Advanced Astronomical Observations 2021
Session 09: Using imexam Package

Kinoshita Daisuke

14 April 2021
publicly accessible version

About this file...

- Important information about this file
  - The author of this file is Kinoshita Daisuke.
  - The original version of this file was used for the course "Advanced Astronomical Observations" (course ID: AS6005) offered at Institute of Astronomy, National Central University from February 2021 to June 2021.
  - The file is provided in the hope that it will be useful, but there is no guarantee for the correctness. Use this file at your own risk.
  - If you are willing to use this file for your study, please feel free to use. I'll be very happy to receive feedback from you.
  - If you are willing to use this file for your teaching, please contact to Kinoshita Daisuke. When you use this file partly or entirely, please mention clearly that the author of the original version is Kinoshita Daisuke. Please help me to improve the contents of this file by sending your feedback.
  - Contact address: `https://www.instagram.com/daisuke23888/`

For this session, we try `imexam` package for Python. The `imexam` package is an Astropy affiliated package for quick image examination for astronomy. Its functionality and key bindings are similar to that of `imexamine` task of IRAF.

1 Installation of imexam package

1.1 About imexam

The `imexam` package is a Python package for simple examination and analysis for astronomical images. Visit the official web pages of `imexam` package, read documents, and learn about `imexam`.

- imexam
  - `https://imexam.readthedocs.io/`
  - `https://pypi.org/project/imexam/`
  - `https://github.com/spacetelescope/imexam`

1.2 Checking whether you have imexam on your computer

Try following to check whether you have imexam on your computer. If the package is successfully loaded, then you have imexam package properly installed on your computer.

```
% python3.9
Python 3.9.2 (default, Feb 21 2021, 12:39:42)
[GCC 7.5.0] on netbsd9
Type "help", "copyright", "credits" or "license" for more information.
>>> import imexam
```

```
/usr/pkg/lib/python3.9/site-packages/ginga/cmap.py:13317: MatplotlibDeprecationW
arning: The global colormaps dictionary is no longer considered public API.
  for name in _cm.cmap_d:
>>>
```

If you see an error message like below, then you do not have imexam on your computer.

```
% python3.9
Python 3.9.2 (default, Mar 27 2021, 17:26:25)
[GCC 7.5.0] on netbsd9
Type "help", "copyright", "credits" or "license" for more information.
>>> import imexam
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ModuleNotFoundError: No module named 'imexam'
>>>
```

If you do not have imexam installed on your computer, read following document and install it on your computer. Note that you also need to install (1) Ginga image viewer, and (2) photutils package.

- imexam
  - https://imexam.readthedocs.io/en/0.9.1/imexam/description.html#how-to-install
  - https://github.com/spacetelescope/imexam/blob/master/README.rst
- ginga
  - https://ginga.readthedocs.io/en/stable/install.html
- photutils
  - https://photutils.readthedocs.io/en/stable/install.html

2 Downloading an image

Make a Python script to download a SDSS image.

Python Code 1: ao2021_s09_01.py

```python
#!/usr/pkg/bin/python3.9

# importing argparse module
import argparse

# importing sys module
import sys

# importing astroquery module
import astroquery.simbad
import astroquery.ned
import astroquery.skyview

# importing astropy module
import astropy.coordinates
import astropy.units

# importing datetime module
import datetime
```

```
# importing ssl module
import ssl

# allow insecure downloading
ssl._create_default_https_context = ssl._create_unverified_context

# date/time
now = datetime.datetime.now ().isoformat ()

# units
u_ha  = astropy.units.hourangle
u_deg = astropy.units.deg

# constructing parser object
desc = "downloading SDSS image"
parser = argparse.ArgumentParser (description=desc)

# adding arguments
list_resolver = ['simbad', 'ned']
list_survey   = ['SDSSu', 'SDSSg', 'SDSSr', 'SDSSi', 'SDSSz']
parser.add_argument ('-r', '--resolver', choices=list_resolver, \
                     default='simbad', help='choice of name resolver')
parser.add_argument ('-s', '--survey', choices=list_survey, \
                     default='SDSSr', help='choice of survey')
parser.add_argument ('-t', '--target', default='', help='target name')
parser.add_argument ('-f', '--fov', type=int, default=1024, \
                     help='field-of-view in pixel')
parser.add_argument ('-o', '--output', default='', help='output file name')

# command-line argument analysis
args = parser.parse_args ()

# input parameters
name_resolver = args.resolver
survey        = args.survey
target_name   = args.target
fov_pix       = args.fov
file_output   = args.output

# checking target name
if (target_name == ''):
    # printing error message
    print ("No target name is given!")
    # exit
    sys.exit ()

# checking output file name
if (file_output == ''):
    # printing error message
    print ("No output file name is given!")
    # exit
    sys.exit ()
elif not (file_output[-5:] == '.fits'):
    # printing error message
    print ("Output file must be FITS file!")
    # exit
    sys.exit ()
```

```
# using name resolver
if (name_resolver == 'simbad'):
    query_result = astroquery.simbad.Simbad.query_object (target_name)
elif (name_resolver == 'ned'):
    query_result = astroquery.ned.Ned.query_object (target_name)

# RA and Dec
RA  = query_result['RA']
Dec = query_result['DEC']

# coordinate
if (name_resolver == 'simbad'):
    coord = astropy.coordinates.SkyCoord (RA[0], Dec[0], unit=(u_ha, u_deg))
elif (name_resolver == 'ned'):
    coord = astropy.coordinates.SkyCoord (RA[0], Dec[0], unit=(u_deg, u_deg))

coord_str = coord.to_string (style='hmsdms')
(coord_ra_str, coord_dec_str) = coord_str.split ()
coord_ra_deg  = coord.ra.deg
coord_dec_deg = coord.dec.deg

# printing coordinate
print ("Target Name: %s" % target_name)
print ("  RA:  %s = %f deg" % (coord_ra_str, coord_ra_deg) )
print ("  Dec: %s = %f deg" % (coord_dec_str, coord_dec_deg) )

# searching image
list_image = astroquery.skyview.SkyView.get_image_list (position=coord, \
                                                         survey=survey)

# printing image list
print ("Available images:")
print (" ", list_image)

# getting image
image = astroquery.skyview.SkyView.get_images (position=coord, survey=survey, \
                                               pixels=fov_pix)

# header and data
image0 = image[0]
header = image0[0].header
data   = image0[0].data

# adding comments in header
header['history'] = "image downloaded from %s" % survey
header['history'] = "image saved on %s" % now

# saving to a FITS file
astropy.io.fits.writeto (file_output, data, header=header)
```

Run the script, and download an image.

```
% chmod a+x ao2021_s09_01.py
% ./ao2021_s09_01.py -h
usage: ao2021_s09_01.py [-h] [-r {simbad,ned}]
                        [-s {SDSSu,SDSSg,SDSSr,SDSSi,SDSSz}] [-t TARGET]
                        [-f FOV] [-o OUTPUT]
```

```
downloading SDSS image

optional arguments:
  -h, --help            show this help message and exit
  -r {simbad,ned}, --resolver {simbad,ned}
                        choice of name resolver
  -s {SDSSu,SDSSg,SDSSr,SDSSi,SDSSz}, --survey {SDSSu,SDSSg,SDSSr,SDSSi,SDSSz}
                        choice of survey
  -t TARGET, --target TARGET
                        target name
  -f FOV, --fov FOV     field-of-view in pixel
  -o OUTPUT, --output OUTPUT
                        output file name

% ./ao2021_s09_01.py -t M35 -o m35_sdss_r.fits
Target Name: M35
  RA:  06h08m54s = 92.225000 deg
  Dec: +24d20m00s = 24.333333 deg
Available images:
  ['https://skyview.gsfc.nasa.gov/tempspace/fits/skv16425405959462.fits']
Downloading https://skyview.gsfc.nasa.gov/tempspace/fits/skv16425411188129.fits
|=========================================| 4.2M/4.2M (100.00%)        14s

% ls -l m35_sdss_r.fits
-rw-r--r--  1 daisuke  taiwan  4207680 Apr 14 01:12 m35_sdss_r.fits
```

3 Trying imexam

Make a Python script to try imexam package.

Python Code 2: ao2021_s09_02.py

```python
#!/usr/pkg/bin/python3.9

# importing argparse module
import argparse

# importing sys module
import sys

# importing time module
import time

# importing imexam module
import imexam

# constructing parser object
desc = "imexam"
parser = argparse.ArgumentParser (description=desc)

# adding arguments
parser.add_argument ('-l', '--log', default='imexam.log', \
                     help='output log file name')
parser.add_argument ('-s', '--sleep', default=60, \
                     help='waiting time for measurements')
parser.add_argument ('file', nargs=1, help='input FITS file name')

# command-line argument analysis
```

```
args = parser.parse_args ()

# input parameters
file_log   = args.log
file_fits  = args.file
time_sleep = args.sleep

# checking log file name
if (file_log == ''):
    # printing error message
    print ("No log file name is given!")
    print ("You need to specify a file name for log file.")
    sys.exit ()

# checking FITS file name
if (file_fits == ''):
    # printing error message
    print ("No FITS file name is given!")
    print ("You need to specify a file name of an input FITS file.")
    sys.exit ()

# connecting to a viewer (ginga)
viewer = imexam.connect (viewer='ginga')

# loading a FITS file
viewer.load_fits (file_fits)

# setting zoom level
viewer.zoom (1)

# setting a log file
viewer.setlog (filename=file_log)

# starting imexam
viewer.imexam ()

# waiting for measurements
time.sleep (time_sleep)

# closing viewer
viewer.close ()
```

Execute the script.

```
% chmod a+x ao2021_s09_02.py
% ./ao2021_s09_02.py -h
/usr/pkg/lib/python3.9/site-packages/ginga/cmap.py:13317: MatplotlibDeprecationW
arning: The global colormaps dictionary is no longer considered public API.
  for name in _cm.cmap_d:
usage: ao2021_s09_02.py [-h] [-l LOG] [-s SLEEP] file

imexam

positional arguments:
  file                  input FITS file name

optional arguments:
  -h, --help            show this help message and exit
```

```
  -l LOG, --log LOG     output log file name
  -s SLEEP, --sleep SLEEP
                        waiting time for measurements

% ./ao2021_s09_02.py m35_sdss_r.fits
```

Then, you see a web browser window like Fig. 1.

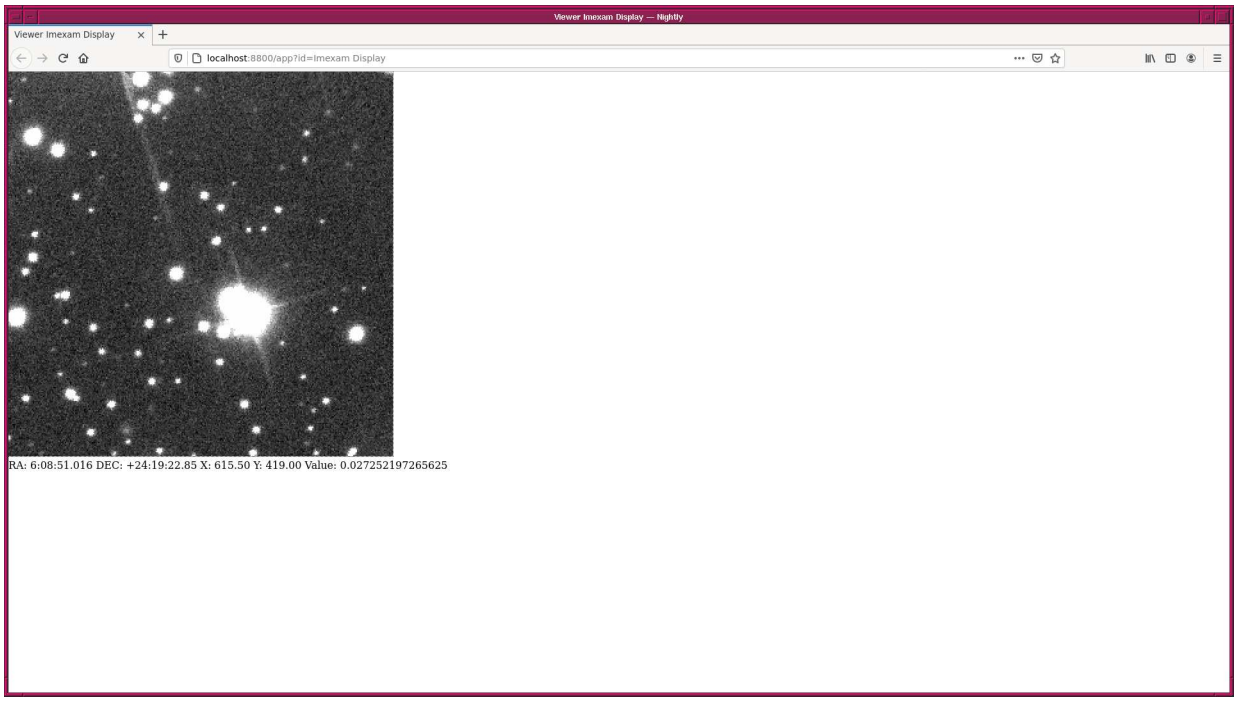

Figure 1: The Ginga HTML5 viewer for imexam.

Try a function of imexam. Type following command, move the mouse cursor to the image, type "i" key to enter "imexam mode", move the mouse cursor to a star, and then type "r" key. You see a plot like Fig. 2.

```
% ./ao2021_s09_02.py m35_sdss_r.fits
```

Show the content of the log file. The result of the measurements, such as centre of the object and FWHM, are recorded in the log file.

```
% cat imexam.log

gauss_center
xc=492.3895     yc=503.9387

radial_profile
Background per pixel: 0.04783093088765076

radial_profile
Max. pix. flux =    20.890
amp =    20.362
fwhm =     3.380
```

Next, try "d" key after entering "imexam mode" by typing "i" key and moving the mouse cursor to an object. Then, you see the result of centre of mass measurements.

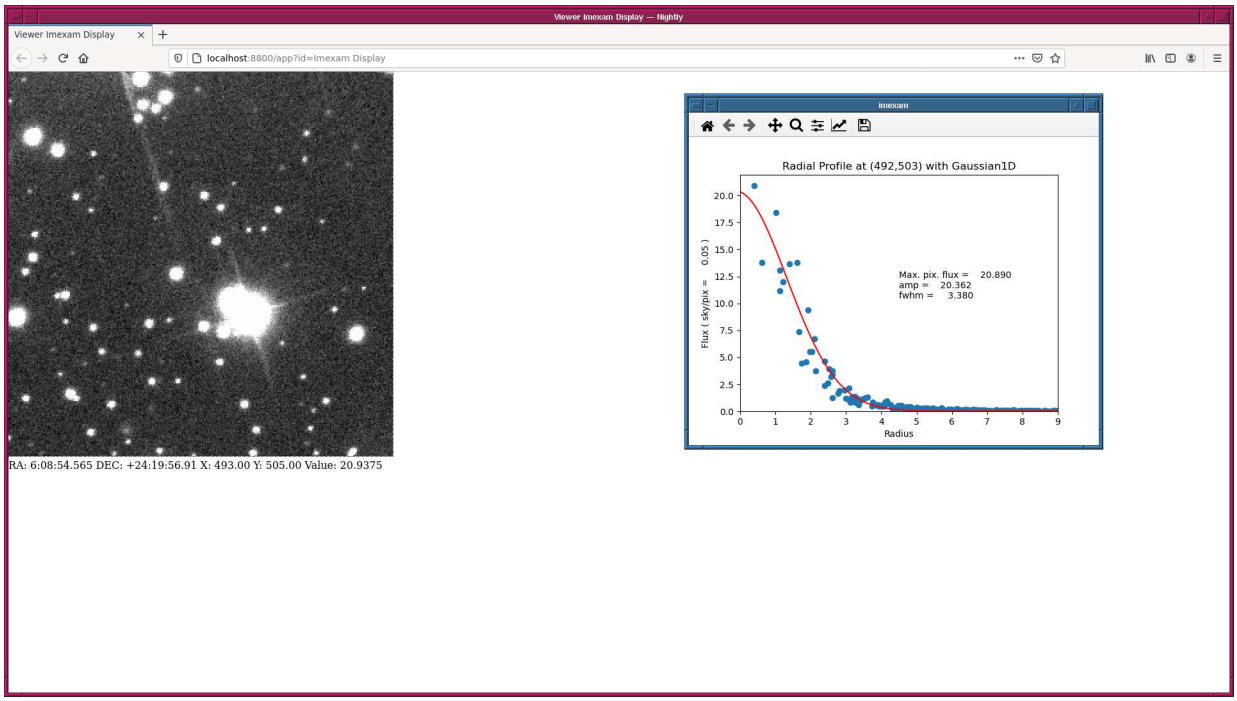

Figure 2: The Ginga HTML5 viewer for imexam.

```
% tail -2 imexam.log
com_center
xc=492.4177     yc=503.9044
```

4 For your training

To know more about `imexam` package, read following document.

- `https://imexam.readthedocs.io/en/0.9.1/imexam/walkthrough.html`
- `https://imexam.readthedocs.io/en/0.9.1/imexam/imexam_command.html`

5 Assignment

None.