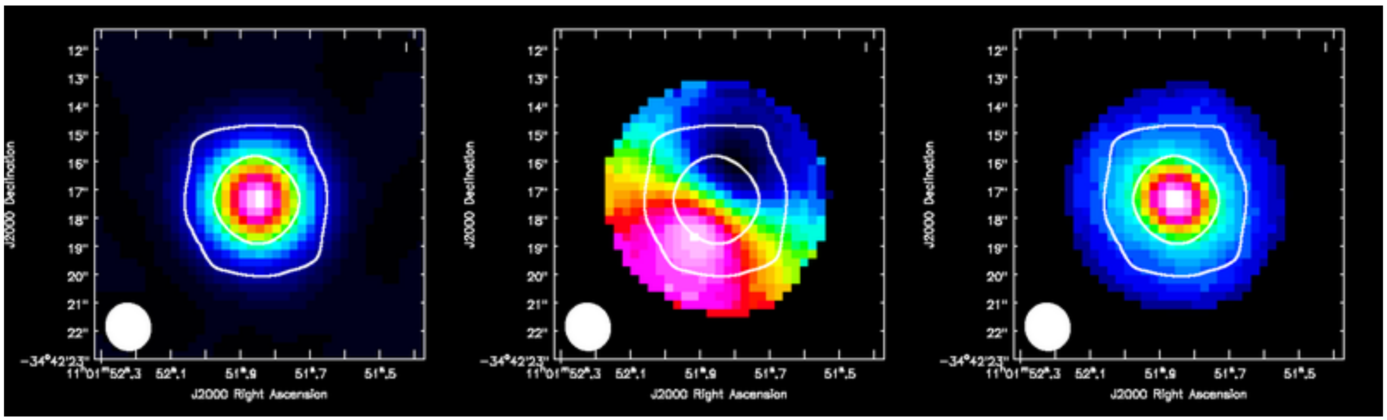
TW Hya CASA
Spectral Line Reduction Tutorial
Imaging and Analysis

Day 4, Wednesday September 9th 12:15



HCO+(4-3) moment maps of TW Hya

Tutors:
Katharine Johnston
Andy Biggs, Sandra Etoka, (Liz Humphreys),
John McKean, Rosita Paladino, Anita Richards and
Lorant Sjouwerman

CASA guides tutorial link

Will follow script given at link below, or on ERIS webpage, or available on data sticks:

www.mpia.de/~johnston/ERIS/TWHya_advanced_script.txt

Extended version of reduction can be found on the CASA guides website:
https://casaguides.nrao.edu

You can copy-paste these commands into CASA as we go along

Which data to use

- If you finished running the calibration script from T5 on Monday, you can use that.
- Or you can use the calibrated data tar file:

`TWHya_corrected.tgz`

You can untar/zip this using: `tar –xvzf FILENAME`

It should contain: `TWHydra_corrected.ms`

Average and split out the data for continuum imaging

This will speed up the clean task

```
os.system('rm -rf TWHydra_cont.ms*')
split(vis='TWHydra_corrected.ms',
      outputvis='TWHydra_cont.ms',
      spw='0~3:7~1273', width=30,
      datacolumn='data')
```

- When averaging your own data, remember not to over-average or you will get bandwidth smearing.
- Calculate the largest bandwidth you can safely average for your required field of view

Check which channels need flagged using plotms

```
plotms(vis='TWHydra_cont.ms', spw='0~3',
       xaxis='channel', yaxis='amp’,
       avgtime='1e8', avgscan=T,
       coloraxis='spw', iteraxis='spw',
       xselfscale=T)
```

Question: which channels need flagged?

Check which channels need flagged using plotms

```
plotms(vis='TWHydra_cont.ms', spw='0~3',
       xaxis='channel', yaxis='amp',
       avgtime='1e8', avgscan=T,
       coloraxis='spw', iteraxis='spw',
       xselfscale=T)
```

Question: which channels need flagged?

Answer:

spw 0:18, 2:23~24 and the end of spw 3

Flag these line/bad channels in continuum data

```
flagdata(vis='TWHydra_cont.ms', mode='manual',
         spw='0:18~18, 2:23~24, 3:33~42')
```

To do: check the flagging worked using plotms again

Estimating the noise for imaging

Run listobs on data:

```
listobs('TWHydra_corrected.ms',
        listfile='TWHydra_corrected.ms.listobs')
```

Can estimate total time on source using script here (or available on data sticks):

www.mpia.de/~johnston/ERIS/time_on_source.py

What is the time on source: ?

```
execfile('time_on_source.py')
```

Estimating the noise for imaging

Measure total time on source using script:

```
time_on_source.py
```

Time on source ~2.4 hr

Can then use the ALMA sensitivity calculator to determine the expected noise (if have internet):

https://almascience.nrao.edu/proposing/sensitivity-calculator

Need: Declination, Obs. frequency, bandwidth of continuum, number of antennas, time on source

Estimating the noise for imaging

Use the ALMA sensitivity calculator to determine the expected noise (if have internet connection):

https://almascience.nrao.edu/proposing/sensitivity-calculator

Need: Declination (-35deg), Obs. Frequency (~350GHz), bandwidth of continuum (3x0.46875GHz), number of antennas (8), time on source (2.4hr)

Expected sensitivity = 0.176 mJy/beam

To do:

Determine the sensitivity for the line observations for 0.32 km/s channels

(Answer: ~11 mJy/beam)

Continuum imaging

```
os.system('rm -rf TWHydra_contall.*')
clean(vis='TWHydra_cont.ms',
      imagename='TWHydra_contall',
      mode='mfs', imagermode='csclean',
      imsize=100, cell=['0.3arcsec'], spw='',
      weighting='briggs', robust=0.5,
      mask='', usescratch=False, interactive=T,
      threshold=‘0.6mJy', niter=10000)
```

Continuum imaging

```
os.system('rm -rf TWHydra_contall.*')
clean(vis='TWHydra_cont.ms',
      imagename='TWHydra_contall',
      mode='mfs', imagermode='csclean',
      imsize=100, cell=['0.3arcsec'], spw='',
      weighting='briggs', robust=0.5,
      mask='', usescratch=False, interactive=T,
      threshold='0.6mJy', niter=10000)
```

mode='mfs' – use multi-frequency synthesis algorithm for continuum imaging

imagermode='csclean' – Cotton-Schwab clean

cell=['0.3arcsec'] – the synthesised beam at 350GHz should be ~1.7”, want 4-5 pixels across the beam

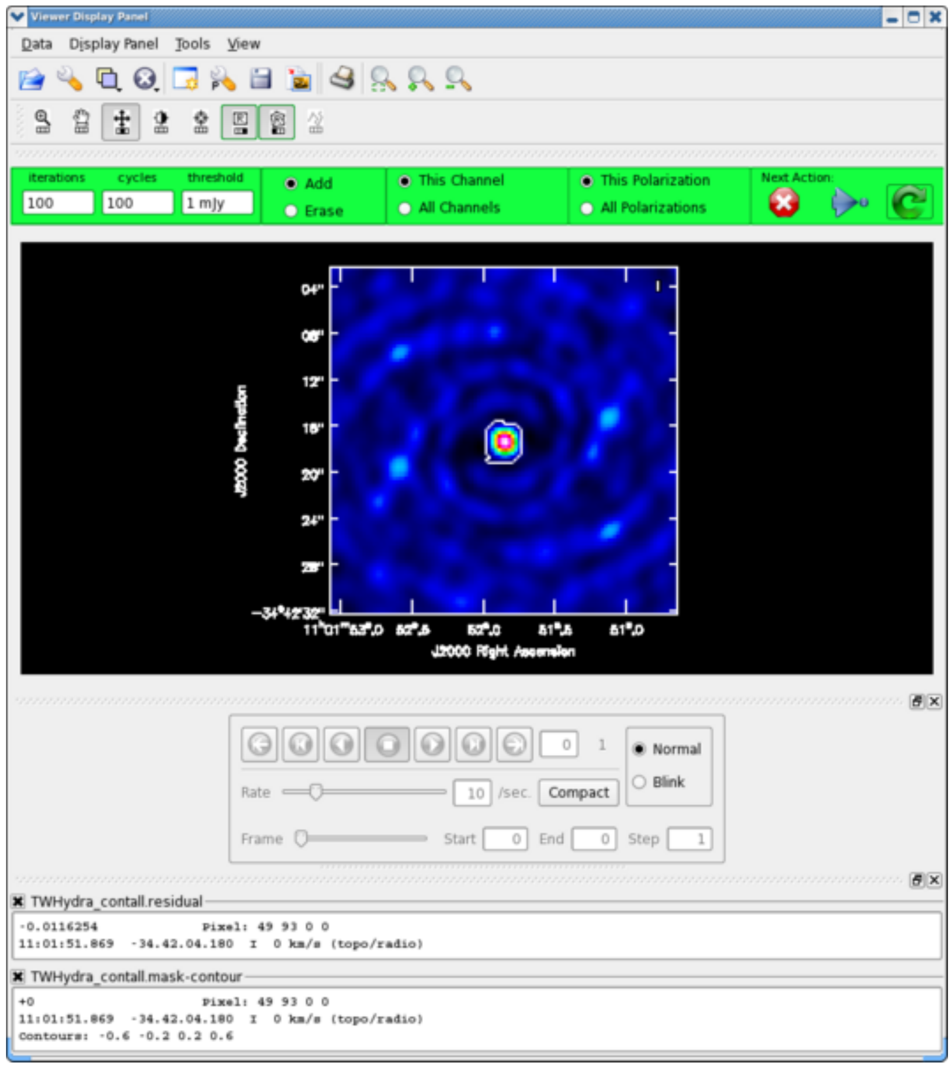
imagesize=100 – primary beam is ~18”, so 0.3”x100 = 30” will cover it

weighting='briggs', robust=0.5 – how you weight the data in uv-space (this is a good compromise)

threshold='0.6mJy' — a threshold for cleaning ~3 x noise

niter=10000 – enough iterations so you reach the threshold first

Continuum imaging



Split out the line data

For the $^{12}CO(3\text{-}2)$:

```
os.system('rm -rf TWHydra_CO3_2.ms*')
split(vis='TWHydra_corrected.ms’,
      outputvis='TWHydra_CO3_2.ms',
      datacolumn=‘data', spw='2')
```

For the HCO^+:

```
os.system('rm -rf TWHydra_HCOplus.ms*')
split(vis='TWHydra_corrected.ms',
      outputvis='TWHydra_HCOplus.ms',
      datacolumn=’data', spw='0')
```

Continuum subtraction

To do: Find the line free channels for both datasets using plotms, e.g. for $^{12}CO(3\text{-}2)$:

```
plotms(vis='TWHydra_CO3_2.ms',
       spw='0',xaxis='channel', yaxis='amp',
       avgtime='1e8', avgscan=T, coloraxis='spw',
       plotfile='CO3_2_channel.png')
```

Continuum subtraction

To do: Find the line free channels for both line datasets using task **plotms**, e.g. for $^{12}CO(3\text{-}2)$:

```
plotms(vis='TWHydra_CO3_2.ms',
       spw='0',xaxis='channel', yaxis='amp',
       avgtime='1e8', avgscan=T, coloraxis='spw',
       plotfile='CO3_2_channel.png')
```

Then subtract them using task **uvcontsub**, e.g.

```
uvcontsub(vis='TWHydra_CO3_2.ms',
          fitorder=1,
          fitspw='0:0:6~630,0:800~1265')
```

Can also use task **imcontsub** to subtract in image plane.

Continuum subtraction

To do:

Plot the continuum subtracted data (*.ms.contsub) as a function of velocity using task plotms.

Parameters you'll need to set:

avgtime and **avgscan** (average over all time and scans)

transform and **freqframe** (transform to LSR velocity frame)

restfreq CO(3-2): 345.79599GHz and HCO+(4-3) 356.7342GHz

Continuum subtraction

To do:

Plot the continuum subtracted data (*.ms.contsub) as a function of velocity using task plotms.

Which would look like this:

```
plotms(vis='TWHydra_CO3_2.ms.contsub',
        xaxis='velocity', yaxis='amp’, avgtime='1e8',
        avgscan=T, transform=T, freqframe='LSRK’,
        restfreq='345.79599GHz', plotrange=[-20,23,0,0],
        plotfile='CO3_2_vel.png')
```

Questions:

Which reference frame would the data be in if freqframe was not set?

Which velocities should we image between? (including line-free channels)

Continuum subtraction

To do:
Plot the continuum subtracted data (*.ms.contsub) as a function of velocity using task plotms.

Which would look like this:

```
plotms(vis='TWHydra_CO3_2.ms.contsub',
       xaxis='velocity', yaxis='amp', avgtime='1e8',
       avgscan=T, transform=T, freqframe='LSRK',
       restfreq='345.79599GHz', plotrange=[-20,23,0,0],
       plotfile='CO3_2_vel.png')
```

Questions:
Which reference frame would the data be in if freqframe was not set?
Which velocities should we image between? (including line-free channels)

Answers:
Velocity reference frame: TOPO; Velocity range: -4 to +8 km/s

$^{12}CO(3-2)$ imaging

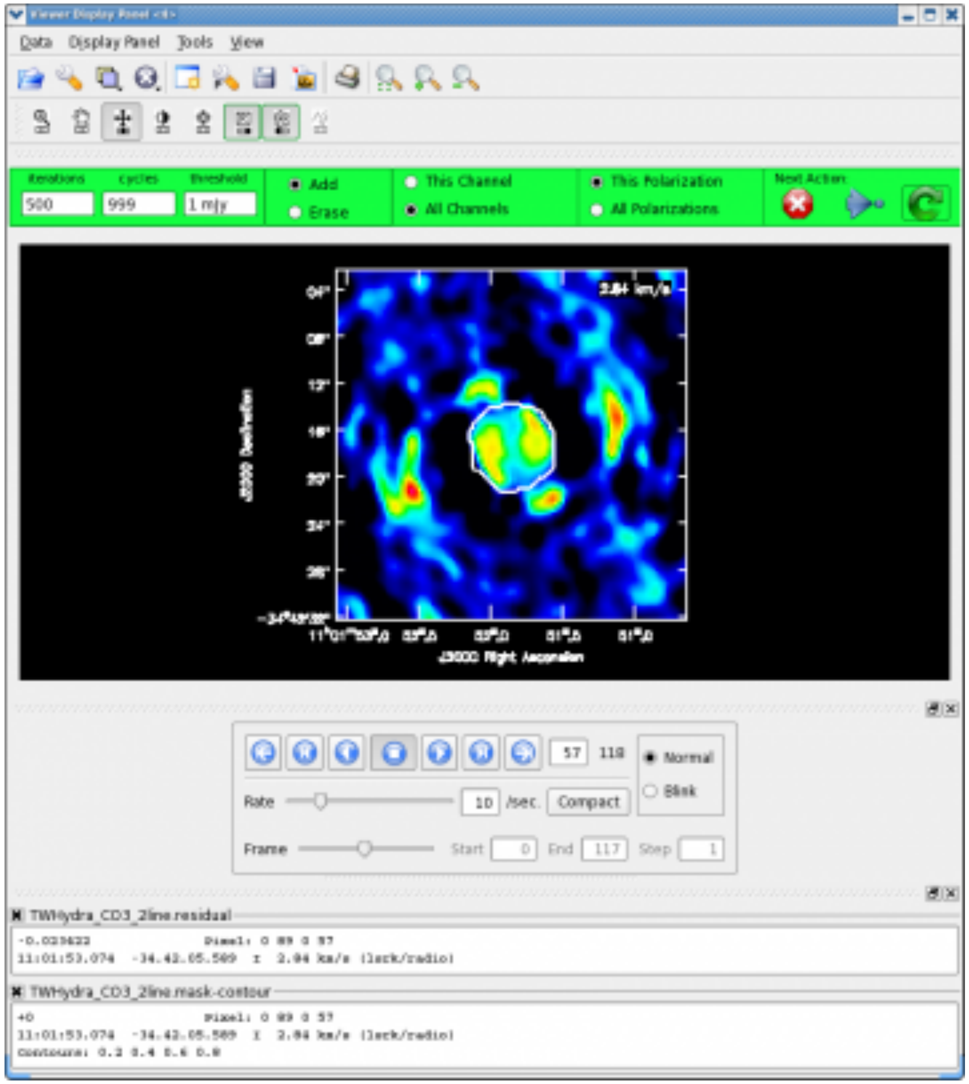
```
os.system('rm -rf TWHydra_CO3_2line.*')
clean(vis='TWHydra_CO3_2.ms.contsub',
      imagename='TWHydra_CO3_2line', imagermode='csclean',
      spw='', imsize=100, cell=['0.3arcsec'],
      mode='velocity', start='-4km/s', width='0.32km/s',
      nchan=40, restfreq='345.79599GHz', outframe='LSRK',
      weighting='briggs', robust=0.5,
      mask='', usescratch=False, interactive=T,
      threshold='33mJy', niter=100000)
```

The velocity resolution (3 x 122 kHz or 0.106km/s = 0.317 km/s)

Enough channels to get to +8.48 km/s

Approx. x3 expected noise

$^{12}CO(3\text{-}2)$ imaging



HCO^+(4-3) imaging

To do: (if you have time)

make an image of HCO^+(4-3)

Rest frequency of HCO+(4-3): 356.7342GHz

Image Analysis

To do: determine the restoring synthesised beam sizes for the two images using the task **imhead**, e.g.

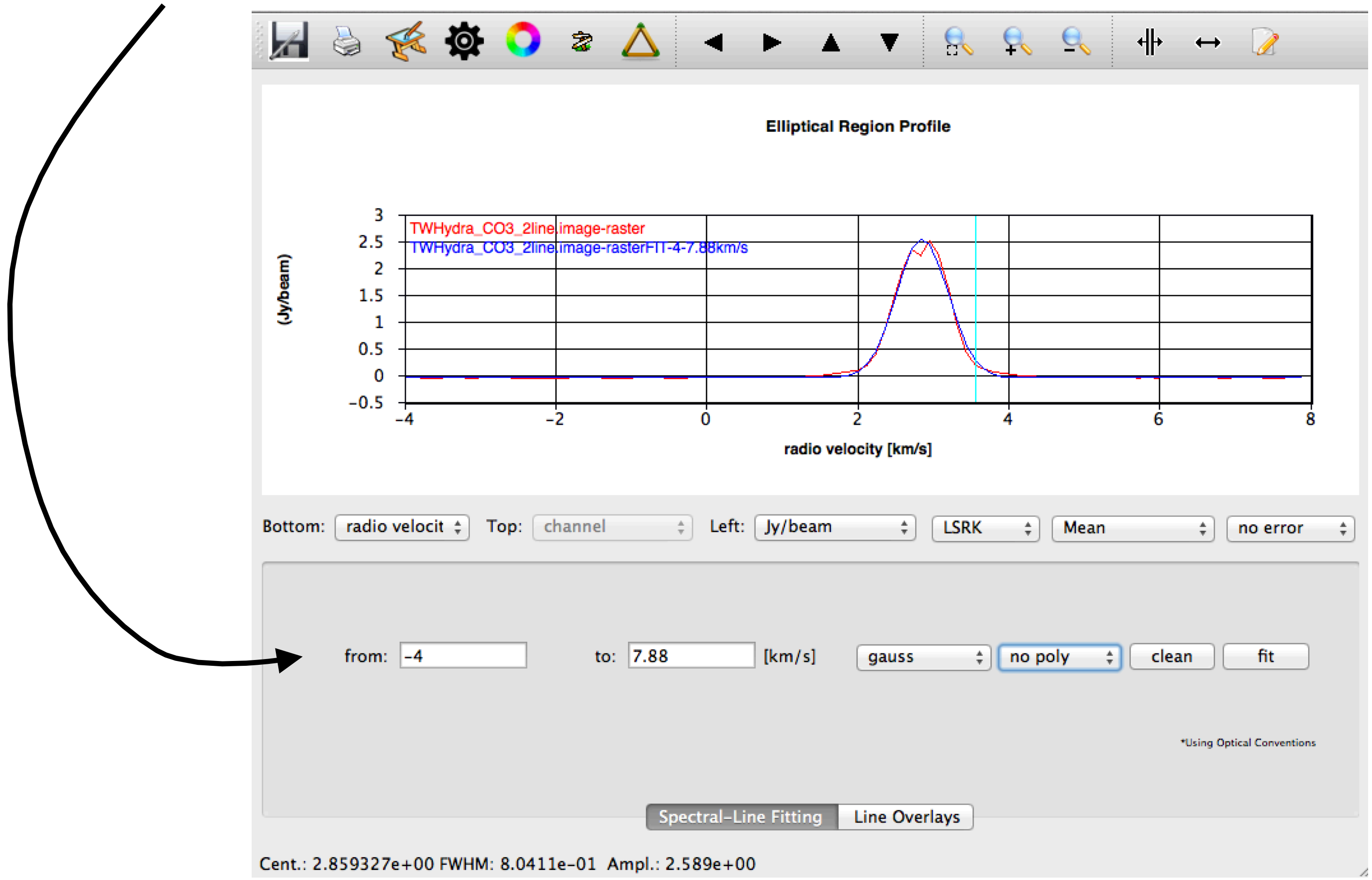
```
imhead("TWHydra_CO3_2line.image")
```

Save line spectra to file using spectral profile tool

- Open line images in viewer, e.g.
  `viewer("TWHydra_CO3_2line.image")`
- Use the Spectral Profile Tool (icon that looks like window with red line in it) to make a spectrum
- Save the spectrum to file (for creating a figure using your favourite software, e.g. python + matplotlib)
- Save elliptical region file using menu -> View -> Regions -> File tab

Line fitting in CASA

Can fit lines with Gaussians using “Spectral-Line fitting” tab in Spectral Profile Tool window



Line fitting in CASA

Can also do fit using command line, e.g.

```
specfit(imagename='TWHydra_CO3_2line.image',
        region='spectrum_region.crtf', poly=-1,
        logresults=True)
```

To do:

Check you get similar results to interactive fitting

Note: you'll need to save a region in the viewer first

Moment maps of line emission

Zero moment map = integrated flux map

First moment map = intensity-weighted velocity

Second moment map = intensity-weighted
velocity dispersion
about the mean

These are made using task **immoments**

RMS noise and spectral extent

First estimate the spectral extent of the $^{12}CO(3\text{-}2)$ emission using the viewer:

```
viewer("TWHydra_CO3_2line.image")
```

To do:

- Estimate the noise in the image by drawing a region in a line-free channel and double clicking in it (results appear in CASA terminal)
- Open the same image as a contour map in the same viewer
- Determine the range of channels which have flux > 5 sigma

RMS noise and spectral extent

RMS noise can also be determined using the task imstat for line-free channels, e.g.

```
results = imstat("TWHydra_CO3_2line.image",
                 chans=“7")
print results
print "  s.d. ", results['sigma']
print "  RMS ", results['rms']
```

Moment maps of line emission

To do:

Make zero moment maps for both lines using task immoments, e.g.

```
os.system('rm -rf TWHydra_CO3_2line.image.mom0')
immoments(imagename='TWHydra_CO3_2line.image',
          outfile='TWHydra_CO3_2line.image.mom0',
          moments=[0], chans='13~32')
```

Your range here

Viewing the moment maps using task imview

```
imview( raster= {'file':'TWHydra_CO3_2line.image.mom0',
                 'range':[-1.,10.]},
        contour={'file':'TWHydra_contall.image',
                 'base':0, 'unit':0.0025,
                 'levels':[3,100]} )
```

Making the first and second moment maps

First moment:

```
os.system('rm -rf TWHydra_CO3_2line.image.mom1')
immoments(imagename='TWHydra_CO3_2line.image',moments=[1],
          outfile='TWHydra_CO3_2line.image.mom1',
          chans='13~32',includepix=[0.5,100])
```

Second moment:

```
os.system('rm -rf TWHydra_CO3_2line.image.mom2')
immoments(imagename='TWHydra_CO3_2line.image',moments=[2],
          outfile='TWHydra_CO3_2line.image.mom2',
          chans='13~32',includepix=[0.5,100])
```

4 or 5 sigma from noise in channel with brightest emission

Viewing and exporting the moment maps

```
imview( raster=[ {'file':'TWHydra_CO3_2line.image.mom0'},
                 {'file':'TWHydra_CO3_2line.image.mom1'},
                 {'file':'TWHydra_CO3_2line.image.mom2'} ],
        contour={'file':'TWHydra_contall.image',
                 'base':0, 'unit':0.0025,
                 'levels':[3,100]} )
```

To do: Export your images using task exportfits, e.g.

```
os.system('rm -rf TWHydra_CO3_2line.image.fits')
exportfits(imagename='TWHydra_CO3_2line.image',
           fitsimage='TWHydra_CO3_2line.image.fits')
```

To do: Check what parameters **velocity=True** and **dropstokes=True** do

Primary beam corrections

- Without correction for the primary beam response (default), images should have roughly constant noise across them…
- …but the flux is incorrect everywhere except the field centre
- To measure fluxes in your images, make sure to correct for the primary beam response first!

Primary beam corrections

You can use the task impbcor:

```
impbcor(imagename='TWHydra_contall.image',
       pbimage='TWHydra_contall.flux’,
       mode=‘divide’,
       outfile='TWHydra_contall.pbcor')
```

Fitting a gaussian to the continuum using task imfit

Fit the continuum emission with a 2D gaussian:

```
imfit(imagename="TWHydra_contall.pbcor",
      box="40,40,60,60”, logfile = "contin_fit.log",
      residual="TWHydra_contall.fitresid")
```

To do:

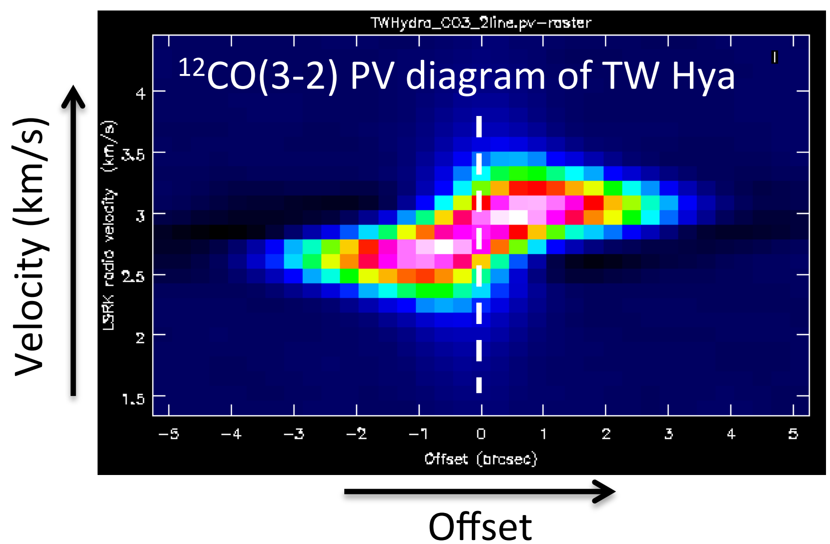
- Check the residual image to make sure the fit was good
- Look at the log file and determine the integrated flux and deconvolved size

Making position-velocity diagrams in viewer and using task impv

- Open one of the image cubes in the viewer
- Click on the P/V tool button 
- Draw a slice across the source (blue to red shifted)
- Go to menu => view => Regions => pV tab
- Click “Generate P/V”
- Change the averaging width and generate again
- Save the image
- (Note down the position angle!)

Making position-velocity diagrams in viewer and using task impv

If you had the full spectral resolution dataset, your pv plot would look like this:



Making position-velocity diagrams in viewer and using task impv

Can also generate pv diagrams using the task impv, e.g.

```
os.system('rm -rf TWHydra_CO3_2line.image.pv')
impv(imagename='TWHydra_CO3_2line.image',
               mode='length', center=[50,49],
               length='10arcsec', width='8arcsec',
               pa='-35deg', chans='12~30',
               outfile='TWHydra_CO3_2line.image.pv')
```

Peak pixel in mom0 map

Position angle determined above (could also fit mom0 emission)

Reprojecting an image using task imregrid

For example, to reproject to Galactic coordinates:

```
imregrid(imagename='TWHydra_CO3_2line.image',
         template='GALACTIC',
         output='TWHydra_CO3_2line.Galactic')
```

Or to reproject to another image header (only example!!):

```
regrid_dict = imregrid(imagename="target.image",
                       template="get")
imregrid(imagename="input.image",
         output="output.image",
         template=regrid_dict)
```

parameters in blue are not real images, just example entries

More analysis tasks…

Can be found by typing "tasklist" in CASA:

```
Analysis
------------------
imcollapse              imsmooth
imcontsub               imstat
imfit                   imsubimage
imhead                  imtrans
immath                  imval
immoments               listvis
impbcor                 rmfit
impv                    slsearch
imrebin                 specsmooth
imreframe               splattotable
imregrid
```

More analysis tasks…

The CASA **toolkit** (from which the CASA tasks are built) can also be used, but is more advanced:

http://casa.nrao.edu/docs/CasaRef/CasaRef.html