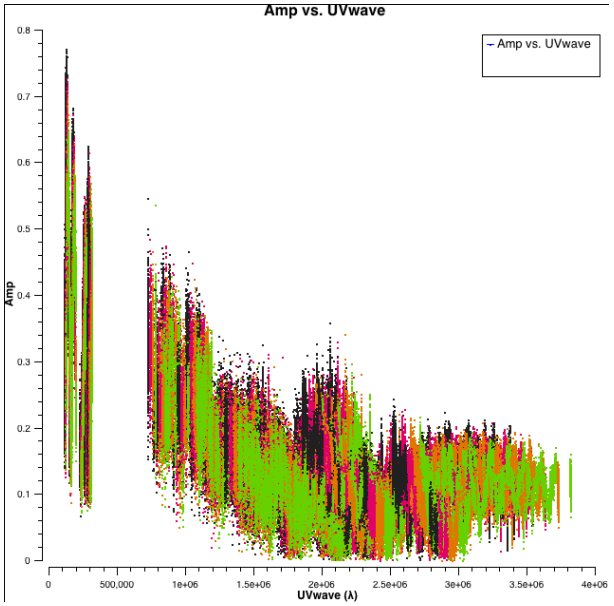
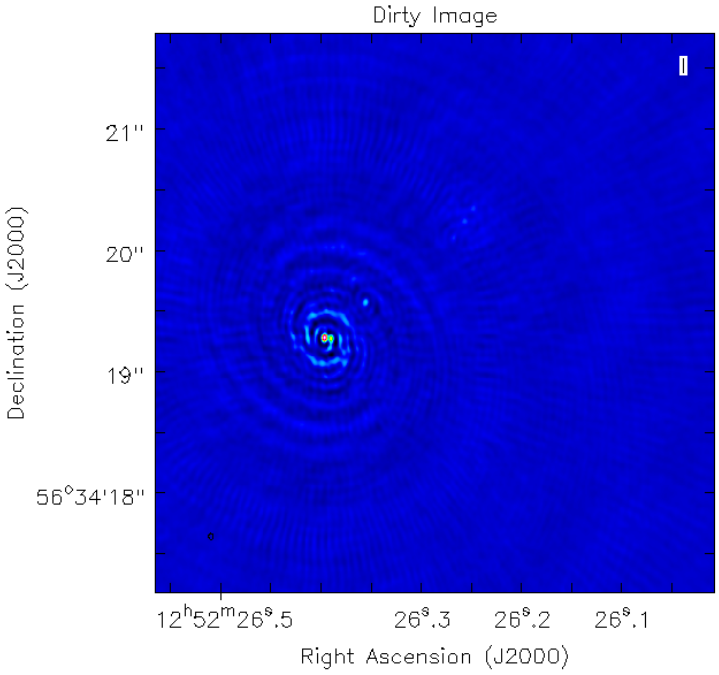
Basic Imaging and Self-Calibration (T4 + T7)

John McKean

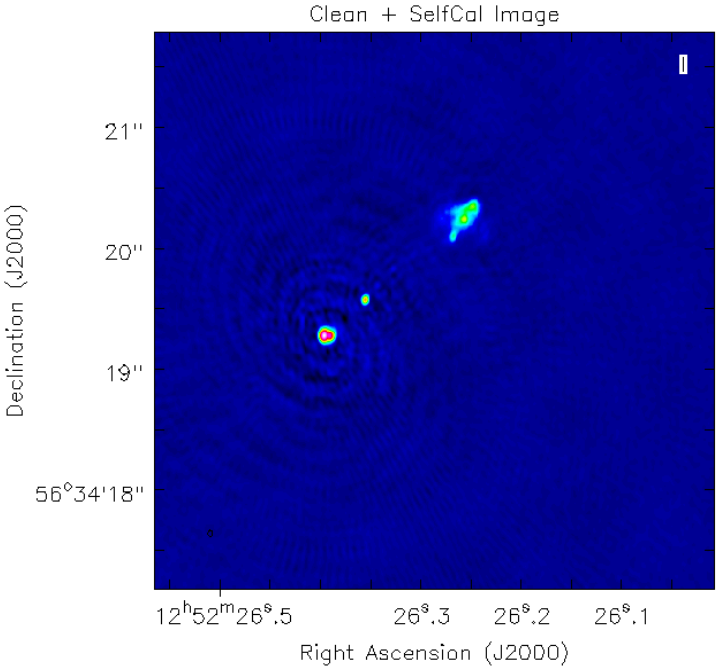
Visibilities



Fourier Transform



Deconvolution



AIM:

1. To make an image by taking the fast Fourier transform of the visibility data.
2. Carry out deconvolution using the CLEAN algorithm with CASA.
3. Use the new model obtained for the sky brightness distribution to carry out self-calibration.

During this process, we will make a ScriptForImaging.py file that can be used within CASA to make images automatically.

In the following "`> command`" is used to show inputs to the terminal and # comment # is used to explain where possible what is going on.

We will use the e-MERLIN data set on J1252+5634 that was edited and calibrated during the earlier tutorials (T2 and T3).

If you had problems during T3, download the calibrated dataset from,

http://almanas.jb.man.ac.uk/amsr/3C277p1/1252+5634.ms.tar

STEP 1 - Set-up the script

We will add our commands to a new ScriptForImaging.py, This script allows us to re-do what we have done, or parts of the process, automatically (useful for checking mistakes). Download the template from,

http://www.astron.nl/~mckean/ScriptForImaging.py

We can edit this file using your favourite text editor, e.g. emacs, pico, etc.

```
> pico ScriptForImaging.py
```

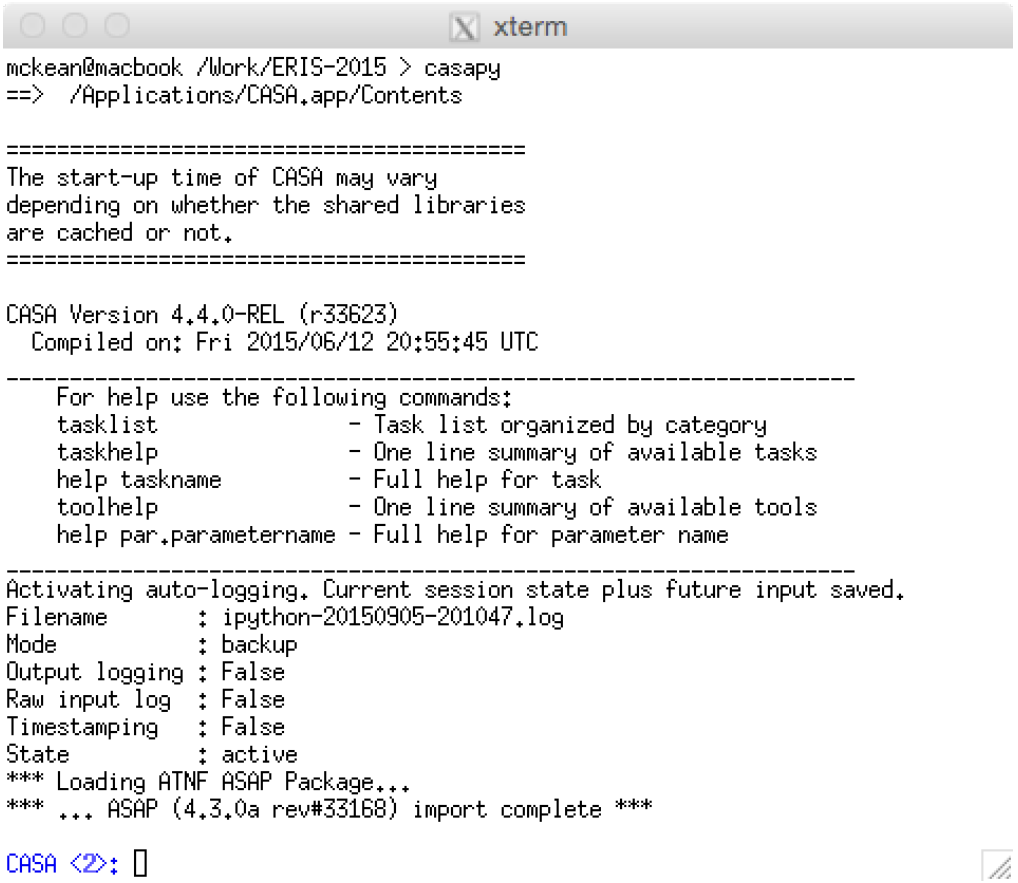
We will edit the script as we go.


ScriptForImaging — Edited
ScriptForImaging.py 〉 No Selection
e-MERLIN imaging script for J1252+5634 (4 spws x 64 channels) in CASA 4.4.0
#Calibration steps
thesteps = [0]
step_title = {0: 'Title of step 0 (casa task)',
1: 'Title of step 1 (casa task)'}
try:
print 'List of steps to be executed ...', mysteps
thesteps = mysteps
except:
print 'global variable mysteps not set.'
if (thesteps==[]):
thesteps = range(0,len(step_title))
print 'Executing all steps: ', thesteps
The Python variable 'mysteps' will control which steps
are executed when you start the script using
execfile('scriptForCalibration.py')
e.g. setting
mysteps = [2,3,4]# before starting the script will make the script execute
only steps 2, 3, and 4
Setting mysteps = [] will make it execute all steps.
print 'Write the value for variables -> run the script from the beginning'
#definitions
msfile = '1252+5634.ms' #ms multisource file
myspw = '0~3' #spw of interest, use myspw = '3' is your computer is slow
# description of step
mystep = 0
if(mystep in thesteps):
casalog.post('Step '+str(mystep)+' '+step_title[mystep],'INFO')
print 'Step ', mystep, step_title[mystep]
# description of step
mystep = 1
if(mystep in thesteps):
casalog.post('Step '+str(mystep)+' '+step_title[mystep],'INFO')
print 'Step ', mystep, step_title[mystep]
Here we enter our steps
Here we enter our variables
Here we will enter our commands
Here we will enter our commands

To start CASA,

```
> casapy      # start CASA #
```

You should see the following in your terminal



To run the script,

```
> mysteps = [0, 1]      # this will run steps 0 and 1 #
> execfile('ScriptForImaging.py')      # this run the script#
```

Nothing will happen because we have no commands yet, but msfile and myspw alias have been set.

STEP 2 - Determine our imaging field-of-view and pixel size

We will make an image by taking the fast Fourier transform (FFT) of the visibility data. This will involve projecting the sky surface brightness distribution onto a regular grid of pixels. We have some choices to make,

1. What is the size of the image that we would like to make?
2. How large should the pixels be?

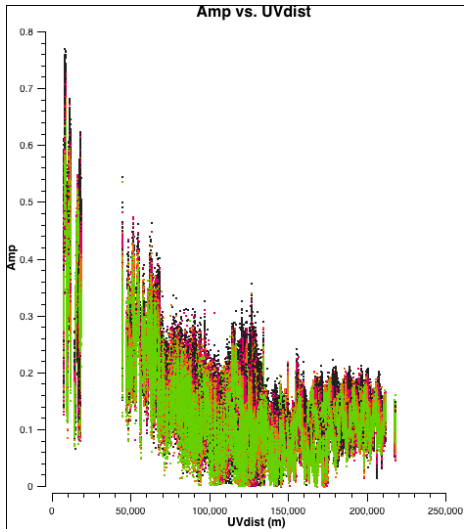
Image size: The visibilities contain information from all of the sources in the field-of-view. Technically we should make an image that is equal to this field-of-view. Our array is 6 antennas that are 25 m in size.

What is the field of view or a 25 m telescope at ~5 GHz?

```
> 3600 * (180 / pi) * (3e8 / 5.265e9) / 25    # arcsec * (rad->deg) * (c / ν) / D #
Out: 470.11921651759855      # Full width half max in arcsec #
```

This should be the field-of-view that we image, but we will use ~5 arcsec for speed.

Pixel Size: We need to Nyqvist sample the data when we projected it onto a regular grid so that we do not lose information. We can estimate pixel-size by considering the longest baseline in our data set using plotms and plotting AMP versus UVDIST (colourise SPW, corr='RR, LL', argchannel = '64').



We see that the longest baseline is at ~220 km. So we can estimate the synthesised beam with,

```
> 3600 * (180 / pi) * (3e8 / 5.265e9) / 220e3
Out: 0.0534734202161501      # max resolution in arcsec #
```

This is approximately what we would expect, so we take 10.7 mas pixels for safety (1/5 sampling).

STEP 3 - Make an image

We will start by making our first image, which will be the FFT of the visibility data. All deconvolution is carried out in CASA using the CLEAN task.

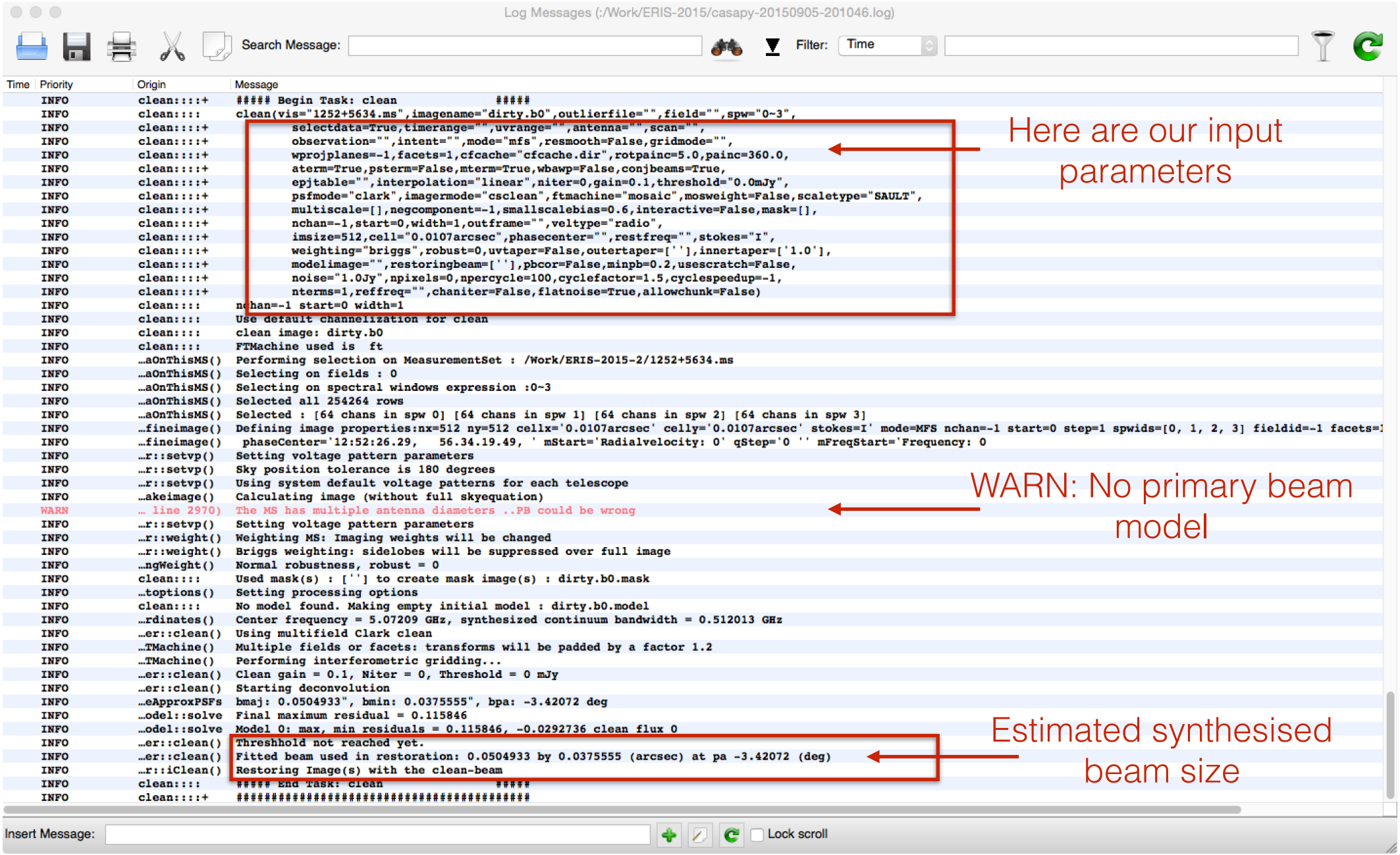
```
> help clean
```

This will give you a full summary of the task and suggested input parameters. Many of them we will not use here for this tutorial.

Start by replacing all of the parameters back to their defaults

```
> default clean

> vis = msfile              # Name of the visibility MS file #
> spw = myspw               # spectral windows that we will use #
> cell = "0.0107arcsec"     # pixel-size we will use #
> imsize = 512              # image size we will use (~5 arcsec) #
> weighting = "briggs"      # set visibility weighting #
> robust = 0                # set robust parameter (balance between nat/uni) #
> niter = 0                 # we will do no convolution #
> imagename = "dirty.b0"    # call your image something #
> inp                       # check your inputs (nothin should be red) #
> go clean                  # run the task #
```

Look at your logger window to view the progress of CLEAN.



Time	Priority	Origin	Message
	INFO	clean::::+	##### Begin Task: clean #####
	INFO	clean::::	clean(vis="1252+5634.ms",imagename="dirty.b0",outlierfile="",field="",spw="0~3",
	INFO	clean::::+	selectdata=True,timerange="",uvrange="",antenna="",scan="",
	INFO	clean::::+	observation="",intent="",mode="mfs",resmooth=False,gridmode="",
	INFO	clean::::+	wprojplanes=-1,facets=1,cfcache="cfcache.dir",rotpainc=5.0,painc=360.0,
	INFO	clean::::+	aterm=True,psterm=False,mterm=True,wbawp=False,conjbeams=True,
	INFO	clean::::+	epjtable="",interpolation="linear",niter=0,gain=0.1,threshold="0.0mJy",
	INFO	clean::::+	psfmode="clark",imagermode="csclean",ftmachine="mosaic",mosweight=False,scaletype="SAULT",
	INFO	clean::::+	multiscale=[],negcomponent=-1,smallscalebias=0.6,interactive=False,mask=[],
	INFO	clean::::+	nchan=-1,start=0,width=1,outframe="",veltype="radio",
	INFO	clean::::+	imsize=512,cell="0.0107arcsec",phasecenter="",restfreq="",stokes="I",
	INFO	clean::::+	weighting="briggs",robust=0,uvtaper=False,outertaper=[''],innertaper=['1.0'],
	INFO	clean::::+	modelimage="",restoringbeam=[''],pbcor=False,minpb=0.2,usescratch=False,
	INFO	clean::::+	noise="1.0Jy",npixels=0,npercycle=100,cyclefactor=1.5,cyclespeedup=-1,
	INFO	clean::::+	nterms=1,reffreq="",chaniter=False,flatnoise=True,allowchunk=False)
	INFO	clean::::	nchan=-1 start=0 width=1
	INFO	clean::::	Use default channelization for clean
	INFO	clean::::	clean image: dirty.b0
	INFO	clean::::	FTMachine used is ft
	INFO	…aOnThisMS()	Performing selection on MeasurementSet : /Work/ERIS-2015-2/1252+5634.ms
	INFO	…aOnThisMS()	Selecting on fields : 0
	INFO	…aOnThisMS()	Selecting on spectral windows expression :0~3
	INFO	…aOnThisMS()	Selected all 254264 rows
	INFO	…aOnThisMS()	Selected : [64 chans in spw 0] [64 chans in spw 1] [64 chans in spw 2] [64 chans in spw 3]
	INFO	…fineimage()	Defining image properties:nx=512 ny=512 cellx='0.0107arcsec' celly='0.0107arcsec' stokes=I' mode=MFS nchan=-1 start=0 step=1 spwids=[0, 1, 2, 3] fieldid=-1 facets=1
	INFO	…fineimage()	phaseCenter='12:52:26.29, 56.34.19.49, ' mStart='Radialvelocity: 0' qStep='0 '' mFreqStart='Frequency: 0
	INFO	…r::setvp()	Setting voltage pattern parameters
	INFO	…r::setvp()	Sky position tolerance is 180 degrees
	INFO	…r::setvp()	Using system default voltage patterns for each telescope
	INFO	…akeimage()	Calculating image (without full skyequation)
	WARN	… line 2970)	The MS has multiple antenna diameters ..PB could be wrong
	INFO	…r::setvp()	Setting voltage pattern parameters
	INFO	…r::weight()	Weighting MS: Imaging weights will be changed
	INFO	…r::weight()	Briggs weighting: sidelobes will be suppressed over full image
	INFO	…ngWeight()	Normal robustness, robust = 0
	INFO	clean::::	Used mask(s) : [''] to create mask image(s) : dirty.b0.mask
	INFO	…toptions()	Setting processing options
	INFO	clean::::	No model found. Making empty initial model : dirty.b0.model
	INFO	…rdinates()	Center frequency = 5.07209 GHz, synthesized continuum bandwidth = 0.512013 GHz
	INFO	…er::clean()	Using multifield Clark clean
	INFO	…TMachine()	Multiple fields or facets: transforms will be padded by a factor 1.2
	INFO	…TMachine()	Performing interferometric gridding...
	INFO	…er::clean()	Clean gain = 0.1, Niter = 0, Threshold = 0 mJy
	INFO	…er::clean()	Starting deconvolution
	INFO	…eApproxPSFs	bmaj: 0.0504933", bmin: 0.0375555", bpa: -3.42072 deg
	INFO	…odel::solve	Final maximum residual = 0.115846
	INFO	…odel::solve	Model 0: max, min residuals = 0.115846, -0.0292736 clean flux 0
	INFO	…er::clean()	Threshhold not reached yet.
	INFO	…er::clean()	Fitted beam used in restoration: 0.0504933 by 0.0375555 (arcsec) at pa -3.42072 (deg)
	INFO	…r::iClean()	Restoring Image(s) with the clean-beam
	INFO	clean::::	##### End Task: clean #####
	INFO	clean::::+	##

Lets look at the output. We have generated 5 images that are all on the same grid

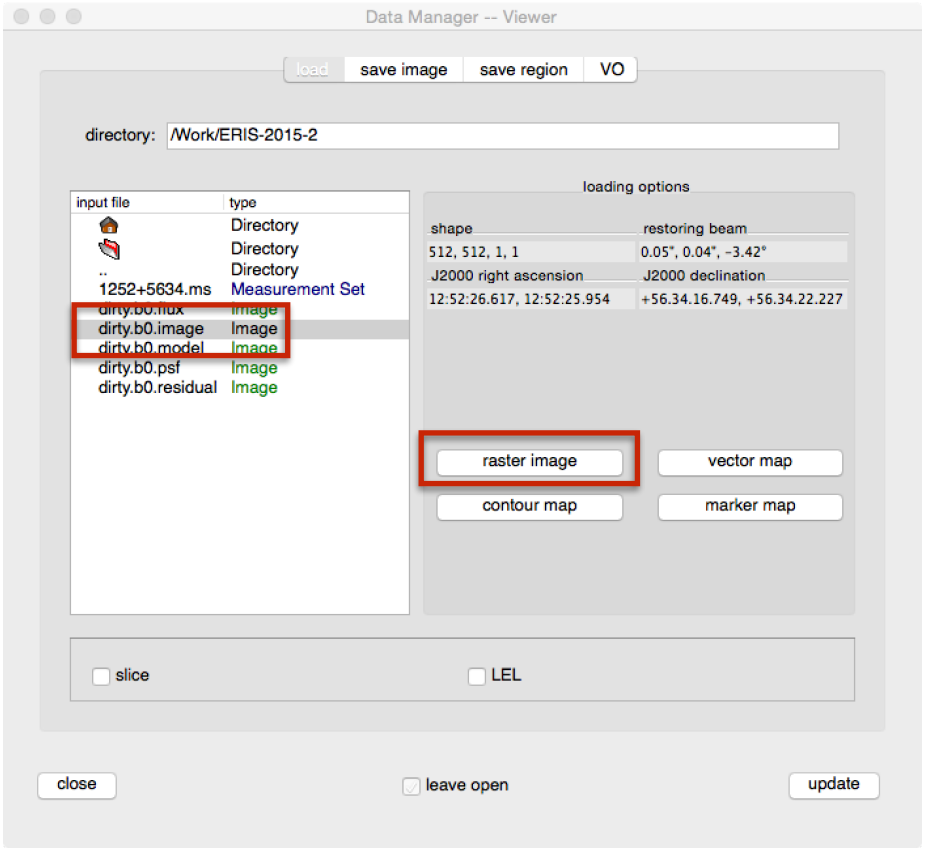
```
dirty.b0.image      # The 'deconvolved' image #
dirty.b0.psf        # The image of the point spread function (FFT of the uv-sampling function) #
dirty.b0.model      # The image containing your model components (delta functions, truncated Gaussians) #
dirty.b0.residual   # The image made by subtracting the model from the visibility of doing an FFT #
dirty.b0.flux       # An image of the expected primary beam response #
```

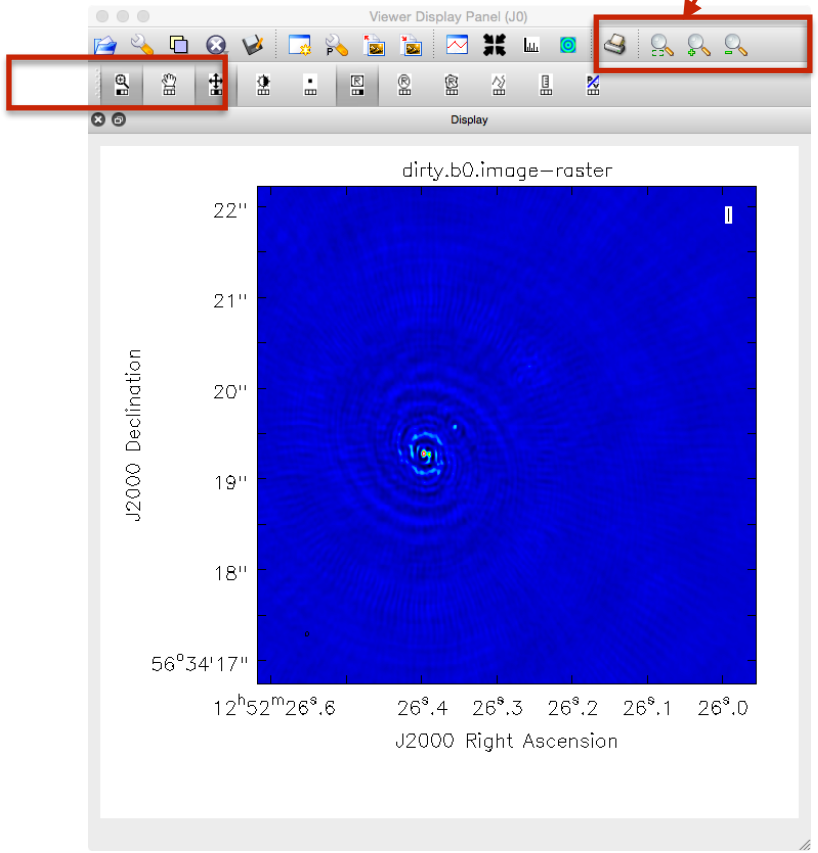
We can look at each of these images using the CASA VIEWER (run interactively or from the command line).

```
> viewer     # start the viewer GUI and DATA MANAGER #
```

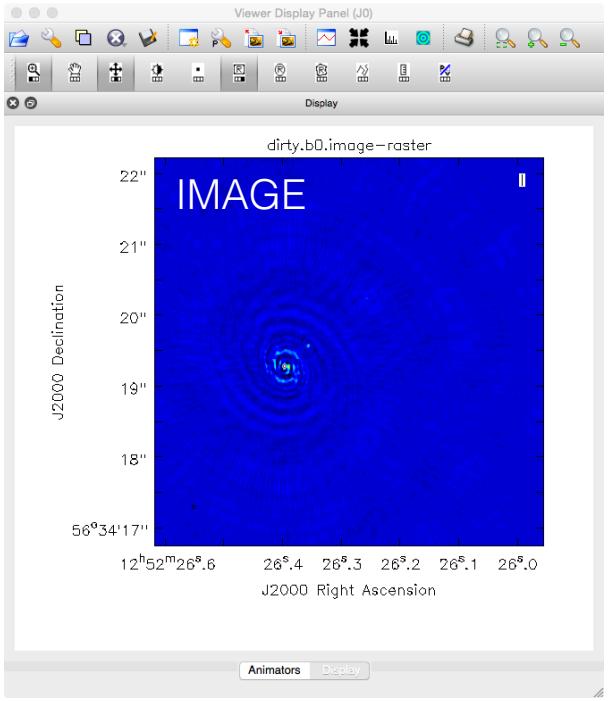
Zoom in/out

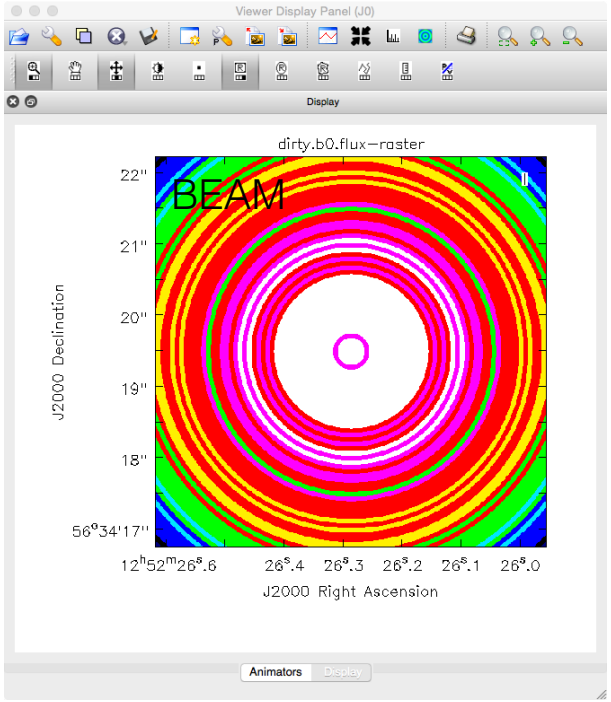
This will start the VIEWER GUI, select the dirty.b0.image and then select raster map.

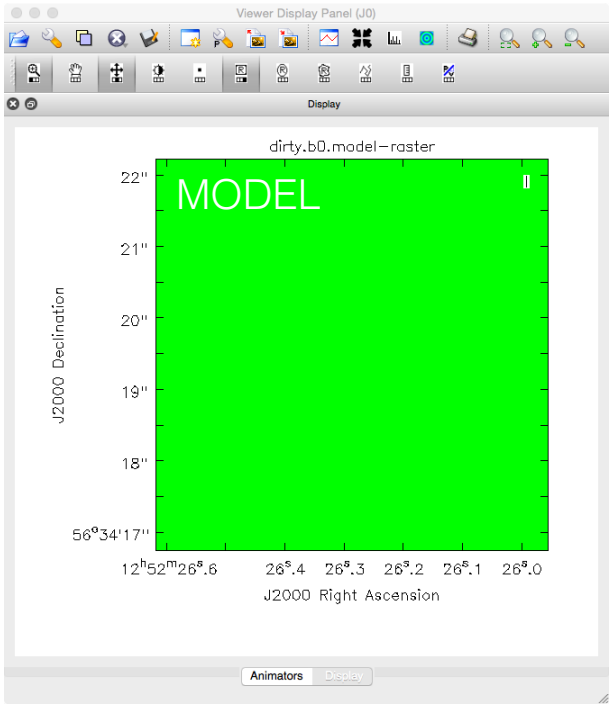


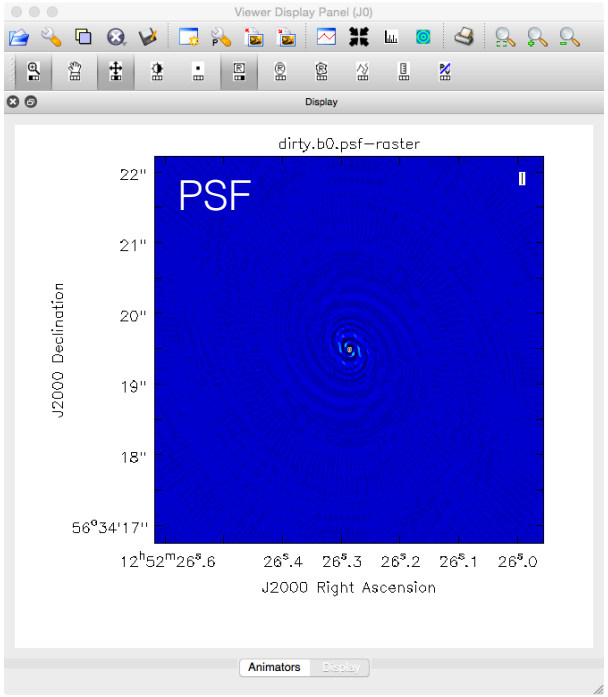


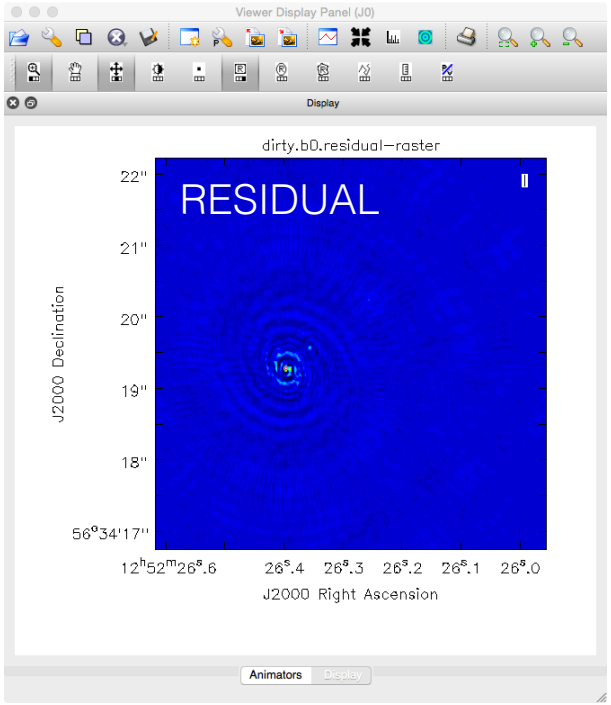
Lets look at each of the output images (either start a new VIEWER or add multiple images to the same VIEWER and use the ANIMATOR option - top menu -> VIEW -> ANIMATOR).











All that seemed to work well, so lets add the parameters of our CLEAN run to our script. Every time we run a task in CASA we generate a, for example clean.last file

```
> !more clean.last
```

and copy the final part to our script, and if we wanted, do what we just did using our script,

```
> mysteps = [0]      # this will run step 0 #
> execfile('ScriptForImaging.py')      # this run the script#
```

ScriptForImaging — Edited

ScriptForImaging.py 〉 No Selection

```
1  # e-MERLIN imaging script for J1252+5634 (4 spws x 64 channels) in CASA 4.4.0
2
3  #Calibration steps
4  thesteps = [0]
5  step_title = {0: 'Make dirty image (clean)',
6                1: 'Make clean image (clean)'}
7
8  try:
9    print 'List of steps to be executed ...', mysteps
10   thesteps = mysteps
11 except:
12   print 'global variable mysteps not set.'
13 if (thesteps==[]):
14   thesteps = range(0,len(step_title))
15   print 'Executing all steps: ', thesteps
16
17
18 # The Python variable 'mysteps' will control which steps
19 # are executed when you start the script using
20 #   execfile('scriptForCalibration.py')
21 # e.g. setting
22 #   mysteps = [2,3,4]# before starting the script will make the script execute
23 # only steps 2, 3, and 4
24 # Setting mysteps = [] will make it execute all steps.
25
26 print 'Write the value for variables -> run the script from the beginning'
27 #definitions
28
29 msfile = '1252+5634.ms' #ms multisource file
30 myspw = '0~3' #spw of interest, use myspw = '3' if your computer is slow
31
32 # description of step
33 mystep = 0
34 if(mystep in thesteps):
35   casalog.post('Step '+str(mystep)+' '+step_title[mystep],'INFO')
36   print 'Step ', mystep, step_title[mystep]
37
38   clean(vis="1252+5634.ms",imagename="dirty.b0",outlierfile="",field="",spw="0~3",selectdata=True,timerange="",
       uvrange="",antenna="",scan="",observation="",intent="",mode="mfs",resmooth=False,gridmode="",wprojplanes=-1
       ,facets=1,cfcache="cfcache.dir",rotpainc=5.0,painc=360.0,aterm=True,psterm=False,mterm=True,wbawp=False,
       conjbeams=True,ejtable="",interpolation="linear",niter=0,gain=0.1,threshold="0.0mJy",psfmode="clark",
       imagermode="csclean",ftmachine="mosaic",mosweight=False,scaletype="SAULT",multiscale=[],negcomponent=-1,
       smallscalebias=0.6,interactive=False,mask=[],nchan=-1,start=0,width=1,outframe="",veltype="radio",imsize=
       512,cell="0.0107arcsec",phasecenter="",restfreq="",stokes="I",weighting="briggs",robust=0,uvtaper=False,
       outertaper=[''],innertaper=['1.0'],modelimage="",restoringbeam=[''],pbcor=False,minpb=0.2,usescratch=False,
       noise="1.0Jy",npixels=0,npercycle=100,cyclefactor=1.5,cyclespeedup=-1,nterms=1,reffreq="",chaniter=False,
       flatnoise=True,allowchunk=False)
39
40
41 # description of step
```

Update our steps

copy clean parameters here (remember to indent)

STEP 4 - What about image weights

So far we have only used robust = 0, but lets try the case of natural and uniform weights (robust = 2 and = -2).

```
> tget clean                # recover the last set of parameters used #
> robust = 2                # set robust parameter to 2 (natural weighting) #
> imagename = “dirty.b2”    # set new image name to make new file #
> go clean                  # start FFT #
```

And once that is completed, we can add the clean.last command to our script. The run with robust = -2

```
> tget clean                 # recover the last set of parameters used #
> robust = -2                # set robust parameter to -2 (uniform weighting) #
> imagename = “dirty.b-2”    # set new image name to make new file #
> go clean                   # start FFT #
```

Note the synthesised beam sizes that are estimated by CASA for the different weights.

Next lets look at the dirty images and psf images using the VIEWER.

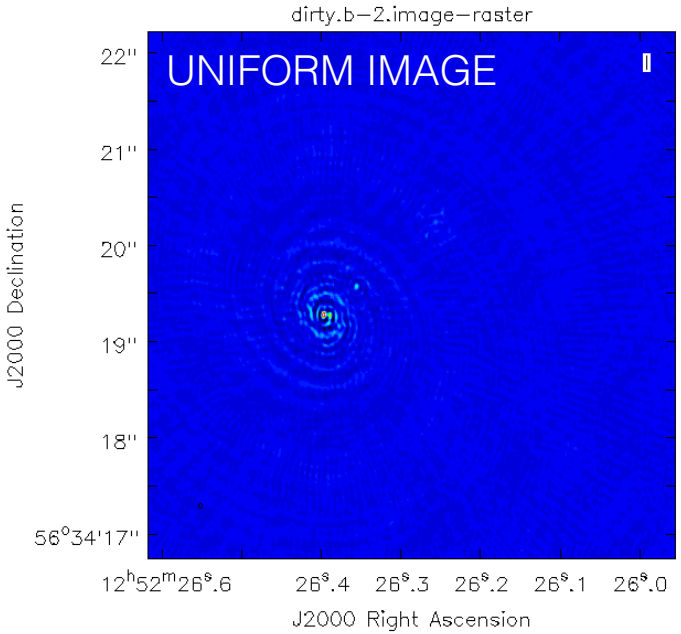
ScriptForImaging

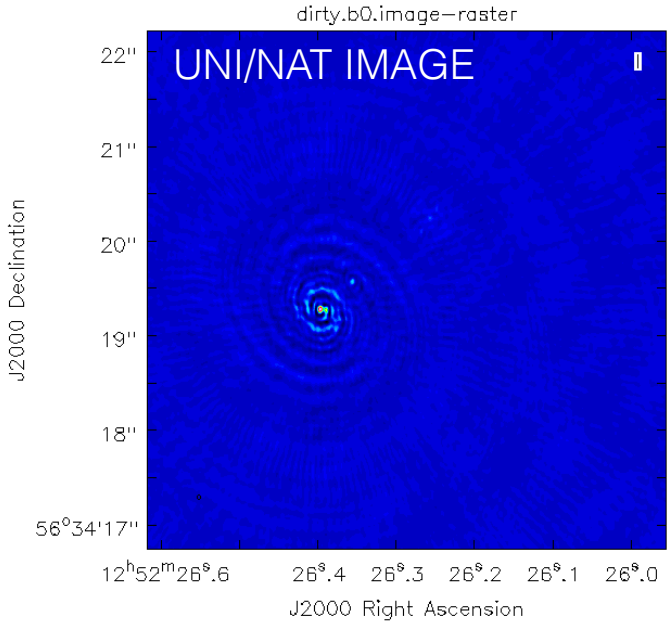
ScriptForImaging.py 〉 No Selection

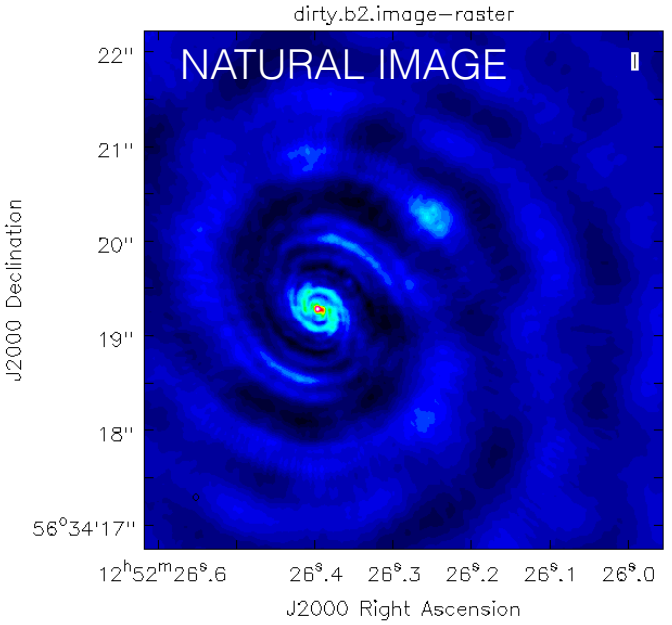
```
# Setting mysteps = [] will make it execute all steps.

print 'Write the value for variables -> run the script from the beginning'
#definitions

msfile = '1252+5634.ms' #ms multisource file
myspw = '0~3' #spw of interest, use myspw = '3' if your computer is slow

# description of step
mystep = 0
if(mystep in thesteps):
  casalog.post('Step '+str(mystep)+' '+step_title[mystep],'INFO')
  print 'Step ', mystep, step_title[mystep]

  clean(vis="1252+5634.ms",imagename="dirty.b0",outlierfile="",field="",spw="0~3",selectdata=True,timerange="",
      uvrange="",antenna="",scan="",observation="",intent="",mode="mfs",resmooth=False,gridmode="",wprojplanes=-1
      ,facets=1,cfcache="cfcache.dir",rotpainc=5.0,painc=360.0,aterm=True,psterm=False,mterm=True,wbawp=False,
      conjbeams=True,epjtable="",interpolation="linear",niter=0,gain=0.1,threshold="0.0mJy",psfmode="clark",
      imagermode="csclean",ftmachine="mosaic",mosweight=False,scaletype="SAULT",multiscale=[],negcomponent=-1,
      smallscalebias=0.6,interactive=False,mask=[],nchan=-1,start=0,width=1,outframe="",veltype="radio",imsize=
      512,cell="0.0107arcsec",phasecenter="",restfreq="",stokes="I",weighting="briggs",robust=0,uvtaper=False,
      outertaper=[''],innertaper=['1.0'],modelimage="",restoringbeam=[''],pbcor=False,minpb=0.2,usescratch=False,
      noise="1.0Jy",npixels=0,npercycle=100,cyclefactor=1.5,cyclespeedup=-1,nterms=1,reffreq="",chaniter=False,
      flatnoise=True,allowchunk=False)

  clean(vis="1252+5634.ms",imagename="dirty.b2",outlierfile="",field="",spw="0~3",selectdata=True,timerange="",
      uvrange="",antenna="",scan="",observation="",intent="",mode="mfs",resmooth=False,gridmode="",wprojplanes=-1
      ,facets=1,cfcache="cfcache.dir",rotpainc=5.0,painc=360.0,aterm=True,psterm=False,mterm=True,wbawp=False,
      conjbeams=True,epjtable="",interpolation="linear",niter=0,gain=0.1,threshold="0.0mJy",psfmode="clark",
      imagermode="csclean",ftmachine="mosaic",mosweight=False,scaletype="SAULT",multiscale=[],negcomponent=-1,
      smallscalebias=0.6,interactive=False,mask=[],nchan=-1,start=0,width=1,outframe="",veltype="radio",imsize=
      512,cell="0.0107arcsec",phasecenter="",restfreq="",stokes="I",weighting="briggs",robust=2,uvtaper=False,
      outertaper=[''],innertaper=['1.0'],modelimage="",restoringbeam=[''],pbcor=False,minpb=0.2,usescratch=False,
      noise="1.0Jy",npixels=0,npercycle=100,cyclefactor=1.5,cyclespeedup=-1,nterms=1,reffreq="",chaniter=False,
      flatnoise=True,allowchunk=False)

  clean(vis="1252+5634.ms",imagename="dirty.b-2",outlierfile="",field="",spw="0~3",selectdata=True,timerange="",
      uvrange="",antenna="",scan="",observation="",intent="",mode="mfs",resmooth=False,gridmode="",wprojplanes=-1
      ,facets=1,cfcache="cfcache.dir",rotpainc=5.0,painc=360.0,aterm=True,psterm=False,mterm=True,wbawp=False,
      conjbeams=True,epjtable="",interpolation="linear",niter=0,gain=0.1,threshold="0.0mJy",psfmode="clark",
      imagermode="csclean",ftmachine="mosaic",mosweight=False,scaletype="SAULT",multiscale=[],negcomponent=-1,
      smallscalebias=0.6,interactive=False,mask=[],nchan=-1,start=0,width=1,outframe="",veltype="radio",imsize=
      512,cell="0.0107arcsec",phasecenter="",restfreq="",stokes="I",weighting="briggs",robust=-2,uvtaper=False,
      outertaper=[''],innertaper=['1.0'],modelimage="",restoringbeam=[''],pbcor=False,minpb=0.2,usescratch=False,
      noise="1.0Jy",npixels=0,npercycle=100,cyclefactor=1.5,cyclespeedup=-1,nterms=1,reffreq="",chaniter=False,
      flatnoise=True,allowchunk=False)


# description of step
mystep = 1
if(mystep in thesteps):
  casalog.post('Step '+str(mystep)+' '+step_title[mystep],'INFO')
  print 'Step ', mystep, step_title[mystep]
```

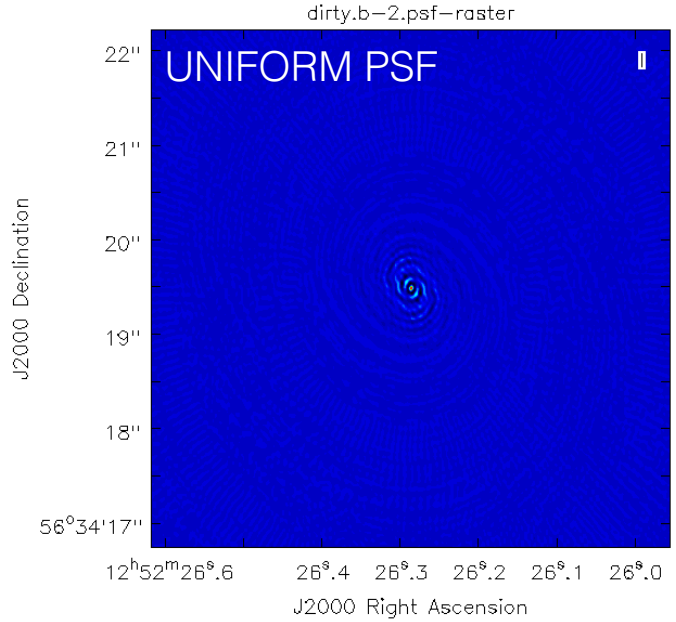
TIP: It is useful to first make a dirty image to see if you choice of pixel size (cell) and image size (imsize) is appropriate given your target observation.

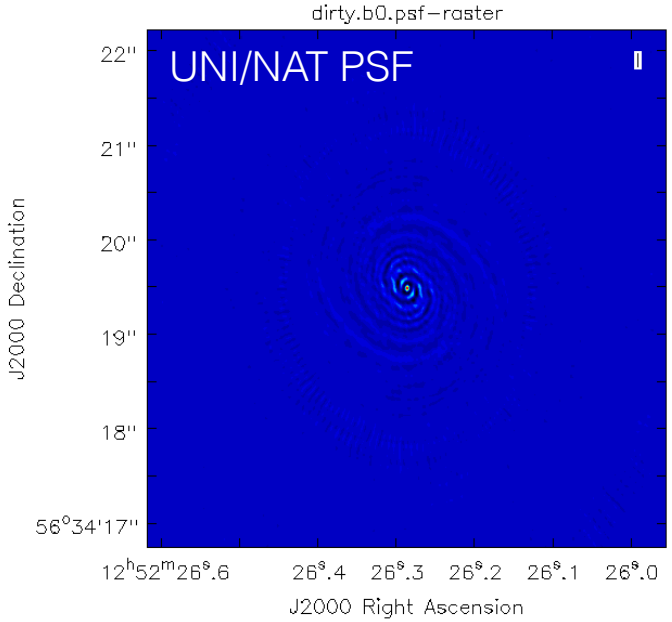
Also, look at the side-lobe structure of the PSF as it will help you when you are de-convolving the image,

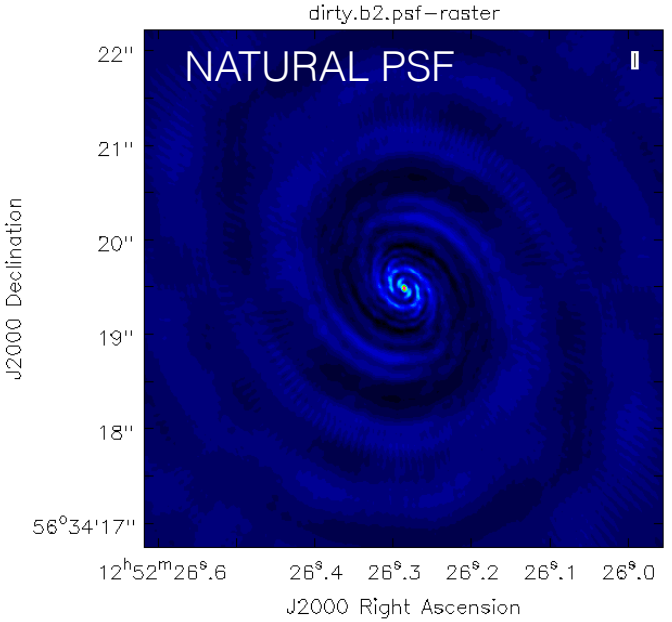












STEP 5 - Deconvolution

The ripples that we see in the dirty images are due to the side-lobe structure of the PSF. This is dependent on the uv-coverage (sampling function) and our choice of weighting. For the remainder of the tutorial, we will use Briggs weighting with robust = 0.

```
> tget clean      # recover the last set of parameters used #
> robust = 0      # set robust parameter to 0 (uniform/natural weighting) #
```

We deconvolve using the CLEAN algorithm, and in this case we will use delta functions to make a model for the source. Other options, for example, truncated Gaussians are possible, but we will not use here.

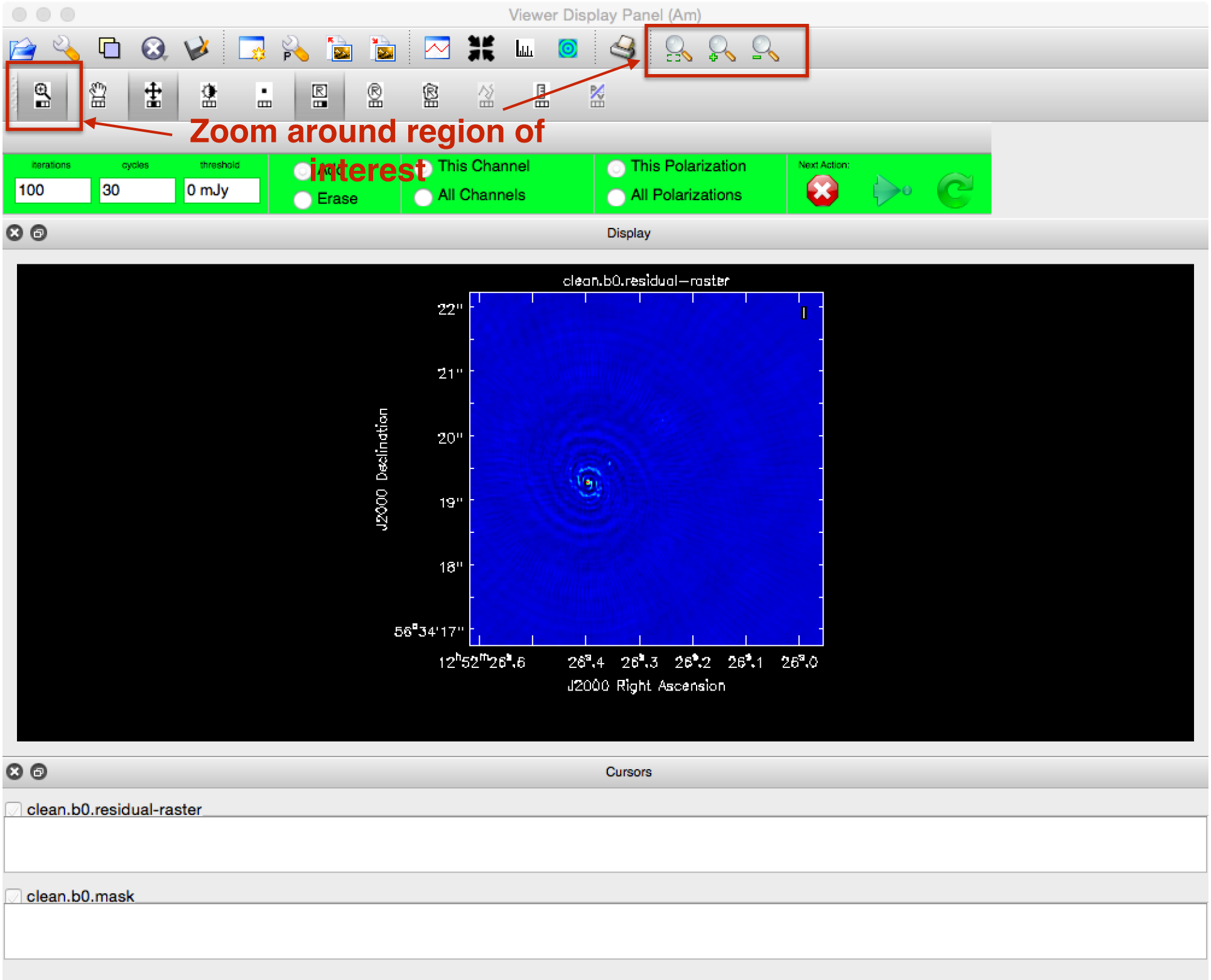
The CLEAN algorithm has the following steps:

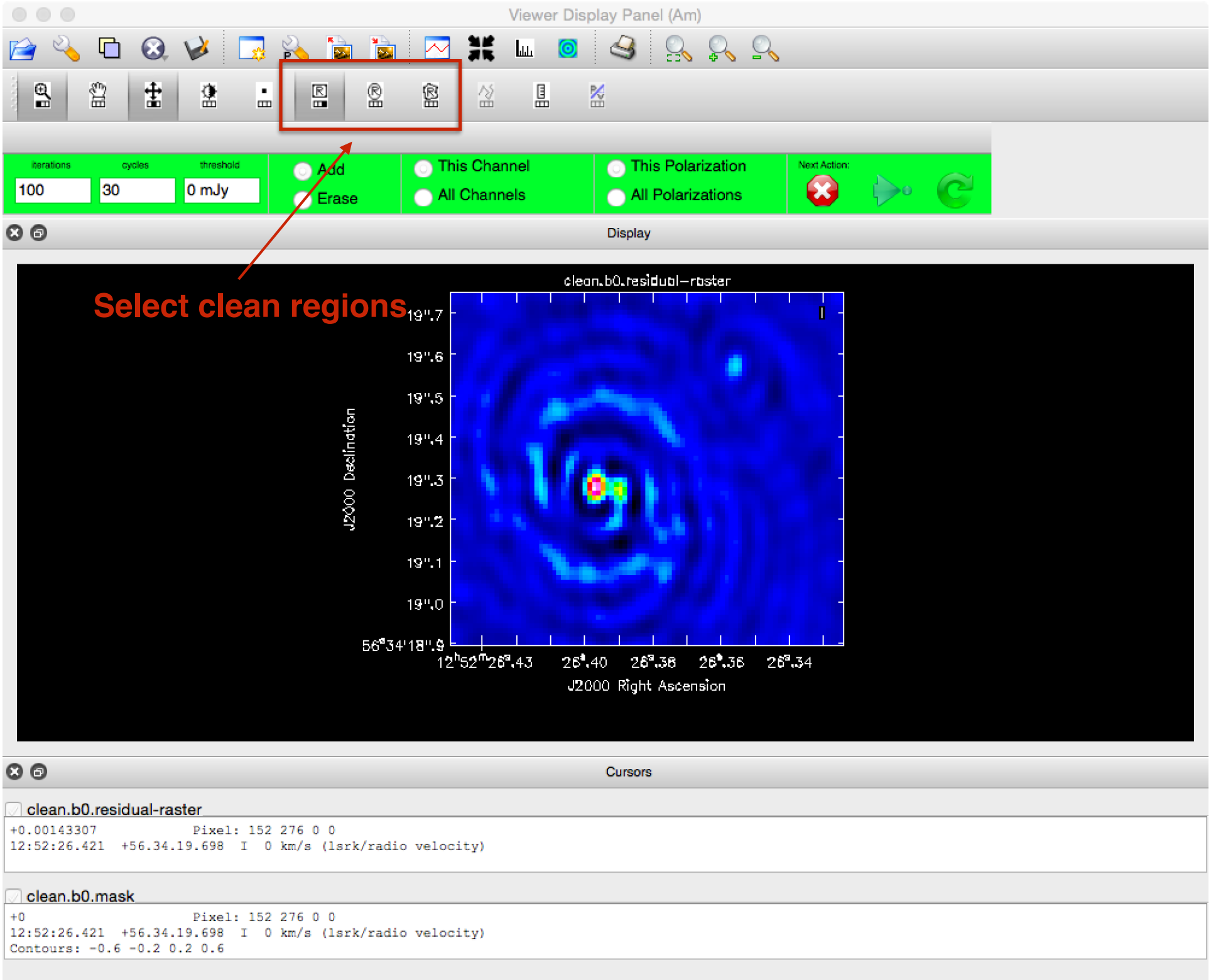
1. Identify the surface brightness peak in the map.
2. Fit a delta function to this position that has a value of the peak surface brightness * gain factor.
3. Subtract the delta function from the image.
4. Identify the next brightness peak and repeat steps 2 and 3 (Minor Cycle).
5. Subtract the collection of delta functions from the uv-data and re image.
6. Repeat steps 1-5 until some threshold is reached.

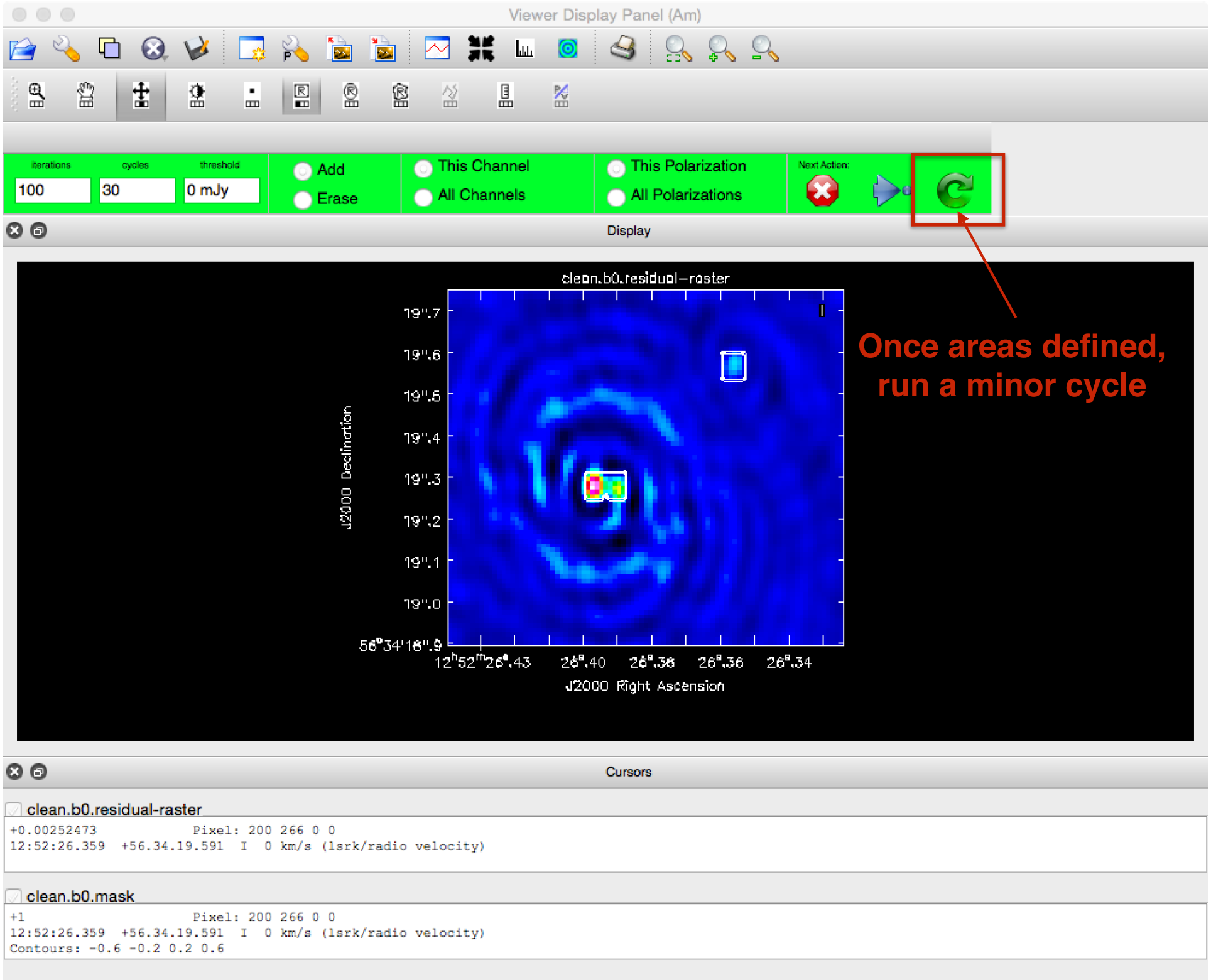
Now we need to define two new parameters for CLEAN

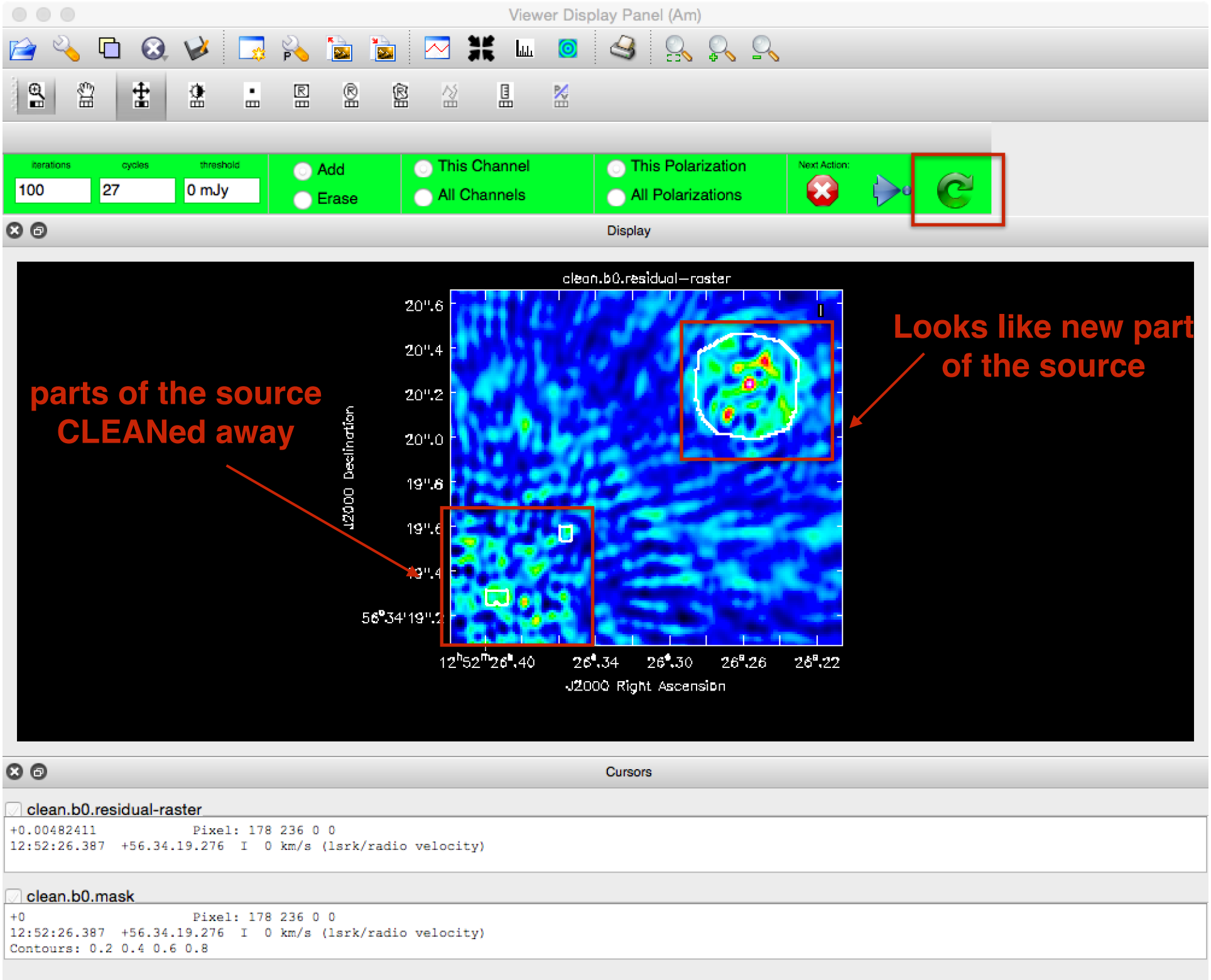
```
> niter = 3000              # number of interactions (trial / error) #
> gain = 0.05               # factor of the peak brightness to be subtracted #
> interactive = T           # to allow interactive cleaning #
> imagename = “clean.b0”    # set new image name to make new file #
> inp                       # review parameters #
> go clean                  # start FFT and deconvolution #
```

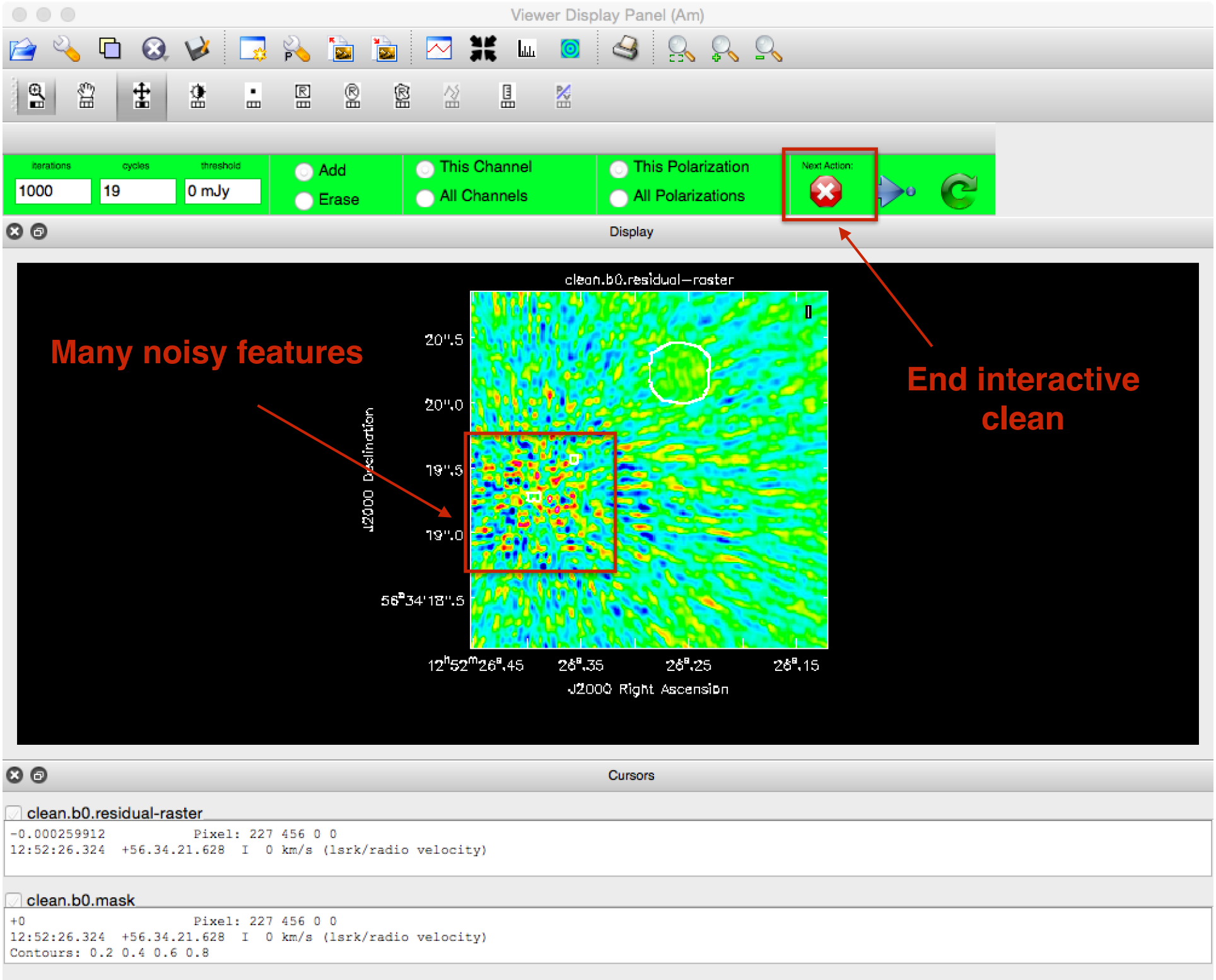
Remember to look at your logger for information.

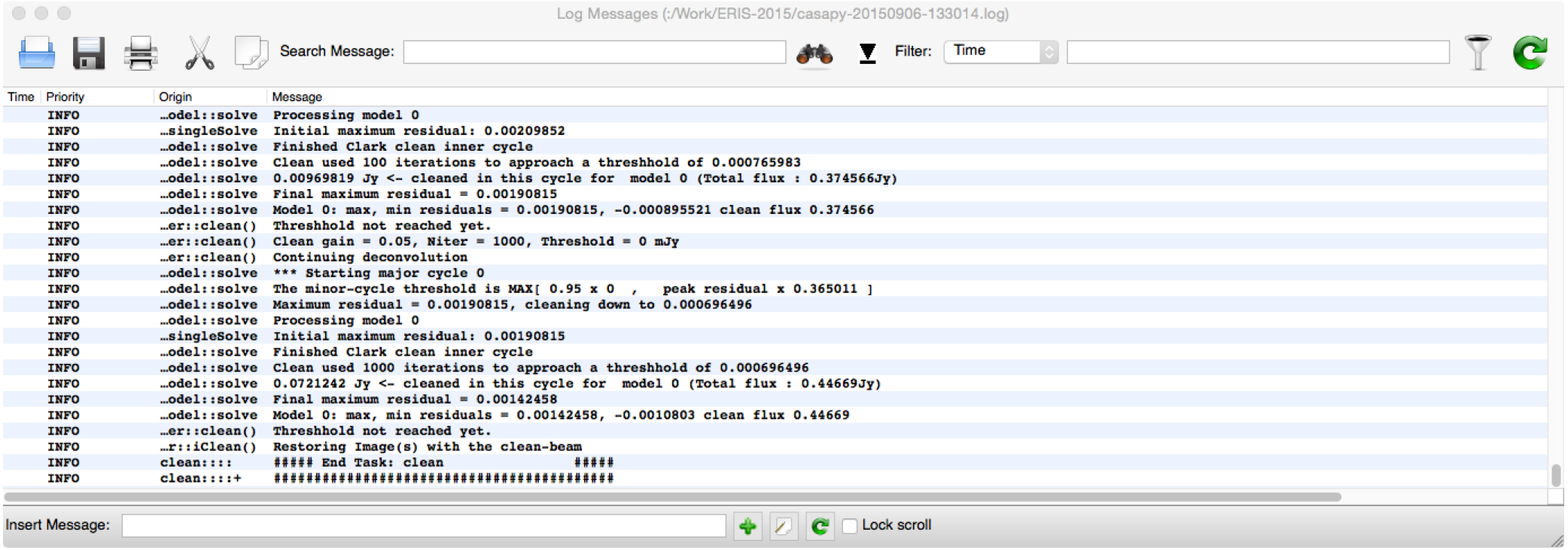

Viewer Display Panel (Am)
Zoom around region of interest
iterations
cycles
threshold
100
30
0 mJy
Erase
This Channel
All Channels
This Polarization
All Polarizations
Next Action:
Display
clean.b0.residual—raster
22''
21''
20''
19''
18''
56°34'17''
J2000 Declination
12h52m26s.6
26s.4
26s.3
26s.2
26s.1
26s.0
J2000 Right Ascension
Cursors
clean.b0.residual-raster
clean.b0.mask


Viewer Display Panel (Am)
iterations
cycles
threshold
100
30
0 mJy
Add
Erase
This Channel
All Channels
This Polarization
All Polarizations
Next Action:
Display
Select clean regions
clean.b0.residual-raster
19''.7
19''.6
19''.5
19''.4
19''.3
19''.2
19''.1
19''.0
56°34'18''.9
J2000 Declination
12h52m26s.43
26s.40
26s.38
26s.36
26s.34
J2000 Right Ascension
Cursors
clean.b0.residual-raster
+0.00143307 Pixel: 152 276 0 0
12:52:26.421 +56.34.19.698 I 0 km/s (lsrk/radio velocity)
clean.b0.mask
+0 Pixel: 152 276 0 0
12:52:26.421 +56.34.19.698 I 0 km/s (lsrk/radio velocity)
Contours: -0.6 -0.2 0.2 0.6


Viewer Display Panel (Am)
iterations
cycles
threshold
100
30
0 mJy
Add
Erase
This Channel
All Channels
This Polarization
All Polarizations
Next Action:
Display
clean.b0.residual-raster
19".7
19".6
19".5
19".4
19".3
19".2
19".1
19".0
56°34'18".9
J2000 Declination
12h52m26s.43
26s.40
26s.38
26s.36
26s.34
J2000 Right Ascension
Once areas defined,
run a minor cycle
Cursors
clean.b0.residual-raster
+0.00252473 Pixel: 200 266 0 0
12:52:26.359 +56.34.19.591 I 0 km/s (lsrk/radio velocity)
clean.b0.mask
+1 Pixel: 200 266 0 0
12:52:26.359 +56.34.19.591 I 0 km/s (lsrk/radio velocity)
Contours: -0.6 -0.2 0.2 0.6


Viewer Display Panel (Am)
iterations
cycles
threshold
100
27
0 mJy
Add
Erase
This Channel
All Channels
This Polarization
All Polarizations
Next Action:
Display
clean.b0.residual—raster
20".6
20".4
20".2
20".0
19".6
19".4
56°34'19".2
J2000 Declination
12h52m26s.40
26s.34
26s.30
26s.26
26s.22
J2000 Right Ascension
parts of the source CLEANed away
Looks like new part of the source
Cursors
clean.b0.residual-raster
+0.00482411 Pixel: 178 236 0 0
12:52:26.387 +56.34.19.276 I 0 km/s (lsrk/radio velocity)
clean.b0.mask
+0 Pixel: 178 236 0 0
12:52:26.387 +56.34.19.276 I 0 km/s (lsrk/radio velocity)
Contours: 0.2 0.4 0.6 0.8


Viewer Display Panel (Am)
iterations
cycles
threshold
1000
19
0 mJy
Add
Erase
This Channel
All Channels
This Polarization
All Polarizations
Next Action:
Display
clean.b0.residual—raster
Many noisy features
End interactive clean
20".5
20".0
19".5
19".0
56°34'18".5
J2000 Declination
12h52m26s.45
26s.35
26s.25
26s.15
J2000 Right Ascension
Cursors
clean.b0.residual-raster
-0.000259912 Pixel: 227 456 0 0
12:52:26.324 +56.34.21.628 I 0 km/s (lsrk/radio velocity)
clean.b0.mask
+0 Pixel: 227 456 0 0
12:52:26.324 +56.34.21.628 I 0 km/s (lsrk/radio velocity)
Contours: 0.2 0.4 0.6 0.8



We end clean when we think we have reached a reasonable noise limit.

Note that we have cleaned a total flux of ~0.45 Jy and the threshold is 0.0007 Jy (we will use these values for running CLEAN non-INTERACTIVELY).

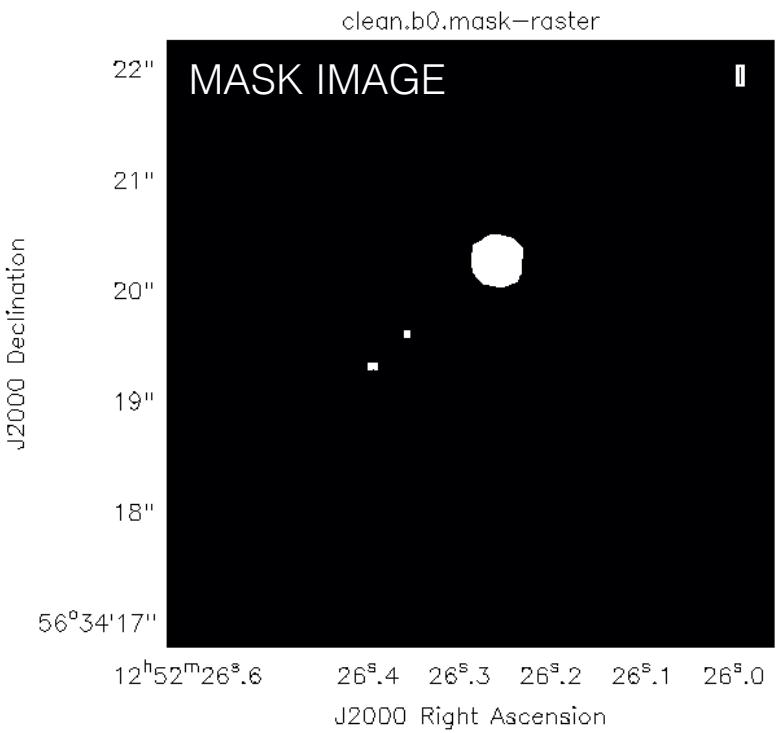
We have also generated a new file,

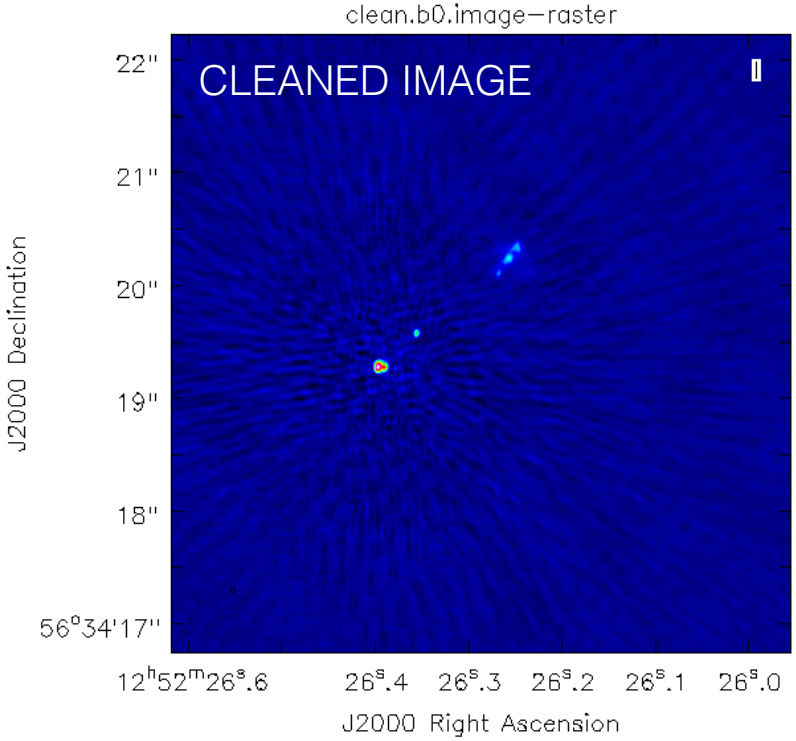
clean.b0.mask # The mask image that defines the CLEAN regions #

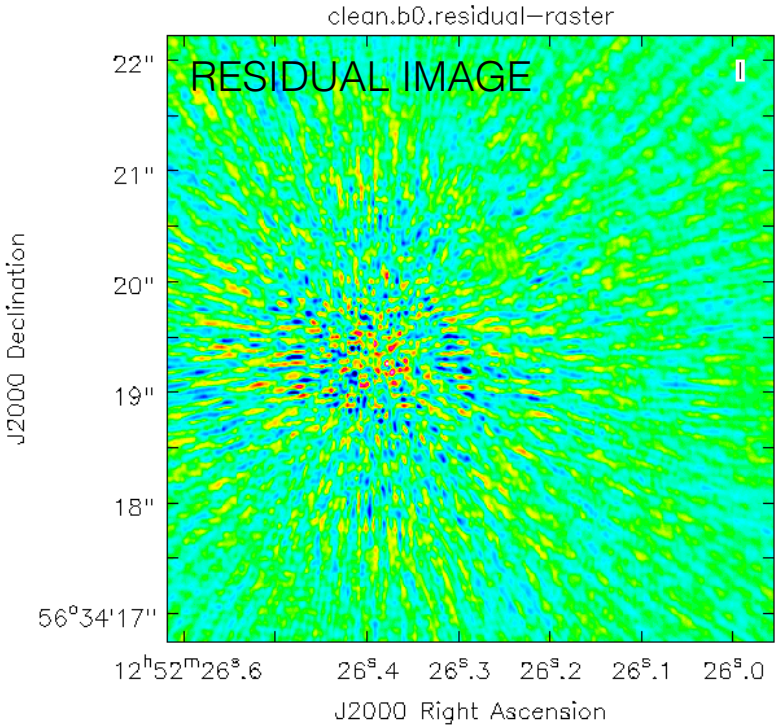
Let's look at the final images using the VIEWER

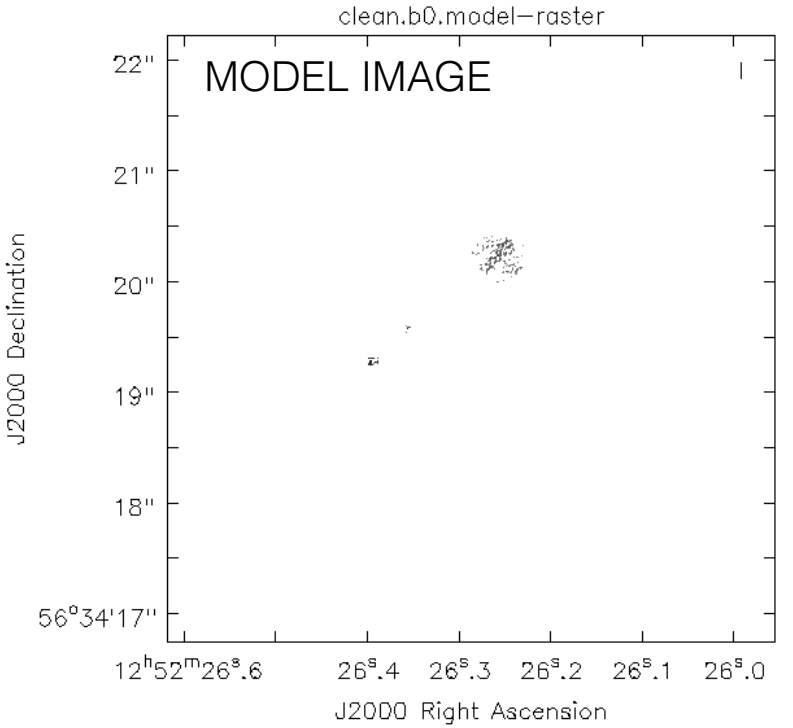
```
> viewer          # start the viewer GUI and DATA MANAGER #
```

Load the RASTER map of the image, model, residual, mask.


clean.b0.mask−raster
MASK IMAGE
22"
21"
20"
19"
18"
56°34'17"
J2000 Declination
12h52m26s.6
26s.4
26s.3
26s.2
26s.1
26s.0
J2000 Right Ascension


clean.b0.image−raster
CLEANED IMAGE
22"
21"
20"
19"
18"
56°34'17"
J2000 Declination
12h52m26s.6
26s.4
26s.3
26s.2
26s.1
26s.0
J2000 Right Ascension


clean.b0.residual−raster
RESIDUAL IMAGE
22"
21"
20"
19"
18"
56°34'17"
J2000 Declination
12h52m26s.6
26s.4
26s.3
26s.2
26s.1
26s.0
J2000 Right Ascension


clean.b0.model−raster
MODEL IMAGE
22"
21"
20"
19"
18"
56°34'17"
J2000 Declination
12h52m26s.6
26s.4
26s.3
26s.2
26s.1
26s.0
J2000 Right Ascension

All that seemed to work well, so lets add the parameters of our CLEAN run to our script. First, we add the threshold, give a new image name and set not to run interactively,

```
> tget clean                      # recover the last set of parameters used #
> imagename = “clean.b0.auto” # set new image name to make new file #
> interactive = F                 # don’t allow interactive cleaning #
> threshold = “0.7mJy”            # set threshold to stop cleaning #
> mask = “clean.b0.mask”          # use of pre-defined mask #
> inp                             # review parameters #
> tput clean                      # save parameters to the .last file #

> !more clean.last
```

and copy the final part to our script (step 2).

```
# description of step
mystep = 1
if(mystep in thesteps):
  casalog.post('Step '+str(mystep)+' '+step_title[mystep],'INFO')
  print 'Step ', mystep, step_title[mystep]

  clean(vis="1252+5634.ms",imagename="clean.b0.auto",outlierfile="",field="",spw="0~3",selectdata=True,timerange="",uvrange="",
      antenna="",scan="",observation="",intent="",mode="mfs",resmooth=False,gridmode="",wprojplanes=-1,facets=1,cfcache="cfcach
      e.dir",rotpainc=5.0,painc=360.0,aterm=True,psterm=False,mterm=True,wbawp=False,conjbeams=True,epjtable="",interpolation="
      linear",niter=3000,gain=0.05,threshold="0.7mJy",psfmode="clark",imagermode="csclean",ftmachine="mosaic",mosweight=False,
      scaletype="SAULT",multiscale=[],negcomponent=-1,smallscalebias=0.6,interactive=False,mask="clean.b0.mask",nchan=-1,start=
      0,width=1,outframe="",veltype="radio",imsize=512,cell="0.0107arcsec",phasecenter="",restfreq="",stokes="I",weighting="bri
      ggs",robust=0,uvtaper=False,outertaper=[''],innertaper=['1.0'],modelimage="",restoringbeam=[''],pbcor=False,minpb=0.2,
      usescratch=False,noise="1.0Jy",npixels=0,npercycle=100,cyclefactor=1.5,cyclespeedup=-1,nterms=1,reffreq="",chaniter=False
      ,flatnoise=True,allowchunk=False)
```

Lets try running everything using our script (this will overwrite our dirty images and make a new clean image). Depending on your computer, this should take about ~5 mins to run.

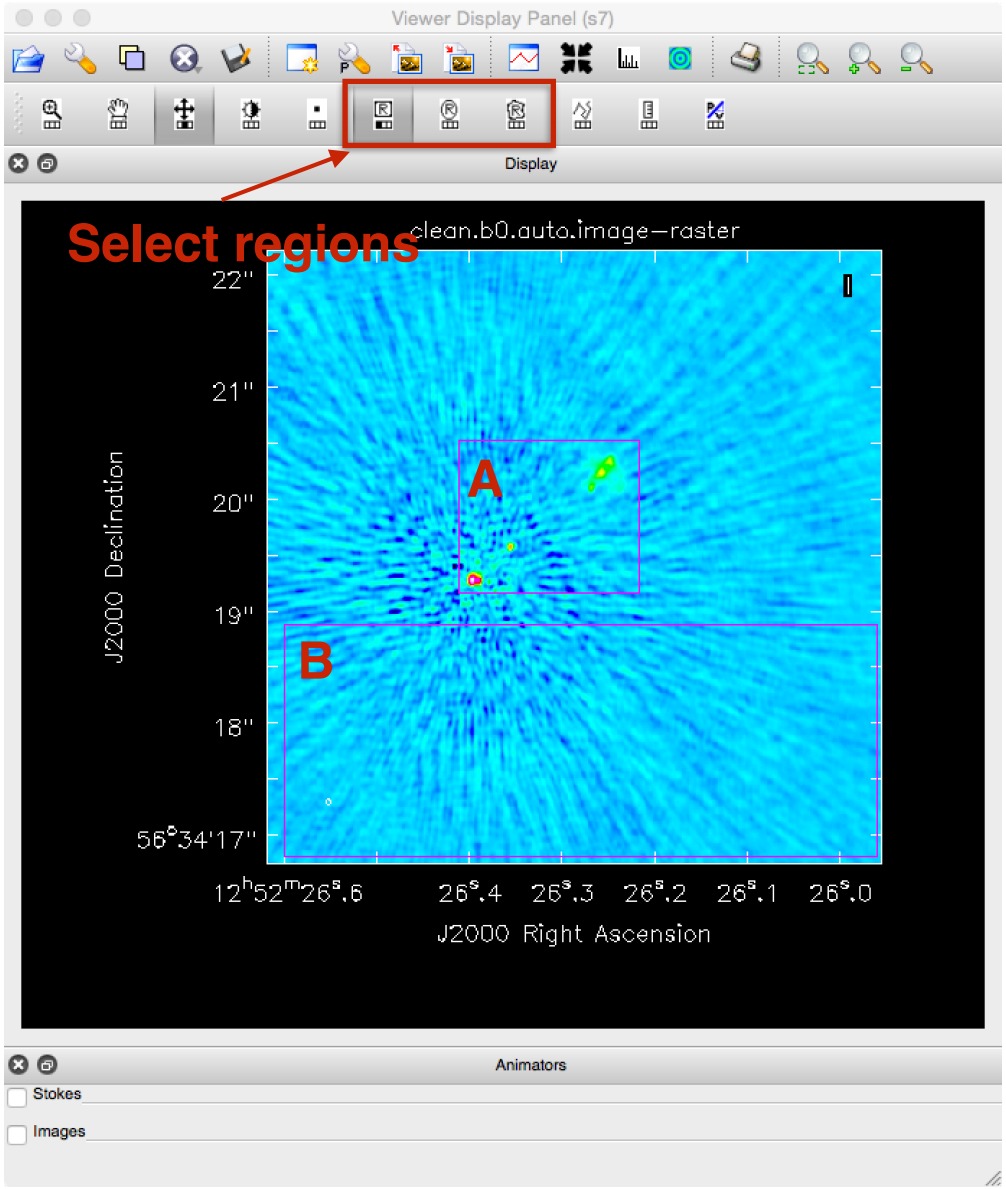
```
> mysteps = [0, 1]      # this will run step 0 #
> execfile(‘ScriptForImaging.py')      # this run the script#
```

STEP 6 - Image properties

We can use the VIEWER to estimate some image statistics based on our new clean image.

```
> viewer       # start the viewer GUI and DATA MANAGER #
```

Load the RASTER “clean.b0.auto.image” map of the VIEWER.



Double click inside the regions to get the statistics.

xterm

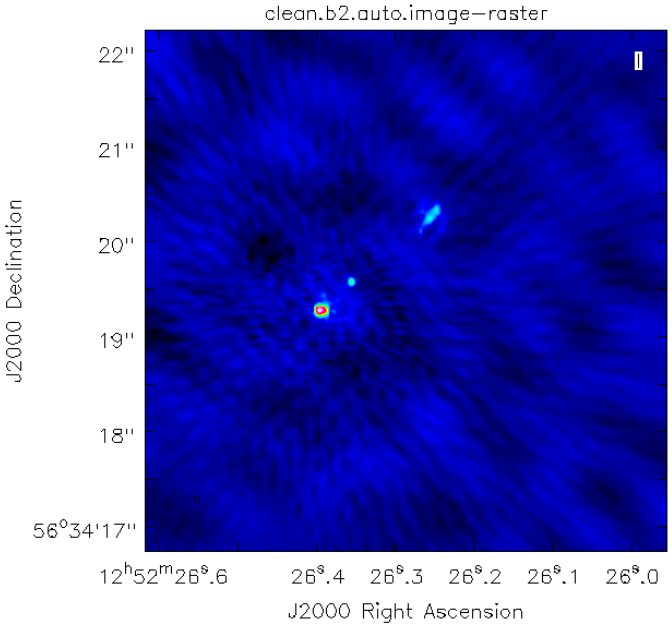
```
CASA <61>: ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---
(clean.b0.auto.image)
A       Stokes        Velocity          Frame        Doppler      Frequency
              I   5.23836km/s           LSRK          RADIO      5.072e+09
BrightnessUnit       BeamArea           Npts            Sum    FluxDensity
       Jy/beam        18.7674          19050   9.921288e+00   5.286446e-01
           Mean            Rms        Std dev        Minimum        Maximum
  5.208025e-04   4.284416e-03   4.252756e-03  -5.100328e-03   1.244205e-01
   region count
              1
---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ----
---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ----
(clean.b0.auto.image)
B       Stokes        Velocity          Frame        Doppler      Frequency
              I   5.23836km/s           LSRK          RADIO      5.072e+09
BrightnessUnit       BeamArea           Npts            Sum    FluxDensity
       Jy/beam        18.7674          95342  -7.816814e-01  -4.165101e-02
           Mean            Rms        Std dev        Minimum        Maximum
 -8.198710e-06   6.026908e-04   6.026382e-04  -4.210846e-03   3.558525e-03
   region count
              1
---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ----
CASA <61>:
```

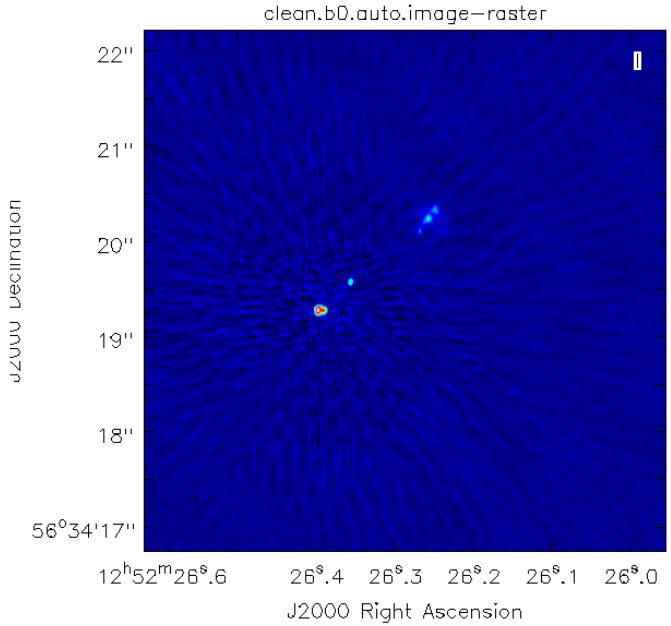
Note the flux-density of our target and the rms noise of the image

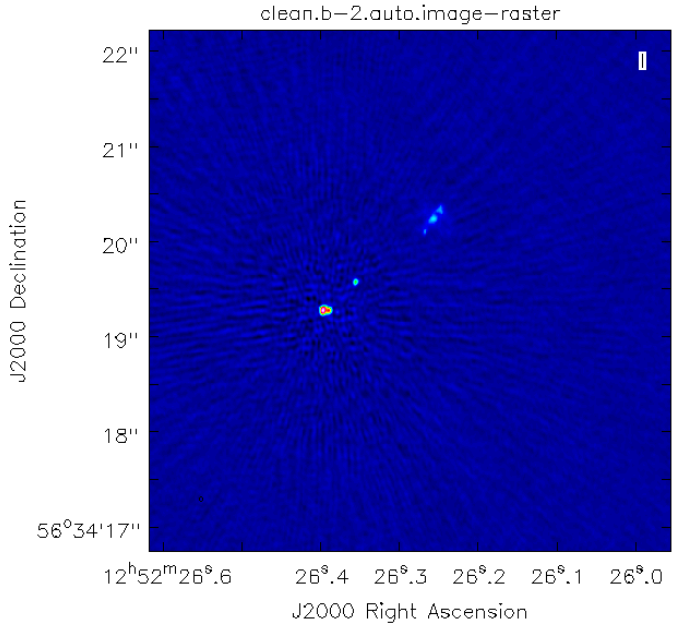
STEP 6 - Student exercise

Try making an image of the source using uniform and natural weighting (robust = 2 and -2), do this by making a new step 2 and 3 in your imaging script, and run it over lunch.

Remember to change the image name, otherwise you will overwrite your previous images.







Measure the flux-density and rms noise of each map, how do they compare.