Breve introducción a “imexam”

G. L. Baume - 2022

1. Introducción

El paquete “imexam” en Python consiste en un conjunto de librerías que permiten la exploración de imágenes o arreglos bidimensionales de datos. “imexam” utiliza diferentes herramientas gráficas y de análisis ya implementadas en Python como por ejemplo el paquete “matplotlib”.

El paquete “imexam” se halla afiliado a “Astropy”. Esto significa que si bien es compatible con “Astropy”, la instalación del paquete “imexam” debe llevarse a cabo de forma independiente. En particular, si se dispone instalada la distribución “Anaconda” de Python, se utiliza el siguiente comando:

```
> conda install -c conda-forge imexam
```

Dentro del paquete “imexam”, el método ”**`imexam()`**” es el que permite realizar la interacción con la imagen (o datos) desplegada. Esta interacción se lleva a cabo ubicando el cursor en la zona deseada de la imagen visualizada y utilizando ciertas “teclas especiales”. Dado que el paquete “imexam” intenta representar la contraparte Python de la tarea “imexamine” de IRAF, dichas “teclas especiales” y sus significados son similares, siendo:

```
2 Make the next plot in a new window
a Aperture sum, with radius region_size
b Return the 2D gauss fit center of the object
c Return column plot
d Return the Center of Mass fit center of the object
e Return a contour plot in a region around the cursor
g Return curve of growth plot
h Return a histogram in the region around the cursor
j 1D [Gaussian1D default] line fit
k 1D [Gaussian1D default] column fit
l Return line plot
m Square region stats, in [region_size],default is median
r Return the radial profile plot
s Save current figure to disk as [plot_name]
t Make a fits image cutout using pointer location
w Display a surface plot around the cursor location
x Return x,y,value of pixel
y Return x,y,value of pixel
```

Cabe notar que si se utiliza el visualizador “Ginga”, para habilitar / deshabilitar el uso de las teclas del método “**`imexam()`**”, es necesario:

- Posicionar el cursor sobre la imagen desplegada

- Utilizar la tecla “**`i`**” para habilitar “imexam” y la tecla “**`q`**” para deshabilitarlo.

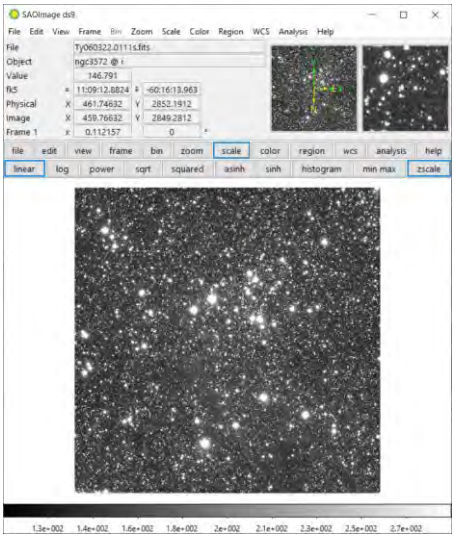
2. Visualizadores

El paquete “imexam” puede utilizarse sin necesidad de interactuar con los datos (imagen o arreglo bidimensional), pero para su uso de forma interactiva es necesario disponer de un programa que provea la interfaz donde se despliegue la imagen (o datos) a analizar. Hasta el momento solo los visualizadores “DS9” y “Ginga” pueden ser utilizados. Ambos visualizadores pueden ser instalados en Linux o en Windows. No obstante, el visualizador “DS9” solo es posible utilizarlo con “imexam” en Linux. O sea que para los usuarios en Windows la única posibilidad es la de utilizar el visualizador “Ginga”. Cabe notar además que el visualizador “Ginga” ha sido desarrollado en Python, siendo entonces más compatible con el paquete “imexam”.

La forma de instalar los visualizadores “DS9” y/o “Ginga” se encuentra descripta en sus respectivos “webpages”:

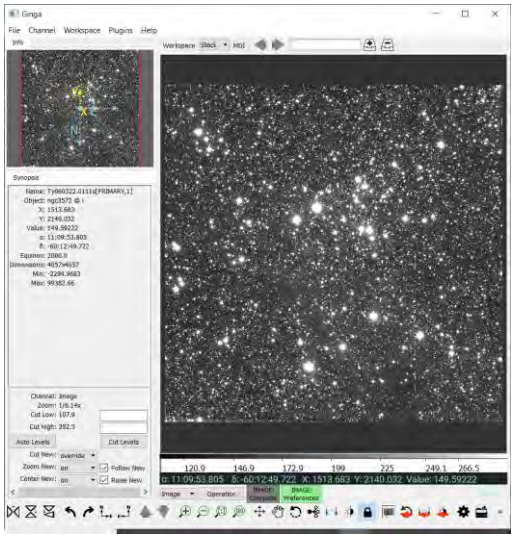
DS9

https://sites.google.com/cfa.harvard.edu/saoimageds9



Ginga

https://ginga.readthedocs.io/en/stable/



3. Uso del paquete “imexam”

Existen diversas formas de utilizar el paquete “imexam”. Algunas de ellas son más simples y otras más elaboradas. Aunque, en general las mas simples suelen ser más limitadas, mientras que las más elaboradas permiten aprovechar todo el potencial de posibilidades que ofrece el paquete.

Se nota además que la disponibilidad y/o simpleza de cada opción de uso dependerá tanto del sistema operativo (Linux o Windows) como del visualizador (“DS9” o “Ginga”) utilizado.

Se pueden enumerar entonces las siguientes formas de operación,

a) Con línea de comandos

Esta opción es más adecuada para ser utilizada en Linux. En Windows suele tener complicaciones por el uso combinado de la interfaz gráfica del visualizador y para el despliegue de los eventuales gráficos asociados.

La manera de emplear esta opción en Linux consiste en:

- Entrar en el entorno “Python” (o eventualmenteen el entorno “Interactive Python”). El comando sería:

```
> python
```

- Dentro del entorno elegido, el paso que sigue es introducir la sucesión de comandos:

```
>>> import imexam
>>> viewer = imexam.connect()
>>> viewer.load_fits(<image_name>)
>>> viewer.imexam()
```

  Ellos permiten cargar del paquete “imexam”, realizar la conexión con el visualizador, cargar un archivo de imagen (o un conjunto de datos bidimensionales) y habilitar los comandos del visualizador

- Luego, con el cursor localizado sobre el visualizador, utilizar las “teclas especiales” del método “**`imexam()`**”

- El paso final es la ejecución de un comando que cierre la conexión con el visualizador

```
>>> viewer.close()
```

Para más detalles del significado y opciones de cada comando, así como otros comandos disponibles ver la documentación de “imexam” (https://imexam.readthedocs.io/en/0.9.1/)

b) Con un “script”

Esta es una opción más práctica que la anterior y consiste en ejecutar un archivo con los comandos deseados. La ejecución se realizará de forma uniforme y siempre con los mismos parámetros. Este caso también es adecuado para el entorno Linux. En Windows existen las mismas complicaciones que en el caso a).

Un ejemplo viene dado por el archivo “**imexa.py**“ (NO “**imexam.py**”), presentado a continuación.

```
#!/usr/bin/env python

# Inport packages
import sys
import os
import os.path
import imexam

# Input image (first argument)
n = len(sys.argv)
if n<2:
    in_name = input("Enter a file name: ")
else:
    in_name = sys.argv[1]
image_name = str(in_name)

# Start DS9 viewer window (only in Linux)
viewer = imexam.connect()

# Load and display image in viewer
viewer.load_fits(image_name)
viewer.scale()

# Interactively examine the image.

# Set logfile
if os.path.isfile('logfile.txt'):
    os.remove('logfile.txt')
else:
    pass
viewer.setlog('logfile.txt')

viewer.unlearn()     # Unlearn all the imexam parameters
viewer.imexam()      # Run imexam
viewer.close()       # Close viewer
```

Este archivo se puede emplear como “script” o como “método de Python”. Al ejecutar este archivo se dispone del acceso a las “teclas especiales” de “imexam” y se produce un archivo (“**logfile.txt**”) con los resultados de la actividad realizada.

En este caso el comando a usar sería simplemente:

```
> ./imexa.py
```

c) Con un “notebook” (en “Jupyter”)

Esta es una variante más elaborada de la “línea de comandos”, aunque es posible utilizarla sin inconvenientes tanto en Linux como en Windows.

Caben destacar algunos aspectos a tener en cuenta al usar esta opción:

- En el caso de estar trabajando en Windows, los gráficos se deben desplegar dentro del notebook. Esto se logra mediante el uso de la “magicword” “**`%matplotlib`**”. De esta forma:
  - Para usar gráficos diferentes para cada requerimiento

```
%matplotlib inline
```

  - Para usar un único gráfico para todos los requerimientos

```
%matplotlib notebook
```

- Una “notebook” requiere el uso de un ambiente para ejecutarla. Dado que el uso de “imexam” de forma interactiva requiere el uso de un visualizador local, la “notebook” debe ser utilizada en “Jupyter” y no es posible utilizar el entorno “Google Colab”.

- El archivo “**`imexa.py`**“ (NO “**`imexam.py`**”) presentado en el caso b) puede utilizarse dentro de un “notebook” como un método de Python de la siguiente forma:

```
from imexa import imexa
imexa('imagen.fits', viewer='ginga')
```

Un ejemplo simple de “notebook”, para el uso del paquete “imexam”, se presenta en el archivo “**`imexa_ginga.ipnb`**”. Como su nombre indica, este “notebook” se halla pensado para ser utilizado con “Ginga”, pero su adaptación a “DS9” es totalmente simple.

d) Con un “plugin” del visualizador

https://github.com/ejeschke/ginga/blob/master/experimental/plugins/Imexam.py

Esta es una opción que por el momento se encuentra en fase experimental y solo disponible para su uso con el visualizador “Ginga” (tanto en Linux como en Windows). Este “plugin” permite incluir las facilidades del paquete “imexam” como una opción adicional de dicho visualizador.

Para emplear esta opción es necesario instalar el “plugin” y esto se hace colocando el archivo “**`Imexam.py`**” (con “**`I`**” mayúscula y con “**`m`**” al final) en la carpeta:

`“$HOME/.ginga/plugins”`

Donde, usualmente, “**`$HOME`**” viene dada por:

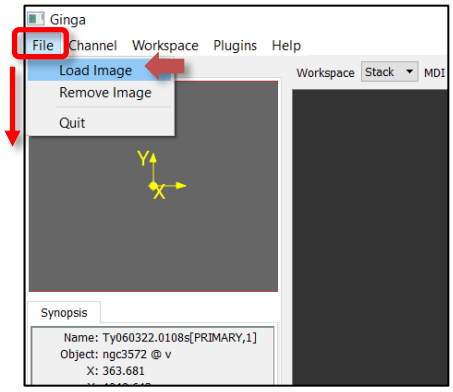
- En Linux: “**`/home/<usuario>/`**”
- En Windows: “**`C:\users\<usuario>\`**”

Para utilizar el “plugin”, es necesario abrir el visualizador “Ginga” con el comando:

```
> ginga --plugins=Imexam
```

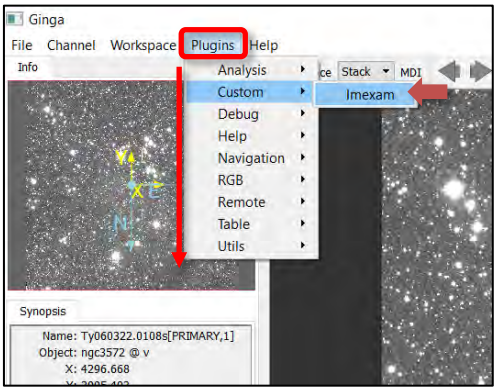
Al abrirse la ventana del programa, cargar la imagen que se desea analizar, realizar los siguientes pasos:

a) “`File > Load Image`”



para cargar una imagen (.fits)

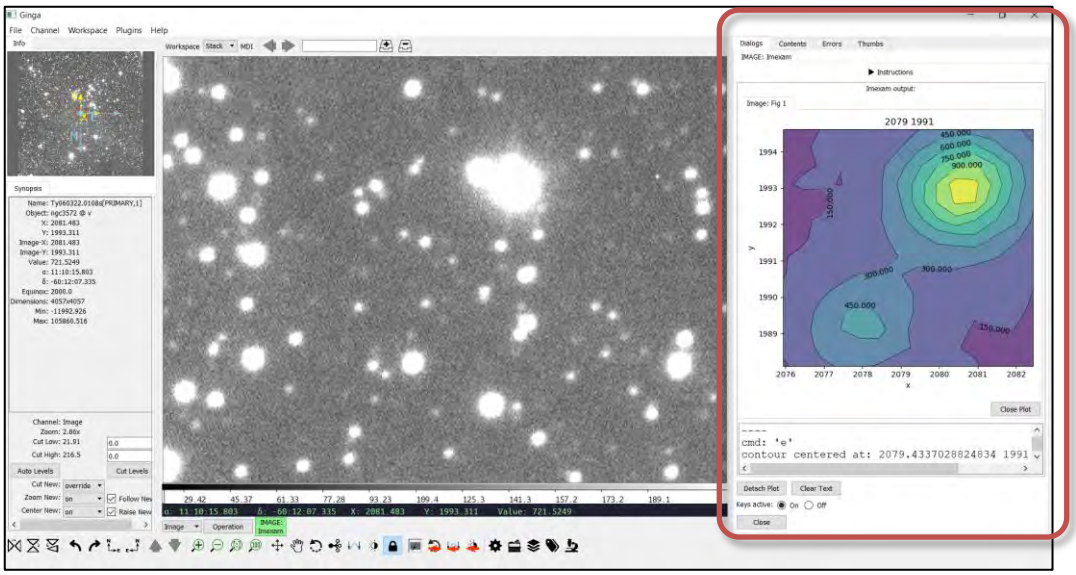
b) “`Plugins > Custom > Imexam`”



para abrir el plugin

Entonces se encontrarán disponibles los comandos usuales de “imexam” en la ventana principal de “Ginga” y en el panel de la derecha se desplegarán los gráficos y parámetros correspondientes:

Ventana IMEXAM



En síntesis, se tienen cuatro formas de utilizar el paquete “imexam”, todas ellas disponibles en Linux y solo las dos últimas en Windows. Las tres primeras pueden utilizar los visualizadores “DS9” o “Ginga” y la última disponible solo para “Ginga”.

Comentarios Finales

El paquete “imexam” es una herramienta que provee varias posibilidades de uso para examinar imágenes (o arreglos bidimensionales). Su flexibilidad de uso y sus diversas posibilidades permiten adaptarlo a diferentes necesidades del usuario y al sistema operativo disponible.

Referencias e información adicional

- Paquete “imexam” de Python
  https://imexam.readthedocs.io/en/0.9.1/
  https://github.com/spacetelescope/imexam

- SAOImageDS9: An image display and visualization tool for astronomical data
  https://sites.google.com/cfa.harvard.edu/saoimageds9

- Ginga: Image Viewer and Toolkit
  https://ginga.readthedocs.io/en/stable/
  https://github.com/ejeschke/ginga

- HST User Documentation: Displaying Imaging Data
  https://hst-docs.stsci.edu/hstdhb/4-hst-data-analysis/4-5-displaying-imaging-data