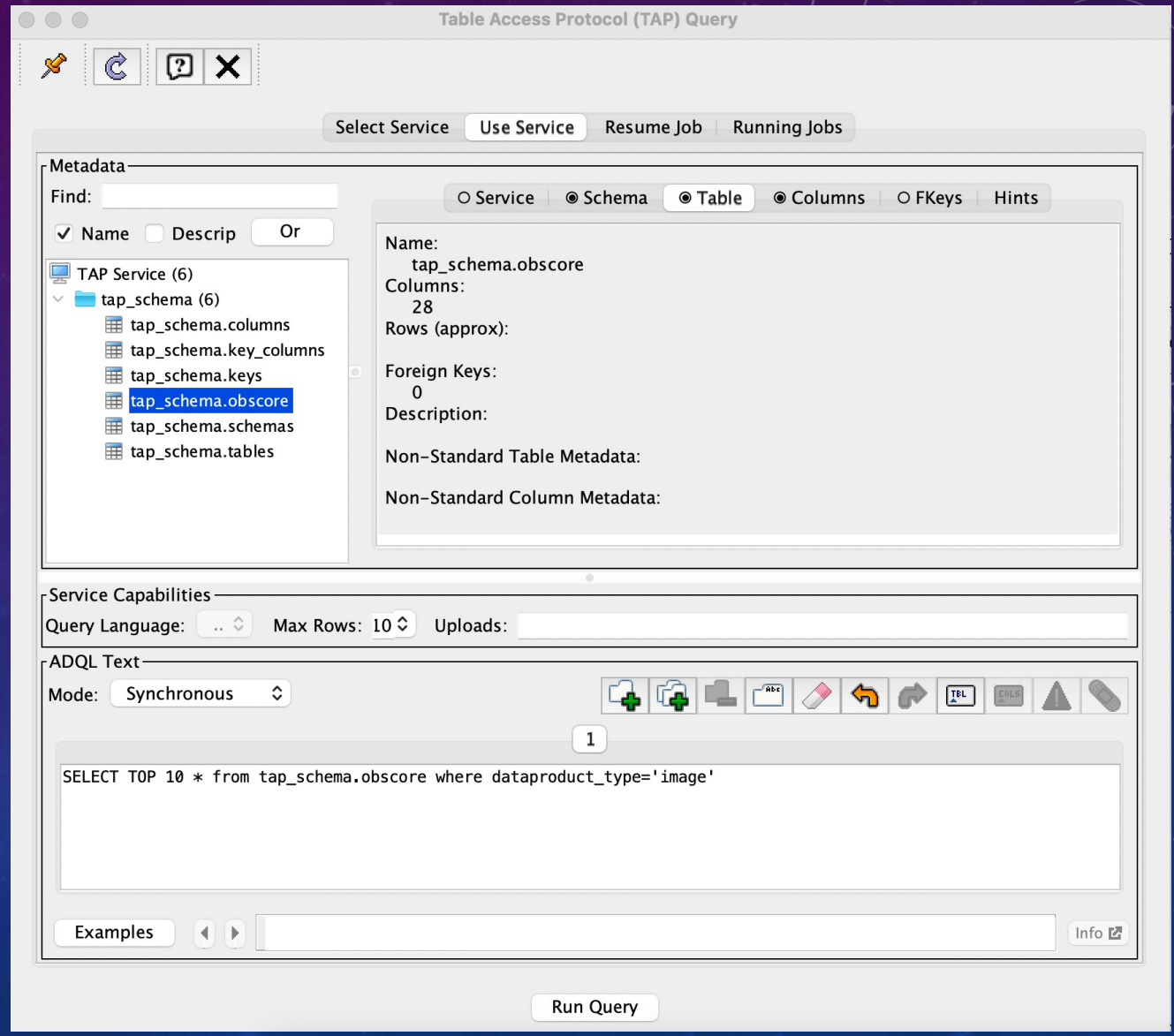
NRAO TAP SERVICE/ VLASS IN THE VIRTUAL OBSERVATORY

MARK LACY

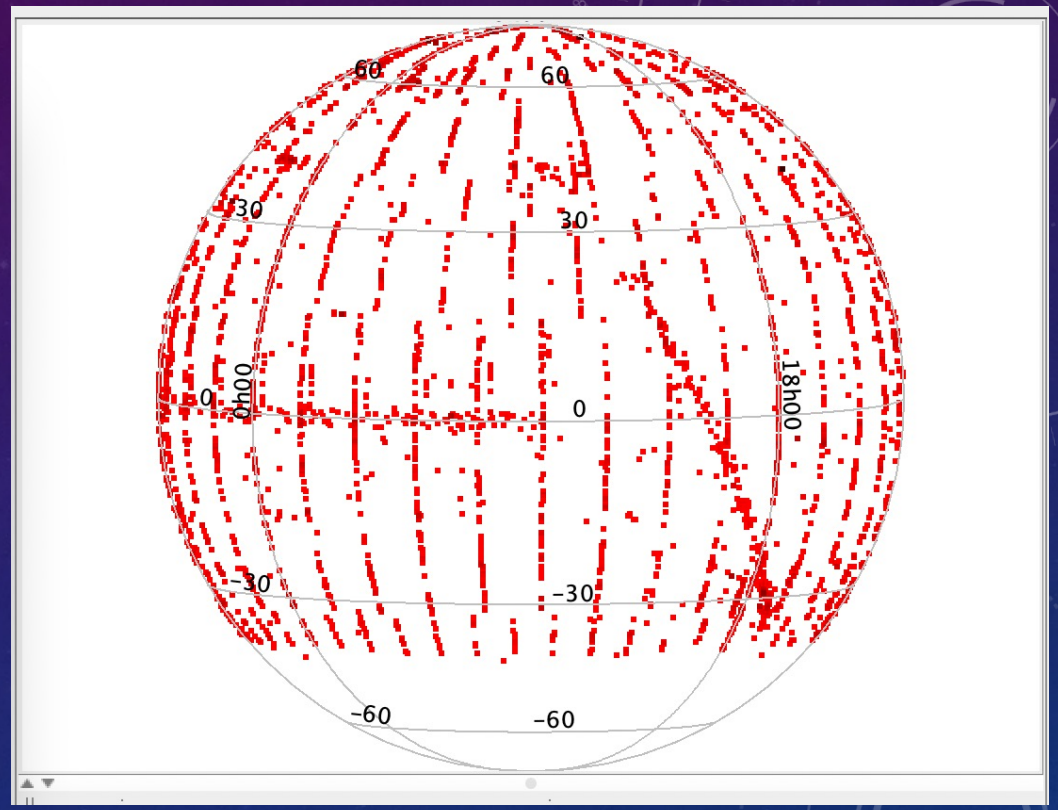
NRAO TAP SERVICE

- NRAO is about to publish a TAP service: https://data-query.nrao.edu/tap (Project scientist John Tobin, developers Stephan Witz and Daniel Lyons).
- This replaces the functionality of the scripted metadata access that used to be available via the legacy archive system and astroquery that was retired last year.
- An single ObsCore table contains both visibility and image metadata from the NRAO archive (VLA visibilities and pipeline images (including VLA Sky Survey), ALMA visibilities and user-driven imaging (ALMA pipeline images to be added later).



IMPLEMENTATION OF NRAO TAP

- Implementation ran into issues that often affect radio data
  - Resolution is a function of observing frequency, no one number for a multi-band visibility dataset (similarly for FOV).
  - Added freq_min and freq_max columns as em_min and em_max are not very useful for radio astronomy (heterodyne receivers).
- Full implementation of ObsCore is thus still a work in progress.
  - No datalink service yet, so access_url is not filled.
- Waiting to fix a couple of minor bugs before registering.
- Nevertheless, the service is a considerable improvement over that from the legacy archive, being accessible from tools such as TOPCAT as well as through scripts via pyVO.



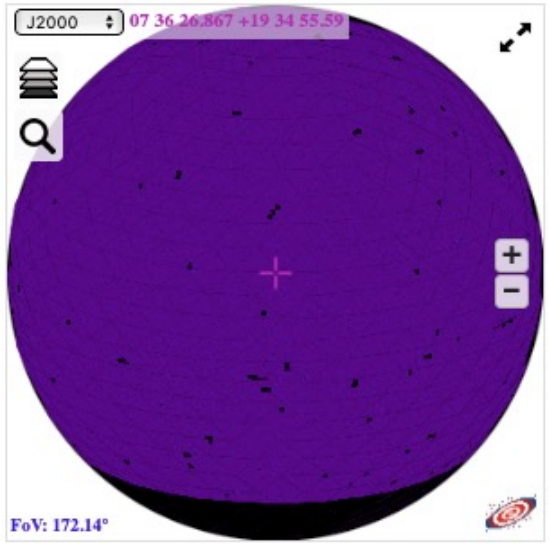
TOP 10000 VLA pointings

THE VLA SKY SURVEY (VLASS) IN THE VO

- The VLA Sky Survey is a three epoch, 3” resolution survey at 3GHz made with the VLA covering the whole sky above Dec. -40 degrees.
- Images cover 1 deg^2 , there are 34,000 images of each type (Quick Look, Single Epoch continuum and cubes over three epochs, plus combined epochs).
  - Really need a good way to get data and cutouts in bulk!
- The image data products are served from both NRAO and CADC, and the first epoch catalog via CDS.
- NRAO’s VO services are still being developed, so we use CADC, CDS and CASDA in the notebooks.

"VLASS-QL-Epoch1-20190905" progressive survey

This Web resource contains HiPS(*) components for **VLASS-QL-Epoch1-20190905** progressive survey.

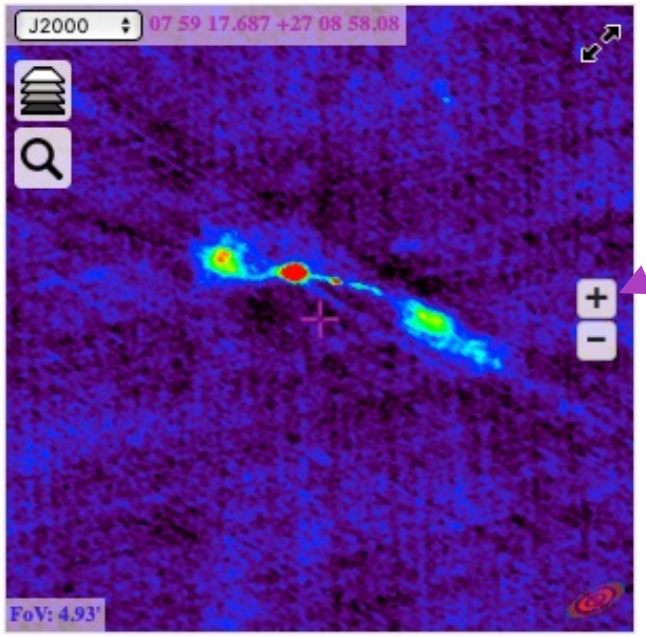


- **Label:** VLASS-QL-Epoch1-20190905
- **Type:** HiPS image
- **Best pixel angular resolution:** 805.2mas
- **Max tile order:** 9 (NSIDE=512)
- **Available encoding tiles:** png fits
- **Tile size:** 512x512
- **FITS tile BITPIX:** -32
- **Processing date:** 2019-09-17T11:54Z
- **HiPS builder:** Aladin/HipsGen v10.044
- **Coordinate frame:** equatorial
- **Sky area:** 81.851% of sky => 33766&#65533;^2
- **Associated coverage map:** MOC
- **Original data access template:** metadata.xml
- **Raw property file:** properties
- **Base URL:** http://archive-new.nrao.edu/vlass/HiPS/VLASS_Epoch1/Quicklook

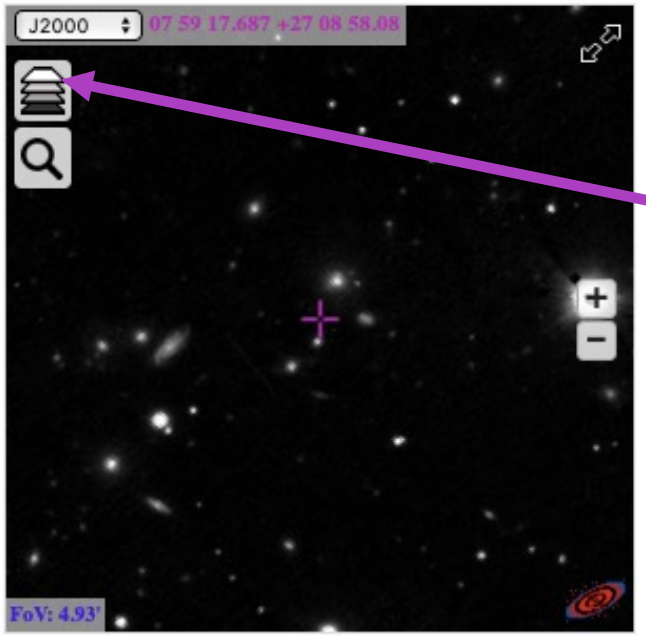
This survey can be displayed by Aladin Lite (see above), by Aladin Desktop client (just open the base URL) or any other HiPS aware clients.

(*) The HiPS technology allows a dedicated client to access an astronomical survey at any location and at any scale. HiPS is based on HEALPix sky tessellation and it is designed for astronomical scientifical usages (low distorsion, true pixel values....) HiPS technical documentation is available here

http://archive-new.nrao.edu/vlass/HiPS



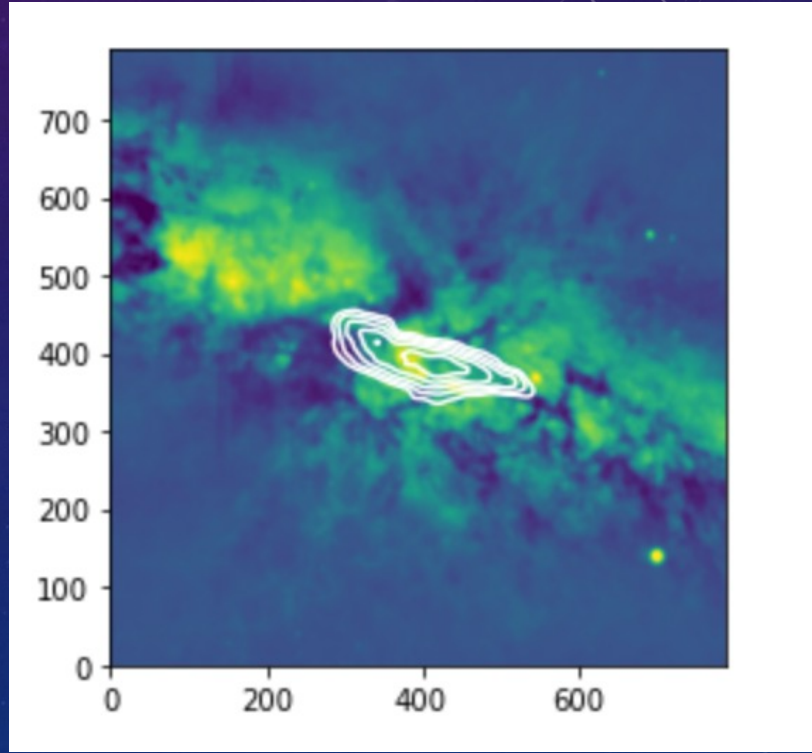
Zoom in on source



Change layer e.g. PanSTARRS g

SCRIPTED QUERIES

- The pyVO python package can be used to make image and catalog searches, and also TAP queries (see https://gitlab.nrao.edu/mlacy/vlass_vo)
- Example image notebook: VO_SIA_SODA_demo.ipynb.
  - Runs SIA2 query at CADC (https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/sia/v2query) for M82
  - Uses Datalink to get a whole VLASS image, and SODA to get a cutout
  - Also gets a CFHT MegaCAM optical cutout via SIA2 and SODA to match the VLASS one.
  - Uses Scipy to interpolate the VLASS and CFHT images to the same grid and overplots VLASS contours on the optical image.
  - All this is done via direct streaming – no awkward saving images to disk and reading them back in!



CODE WALK-THROUGH

In [1]:
```
import numpy as np
import pyvo as vo
from astropy.coordinates import SkyCoord
import astropy.units as u
import urllib.request
import urllib.parse
import astropy.io.fits as fits
import matplotlib.pyplot as plt
import scipy.interpolate as si
from matplotlib.colors import SymLogNorm
```

Get the VLASS image of M82 using the Simple Image Access version 2 protocol (SIAv2)

In [2]:
```
pos=SkyCoord.from_name('M82')
```

To submit an SIAv2 query we need to express the position and radius as a tuple in decimal degrees

In [3]:
```
print(pos)
pst=(pos.ra.value,pos.dec.value,0.02)
print(pst)
```

```
<SkyCoord (ICRS): (ra, dec) in deg
    (148.9684583, 69.6797028)>
(148.9684583, 69.6797028, 0.02)
```

In [4]:
```
sia2_return=vo.dal.imagesearch2("https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/sia/v2query",pos=pst,collection="VLASS")
```

In [5]:
```
print(sia2_return.fieldnames)
print(sia2_return['access_url'])
```

```
('calib_level', 's_ra', 's_dec', 's_fov', 's_region', 'obs_publisher_did', 'obs_collection', 'facility_name', 'instr
ument_name', 'obs_id', 'dataproduct_type', 'obs_release_date', 'target_name', 's_resolution', 's_xel1', 's_xel2', 't
_min', 't_max', 't_exptime', 't_resolution', 't_xel', 'em_min', 'em_max', 'em_res_power', 'em_xel', 'pol_xel', 'acce
ss_url', 'access_estsize', 'em_ucd', 'pol_states', 'o_ucd', 'access_format', 'core_id', 'lastModified')
['https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/datalink?runid=a0nwfpjp3gw4i8m1&ID=ivo%3A%2F%2Fcadc.nrc.ca%2F
VLASS%3FVLASS2.1.T28t05.J095548%2B693000%2FVLASS2.1.T28t05.J095548%2B693000.quicklook'
 'https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/datalink?runid=a0nwfpjp3gw4i8m1&ID=ivo%3A%2F%2Fcadc.nrc.ca%2F
VLASS%3FVLASS1.1.T28t05.J095548%2B693000%2FVLASS1.1.T28t05.J095548%2B693000.quicklook']
```

Pick the first row (the VLASS2.1 image). Run a datalink query to return links to the relevant files in the CADC archive

In [6]:
```
row=sia2_return[0]
datalink=row.getdatalink()
print(datalink.fieldnames)
print(datalink['description'])
print(datalink['semantics'])
```

```
('ID', 'access_url', 'service_def', 'error_message', 'semantics', 'description', 'content_type', 'content_length', '
readable')
['download ad:VLASS/VLASS2.1.ql.T28t05.J095548+693000.10.2048.v1.I.iter1.image.pbcor.tt0.rms.subim.fits'
 'download ad:VLASS/VLASS2.1.ql.T28t05.J095548+693000.10.2048.v1.I.iter1.image.pbcor.tt0.subim_prev.jpg'
 'download ad:VLASS/VLASS2.1.ql.T28t05.J095548+693000.10.2048.v1.I.iter1.image.pbcor.tt0.subim_prev_256.jpg'
 'download ad:VLASS/VLASS2.1.ql.T28t05.J095548+693000.10.2048.v1.I.iter1.image.pbcor.tt0.subim.fits'
 'SODA-sync cutout of ad:VLASS/VLASS2.1.ql.T28t05.J095548+693000.10.2048.v1.I.iter1.image.pbcor.tt0.subim.fits'
 'SODA-async cutout of ad:VLASS/VLASS2.1.ql.T28t05.J095548+693000.10.2048.v1.I.iter1.image.pbcor.tt0.subim.fits'
 'single download containing all files (previews and thumbnails excluded)']
['#auxiliary' '#preview' '#thumbnail' '#this' '#cutout' '#cutout'
 '#package']
```

To identify the primary image data ('#this') we can use the semantics and just get the image data row:

In [7]:
```
datalink2=next(row.getdatalink().bysemantics('#this'))
print(datalink2['access_url'])
```

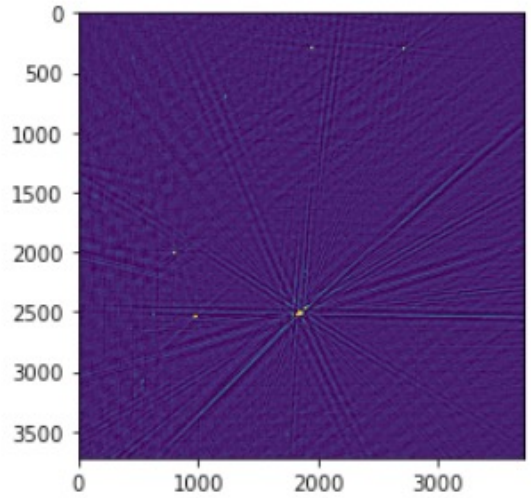
```
https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/data/pub/VLASS/VLASS2.1.ql.T28t05.J095548+693000.10.2048.v1.I.iter1.image.pbcor.tt0.subim.fits
```

Now we can just get the image and stream it direct into memory

In [8]:
```
fitsfileurl=datalink2["access_url"]
with urllib.request.urlopen(fitsfileurl) as f:
    hdu=fits.open(f)
```

In [9]:
```
data=np.squeeze(hdu[0].data)
plt.imshow(data,vmin=-0.0001,vmax=0.001)
```

Out[9]:
```
<matplotlib.image.AxesImage at 0x7fcd29c5ef10>
```



Using the SODA service we can go one better and get a cutout around our region of interest. Integrating this into pyVO is still a work in progress, so we will need to construct our own SODA query using the link in the datalink description and the syntax that we can get from https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/#!/CAOM_SODA_service/get_sync, namely https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/sync?ID=ad%3AVLASS%2FVLASS2.1.ql.T28t05.J095548%2B693000.10.2048.v1.I.iter1.image.pbcor.tt0.subim.fits&CIRCLE=148.9684583%2069.6797028%200.1

In [10]:
```
print(datalink['description'][4])
```

```
SODA-sync cutout of ad:VLASS/VLASS2.1.ql.T28t05.J095548+693000.10.2048.v1.I.iter1.image.pbcor.tt0.subim.fits
```

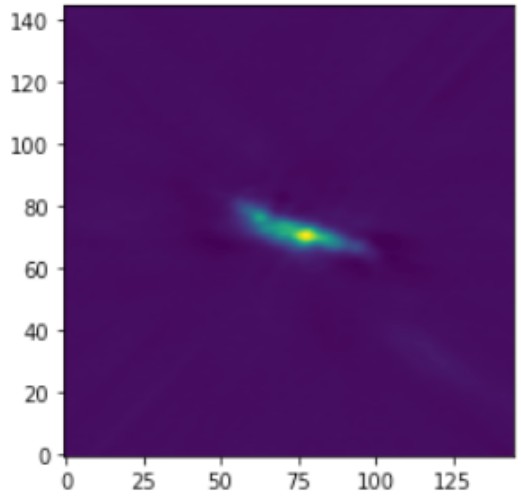
the description gives the location of the fits file as ad:VLASS/VLASS2.1.ql.T28t05.J095548+693000.10.2048.v1.I.iter1.image.pbcor.tt0.subim.fits we can url-ify it and add on the cutout position and size

In [14]:
```
pathurl='https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/sync?ID='
path=urllib.parse.quote('ad:VLASS/VLASS2.1.ql.T28t05.J095548+693000.10.2048.v1.I.iter1.image.pbcor.tt0.subim.fits',sa
print(path)
cutout='CIRCLE='+str(pst[0])+'%20'+str(pst[1])+'%20'+str(pst[2])
print(cutout)
#construct the query:
sodaurl=pathurl+path+'&'+cutout
print(sodaurl)
```

```
ad%3AVLASS%2FVLASS2.1.ql.T28t05.J095548%2B693000.10.2048.v1.I.iter1.image.pbcor.tt0.subim.fits
CIRCLE=148.9684583%2069.6797028%200.02
https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/sync?ID=ad%3AVLASS%2FVLASS2.1.ql.T28t05.J095548%2B693000.10.2048.v1.I.iter1.image.pbcor.tt0.subim.fits&CIRCLE=148.9684583%2069.6797028%200.02
```

In [15]:
```
with urllib.request.urlopen(sodaurl) as g:
    hdu2=fits.open(g)
data2=np.squeeze(hdu2[0].data)
plt.imshow(data2,origin='lower')
```

Out[15]:
```
<matplotlib.image.AxesImage at 0x7fcd481ca640>
```



The power of the VO is that we can query other data collections in the same way as we did for VLASS. So let's see if there is a CFHT image of it in the CTFHTMEGAPIPE collection

In [16]:
```
sia2_return2=vo.dal.imagesearch2("https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/sia/v2query",pos=pst, collection="CFHTM
```

In [17]:
```
print(sia2_return2.fieldnames)
print(sia2_return2['access_url'])
```

```
('calib_level', 's_ra', 's_dec', 's_fov', 's_region', 'obs_publisher_did', 'obs_collection', 'facility_name', 'instr
ument_name', 'obs_id', 'dataproduct_type', 'obs_release_date', 'target_name', 's_resolution', 's_xel1', 's_xel2', 't
_min', 't_max', 't_exptime', 't_resolution', 't_xel', 'em_min', 'em_max', 'em_res_power', 'em_xel', 'pol_xel', 'acce
ss_url', 'access_estsize', 'em_ucd', 'pol_states', 'o_ucd', 'access_format', 'core_id', 'lastModified')
['https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/datalink?runid=kv7tal7ay1aw802c&ID=ivo%3A%2F%2Fcadc.nrc.ca%2F
CFHTMEGAPIPE%3FG008.150.130%2B69.495%2FG008.150.130%2B69.495.R'
 'https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/datalink?runid=kv7tal7ay1aw802c&ID=ivo%3A%2F%2Fcadc.nrc.ca%2F
CFHTMEGAPIPE%3FG008.147.666%2B69.505%2FG008.147.666%2B69.505.I'
 'https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/datalink?runid=kv7tal7ay1aw802c&ID=ivo%3A%2F%2Fcadc.nrc.ca%2F
CFHTMEGAPIPE%3FMegaPipe.104.319%2FMegaPipe.104.319.Z.MP9801'
 'https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/datalink?runid=kv7tal7ay1aw802c&ID=ivo%3A%2F%2Fcadc.nrc.ca%2F
CFHTMEGAPIPE%3FMegaPipe.104.319%2FMegaPipe.104.319.U.MP9301'
 'https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/datalink?runid=kv7tal7ay1aw802c&ID=ivo%3A%2F%2Fcadc.nrc.ca%2F
CFHTMEGAPIPE%3FMegaPipe.104.319%2FMegaPipe.104.319.G.MP9401'
 'https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/datalink?runid=kv7tal7ay1aw802c&ID=ivo%3A%2F%2Fcadc.nrc.ca%2F
CFHTMEGAPIPE%3FMegaPipe.104.319%2FMegaPipe.104.319.I.MP9701'
 'https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/datalink?runid=kv7tal7ay1aw802c&ID=ivo%3A%2F%2Fcadc.nrc.ca%2F
CFHTMEGAPIPE%3FG008.147.666%2B69.505%2FG008.147.666%2B69.505.R'
 'https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/datalink?runid=kv7tal7ay1aw802c&ID=ivo%3A%2F%2Fcadc.nrc.ca%2F
CFHTMEGAPIPE%3FG008.150.130%2B69.495%2FG008.150.130%2B69.495.I'
 'https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/datalink?runid=kv7tal7ay1aw802c&ID=ivo%3A%2F%2Fcadc.nrc.ca%2F
CFHTMEGAPIPE%3FG021.150.130%2B69.495%2FG021.150.130%2B69.495.G'
 'https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/datalink?runid=kv7tal7ay1aw802c&ID=ivo%3A%2F%2Fcadc.nrc.ca%2F
CFHTMEGAPIPE%3FG021.150.130%2B69.495%2FG021.150.130%2B69.495.I'
 'https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/datalink?runid=kv7tal7ay1aw802c&ID=ivo%3A%2F%2Fcadc.nrc.ca%2F
CFHTMEGAPIPE%3FG021.150.130%2B69.495%2FG021.150.130%2B69.495.R'
 'https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/datalink?runid=kv7tal7ay1aw802c&ID=ivo%3A%2F%2Fcadc.nrc.ca%2F
CFHTMEGAPIPE%3FG021.150.130%2B69.495%2FG021.150.130%2B69.495.U'
 'https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/datalink?runid=kv7tal7ay1aw802c&ID=ivo%3A%2F%2Fcadc.nrc.ca%2F
CFHTMEGAPIPE%3FMegaPipe.104.319%2FMegaPipe.104.319.R.MP9601']
```

In [18]:
```
#let's get a u-band image cutout
row=sia2_return2[11]
datalink=row.getdatalink()
print(datalink.fieldnames)
print(datalink['access_url'])
print(datalink['description'])
print(datalink['semantics'])
```

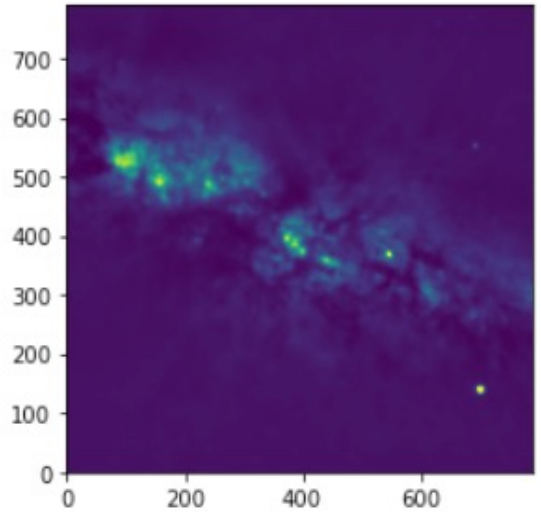
```
('ID', 'access_url', 'service_def', 'error_message', 'semantics', 'description', 'content_type', 'content_length', '
readable')
['https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/raven/files/cadc:CFHTSG/G021.150.130+69.495.U.weight.fits.fz'
 'https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/raven/files/cadc:CFHTSG/G021.150.130+69.495.U.cat'
 'https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/raven/files/cadc:CFHTSG/G021.150.130+69.495.U.fits'
 '' ''
 'https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/pkg?ID=ivo%3A%2F%2Fcadc.nrc.ca%2FCFHTMEGAPIPE%3FG021.150.130%
2B69.495%2FG021.150.130%2B69.495.U']
['download cadc:CFHTSG/G021.150.130+69.495.U.weight.fits.fz'
 'download cadc:CFHTSG/G021.150.130+69.495.U.cat'
 'download cadc:CFHTSG/G021.150.130+69.495.U.fits'
 'SODA-sync cutout of cadc:CFHTSG/G021.150.130+69.495.U.fits'
 'SODA-async cutout of cadc:CFHTSG/G021.150.130+69.495.U.fits'
 'single download containing all files (previews and thumbnails excluded)']
['#auxiliary' '#auxiliary' '#this' '#cutout' '#cutout' '#package']
```

In [19]:
```
#put together a SODA query:
path=urllib.parse.quote('cadc:CFHTSG/G021.150.130+69.495.U.fits',safe='')
print(path)
sodaurl=pathurl+path+'&'+cutout
print(sodaurl)
```

```
cadc%3ACFHTSG%2FG021.150.130%2B69.495.U.fits
https://ws.cadc-ccda.hia-iha.nrc-cnrc.gc.ca/caom2ops/sync?ID=cadc%3ACFHTSG%2FG021.150.130%2B69.495.U.fits&CIRCLE=148
.9684583%2069.6797028%200.02
```

In [20]:
```
with urllib.request.urlopen(sodaurl) as u:
    hdu3=fits.open(u)
data3=np.squeeze(hdu3[0].data)
plt.imshow(data3,origin='lower',vmin=-50,vmax=1000)
print(np.min(data3),np.max(data3),np.median(data3))
```

```
-70.24255 3884.4214 4.1544876
```

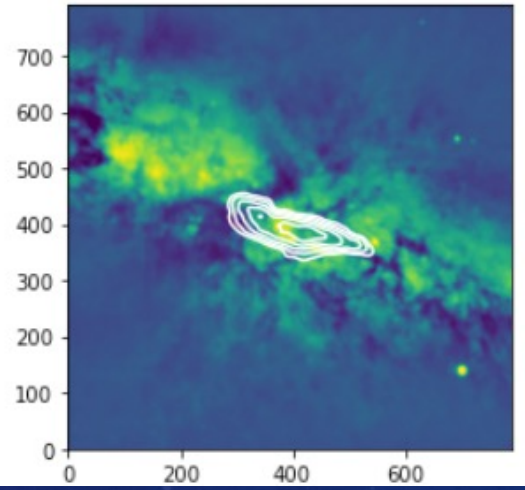


In [21]:
```
print(np.shape(data2[0]))
xv=np.arange(145)
yv=np.arange(145)
f=si.interp2d(xv,yv,data2,kind='linear')
print(np.shape(data3))
xn=np.arange(790)*.184
yn=np.arange(790)*.184
data2new=f(xn,yn)
print(np.shape(data2new))
plt.imshow(data3,origin='lower',norm=SymLogNorm(50,vmin=-50,vmax=1000))
plt.contour(data2new,levels=[0.005,0.01,0.02,0.04,0.08],colors='white',origin='lower')
#plt.imshow(data2new)
```

```
(145,)
(790, 790)
(790, 790)
```

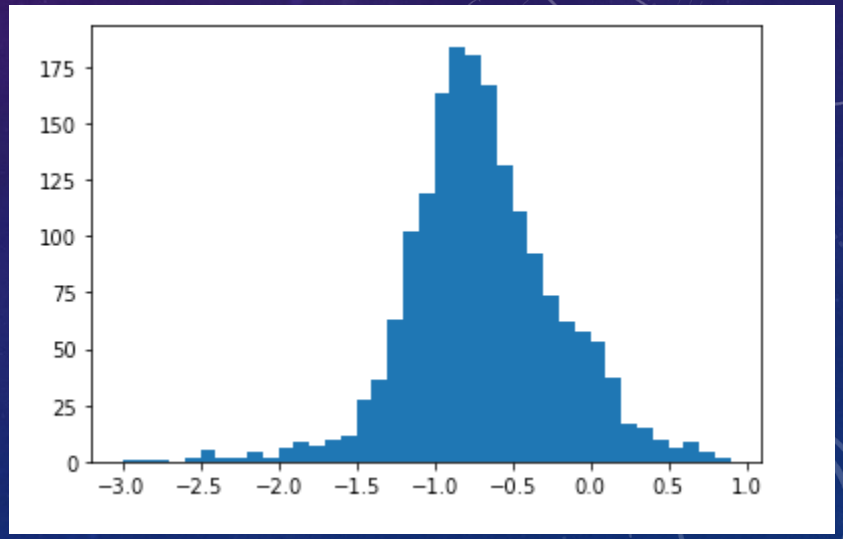
```
<ipython-input-21-e46fe848c9c1>:10: MatplotlibDeprecationWarning: default base will change from np.e to 10 in 3.4.
To suppress this warning specify the base keyword argument.
  plt.imshow(data3,origin='lower',norm=SymLogNorm(50,vmin=-50,vmax=1000))
```

Out[21]:
```
<matplotlib.contour.QuadContourSet at 0x7fcd4826fa90>
```



SCRIPTED QUERIES – CONE SEARCH

- Example image notebook: Cone_search_demo.ipynb.
- Extract a 4 degree radius cutout from the VLASS catalog of Gordon et al. held at CDS in France using SCS (http://vizier.cds.unistra.fr/viz-bin/conesearch/J/ApJS/255/30/comp?)
- Use TAP to extract the same region from the RACS survey from the CASDA archive at CSIRO in Australia (https://casda.csiro.au/casda_vo_tools/tap).
- Use astropy coordinates to match the VLASS and RACS sources and construct a spectral index distribution.



SUMMARY

- NRAO has a new TAP service. Looking forward to later implementations of more services (Datalink, SIA, SODA etc).
- For the VLA Sky Survey, NRAO is producing science ready data products.
  - Images and catalogs are served from VO-complaint data centers (e.g. CADC, CDS), so can write example notebooks for researchers to use for VLASS downloads, optical/radio overlays etc.
  - Will be used by student programs (NRAO's RADIAL project, for example).