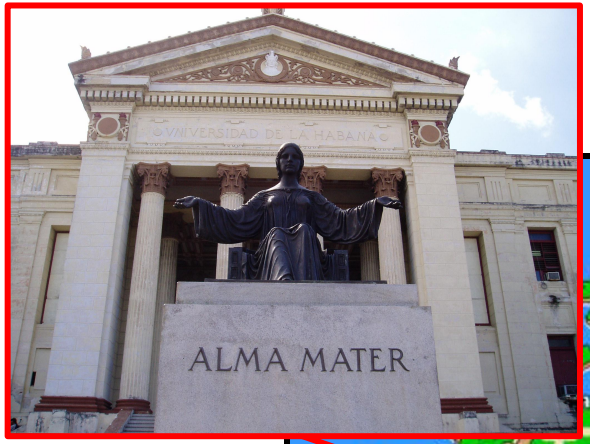
Mi CASA

Venturas y desventuras de una radio-astrónoma en Python

Licenciatura en Física



Máster en Física y Matemáticas
Especialidad en Astrofísica

UNIVERSIDAD DE GRANADA

Máster en Física y Matemáticas
Especialidad en Astrofísica

UNIVERSIDAD DE GRANADA

Instituto de Astrofísica de Andalucía

Doctorado en Física y Matemáticas
Especialidad en Astrofísica

UNIVERSIDAD DE GRANADA

Instituto de Astrofísica de Andalucía

My Drive - Google Drive | CASA - Google Slides | Instituto de Astrofísica

https://www.iaa.csic.es/#

Search

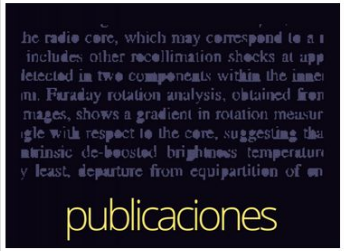
Most Visited | Getting Started | Latest Headlines | From Google Chrome | Plex

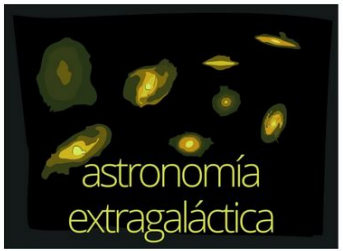
INSTITUTO DE ASTROFÍSICA DE ANDALUCÍA, IAA-CSIC

ACTUALIDAD INVESTIGACIÓN TECNOLOGÍA INSTITUTO DIVULGACIÓN OBSERVATORIOS SITEMAP CONTACTO

Investigación

Formación | Ciencia | Destacados





Estudiamos cómo nacen las estrellas

Filamento de gas denso en Orion

Image: R. Friesen, Dunlap Institute; J. Pineda, MPE; GBO/AUI/NSF

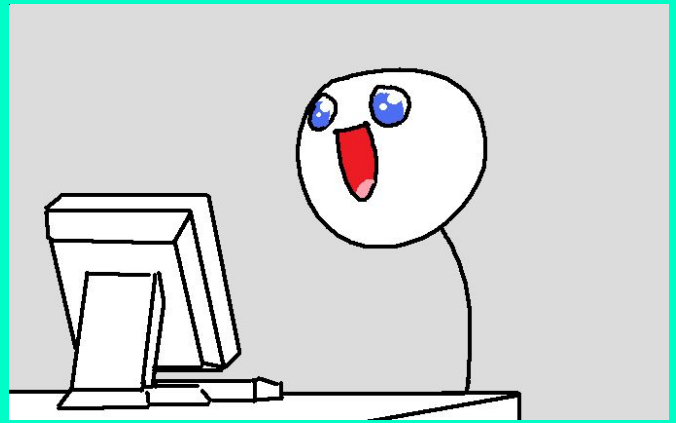
¿Python?

Python

C++

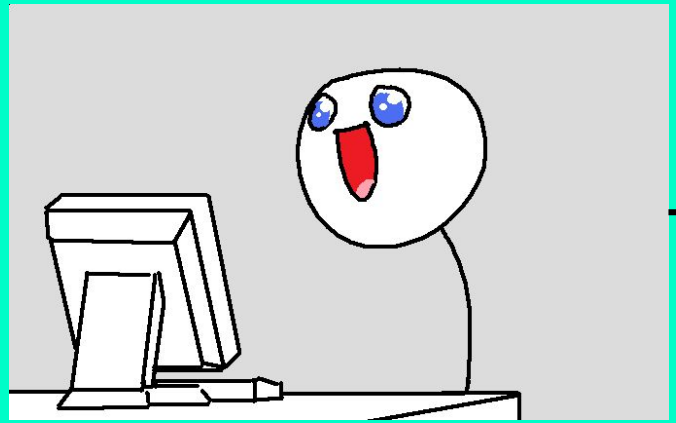
Radio interferómetro ALMA

Radio astrónomo feliz



Radio interferómetro ALMA

Radio astrónomo feliz



PYTHON

CASA

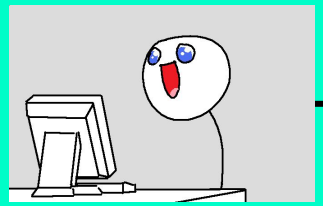
Common Astronomy Software Applications

C++

MAGIA

Radio interferómetro ALMA

Radio astrónomo feliz



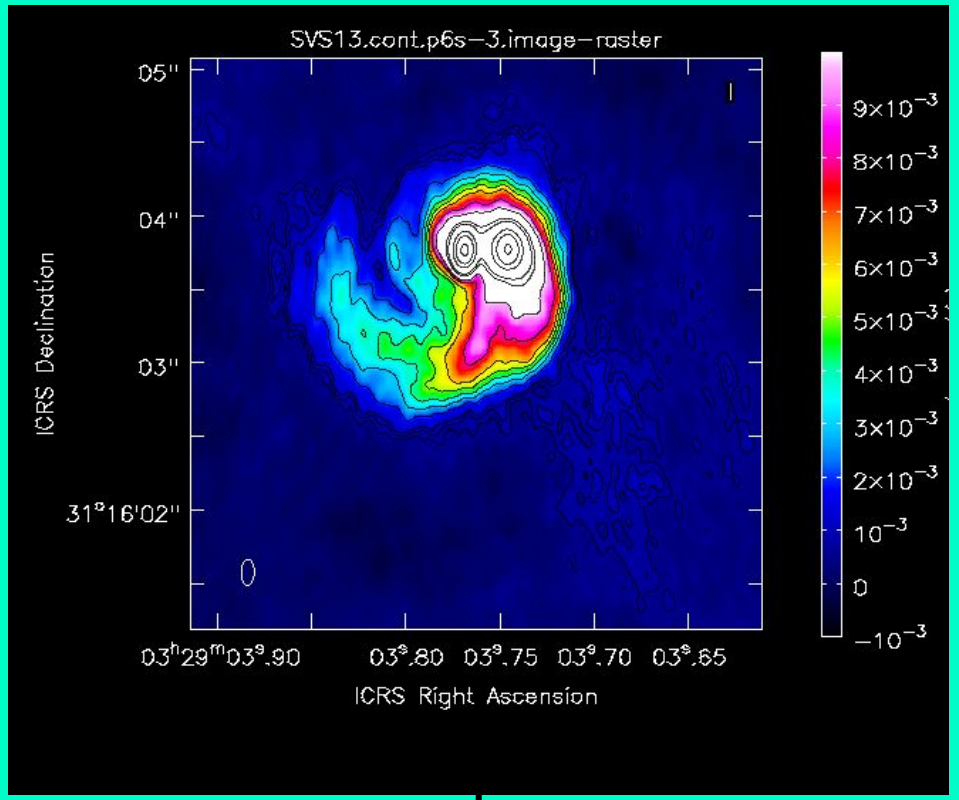
PYTHON

CASA

Common Astronomy Software Applications

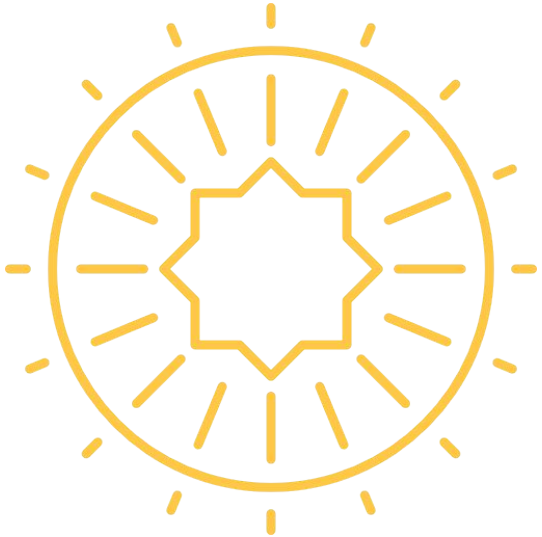
C++

MAGIA

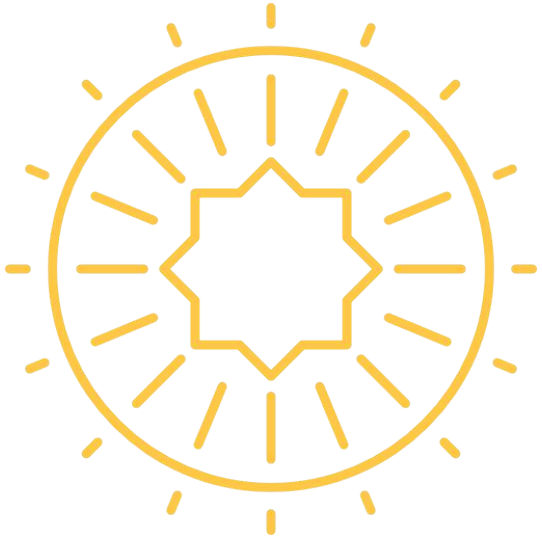


Radio-interferometría

Óptico



Óptico



Radio

Icons made by Freepik from www.flaticon.com

Nube molecular vista en

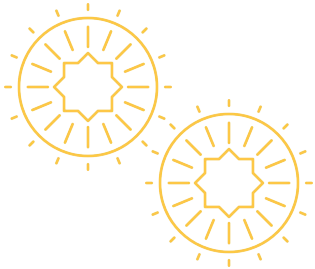
óptico **infrarrojo**

Alves et al. 2001, Nature, 409, 159

RADIO

- La emisión radio atraviesa el material alrededor de la protoestrella
- Se puede detectar desde la superficie de la tierra

Resolución angular ∝ diámetro antena



Icons made by Freepik from www.flaticon.com

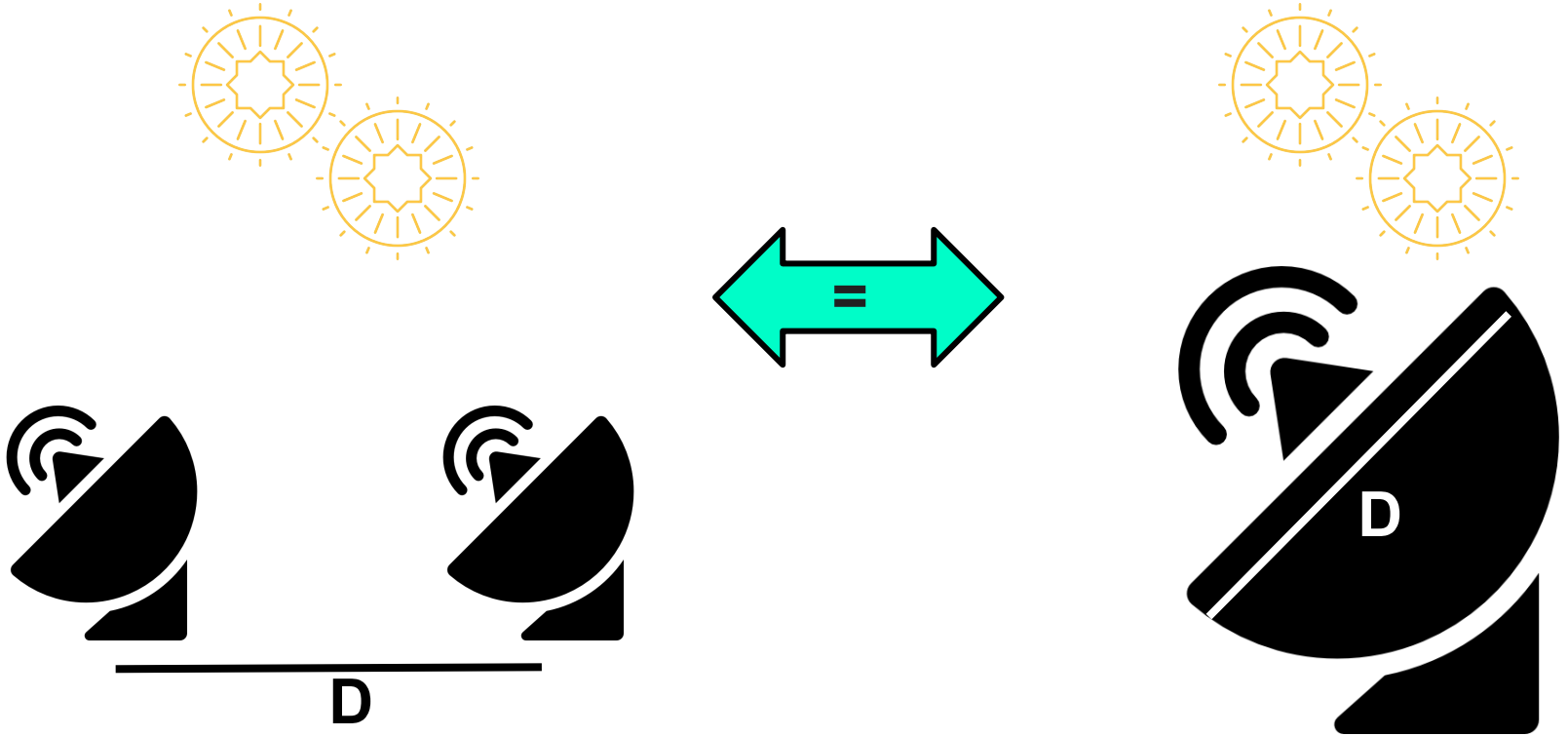
Arecibo Observatory (305 m)

Puerto Rico
1963

Five Hundred Meter Aperture Spherical Telescope (FAST)

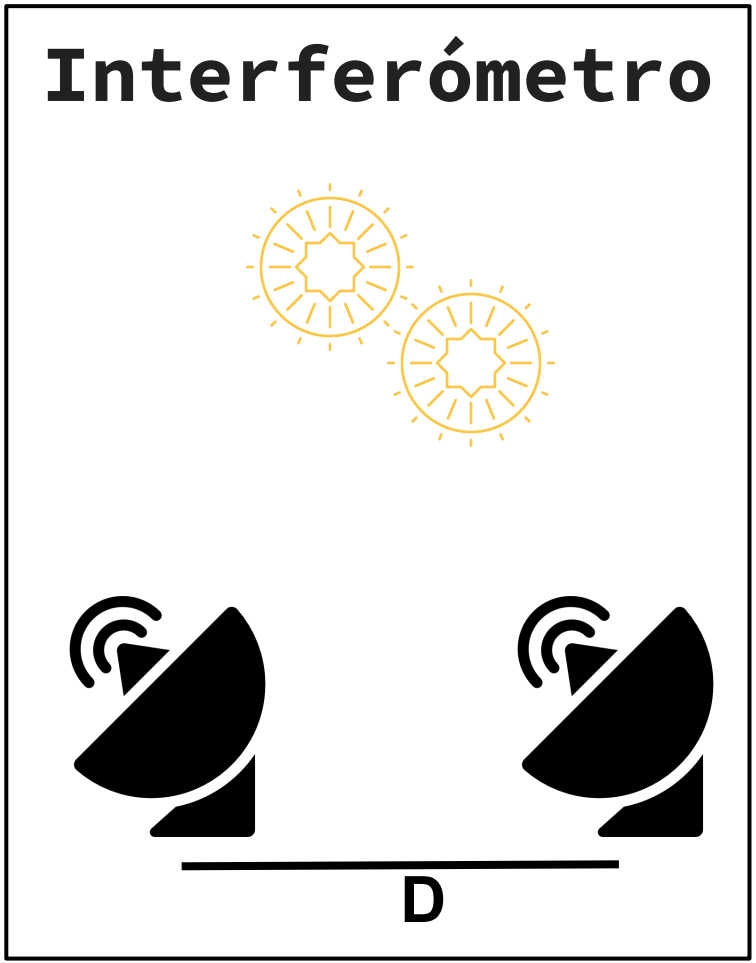
China 2016

Resolución angular ∝ diámetro antena

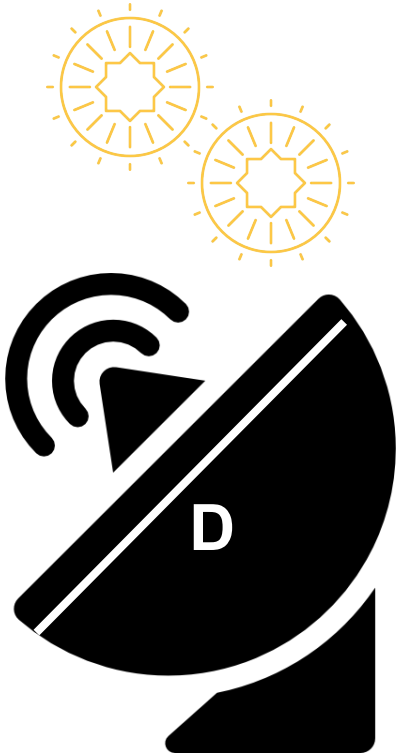


Icons made by Freepik from www.flaticon.com

Interferómetro



=



RADIO

- La emisión radio atraviesa el material alrededor de la protoestrella
- Se puede detectar desde la superficie de la tierra

INTERFEROMETRÍA

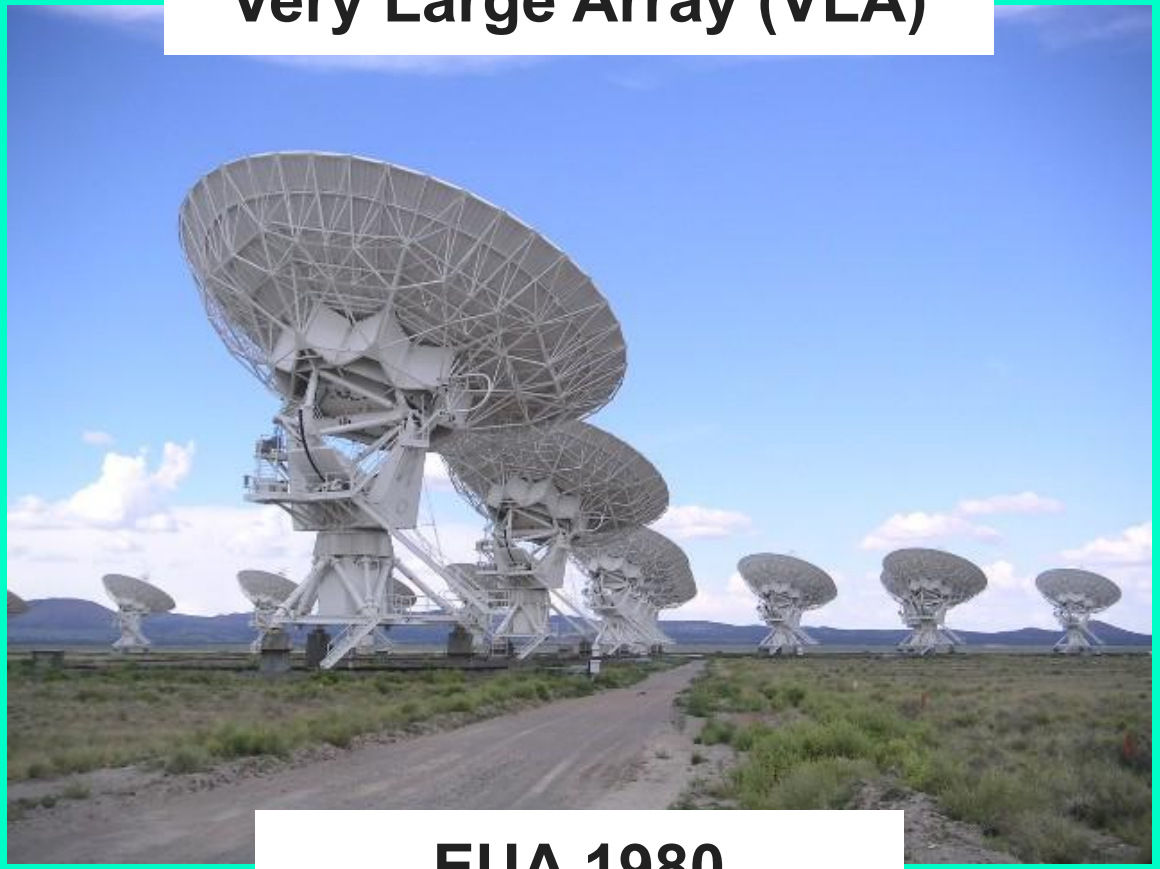
- Varias antenas que observan el mismo objeto se combinan para alcanzar una mejor resolución angular. Es equivalente a observar con una antena de tamaño igual a la separación máxima entre antenas.

Contacto (1997)

Realidad vs ficción

27 x 25 m
Bmax = 36.4 km

Very Large Array (VLA)



EUA 1980

Atacama Large Millimeter/submillimeter Array (ALMA)

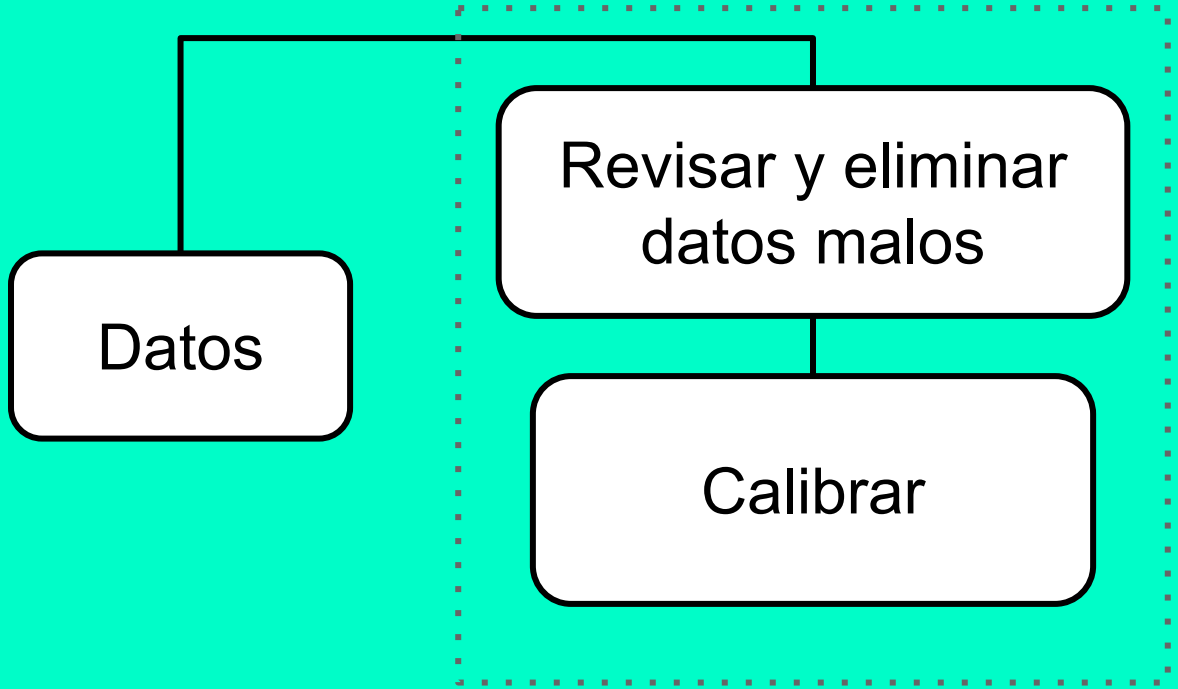
54 x 12 m + 12 x 7m
Bmax = 16 km

Chile 2011

Reducción de datos

Datos

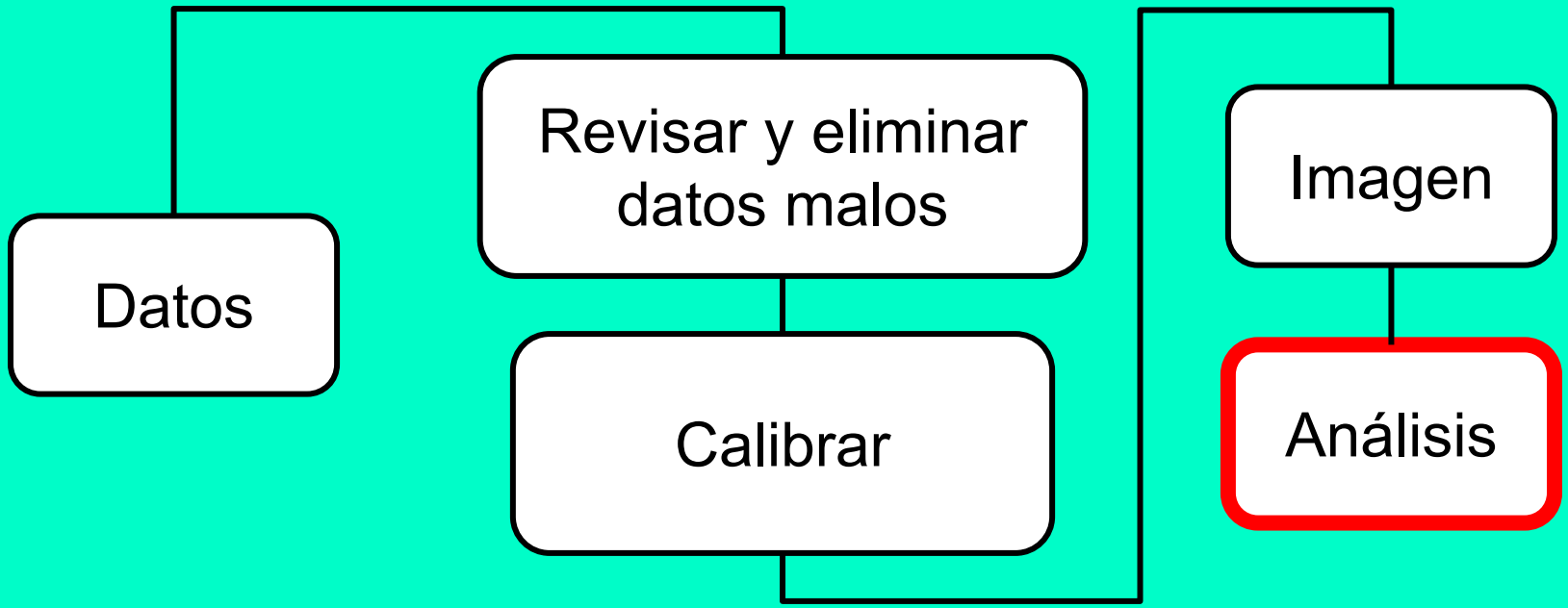
Reducción de datos

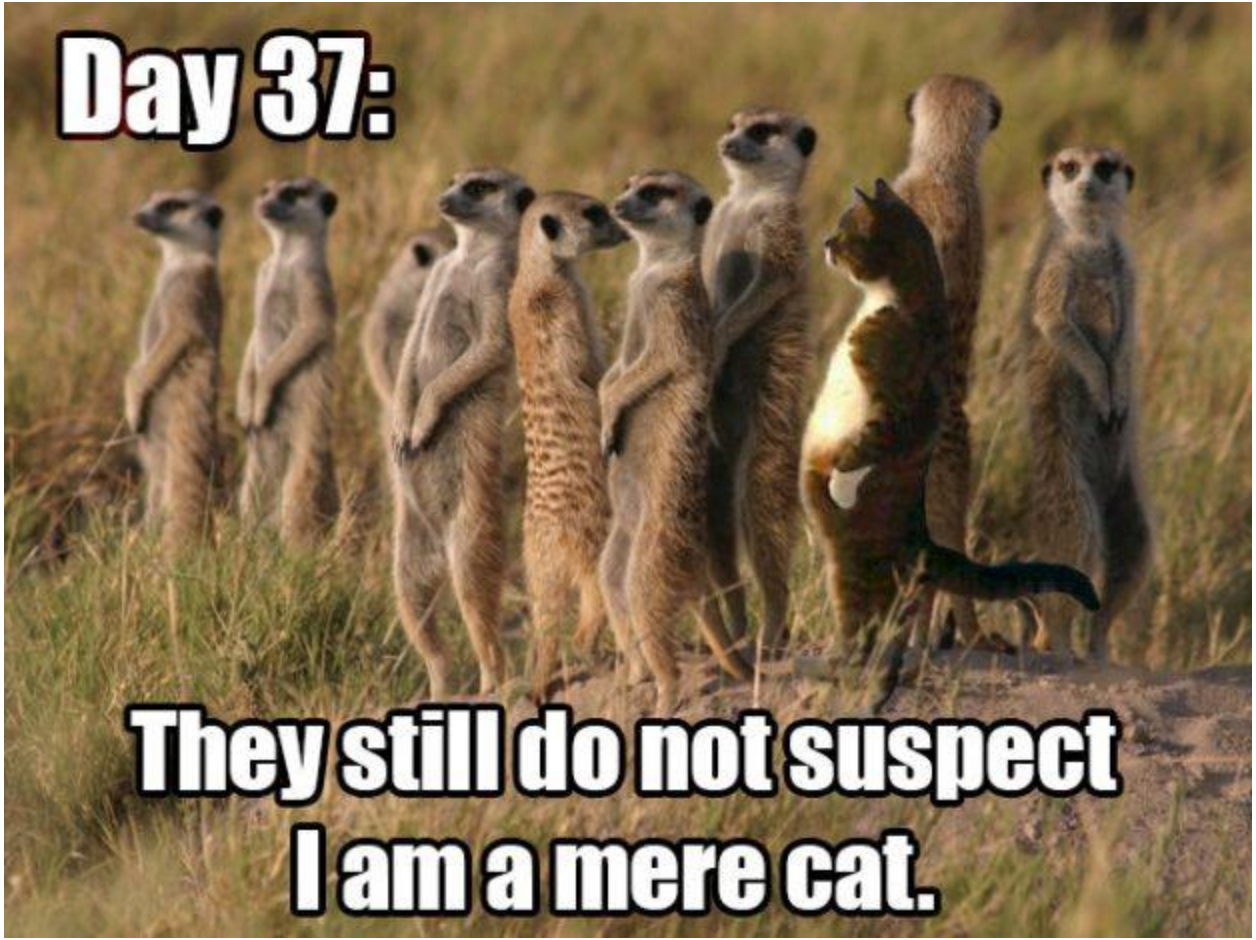


Reducción de datos

Reducción de datos

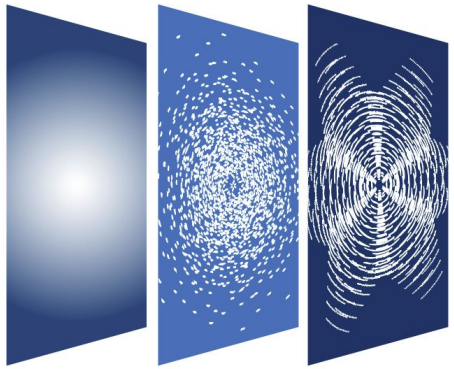
Reducción de datos





From Thinking in the Open

PYTHON



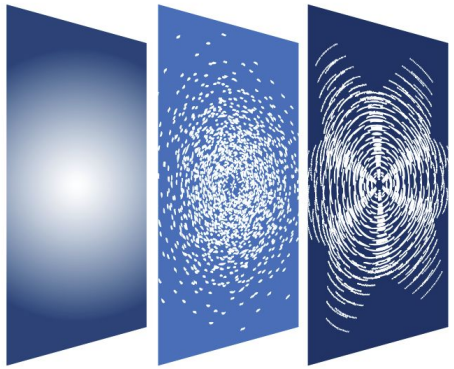
CASA

Common Astronomy Software Applications

C++

```
anika@battlestar-galactica: ~/Documents/Trabajo/PresPyDay
OK, no bytes read while starting xvfb, timeout?...10859...
OK, no bytes read while starting xvfb, timeout?...10859...
OK, no bytes read while starting xvfb, timeout?...10859...
OK, no bytes read while starting xvfb, timeout?...10859...
10854-casaviewer-svr: cannot connect to X server localhost:86.0

_____________________________________________________________________
    For help use the following commands:
    tasklist                - Task list organized by category
    taskhelp                - One line summary of available tasks
    help taskname           - Full help for task
    toolhelp                - One line summary of available tools
    help par.parametername - Full help for parameter name
_____________________________________________________________________
Activating auto-logging. Current session state plus future input saved.
Filename       : ipython-20181211-183852.log
Mode           : backup
Output logging : False
Raw input log  : False
Timestamping   : False
State          : active
*** Loading ATNF ASAP Package...
*** ... ASAP (rev#36500) import complete ***

CASA <2>:
```

PYTHON



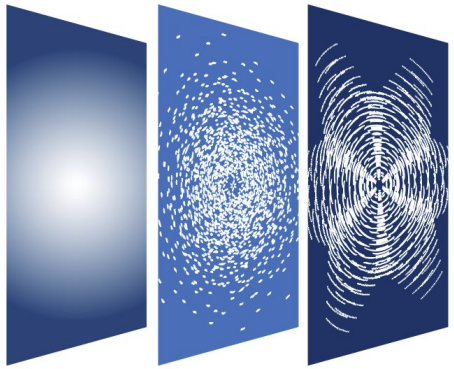
CASA

Common Astronomy Software Applications

C++

```
anika@battlestar-galactica: ~/Documents/Trabajo/PresPyDay
OK, no bytes read while starting xvfb, timeout?...10859...
OK, no bytes read while starting xvfb, timeout?...10859...
OK, no bytes read while starting xvfb, timeout?...10859...
OK, no bytes read while starting xvfb, timeout?...10859...
10854-casaviewer-svr: cannot connect to X server localhost:86.0

________________________________________________________________________
    For help use the following commands:
    tasklist                - Task list organized by category
    taskhelp                - One line summary of available tasks
    help taskname           - Full help for task
    toolhelp                - One line summary of available tools
    help par.parametername - Full help for parameter name
________________________________________________________________________
Activating auto-logging. Current session state plus future input saved.
Filename       : ipython-20181211-183852.log
Mode           : backup
Output logging : False
Raw input log  : False
Timestamping   : False
State          : active
*** Loading ATNF ASAP Package...
*** ... ASAP (rev#36500) import complete ***

CASA <2>: clean(vis='svs13.split',imagename='svs13.im',niter=500')
```

PYTHON



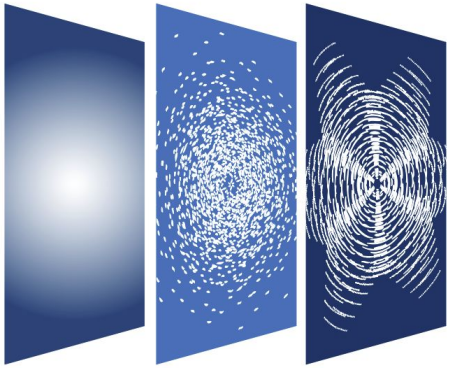
CASA

Common Astronomy Software Applications

C++

```
anika@battlestar-galactica: ~/Documents/Trabajo/PresPyDay
OK, no bytes read while starting xvfb, timeout?...10859...
OK, no bytes read while starting xvfb, timeout?...10859...
OK, no bytes read while starting xvfb, timeout?...10859...
OK, no bytes read while starting xvfb, timeout?...10859...
10854-casaviewer-svr: cannot connect to X server localhost:86.0

________________________________________________________________________
    For help use the following commands:
    tasklist                - Task list organized by category
    taskhelp                - One line summary of available tasks
    help taskname           - Full help for task
    toolhelp                - One line summary of available tools
    help par.parametername - Full help for parameter name
________________________________________________________________________
Activating auto-logging. Current session state plus future input saved.
Filename       : ipython-20181211-183852.log
Mode           : backup
Output logging : False
Raw input log  : False
Timestamping   : False
State          : active
*** Loading ATNF ASAP Package...
*** ... ASAP (rev#36500) import complete ***

CASA <2>: execfile('findsources.py')
```

PYTHON

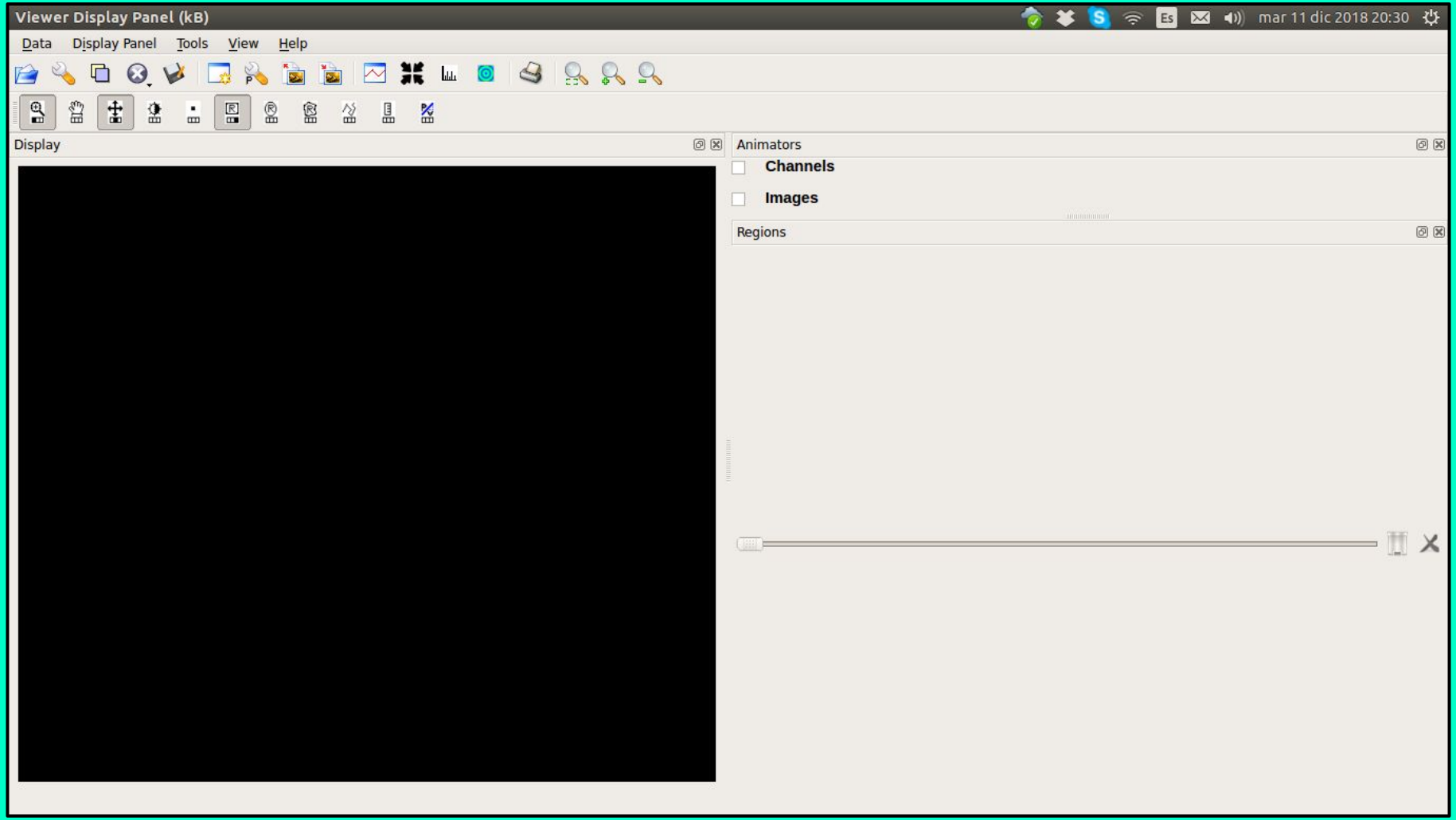


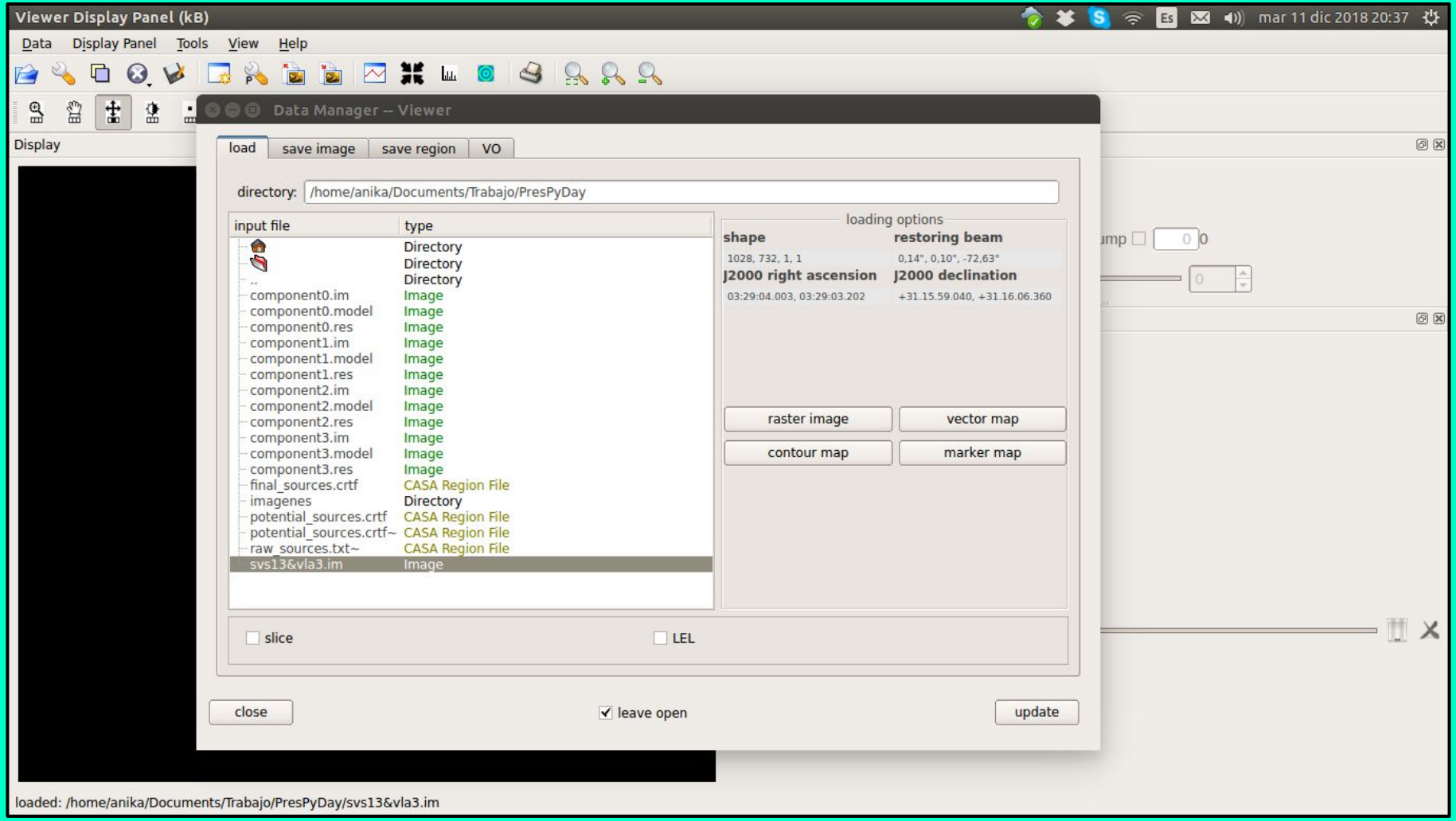
CASA

Common Astronomy Software Applications

C++

```
anika@battlestar-galactica: ~/Documents/Trabajo/PresPyDay
OK, no bytes read while starting xvfb, timeout?...10859...
OK, no bytes read while starting xvfb, timeout?...10859...
OK, no bytes read while starting xvfb, timeout?...10859...
OK, no bytes read while starting xvfb, timeout?...10859...
10854-casaviewer-svr: cannot connect to X server localhost:86.0

_____________________________________________________________________
    For help use the following commands:
    tasklist                - Task list organized by category
    taskhelp                - One line summary of available tasks
    help taskname           - Full help for task
    toolhelp                - One line summary of available tools
    help par.parametername - Full help for parameter name
_____________________________________________________________________
Activating auto-logging. Current session state plus future input saved.
Filename       : ipython-20181211-183852.log
Mode           : backup
Output logging : False
Raw input log  : False
Timestamping   : False
State          : active
*** Loading ATNF ASAP Package...
*** ... ASAP (rev#36500) import complete ***

CASA <2>: viewer
```


Viewer Display Panel (kB)
mar 11 dic 2018 20:30
Data
Display Panel
Tools
View
Help
Display
Animators
Channels
Images
Regions

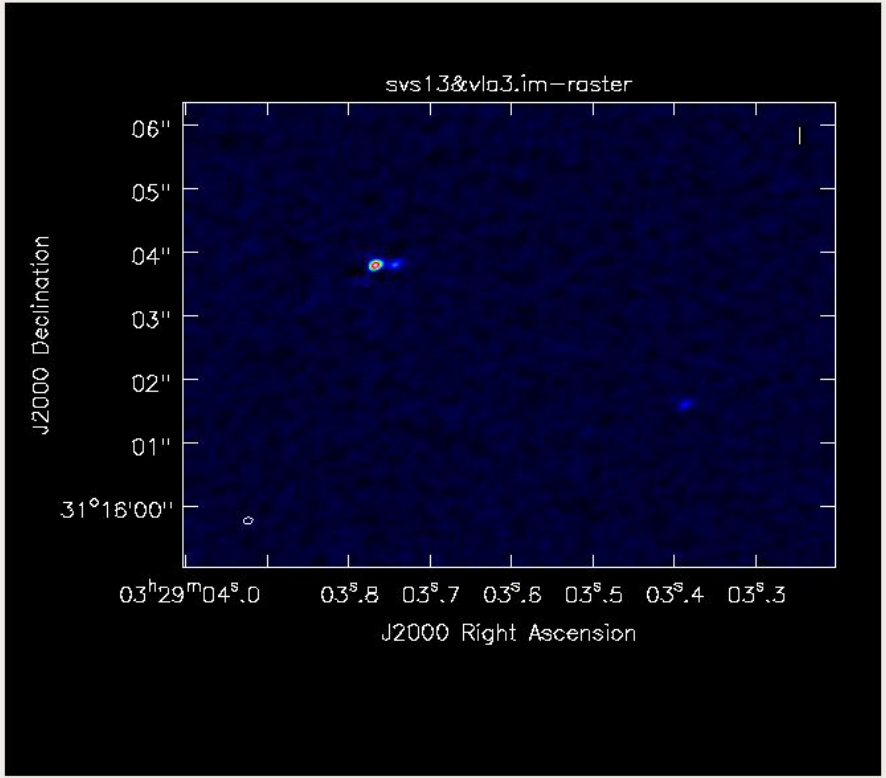

Viewer Display Panel (kB)
mar 11 dic 2018 20:37
Data Display Panel Tools View Help
Display
Data Manager -- Viewer
load
save image
save region
VO
directory: /home/anika/Documents/Trabajo/PresPyDay
input file
type
Directory
Directory
Directory
component0.im Image
component0.model Image
component0.res Image
component1.im Image
component1.model Image
component1.res Image
component2.im Image
component2.model Image
component2.res Image
component3.im Image
component3.model Image
component3.res Image
final_sources.crtf CASA Region File
imagenes Directory
potential_sources.crtf CASA Region File
potential_sources.crtf~ CASA Region File
raw_sources.txt~ CASA Region File
svs13&vla3.im Image
loading options
shape
restoring beam
1028, 732, 1, 1
0,14", 0,10", -72,63°
J2000 right ascension
J2000 declination
03:29:04.003, 03:29:03.202
+31.15.59.040, +31.16.06.360
raster image
vector map
contour map
marker map
slice
LEL
close
leave open
update
loaded: /home/anika/Documents/Trabajo/PresPyDay/svs13&vla3.im

Viewer Display Panel (kB)

mar 11 dic 2018 20:32

Data Display Panel Tools View Help

Display



Animators

Stokes

Images

Rate: 10 Jump 0 0

0 0

Regions

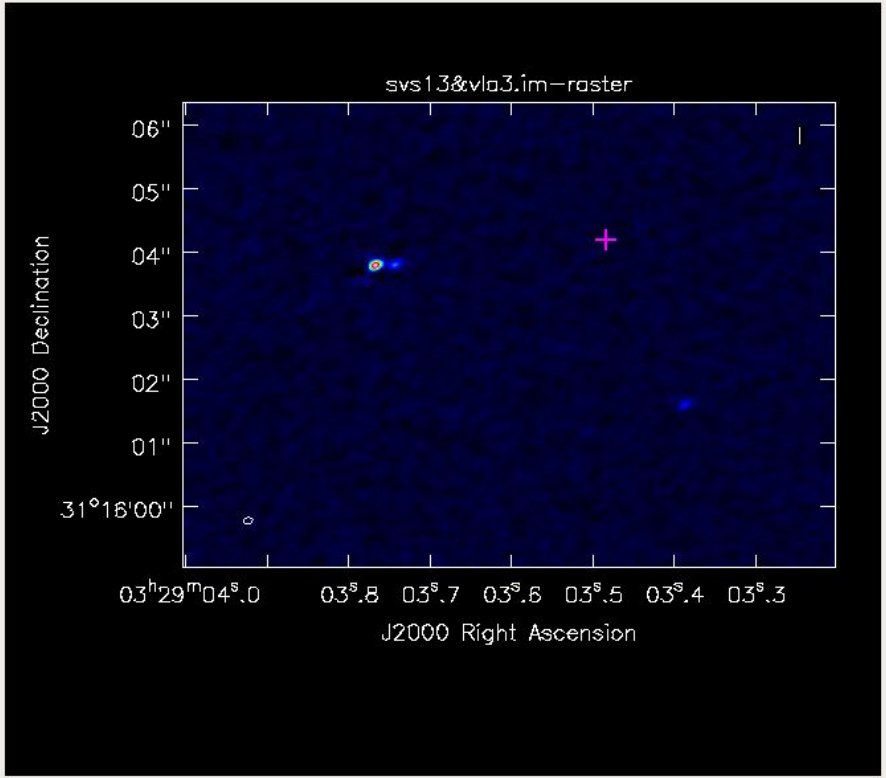
loaded: /home/anika/Documents/Trabajo/PresPyDay/svs13&vla3.im

Viewer Display Panel (kB)

mar 11 dic 2018 20:34

Data Display Panel Tools View Help

Display



Animators

Stokes

Images

Rate: 10 Jump 0 0

0 0

Regions

Properties Statistics File Histogram

point frames 0 1 selected annotation

coordinates line text

system J2000

bounding box (width X height) 0 X 0 "

center x 03:29:03.483 units sexagesimal

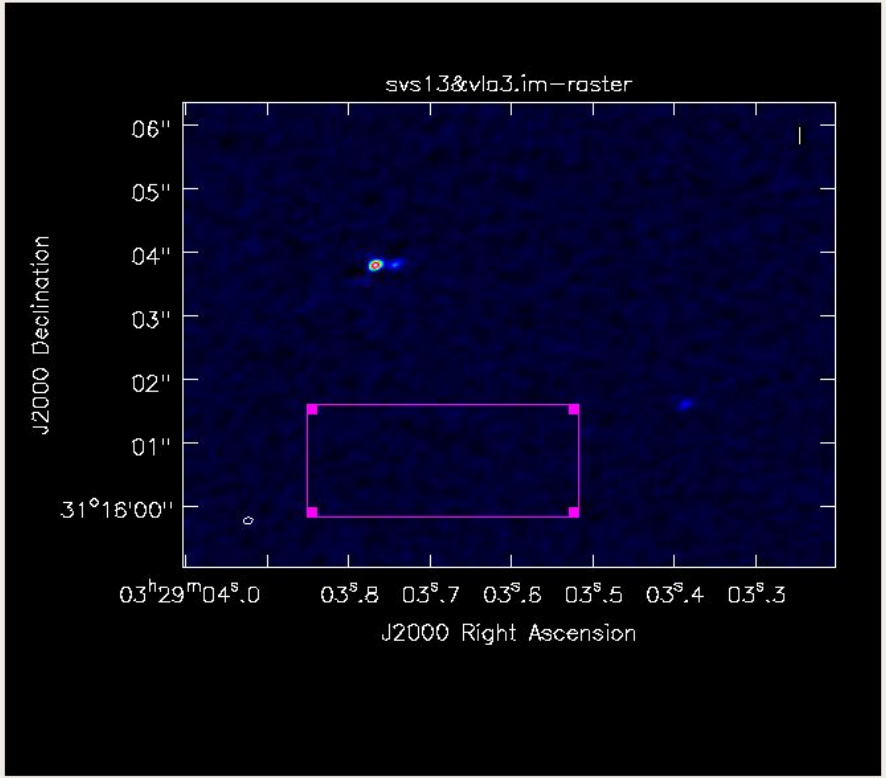
center y 31.16.04.186 units sexagesimal

Viewer Display Panel (kB)

mar 11 dic 2018 20:33

Data Display Panel Tools View Help

Display



Animators

Stokes

Images

Rate: 10 Jump 0 0

0 0

Regions

Properties Statistics Fit File Histogram

svs13vla3.im

Stokes	Velocity	Frame	Doppler
I	35.5191km/s	LSRK	RADIO
Frequency	**BrightnessUnit**	**BeamArea**	**Npts**
3.29966e+10	Jy/beam	162.199	74976
Sum	**FluxDensity**	**Mean**	**Rms**
2,477836e-02	1,527652e-04	3,304839e-07	1,005348e-05
Std dev	**Minimum**	**Maximum**	**region count**
1,004811e-05	-3,665197e-05	3,739555e-05	1

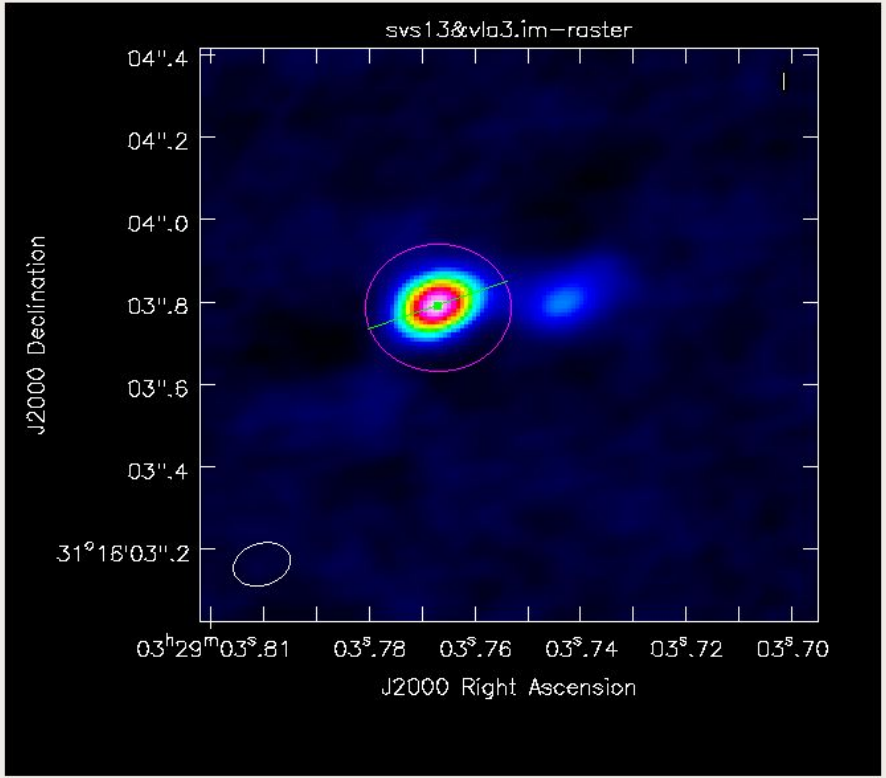
next

Viewer Display Panel (kB)

mar 11 dic 2018 20:35

Data Display Panel Tools View Help

Display



Animators

Stokes

Images

Rate: 10 Jump 0 0

0 0

Regions

Properties Statistics Fit File Histogram

svs13vla3.im

Ra_J2000	Dec_J2000	W-Majorax	W-Minorax
03:29:03.767	31.16.03.79	0.170551 arcsec	0.116812 arcsec
W-Posang	**Xcen**	**Ycen**	**I-Majorax**
108.953 deg	303.198 pix	475.451 pix	17.0551 pix
I-Minorax	**I-Posang**	**Radeg**	**Decdeg**
11.6812 pix	18.9526 deg	412.266 deg	31.2677 deg
IntegrFlux	**PeakFlux**	**Converged**	
0.00133877 Jy	0.000961938 Jy/bea	YES	

next

gaussfit sky component mark center green

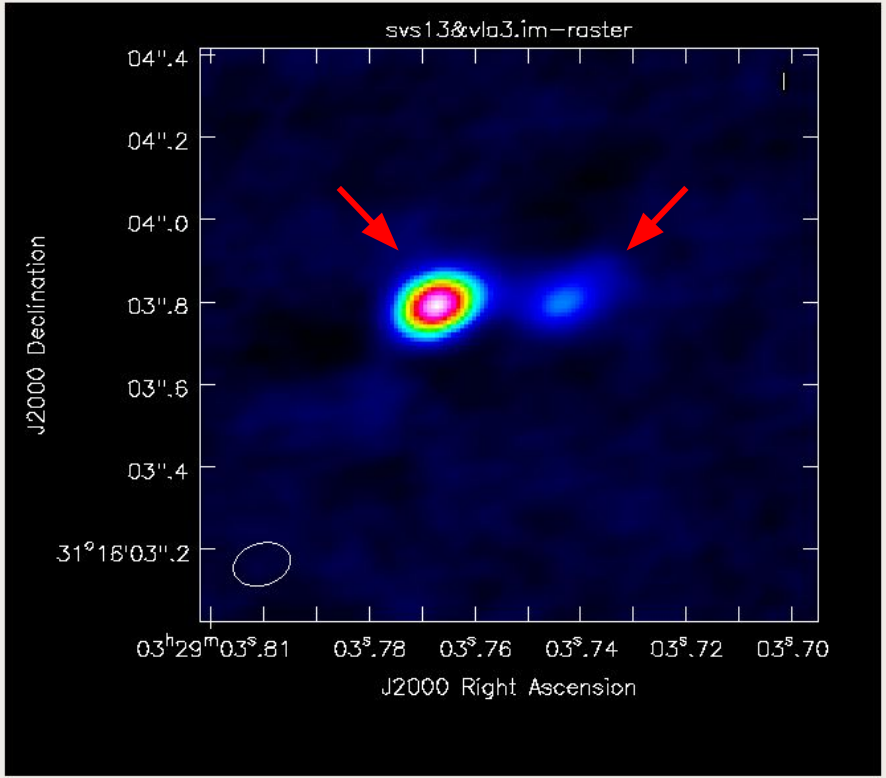
loaded: /home/anika/Documents/Trabajo/PresPyDay/svs13&vla3.im

Viewer Display Panel (kB)

mar 11 dic 2018 20:35

Data Display Panel Tools View Help

Display



Animators

Stokes

Images

Rate: 10 Jump 0 0

0 0

Regions

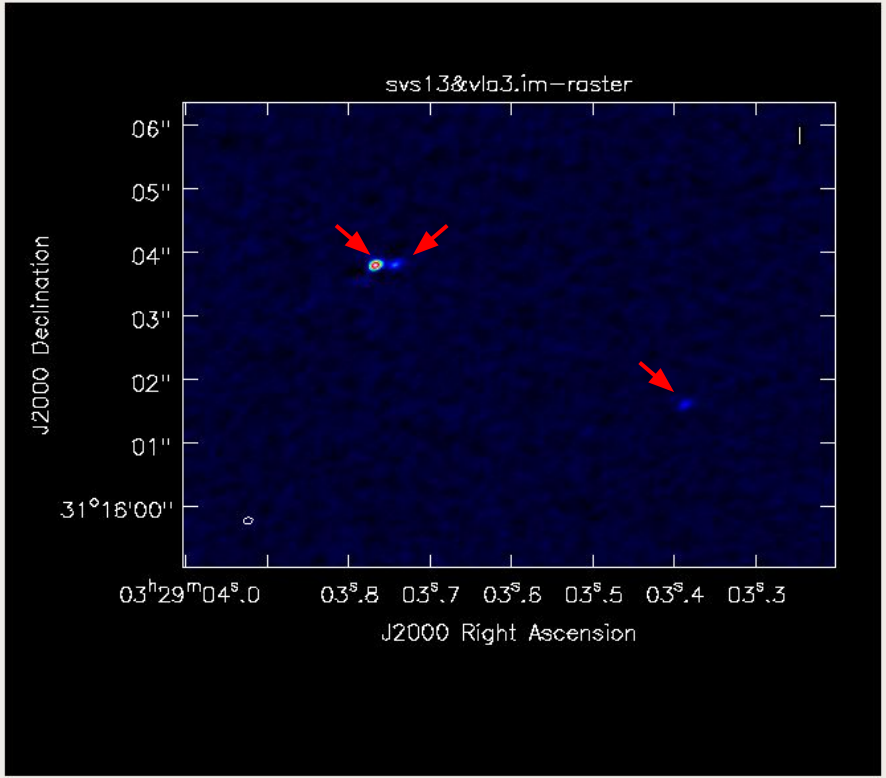
loaded: /home/anika/Documents/Trabajo/PresPyDay/svs13&vla3.im

Viewer Display Panel (kB)

mar 11 dic 2018 20:32

Data Display Panel Tools View Help

Display



Animators

Stokes

Images

Rate: 10 Jump 0 0

0 0

Regions

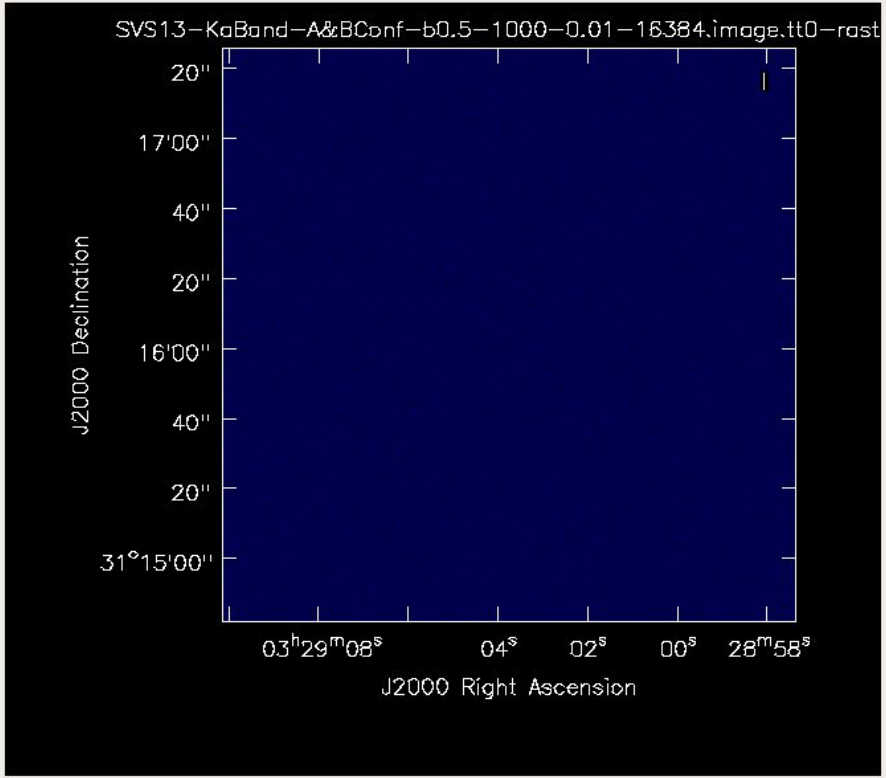
loaded: /home/anika/Documents/Trabajo/PresPyDay/svs13&vla3.im

Viewer Display Panel (Tp)

mar 11 dic 2018 21:08

Data Display Panel Tools View Help

Display



Animators

Stokes

Images

Regions

?

loaded: /home/anika/Documents/Trabajo/DATA/SVS13/Imaging/A&B/Ka/SVS13-KaBand-A&BConf-b0.5-1000-0.01-16384.image.tt0

Tareas

En cada mapa:

- **Identificar las fuentes**

Para cada fuente:

- **Hacer una sub-imagen**
- **Determinar su posición y flujo**

Tareas

En cada mapa:

- **Identificar las fuentes**

Para cada fuente:

- **Hacer una sub-imagen**
- **Determinar su posición y flujo**

49 fuentes en 5 mapas

Script

En cada mapa:

- **Identificar las fuentes**

Para cada fuente:

- **Hacer una sub-imagen**
- **Determinar su posición y flujo**

Script

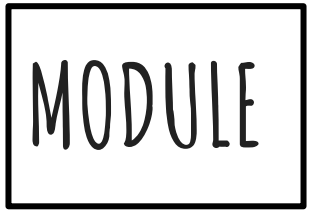
- Escrito en Python
- Usa tareas de CASA y de Python

En cada mapa:

- **Identificar las fuentes**

Para cada fuente:

- **Hacer una sub-imagen**
- **Determinar su posición y flujo**




```
from taskinit import *

def my_function1(arg1, arg2, arg=A):
    # magic
    return value
.
.
.
def my_functionN(arg1, arg2, arg=A):
    # magic
    return value
```


```
import my_module

x=my_module.my_function1(arg1, arg2)
.
.
.
z=my_module.my_functionN(arg1, arg2)
```

```
def my_function(arg1, arg2, arg=A):
    # magic
    return value
```

1. Dada una imagen, encuentre las fuentes en ella y sus coordenadas.
2. Dada una lista de posiciones, las escriba en una región de CASA.
3. Dada una lista de posiciones, haga sub-imágenes centradas en cada una de ellas.
4. Dada una imagen, haga un ajuste gaussiano de las fuentes en ella.
5. Extraiga la información relevante de la salida del ajuste gaussiano y lo escriba en un archivo.
6. Convierta coordenadas en radianes a sexagesimal.

```
def my_function(arg1, arg2, arg=A):
    # magic
    return value
```

1. Dada una imagen, encuentre las fuentes en ella y sus coordenadas.
2. Dada una lista de posiciones, las escriba en una región de CASA.
3. Dada una lista de posiciones, haga sub-imágenes centradas en cada una de ellas.
4. Dada una imagen, haga un ajuste gaussiano de las fuentes en ella.
5. Extraiga la información relevante de la salida del ajuste gaussiano y lo escriba en un archivo.
6. Convierta coordenadas en radianes a sexagesimal.

```
def my_function(arg1, arg2, arg=A):
    # magic
    return value
```

1. **Dada una imagen, encuentre las fuentes en ella y sus coordenadas.**
2. Dada una lista de posiciones, las escriba en una región de CASA.
3. Dada una lista de posiciones, haga sub-imágenes centradas en cada una de ellas.
4. Dada una imagen, haga un ajuste gaussiano de las fuentes en ella.
5. Extraiga la información relevante de la salida del ajuste gaussiano y lo escriba en un archivo.
6. Convierta coordenadas en radianes a sexagesimal.

```
def find_sources(image_name, image_rms, components_level=10):

    ia.open(image_name)

    stats=ia.statistics()
    flux_max=stats['max'][0]
    cutoff = components_level*(image_rms/flux_max) #fracción
    del valor máximo hasta donde buscará componentes

    potential_sources = ia.findsources(cutoff=cutoff,
    nmax=10000)

    ia.close()
```

```
ia.findsources(cutoff, nmax, ...)
```


(Image analysis tool)

```
ia.findsources(cutoff, nmax, ...)
```

`ia.findsources(`**`cutoff`**`, nmax, ...)`

- rms
- valor máximo

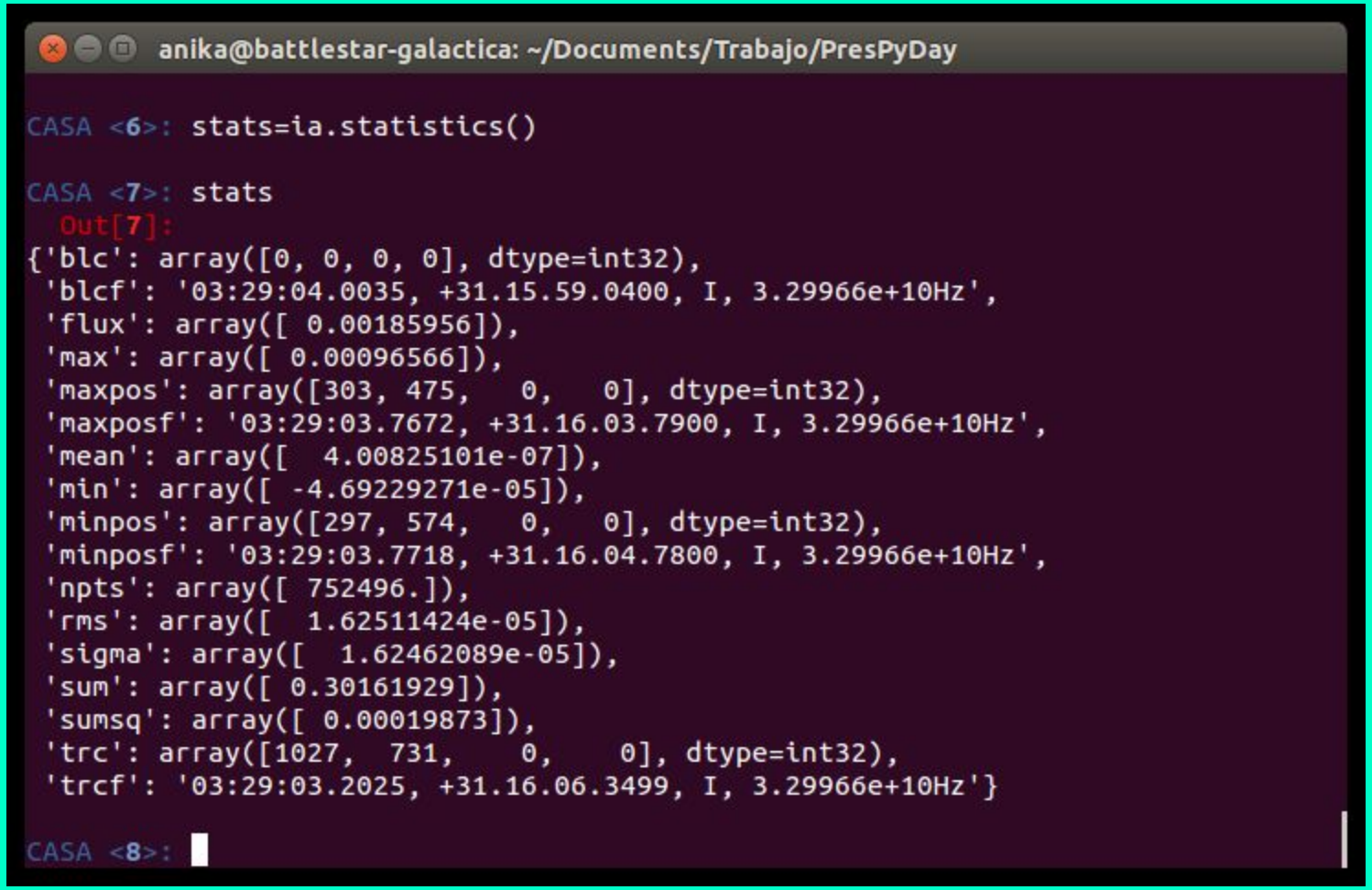
```
ia.findsources(cutoff, nmax, ...)
```

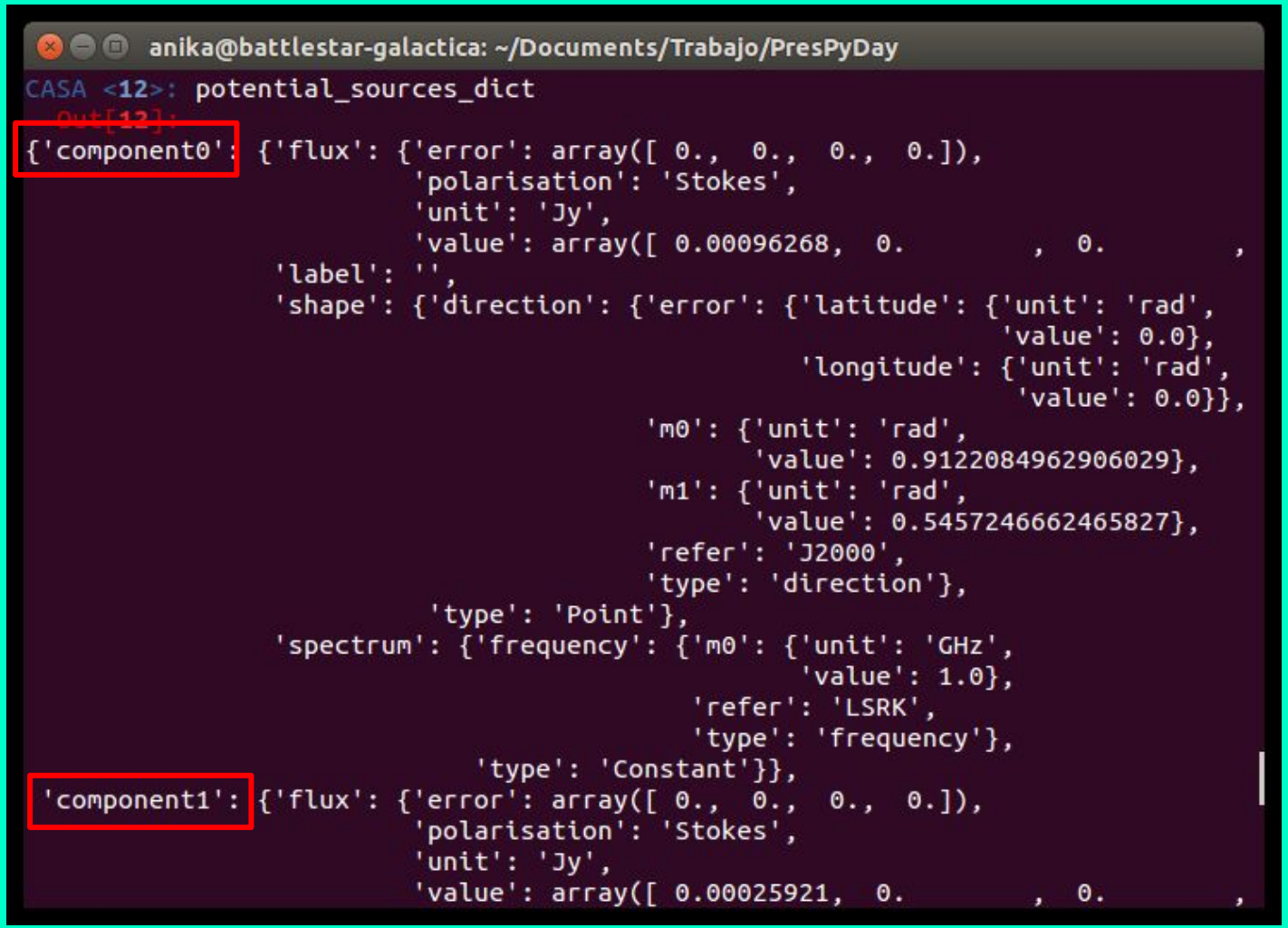
```
def find_sources(image_name, image_rms, components_level=10):

    ia.open(image_name)

    stats=ia.statistics()
    flux_max=stats['max'][0]
    cutoff = components_level*(image_rms/flux_max) #fracción
    del valor máximo hasta donde buscará componentes

    potential_sources = ia.findsources(cutoff=cutoff,
    nmax=10000)

    ia.close()
```

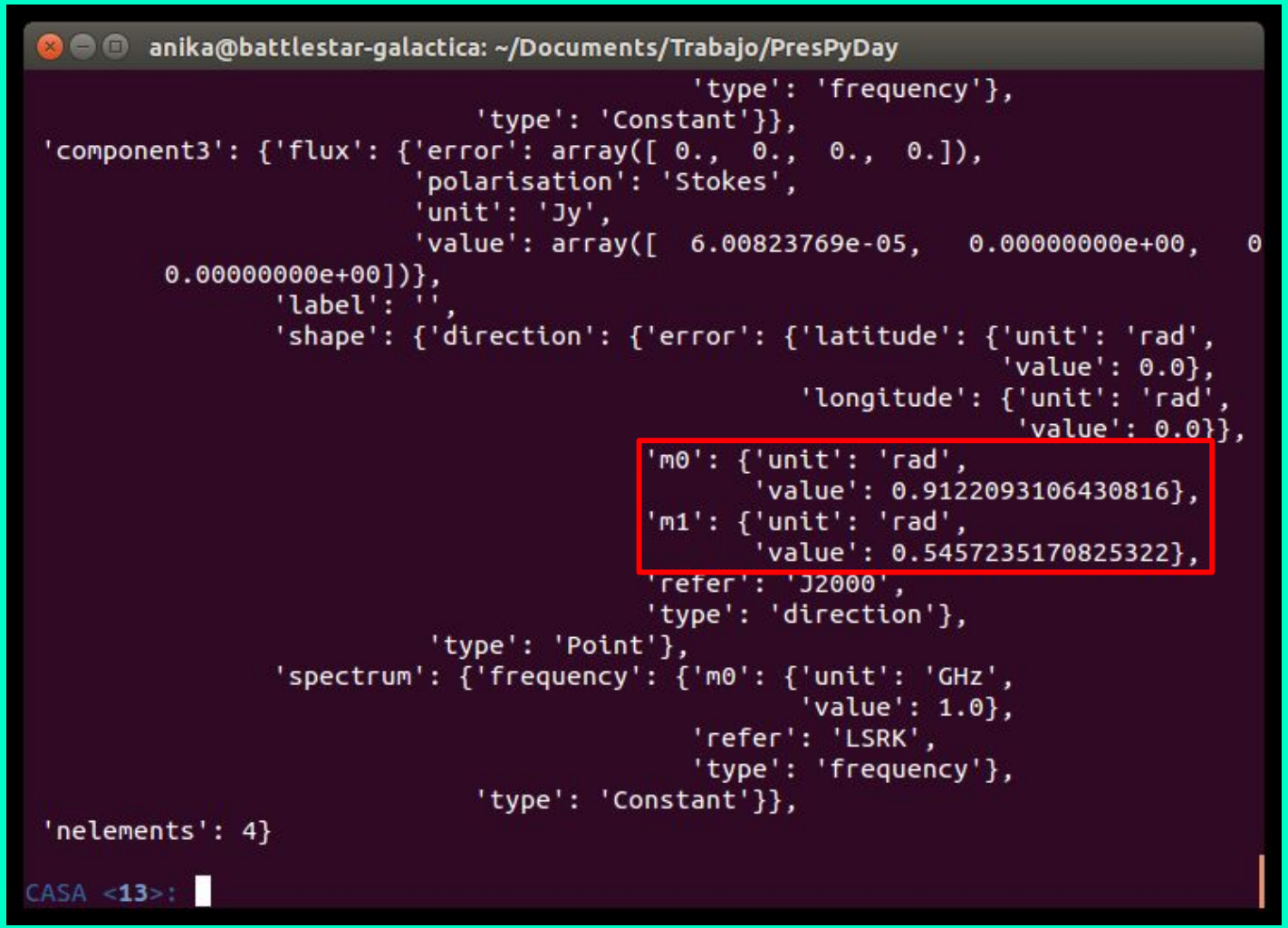
```
def find_sources(image_name, image_rms, components_level=10):

    ia.open(image_name)

    stats=ia.statistics()
    flux_max=stats['max'][0]
    cutoff = components_level*(image_rms/flux_max) #fracción
    del valor máximo hasta donde buscará componentes

    potential_sources = ia.findsources(cutoff=cutoff,
    nmax=10000)

    ia.close()
```

Diccionario

dict = {key1:val1, key2:val2...}


anika@battlestar-galactica: ~/Documents/Trabajo/PresPyDay
CASA <6>: stats=ia.statistics()
CASA <7>: stats
Out[7]:
{'blc': array([0, 0, 0, 0], dtype=int32),
'blcf': '03:29:04.0035, +31.15.59.0400, I, 3.29966e+10Hz',
'flux': array([0.00185956]),
'max': array([0.00096566]),
'maxpos': array([303, 475, 0, 0], dtype=int32),
'maxposf': '03:29:03.7672, +31.16.03.7900, I, 3.29966e+10Hz',
'mean': array([4.00825101e-07]),
'min': array([-4.69229271e-05]),
'minpos': array([297, 574, 0, 0], dtype=int32),
'minposf': '03:29:03.7718, +31.16.04.7800, I, 3.29966e+10Hz',
'npts': array([752496.]),
'rms': array([1.62511424e-05]),
'sigma': array([1.62462089e-05]),
'sum': array([0.30161929]),
'sumsq': array([0.00019873]),
'trc': array([1027, 731, 0, 0], dtype=int32),
'trcf': '03:29:03.2025, +31.16.06.3499, I, 3.29966e+10Hz'}
CASA <8>:

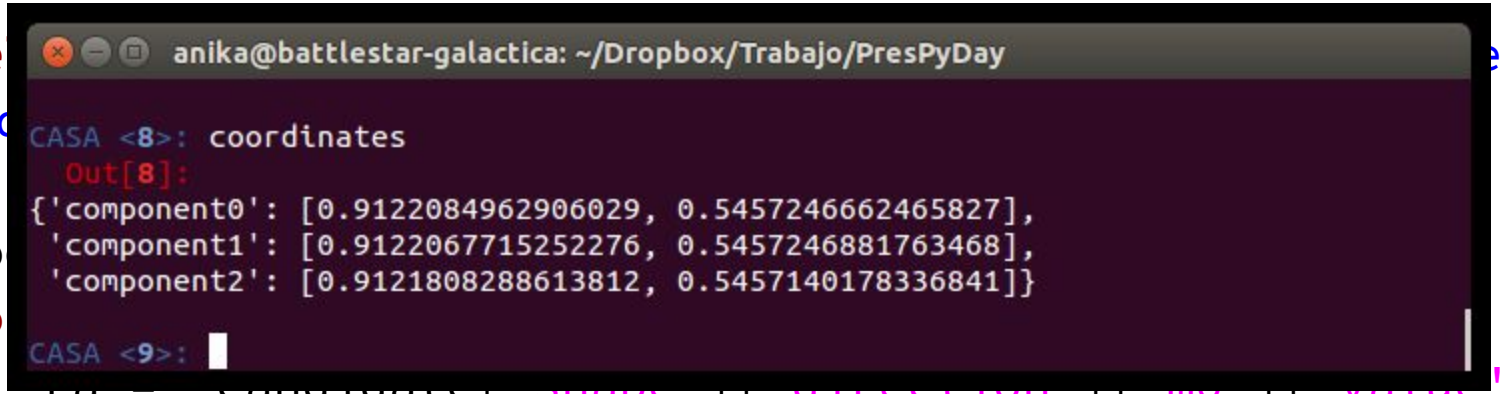

anika@battlestar-galactica: ~/Documents/Trabajo/PresPyDay
CASA <12>: potential_sources_dict
Out[12]:
{'component0': {'flux': {'error': array([0., 0., 0., 0.]),
'polarisation': 'Stokes',
'unit': 'Jy',
'value': array([0.00096268, 0.   , 0.   ,
'label': '',
'shape': {'direction': {'error': {'latitude': {'unit': 'rad',
'value': 0.0},
'longitude': {'unit': 'rad',
'value': 0.0}},
'm0': {'unit': 'rad',
'value': 0.9122084962906029},
'm1': {'unit': 'rad',
'value': 0.5457246662465827},
'refer': 'J2000',
'type': 'direction'},
'type': 'Point'},
'spectrum': {'frequency': {'m0': {'unit': 'GHz',
'value': 1.0},
'refer': 'LSRK',
'type': 'frequency'},
'type': 'Constant'}},
'component1': {'flux': {'error': array([0., 0., 0., 0.]),
'polarisation': 'Stokes',
'unit': 'Jy',
'value': array([0.00025921, 0.   , 0.   ,


anika@battlestar-galactica: ~/Documents/Trabajo/PresPyDay
'type': 'frequency'},
'type': 'Constant'}},
'component3': {'flux': {'error': array([0., 0., 0., 0.]),
'polarisation': 'Stokes',
'unit': 'Jy',
'value': array([6.00823769e-05,   0.00000000e+00,   0
0.00000000e+00])},
'label': '',
'shape': {'direction': {'error': {'latitude': {'unit': 'rad',
'value': 0.0},
'longitude': {'unit': 'rad',
'value': 0.0}},
'm0': {'unit': 'rad',
'value': 0.9122093106430816},
'm1': {'unit': 'rad',
'value': 0.5457235170825322},
'refer': 'J2000',
'type': 'direction'},
'type': 'Point'},
'spectrum': {'frequency': {'m0': {'unit': 'GHz',
'value': 1.0},
'refer': 'LSRK',
'type': 'frequency'},
'type': 'Constant'}},
'nelements': 4}
CASA <13>:

```
def find_sources(image_name, image_rms, components_level=10):
    .
    .
    .
    del potential_sources['nelements'] #elimino este elemento
para quedarme solo con las componentes que ha encontrado

    coordinates={}
    for name, candidate in potential_sources.items():
        ra =  candidate['shape']['direction']['m0']['value']
        dec = candidate['shape']['direction']['m1']['value']
        coordinates[name]=[ra,dec]
    return coordinates
```

```
def find_sources(image_name, image_rms, components_level=10):
    .
    .
    .
    del potential_sources['nelements'] #elimino este elemento
para quedarme solo con las componentes que ha encontrado

    coordinates={}
    for name, candidate in potential_sources.items():
        ra =  candidate['shape']['direction']['m0']['value']
        dec = candidate['shape']['direction']['m1']['value']
        coordinates[name]=[ra,dec]
    return coordinates
```

```
def find_sources(image_name, image_rms, components_level=10):
    .
    .
    .
    de                                                    ento
para c

    co
    fo
        ra = candidate['shape']['direction']['m0']['value']
        dec = candidate['shape']['direction']['m1']['value']
        coordinates[name]=[ra,dec]
    return coordinates
```



```
def my_function(arg1, arg2, arg=A):
    # magic
    return value
```

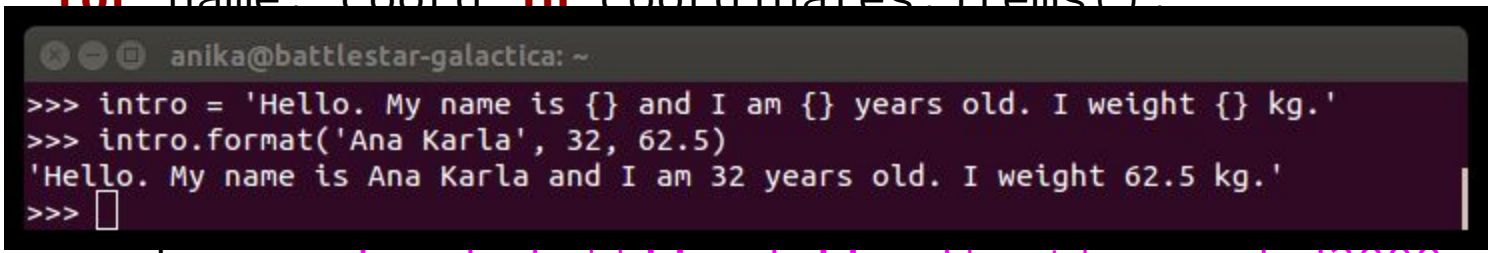
1. Dada una imagen, encuentre las fuentes en ella y sus coordenadas. ✅
2. **Dada una lista de posiciones, las escriba en una región de CASA.**
3. Dada una lista de posiciones, haga sub-imágenes centradas en cada una de ellas.
4. Dada una imagen, haga un ajuste gaussiano de las fuentes en ella.
5. Extraiga la información relevante de la salida del ajuste gaussiano y lo escriba en un archivo.
6. Convierta coordenadas en radianes a sexagesimal.

```
def write_sources(coordinates, out_file):

    with open(out_file+'.crtf','w') as f:
        f.write('#CRTFv0 CASA Region Text Format version 0\n')

        for name, coord in coordinates.items():
            ra =  coord[0]
            dec = coord[1]
            label =  name.replace('component','')
            line = 'symbol [[{}rad,{}rad], +] coord=J2000,
            linewidth=3, linestyle=-, symsize=4, symthick=1,
            color=red, font="DejaVu Sans", fontsize=11,
            fontstyle=normal, usetex=false, label="{}",
            labelcolor=red, labelpos=top\n'
            f.write(line.format(ra,dec,label))
```

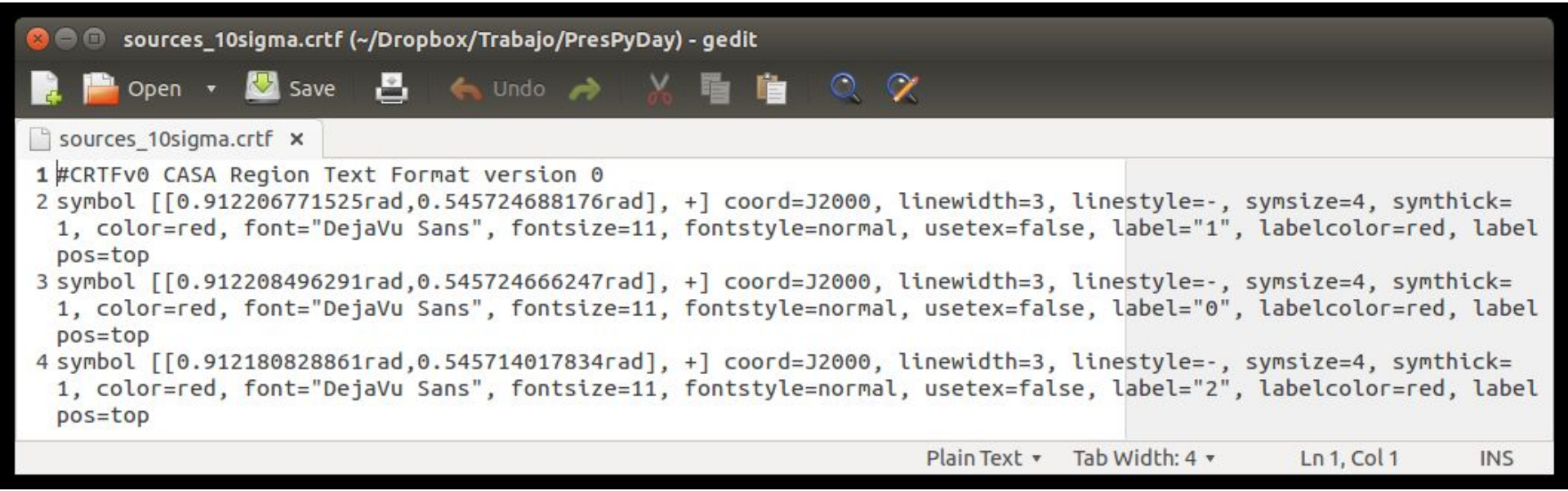
```
def write_sources(coordinates, out_file):

    with open(out_file+'.crtf','w') as f:
        f.write('#CRTFv0 CASA Region Text Format version 0\n')

        for name, coord in coordinates.items():
            ra =  coord[0]
            dec = coord[1]
            label =  name.replace('component','')
            line = 'symbol [[{}rad,{}rad], +] coord=J2000,
            linewidth=3, linestyle=-, symsize=4, symthick=1,
            color=red, font="DejaVu Sans", fontsize=11,
            fontstyle=normal, usetex=false, label="{}",
            labelcolor=red, labelpos=top\n'
            f.write(line.format(ra,dec,label))
```

```
def write_sources(coordinates, out_file):

    with open(out_file+'.crtf','w') as f:
        f.write('#CRTFv0 CASA Region Text Format version 0\n')

        for name, coord in coordinates.items():
            ra =  coord[0]
            dec = coord[1]
            label =  name.replace('component','')
            line = 'symbol [[{}rad,{}rad], +] coord=J2000,
            linewidth=3, linestyle=-, symsize=4, symthick=1,
            color=red, font="DejaVu Sans", fontsize=11,
            fontstyle=normal, usetex=false, label="{}",
            labelcolor=red, labelpos=top\n'
            f.write(line.format(ra,dec,label))
```

```
def write_sources(coordinates, out_file):

    with open(out_file+'.crtf','w') as f:
        f.write('#CRTFv0 CASA Region Text Format version 0\n')

        for name, coord in coordinates.items():
            ra = coord.ra.to_string(unit=u.rad, decimal=True)
            dec = coord.dec.to_string(unit=u.rad, decimal=True)
            label = name
            line = 'symbol [[{}rad,{}rad], +] coord=J2000,
            linewidth=3, linestyle=-, symsize=4, symthick=1,
            color=red, font="DejaVu Sans", fontsize=11,
            fontstyle=normal, usetex=false, label="{}",
            labelcolor=red, labelpos=top\n'
            f.write(line.format(ra,dec,label))
```



```
def write_sources(coordinates, out_file):

    with open(out_file+'.crtf','w') as f:
```

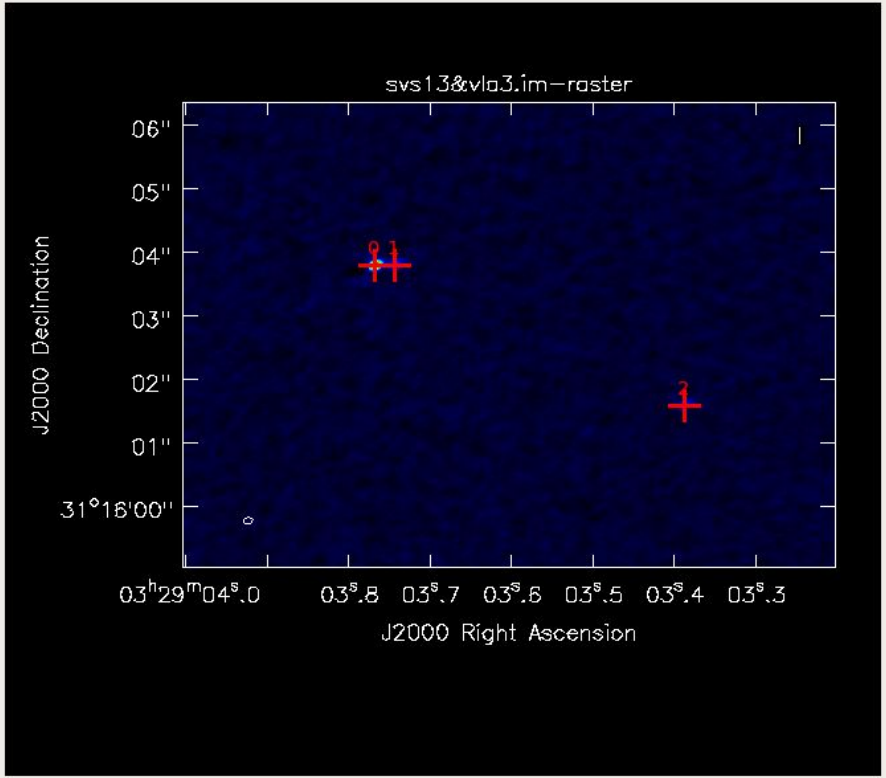



```
            fontstyle=normal, usetex=false, label="{}",
            labelcolor=red, labelpos=top\n'
            f.write(line.format(ra,dec,label))
```

Viewer Display Panel (Pg)

Data Display Panel Tools View Help

Display



Animators

Stokes

Images

Regions

Properties Statistics File Histogram

point frames 0 1 selected annotation

coordinates line text

system

J2000

bounding box (width X height)

0 0 "

center x units

03:29:03.387 sexagesimal

center y units

31.16.01.596 sexagesimal

loaded: /home/anika/Dropbox/Trabajo/PresPyDay/svs13&vla3.im

```
def my_function(arg1, arg2, arg=A):
    # magic
    return value
```

1. Dada una imagen, encuentre las fuentes en ella y sus coordenadas.✅
2. Dada una lista de posiciones, las escriba en una región de CASA. ✅
3. **Dada una lista de posiciones, haga sub-imágenes centradas en cada una de ellas.**
4. Dada una imagen, haga un ajuste gaussiano de las fuentes en ella.
5. Extraiga la información relevante de la salida del ajuste gaussiano y lo escriba en un archivo.
6. Convierta coordenadas en radianes a sexagesimal.

```
def make_subimages(image_name, coordinates, sub_image_size=50):
    ia.open(image_name)
    for name, coord in coordinates.items():
        ra = coord[0]
        dec = coord[1]
        ctr_pix = ia.topixel([ra,dec])['numeric'][0:2]
        bottom = [ctr_pix[0] - sub_image_size/2, ctr_pix[1] -
        sub_image_size/2]
        top = [ctr_pix[0] + sub_image_size/2, ctr_pix[1] +
        sub_image_size/2]
        box = rg.box(blc = bottom , trc = top)
        ia.subimage(outfile=name+'.im', region=box,
        overwrite=True)
    ia.close()
```

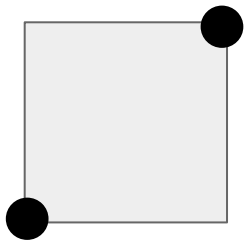
```
ia.subimage(region, ...)
```

`ia.subimage(`**`region`**`, ...)`

- = rg.box(blc, trc) (en pixeles)

(Region manipulation tool)

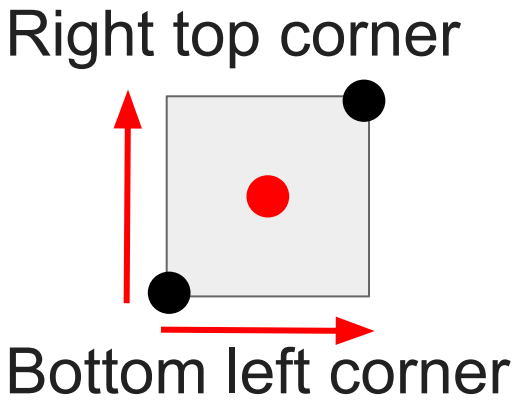
Right top corner



Bottom left corner

`ia.subimage(`**`region`**`, ...)`

- pixel central
- lado del cuadrado

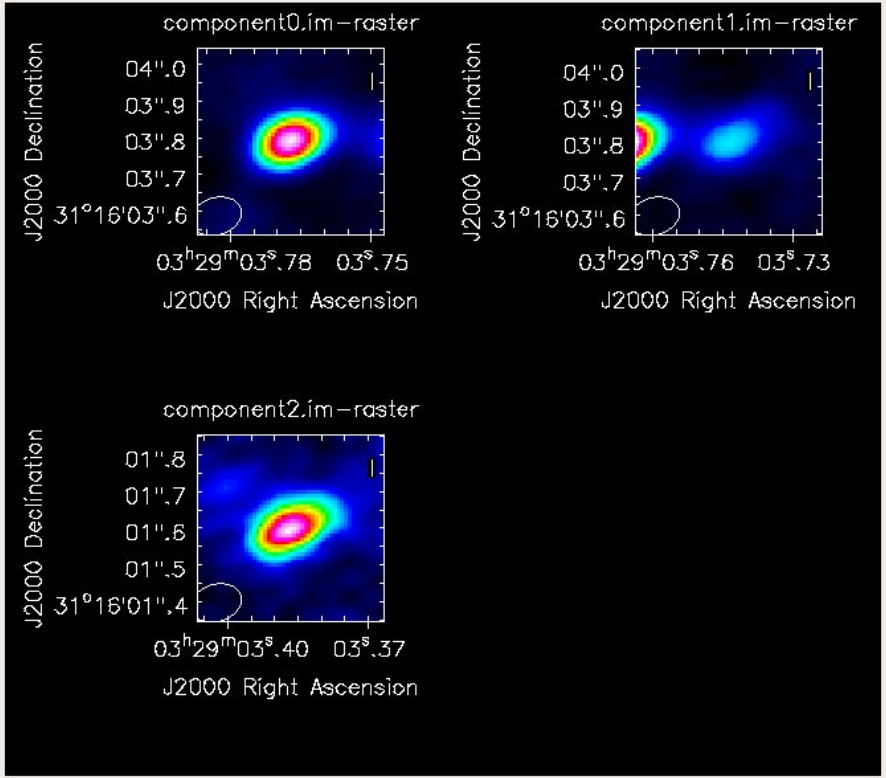


```
def make_subimages(image_name, coordinates, sub_image_size=50):
    ia.open(image_name)
    for name, coord in coordinates.items():
        ra = coord[0]
        dec = coord[1]
        ctr_pix = ia.topixel([ra,dec])['numeric'][0:2]
        bottom = [ctr_pix[0] - sub_image_size/2, ctr_pix[1] -
        sub_image_size/2]
        top = [ctr_pix[0] + sub_image_size/2, ctr_pix[1] +
        sub_image_size/2]
        box = rg.box(blc = bottom , trc = top)
        ia.subimage(outfile=name+'.im', region=box,
        overwrite=True)
    ia.close()
```

Viewer Display Panel (Pg)

Data Display Panel Tools View Help

Display



Animators

Stokes

Images

Rate: 10 Jump 0 3

0 2

Regions

loaded: /home/anika/Dropbox/Trabajo/PresPyDay/component2.im

```
def my_function(arg1, arg2, arg=A):
    # magic
    return value
```

1. Dada una imagen, encuentre las fuentes en ella y sus coordenadas. ✅
2. Dada una lista de posiciones, las escriba en una región de CASA. ✅
3. Dada una lista de posiciones, haga sub-imágenes centradas en cada una de ellas. ✅
4. **Dada una imagen, haga un ajuste gaussiano de las fuentes en ella.**
5. Extraiga la información relevante de la salida del ajuste gaussiano y lo escriba en un archivo.
6. Convierta coordenadas en radianes a sexagesimal.

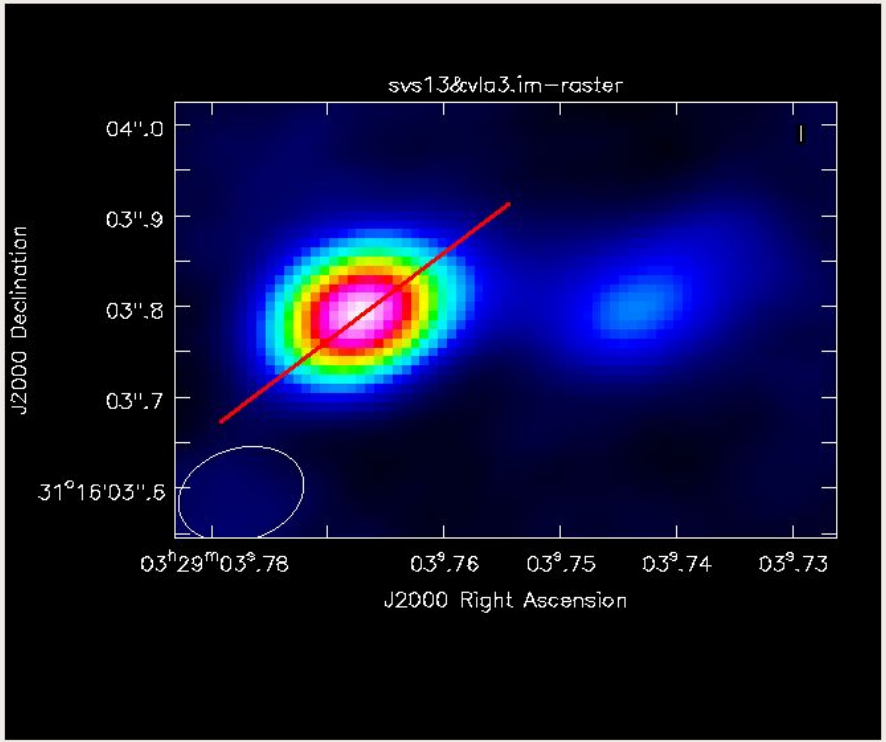
Viewer Display Panel (GF)

jue 14 mar 2019 22:38

Viewer Display Panel (GF)

Data Display Panel Tools View Help

Display



Animators

Stokes

Images

Cursors

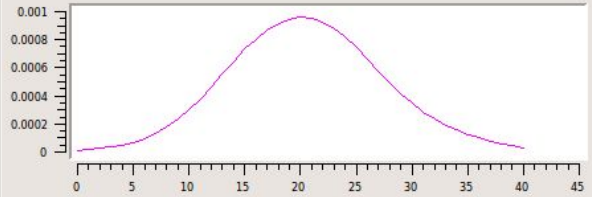
svs13vla3.im-raster

+2.7913e-05 Jy/beam Pixel: 354 482 0 0
03:29:03.728 +31.16.03.856 I 35.5191 km/s (topo/radio velocity)

Regions

Properties | Spatial Profile | File

svs13vla3.im



Spatial Profile Tool... Next

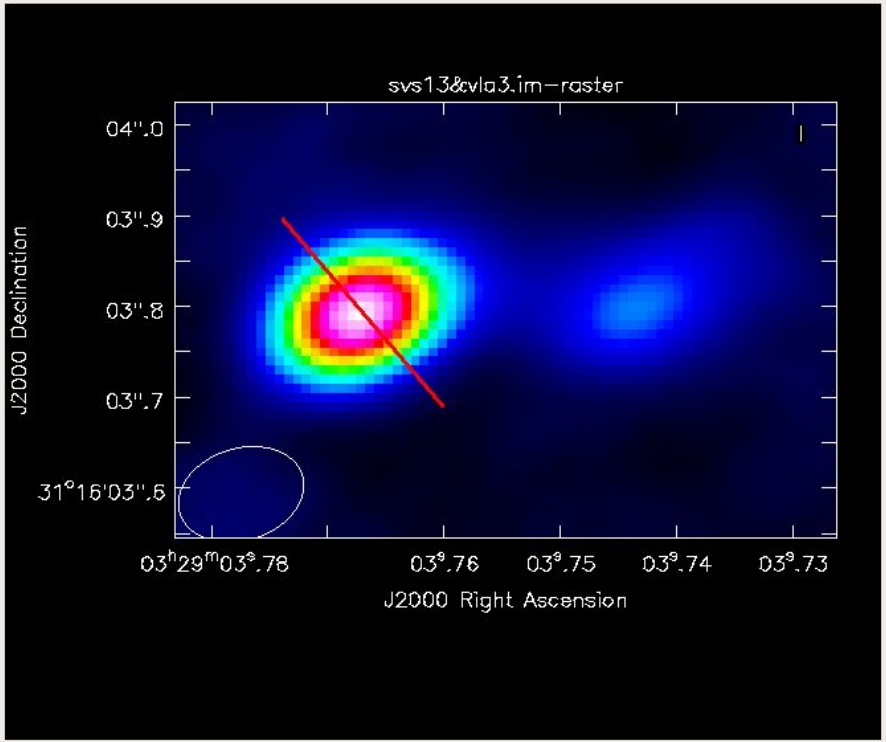
Viewer Display Panel (GF)

jue 14 mar 2019 22:37

Viewer Display Panel (GF)

Data Display Panel Tools View Help

Display



Animators

Stokes

Images

Cursors

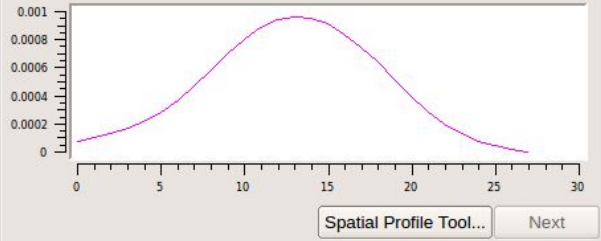
svs13vla3.im-raster

+2.80698e-05 Jy/beam Pixel: 353 469 0 0
03:29:03.728 +31.16.03.735 I 35.5191 km/s (topo/radio velocity)

Regions

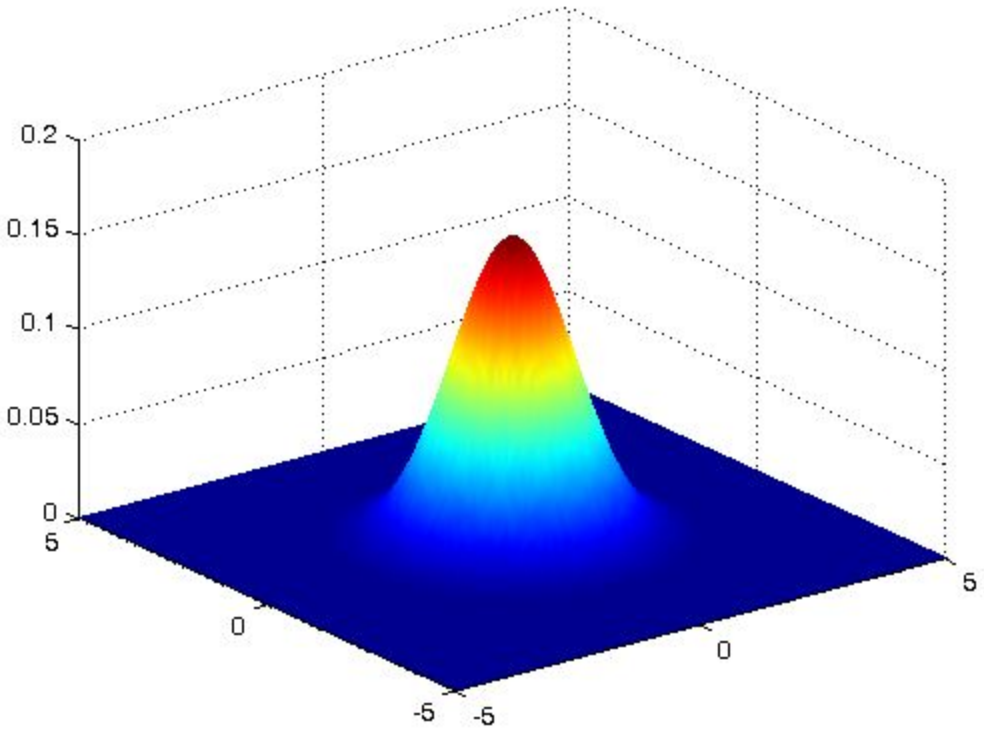
Properties | Spatial Profile | File

svs13vla3.im



Spatial Profile Tool... Next

2D-GAUSSIAN

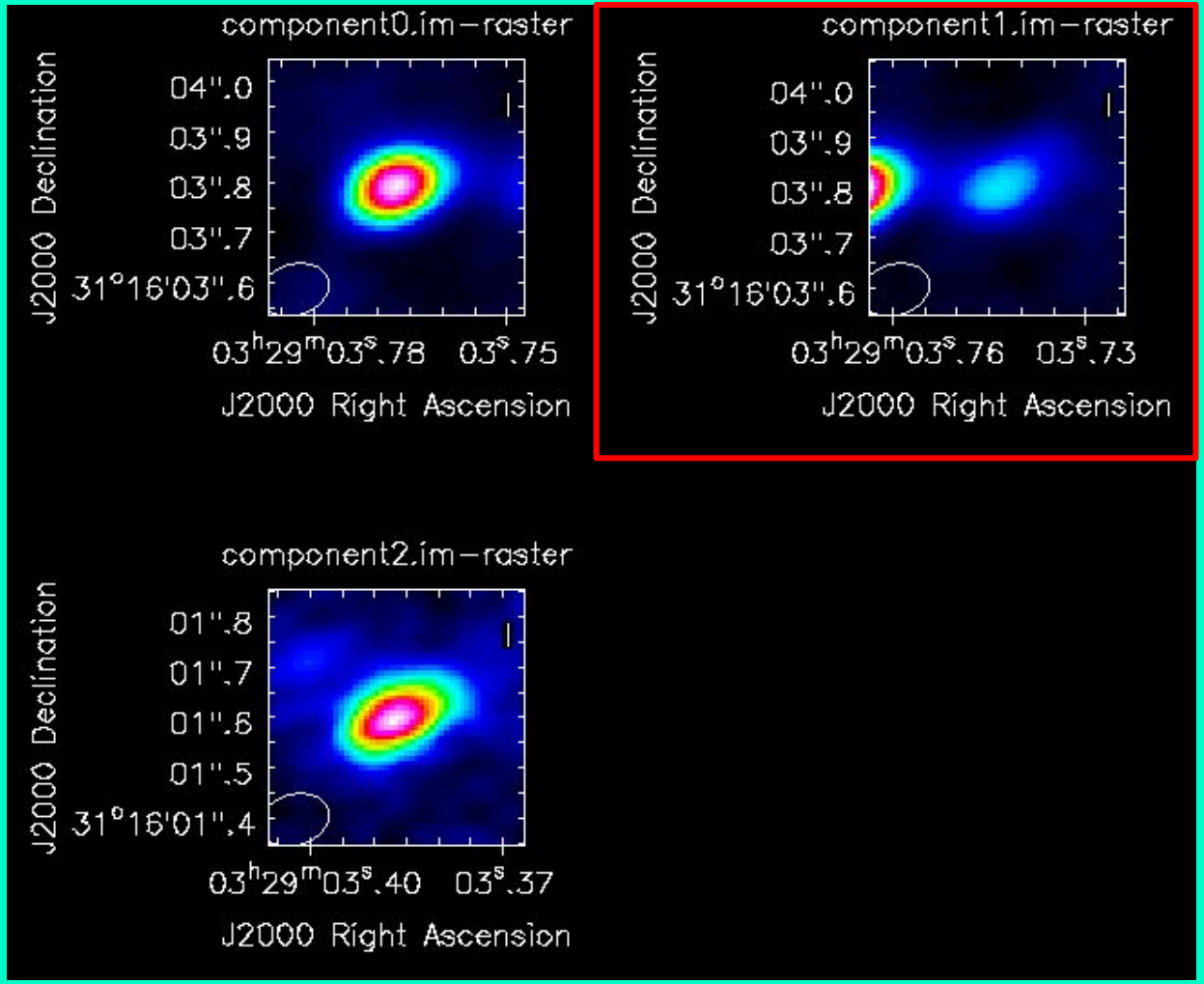


http://campar.in.tum.de/Chair/HaukeHeibelGaussianDerivatives

```
#es necesario haber creado regiones para ajustar, se pueden
usar estimados iniciales
def gaussian_fits(image_name, region, estimate=None):
    ia.open(image_name)
    output_name=image_name.split('.')[0]
    if estimate is None:
        fit_comp=ia.fitcomponents(model=output_name+'.model',r
        egion=region,residual=output_name+'.res',logfile=outpu
        t_name+'.log')
    else:
        fit_comp=ia.fitcomponents(model=output_name+'.model',r
        egion=region,residual=output_name+'.res',logfile=outpu
        t_name+'.log',estimates=estimate)
    ia.close()
    return fit_comp
```

```
ia.fitcomponents(region, estimates,
   model, residual, logfile, ...)
```

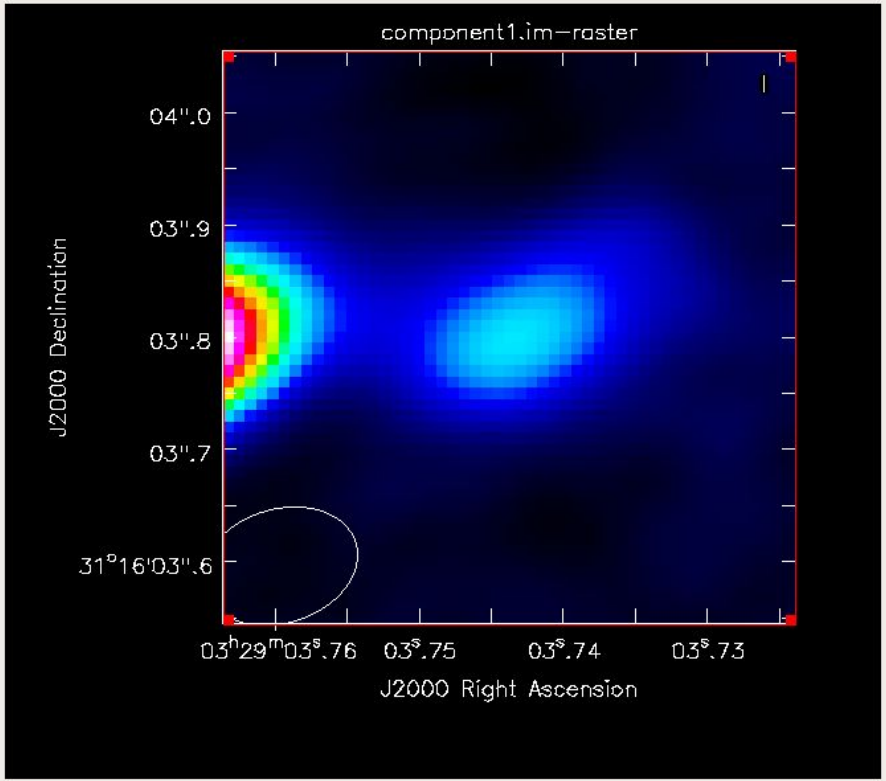
```
ia.fitcomponents(region, estimates,
   model, residual, logfile, ...)
```


component0.im−raster
J2000 Declination
04".0
03".9
03".8
03".7
31°16'03".6
03h29m03s.78
03s.75
J2000 Right Ascension
component1.im−raster
J2000 Declination
04".0
03".9
03".8
03".7
31°16'03".6
03h29m03s.76
03s.73
J2000 Right Ascension
component2.im−raster
J2000 Declination
01".8
01".7
01".6
01".5
31°16'01".4
03h29m03s.40
03s.37
J2000 Right Ascension

Viewer Display Panel (Pg)

Data Display Panel Tools View Help

Display



Animators

Stokes

Images

Rate: 10 Jump 0 1

0 0

Regions

Properties Statistics Fit File Histogram

component1.im
Converged
NO

next

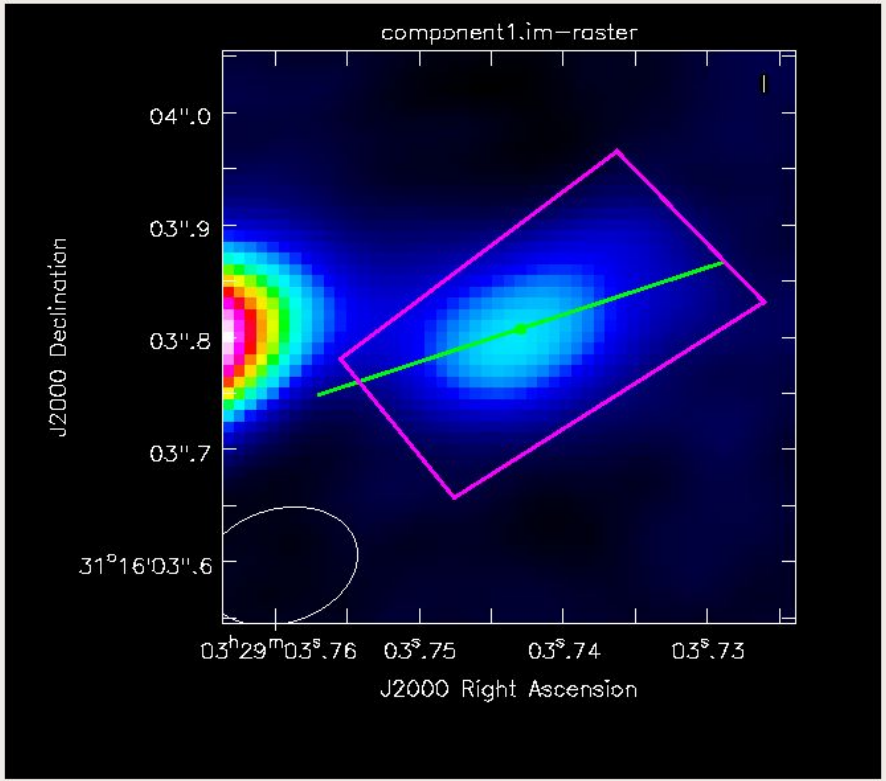
gaussfit sky component mark center green

Viewer Display Panel (Pg)

(1:36, 32%) dom 16 dic 2018 17:13

Data Display Panel Tools View Help

Display



Animators

Stokes

Images

Rate: 10 Jump 0 1

0 0

Regions

Properties Statistics Fit File Histogram

component1.im

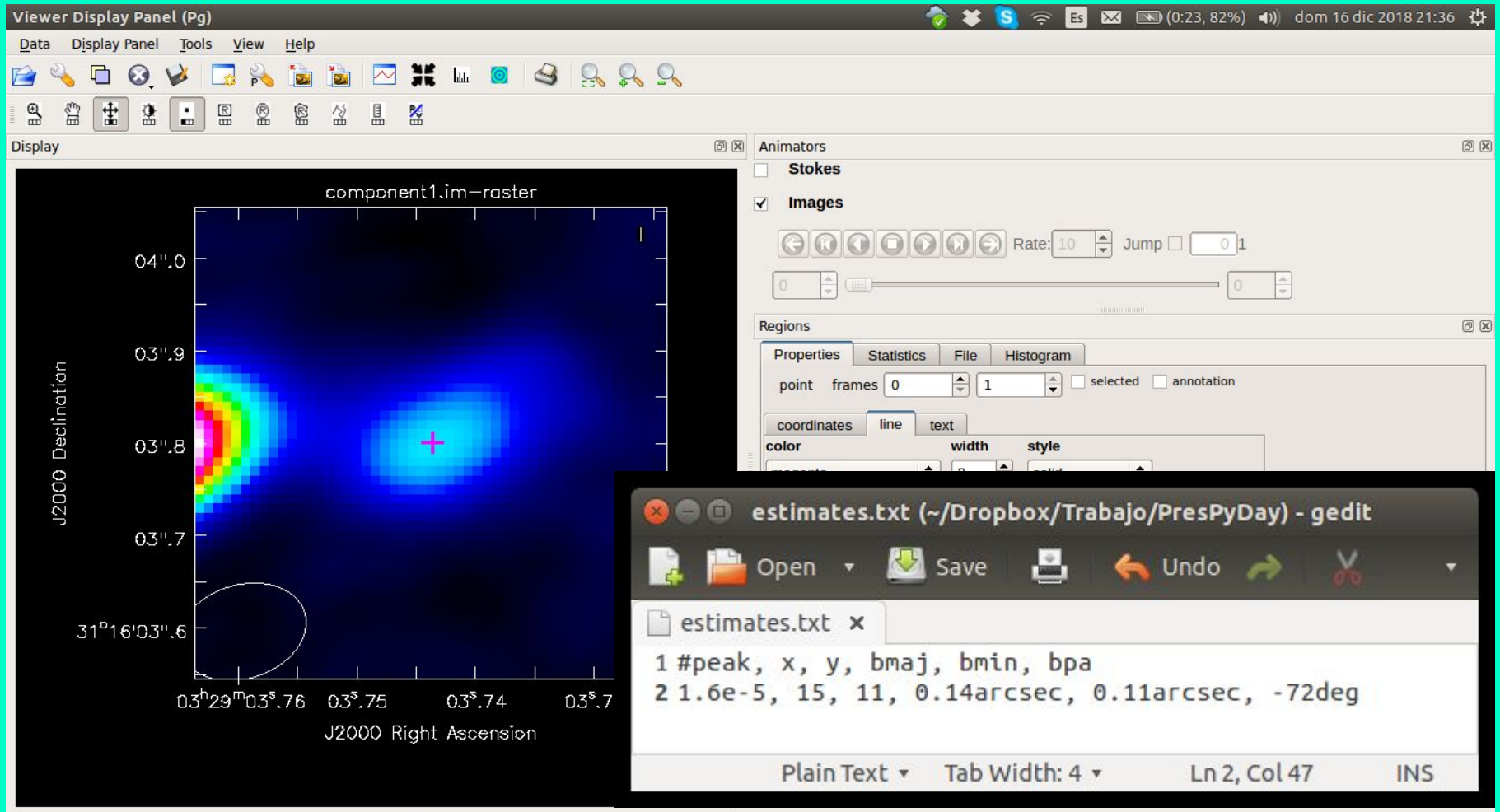
Ra_J2000	Dec_J2000	W-Majorax	W-Minorax
03:29:03.743	31.16.03.81	0.236429 arcsec	0.126545 arcsec
W-Posang	**Xcen**	**Ycen**	**I-Majorax**
108.218 deg	26.1161 pix	25.8176 pix	23.6429 pix
I-Minorax	**I-Posang**	**Radeg**	**Decdeg**
12.6545 pix	18.2178 deg	412.266 deg	31.2677 deg
IntegrFlux	**PeakFlux**	**Converged**	
0.000537481 Jy	0.000257158 Jy/bea	YES	

next

gaussfit sky component mark center green

loaded: /home/anika/Dropbox/Trabajo/PresPyDay/component2.im

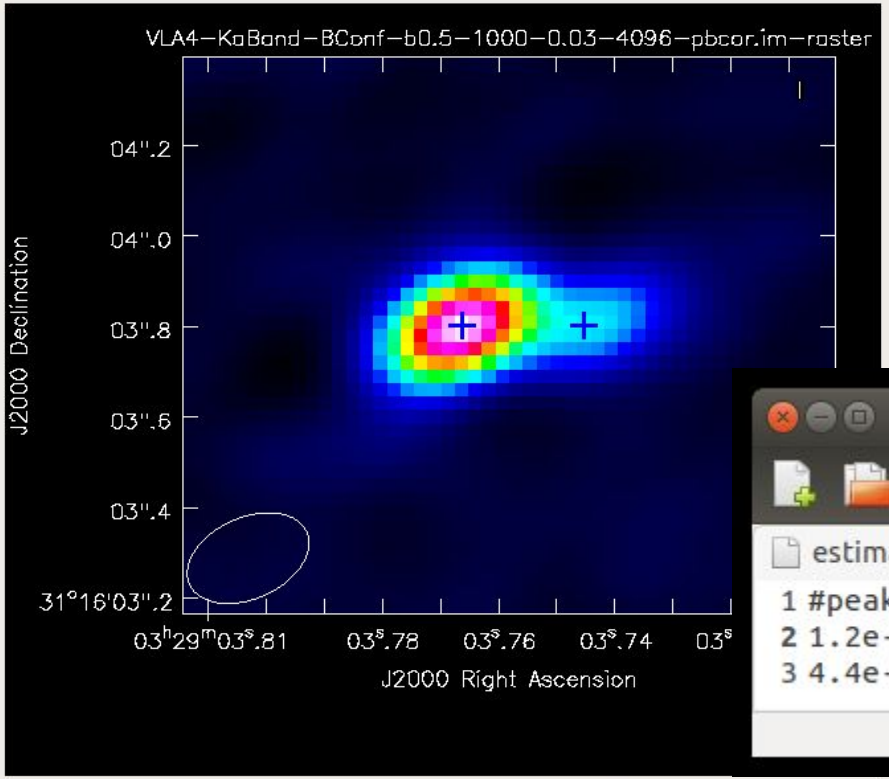
```
ia.fitcomponents(region, estimates,
   model, residual, logfile, ...)
```


Viewer Display Panel (Pg)
Data
Display Panel
Tools
View
Help
Display
component1.im−raster
04".0
03".9
03".8
03".7
31°16'03".6
J2000 Declination
03h29m03s.76
03s.75
03s.74
J2000 Right Ascension
Animators
Stokes
Images
Rate:
Jump
Regions
Properties
Statistics
File
Histogram
point
frames
selected
annotation
coordinates
line
text
color
width
style
estimates.txt (~/Dropbox/Trabajo/PresPyDay) - gedit
Open
Save
Undo
estimates.txt
1 #peak, x, y, bmaj, bmin, bpa
2 1.6e-5, 15, 11, 0.14arcsec, 0.11arcsec, -72deg
Plain Text
Tab Width: 4
Ln 2, Col 47
INS
loaded: /home/anika/Dropbox/Trabajo/PresPyDay/component2.im

Viewer Display Panel (ql)

Data Display Panel Tools View Help

Display



Animators

Stokes

Images

Cursors

VLA4-KaBand-BConf-b0.5-1000-0.03-4096-pbcor.im-raster

-8.13962e-06 Jy/beam Pixel: 69 36 0 0
03:29:03.703 +31.16.03.761 I 35.5191 km/s (topo/radio velocity)

Regions

Properties Statistics File Histogram

point frames 0 1 selected annotation

coordinates line text

estimates.txt (~/Dropbox/Trabajo/PresPyDay) - gedit

Open Save Undo

estimates.txt

```
1 #peak, x, y, bmaj, bmin, bpa
2 1.2e-3, 42, 38, 0.28arcsec, 0.18arcsec, -66deg
3 4.4e-4, 51, 38, 0.28arcsec, 0.18arcsec, -66deg
```

Plain Text Tab Width: 4 Ln 2, Col 17 INS

loaded: /home/anika/Documents/Trabajo/DATA/SVS13/Imaging/B/KaBand-BConf/VLA4-KaBand-BConf-b0.5-1000-0.03-4096-pbcor.im

```
ia.fitcomponents(region, estimates,
   model, residual, logfile, ...)
```

- Imágenes de CASA

```
ia.fitcomponents(region, estimates,
   model, residual, logfile, ...)
```

- Archivo de texto

ia.fit … mates,
mod … ..)

- A …

component0.log (~/Dropbox/Trabajo/PresPyDay) - gedit

Open ▾ Save Undo

component0.log ×

```
1 ****** Fit performed at Sat Dec 15 20:09:57 2018******
2
3 Input parameters ---
4       --- imagename:           /home/anika/Documents/Trabajo/PresPyDay/component0.im
5       --- region:
6       --- channel:             0
7       --- stokes:              I
8       --- mask:
9       --- include pixel range: []
10      --- exclude pixel range: []
11 *** Details of fit for channel number 0
12 Number of pixels used in fit: 900
13 Input and residual image statistics (to be used as a rough guide only as to goodness of
   fit)
14      --- Standard deviation of input image: 0.000245708 Jy
15      --- Standard deviation of residual image: 2.58141e-05 Jy
16      --- RMS of input image: 0.000341231 Jy
17      --- RMS of residual image: 2.59866e-05 Jy
18
19 Fit on component0.im component 0
20 Position ---
21      --- ra:    03:29:03.76705 +/- 0.00017 s (0.00223 arcsec along great circle)
22      --- dec: +031.16.03.79444 +/- 0.00136 arcsec
23      --- ra:  25.14 +/- 0.22 pixels
24      --- dec: 25.44 +/- 0.14 pixels
25
26 Image component size (convolved with beam) ---
27      --- major axis FWHM:     168.7 +/- 5.5 marcsec
28      --- minor axis FWHM:     116.9 +/- 2.8 marcsec
29      --- position angle: 109.5 +/- 2.6 deg
30
```

Plain Text ▾ Tab Width: 4 ▾ Ln 1, Col 1 INS

```
#es necesario haber creado regiones para ajustar, se pueden
usar estimados iniciales
def gaussian_fits(image_name, region, estimate=None):
    ia.open(image_name)
    output_name=image_name.split('.')[0]
    if estimate is None:
        fit_comp=ia.fitcomponents(model=output_name+'.model',r
        egion=region,residual=output_name+'.res',logfile=outpu
        t_name+'.log')
    else:
        fit_comp=ia.fitcomponents(model=output_name+'.model',r
        egion=region,residual=output_name+'.res',logfile=outpu
        t_name+'.log',estimates=estimate)
    ia.close()
    return fit_comp
```

```
#es necesario haber creado regiones para ajustar, se pueden
usa
de
```

anika@battlestar-galactica: ~/Dropbox/Trabajo/PresPyDay

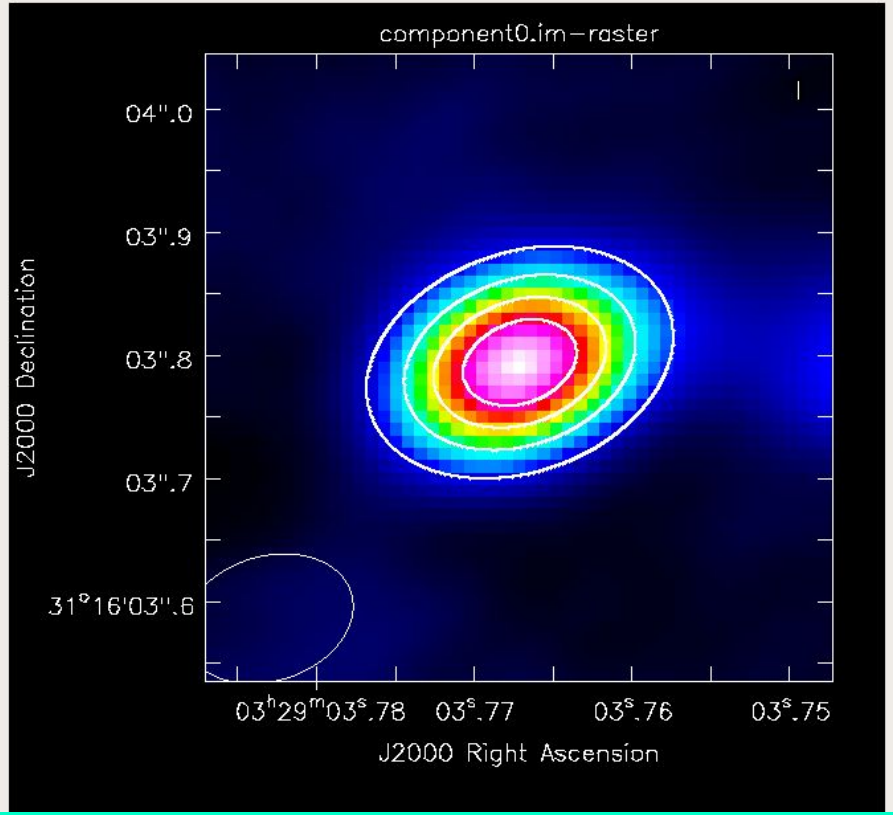
```
CASA <25>: fit
  Out[25]:
{'converged': array([ True], dtype=bool),
 'deconvolved': {'component0': {'beam': {'beamarcsec': {'major': {'unit': 'arcsec',
                                                                  'value': 0.14142018556
59485},
                                                        'minor': {'unit': 'arcsec',
                                                                  'value': 0.10122139751
911163},
                                                        'positionangle': {'unit': 'deg',
                                                                          'value': 107.3
6955261230469}},
                                        'beampixels': 162.19899248459262,
                                        'beamster': 3.812394952365767e-13},
                                'flux': {'error': array([  5.89671940e-05,   0.00000000e
+00,   0.00000000e+00,
         0.00000000e+00]),
                                        'polarisation': 'Stokes',
                                        'unit': 'Jy',
                                        'value': array([ 0.00133108,  0.        ,  0.
      ,  0.        ])},
                                'ispoint': False,
```

```
    return fit_comp
```

Viewer Display Panel (ql)

Data Display Panel Tools View Help

Display



Animators

Stokes

Images

Rate: 10 Jump 0 2

0 1

Cursors

component0.im-raster

-2.71661e-06 Jy/beam Pixel: 4 50 0 0
03:29:03.784 +31.16.04.043 I 35.5191 km/s (topo/radio velocity)

component0.model

+2.28573e-12 Jy/beam Pixel: 4 50 0 0
03:29:03.784 +31.16.04.043 I 35.5191 km/s (topo/radio velocity)
Contours: 0.000193 0.000385 0.000578 0.00077

Regions

(0:53, 95%) dom 16 dic 2018 22:22

```
def my_function(arg1, arg2, arg=A):
    # magic
    return value
```

1. Dada una imagen, encuentre las fuentes en ella y sus coordenadas. ✅
2. Dada una lista de posiciones, las escriba en una región de CASA. ✅
3. Dada una lista de posiciones, haga sub-imágenes centradas en cada una de ellas. ✅
4. Dada una imagen, haga un ajuste gaussiano de las fuentes en ella. ✅
5. Extraiga la información relevante de la salida del ajuste gaussiano y lo escriba en un archivo.
6. Convierta coordenadas en radianes a sexagesimal.

MODULE

```
from taskinit import *

def find_sources(image_name, image_rms, components_level=10):
    # magic
    return coordinates

def write_sources(coordinates, out_file):
    # magic

def make_subimages(image_name, coordinates, sub_image_size=50):
    # magic
```

MODULE

```
def gaussian_fits(image_name, region, estimate=None):
    # magic
    return fit_comp

def write_fits(names, gaussian_fits):
    # magic
    return final_coordinates

def rad2sex(ar, dec):
    # magic
    return (ar,dec)
```

SCRIPT

```
import mi_casa
import time

image_name = 'svs13&vla3.im'
image_rms = 1e-5 #Jy/beam
component_level=10
sub_image_size = 50 #pixeles

coordinates=mi_casa.find_sources(image_name, image_rms,
component_level)
mi_casa.make_subimages(image_name, coordinates, sub_image_size)
```

SCRIPT

```
import mi_casa
import time

image_name = 'svs13&vla3.im'
image_rms = 1e-5 #Jy/beam
component_level=10
sub_image_size = 50 #pixeles

coordinates=mi_casa.find_sources(image_name, image_rms,
component_level)
mi_casa.make_subimages(image_name, coordinates, sub_image_size)
```

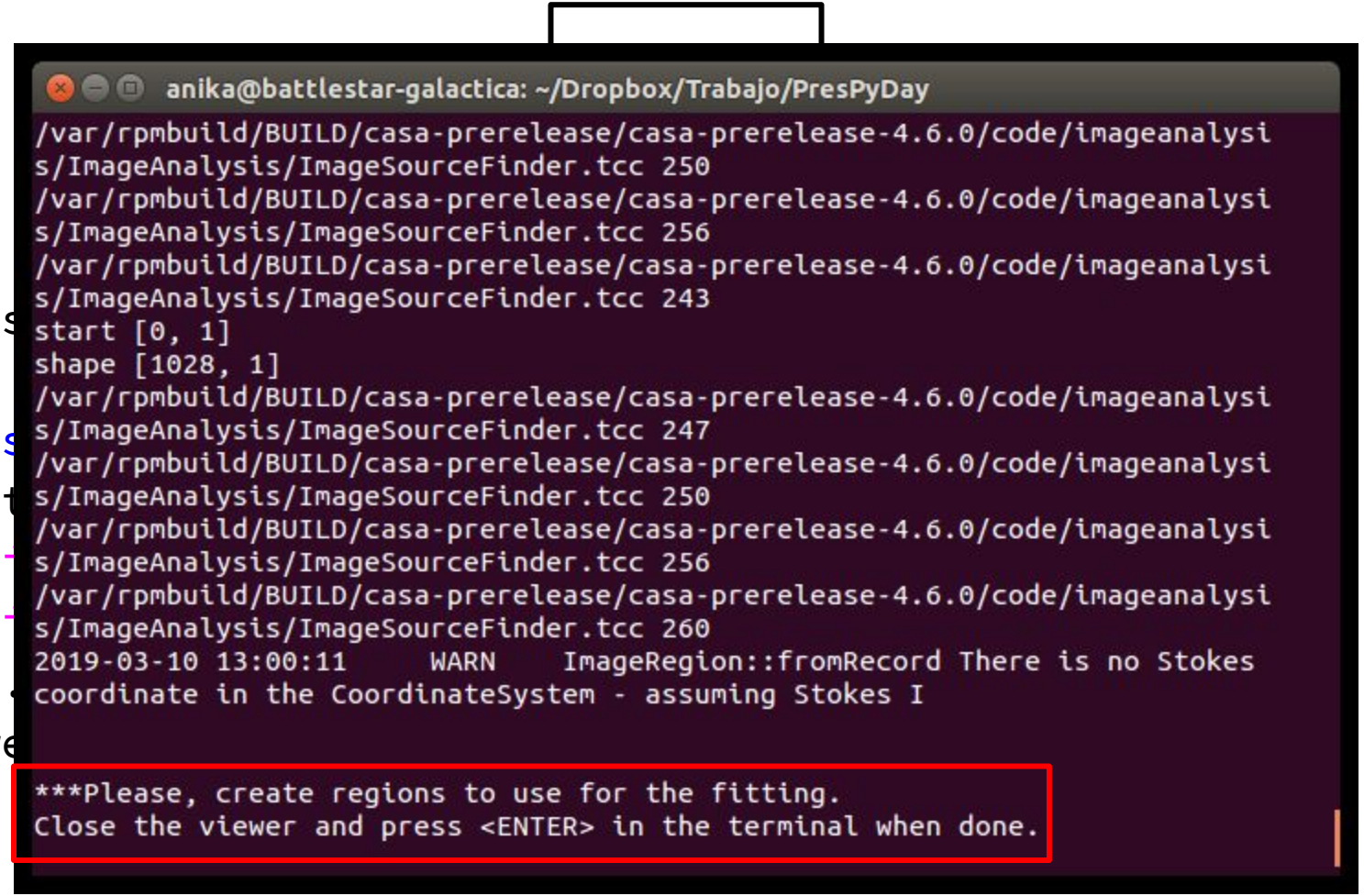
SCRIPT

```
import mi_casa
import time

image_name = 'svs13&vla3.im'
image_rms = 1e-5 #Jy/beam
component_level=10
sub_image_size = 50 #pixeles

coordinates=mi_casa.find_sources(image_name, image_rms,
component_level)
mi_casa.make_subimages(image_name, coordinates, sub_image_size)
```

SCRIPT

```
names = coordinates.keys()

#pausa para hacer regiones para el ajuste
print('\n\n***Please, create regions to use for the
fitting.\nClose the viewer and press <ENTER> in the
terminal when done.')
time.sleep(5) #pausa 5 segundos
viewer()
wait = raw_input()
```

```
names
#paus
print                                                    the
fitt                                                     the
term
time.
viewe
wait
```

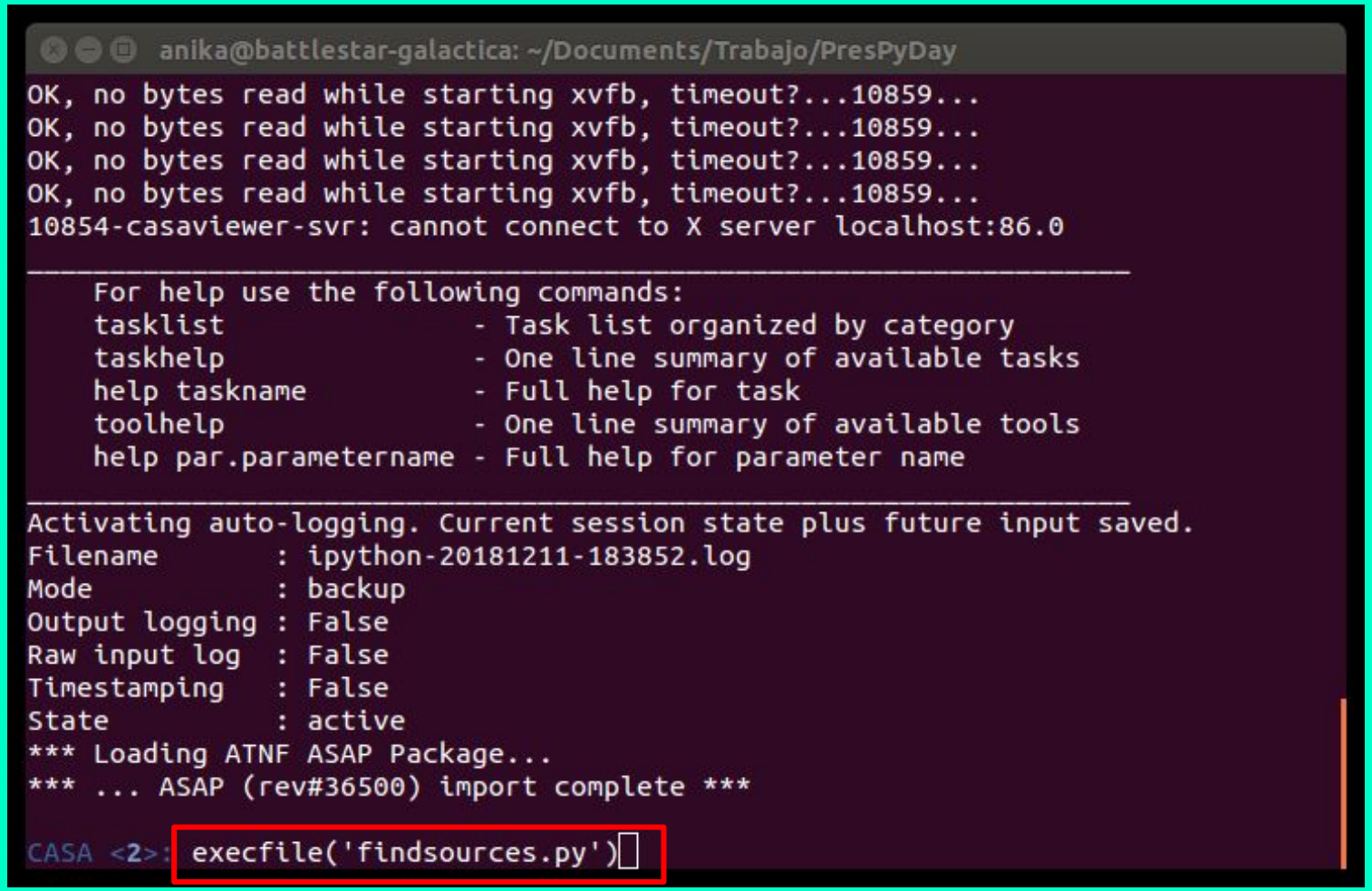


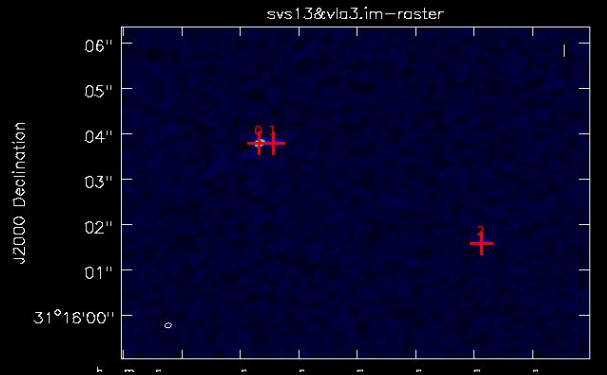
SCRIPT

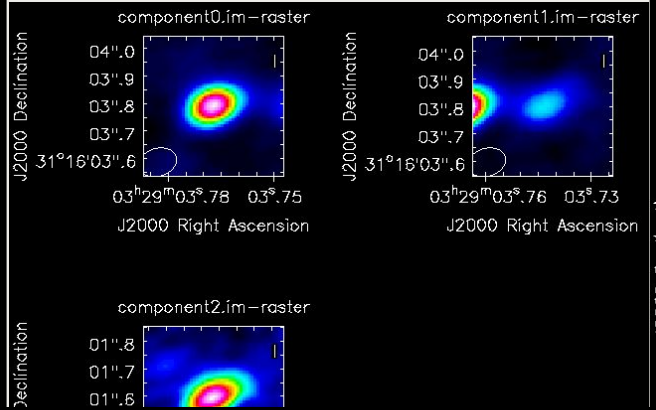
```
fits = []
for name in names:
    fit=mi_casa.gaussian_fits(name+'.im', name+'.crtf')
    fits.append(fit)

final_coordinates=mi_casa.write_fits(names,fits)
mi_casa.write_sources(final_coordinates,
'sources_{}sigma_fits'.format(component_level))
```

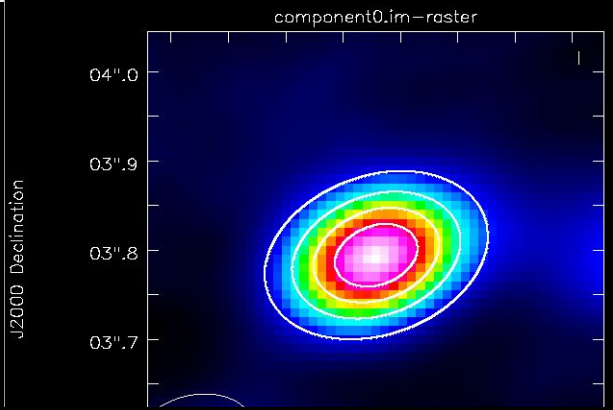
SCRIPT

```
fits = []
for name in names:
    fit=mi_casa.gaussian_fits(name+'.im', name+'.crtf')
    fits.append(fit)

final_coordinates=mi_casa.write_fits(names,fits)
mi_casa.write_sources(final_coordinates,
'sources_{}sigma_fits'.format(component_level))
```


anika@battlestar-galactica: ~/Documents/Trabajo/PresPyDay
OK, no bytes read while starting xvfb, timeout?...10859...
OK, no bytes read while starting xvfb, timeout?...10859...
OK, no bytes read while starting xvfb, timeout?...10859...
OK, no bytes read while starting xvfb, timeout?...10859...
10854-casaviewer-svr: cannot connect to X server localhost:86.0
For help use the following commands:
tasklist - Task list organized by category
taskhelp - One line summary of available tasks
help taskname - Full help for task
toolhelp - One line summary of available tools
help par.parametername - Full help for parameter name
Activating auto-logging. Current session state plus future input saved.
Filename : ipython-20181211-183852.log
Mode : backup
Output logging : False
Raw input log : False
Timestamping : False
State : active
*** Loading ATNF ASAP Package...
*** ... ASAP (rev#36500) import complete ***
CASA <2>: execfile('findsources.py')







gaussian_fits.txt (~/Dropbox/Trabajo/PresPyDay) - gedit

Open Save Undo

gaussian_fits.txt

```
1 Freq(GHz)   source  AR(h:m:s)+-Error(sec)   Dec(d:m:s)+-Error(arcsec)   Flux+-Error(mJy)
      Major_axis+-Error(arcsec)   Minor_axis+-Error(arcsec)   PA+-Error(deg)
2 32.9966 component0  3:29:3.767+-0.0204  31:29:0.063+-0.002  1.3311+-0.059   0.09+-0.01
  0.06+-0.01  113.65+-13.81
3 32.9966 component1  3:29:3.7429+-0.0407 31:29:0.062+-0.006  0.5375+-0.0367  0.19+-0.02
  0.08+-0.01  108.49+-3.48
4 32.9966 component2  3:29:3.3864+-0.0325 31:29:0.056+-0.004  0.3249+-0.0192  0.14+-0.01
  0.06+-0.01  117.65+-5.27
```

Plain Text Tab Width: 4 Ln 1, Col 1 INS

Resumen

- Radio-interferometría
- CASA (tasks, tools, viewer)
- Python
  - Funciones (propias y built-in)
  - Listas, diccionarios, strings
  - Archivos de texto

Mi CASA

venturas y desventuras de una radioastrónoma en Python