Image Analysis with CASA

Jinshi Sai
Postdoc Fellow (ASIAA, TW-ARC)

ALMA Imaging Workshop@ASIAA, Feb 13–14, 2025

Table of Contents

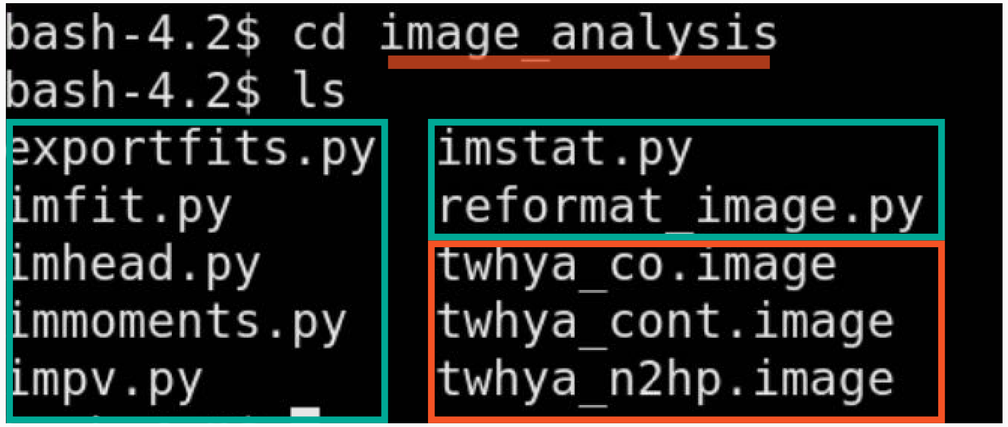
- Introduction to Image Analysis with CASA
  - CASA Image format/Checking image header
- Statistics/Reformat of images
- Continuum analysis
  - Two dimensional Gaussian fit
- Line analysis
  - Moment maps
  - Position-velocity (PV) diagrams

*Brief tutorial is provided by NRAO:
https://casaguides.nrao.edu/index.php/First_Look_at_Image_Analysis_CASA_6.6.1

Table of Contents

- Introduction to Image Analysis with CASA
  - CASA Image format/Checking image header
- Statistics/Reformat of images
- Continuum analysis
  - Two dimensional Gaussian fit
- Line analysis
  - Moment maps
  - Position-velocity (PV) diagrams

Some examples are here!



*Brief tutorial is provided by NRAO:
https://casaguides.nrao.edu/index.php/First_Look_at_Image_Analysis_CASA_6.6.1

Table of Contents

*Brief tutorial is provided by NRAO:
https://casaguides.nrao.edu/index.php/First_Look_at_Image_Analysis_CASA_6.6.1

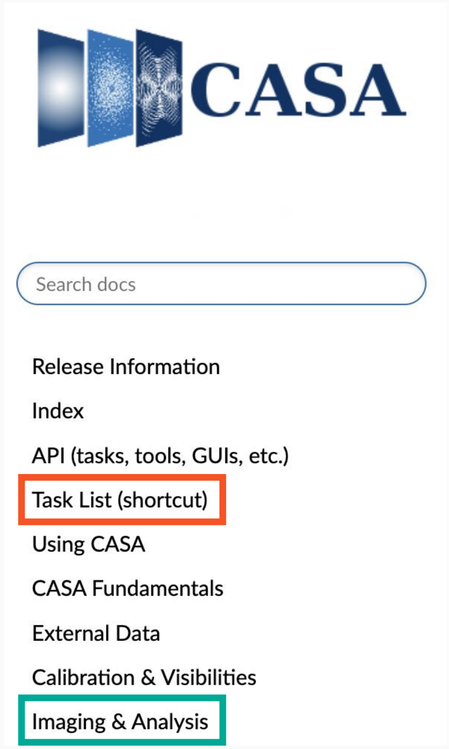
Image Analysis with CASA

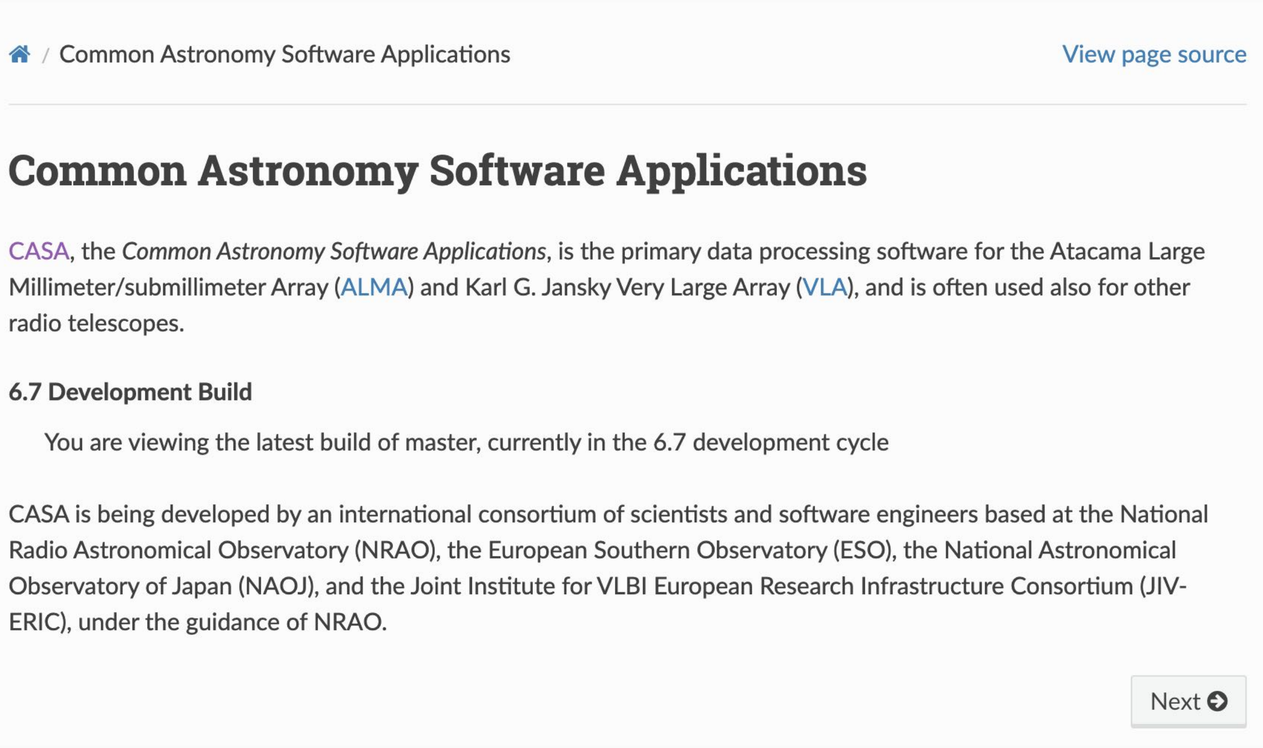
- CASA provides various command-line tasks for image analyses
- Advantage is you can keep log, be more exact in numbers, easily redo analyses, and execute many tasks once running scripts

Google “CASA docs” or https://casadocs.readthedocs.io/en/latest/imaging.html

Task list

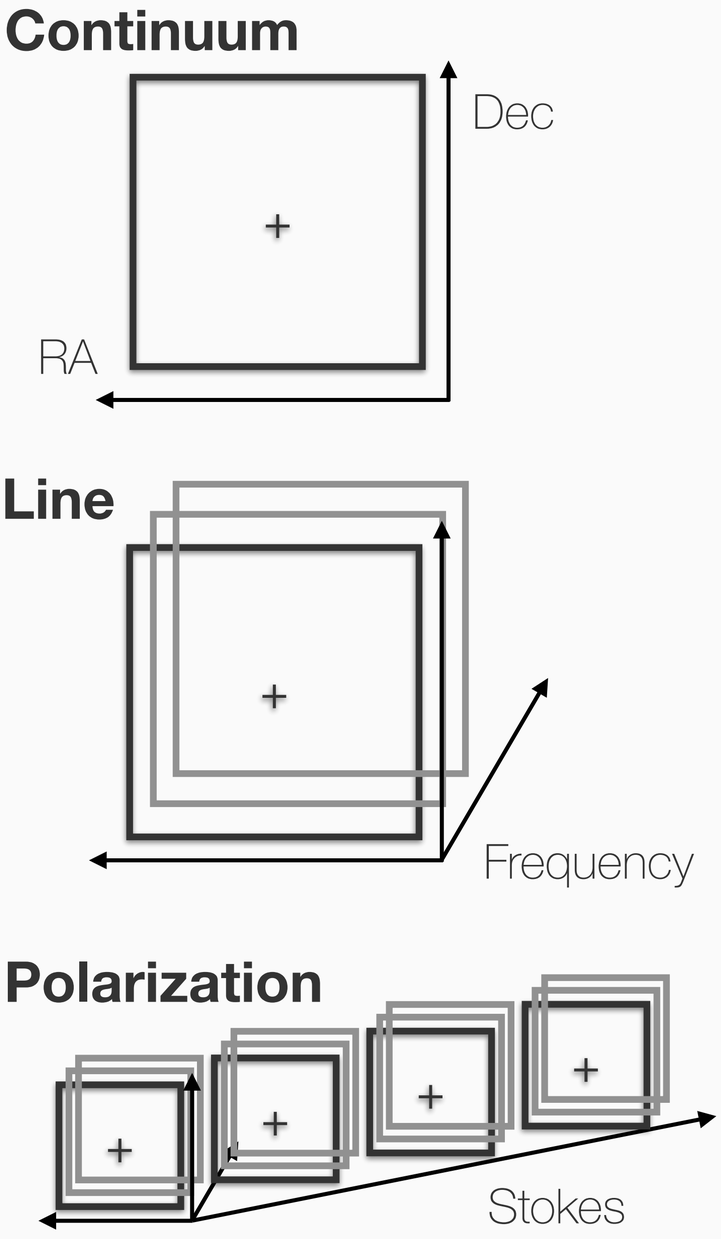
Instruction for imaging & analysis





CASA Image Format

- Meta data are contained in Header
  - Information of observations—observing date, target name, etc
  - Coordinate systems—absolute & pixel coordinates of images
  - Intensity units, restoring beam size, spectral & stokes parameters
- CASA images typically have four axes
  - Spatial coordinate (RA/GLON, DEC/GLAT), stokes, and frequency
  - Non-polarization data have only Stokes I
  - Continuum data have only single frequency point



Checking Image Header

Task-imhead

- Meta data of images are contained in Header
- Usage:

  imhead(image_file)

```
CASA <1>: f = 'twhya_cont.image'
     ...:
     ...: # check header
     ...: imhead(f)
Out[1]:
{'axisnames': array(['Right Ascension', 'Declination', 'Stokes', 'Frequency'],
       dtype='<U15'),
 'axisunits': array(['rad', 'rad', '', 'Hz'], dtype='<U3'),
 'defaultmask': 'mask0',
 'hasmask': True,
 'imagetype': 'Intensity',
 'incr': array([-4.84813681e-07,  4.84813681e-07,  1.00000000e+00,  2.34445115
,
 'masks': array(['mask0'], dtype='<U5'),
 'messages': array([], dtype='<U5'),
 'ndim': 4,
 'refpix': array([125., 125.,   0.,   0.]),
 'refval': array([ 2.88792330e+00, -6.05713443e-01,  1.00000000e+00,  3.726369
]),
 'restoringbeam': {'major': {'unit': 'arcsec', 'value': 0.6092361807823181},
  'minor': {'unit': 'arcsec', 'value': 0.45245257019996643},
  'positionangle': {'unit': 'deg', 'value': -51.57859802246094}},
 'shape': array([250, 250,   1,   1]),
 'tileshape': array([250, 250,   1,   1]),
 'unit': 'Jy/beam'}
```

Checking Image Header

Task imhead

- More information can be obtained with mode='list'

```
CASA <19>: imhead('twhya_n2hp.image', mode='list')
Out[19]:
{'beammajor': {'unit': 'arcsec', 'value': 0.6148935726253035},
 'beamminor': {'unit': 'arcsec', 'value': 0.4531960312291376},
 'beampa': {'unit': 'rad', 'value': -0.9142064866512125},
 'bunit': 'Jy/beam',
 'cdelt1': -4.84813681109536e-07,
 'cdelt2': 4.84813681109536e-07,
 'cdelt3': 1.0,
 'cdelt4': -621550.8096313477,
 'crpix1': 125.0,
 'crpix2': 125.0,
 'crpix3': 0.0,
 'crpix4': 0.0,
 'crval1': 2.887923300334642,
 'crval2': -0.6057134431858937,
 'crval3': 1.0,
 'crval4': 372672489999.99994,
 'ctype1': 'Right Ascension',
 'ctype2': 'Declination',
 'ctype3': 'Stokes',
 'ctype4': 'Frequency',
 'cunit1': 'rad',
 'cunit2': 'rad',
 'cunit3': '',
 'cunit4': 'Hz',
 'datamax': 0.29089340567588806,
 'datamin': -0.11979858577251434,
 'date-obs': '2012/11/19/07:56:26.544000',
 'equinox': 'J2000',
 'imtype': 'Intensity',
 'masks': array(['mask0'], dtype='<U5'),
 'maxpixpos': array([122, 135,   0,   5]),
 'maxpos': '11:01:51.820 -34.42.16.366 I 3.726694e+11Hz',
 'minpixpos': array([142, 139,   0,   6]),
 'minpos': '11:01:51.658 -34.42.15.966 I 3.726688e+11Hz',
 'object': 'TW Hya',
 'observer': 'cqi',
 'projection': 'SIN',
 'reffreqtype': 'LSRK',
 'restfreq': array([3.7267249e+11]),
 'shape': array([250, 250,   1,  15]),
 'telescope': 'ALMA'}
```

Checking Image Header

`Task imhead`

- Various modes

From CASA docs

Values	**Description**
add	Add a new metadata value to the image.
del	Delete a key or reset its value to a fiducial value if possible. Ignores *hdvalue* parameter.
get	Return the specified keyword value. Ignores the *hdvalue* parameter.
history	Log image history. Ignores the *hdkey* and *hdvalue* parameters.
list	Show supported keywords and their values. Ignores the *hdkey* and *hdvalue* parameters.
put	Modify the value associated with the keyword using the specified value.
summary	Log a summary of the image and return a dictionary of various metadata values. Ignores the *hdkey* and *hdvalue* parameters.

```
CASA <6>: f = 'twhya_n2hp.image/'

CASA <7>: imhead(f, mode = 'get', hdkey = 'restfreq')
Out[7]: {'unit': 'Hz', 'value': 372672489999.99994}
```

Checking Image Header

`Task imhead`

- Various modes

From CASA docs

Values	Description
add	Add a new metadata value to the image.
del	Delete a key or reset its value to a fiducial value if possible. Ignores *hdvalue* parameter.
get	Return the specified keyword value. Ignores the *hdvalue* parameter.
history	Log image history. Ignores the *hdkey* and *hdvalue* parameters.
list	Show supported keywords and their values. Ignores the *hdkey* and *hdvalue* parameters.
put	Modify the value associated with the keyword using the specified value.

```
CASA <21>: # modify header value
      ...: imhead('twhya_n2hp.image', mode = 'put', hdkey = 'observer', hdvalue = 'Me')

CASA <22>: # modify header value
      ...: imhead('twhya_n2hp.image', mode = 'get', hdkey = 'observer')
Out[22]: 'Me'
```

Tips for Starting up —before going into more analyses

- Type taskhelp to look into CASA tasks

```
CASA <58>: taskhelp
--------------------------------------------------------------------------------
CASA tasks
--------------------------------------------------------------------------------
> analysis
--------------------------------------------------------------------------------
    imbaseline : Image-based baseline subtraction for single-
    imcollapse : Collapse image along one axis, aggregating p
     imcontsub : Estimates and subtracts continuum emission f
         imdev : Create an image that can represent the stati
```

- Use help command to recall/learn usage of tasks

```
CASA <59>: help(imhead)
```

```
--------- parameter descriptions -----------------------------

imagename Input image cube.
          Default: none

             Example: imagename='ngc5921_task.image'
mode      Mode of operation.
          Default: summary
          Options: "add", "del", "get", "history", "list",
          "put", or "summary".
```

- Visit CASA docs to learn more

Table of Contents

Getting Image Statistics

Task_imstat

- Statistical information of image is calculated
  - Flux, maximum intensity, median, standard deviation, rms, etc.

- Computed information is returned as Python dictionary

```
CASA <9>: stat = imstat(imagename = 'twhya_cont.image')

CASA <10>: stat
Out[10]:
{'blc': array([0, 0, 0, 0]),
 'blcf': '11:01:52.810, -34.42.29.866, I, 3.72637e+11Hz',
 'flux': array([1.98294114]),
 'max': array([0.39978448]),
 'maxpos': array([120, 127,   0,   0]),
 'maxposf': '11:01:51.837, -34.42.17.166, I, 3.72637e+11Hz',
 'mean': array([0.00144245]),
 'medabsdevmed': array([0.00136992]),
 'median': array([3.91167487e-05]),
 'min': array([-0.00869298]),
 'minpos': array([ 85, 191,   0,   0]),
 'minposf': '11:01:52.120, -34.42.10.766, I, 3.72637e+11Hz',
 'npts': array([42937.]),
 'q1': array([-0.00133103]),
 'q3': array([0.00140875]),
 'quartile': array([0.00273978]),
 'rms': array([0.01665415]),
 'sigma': array([0.01659175]),
 'sum': array([61.93455052]),
 'sumsq': array([11.90902983]),
 'trc': array([249, 249,   0,   0]),
 'trcf': '11:01:50.790, -34.42.04.966, I, 3.72637e+11Hz'}
```

Getting Image Statistics

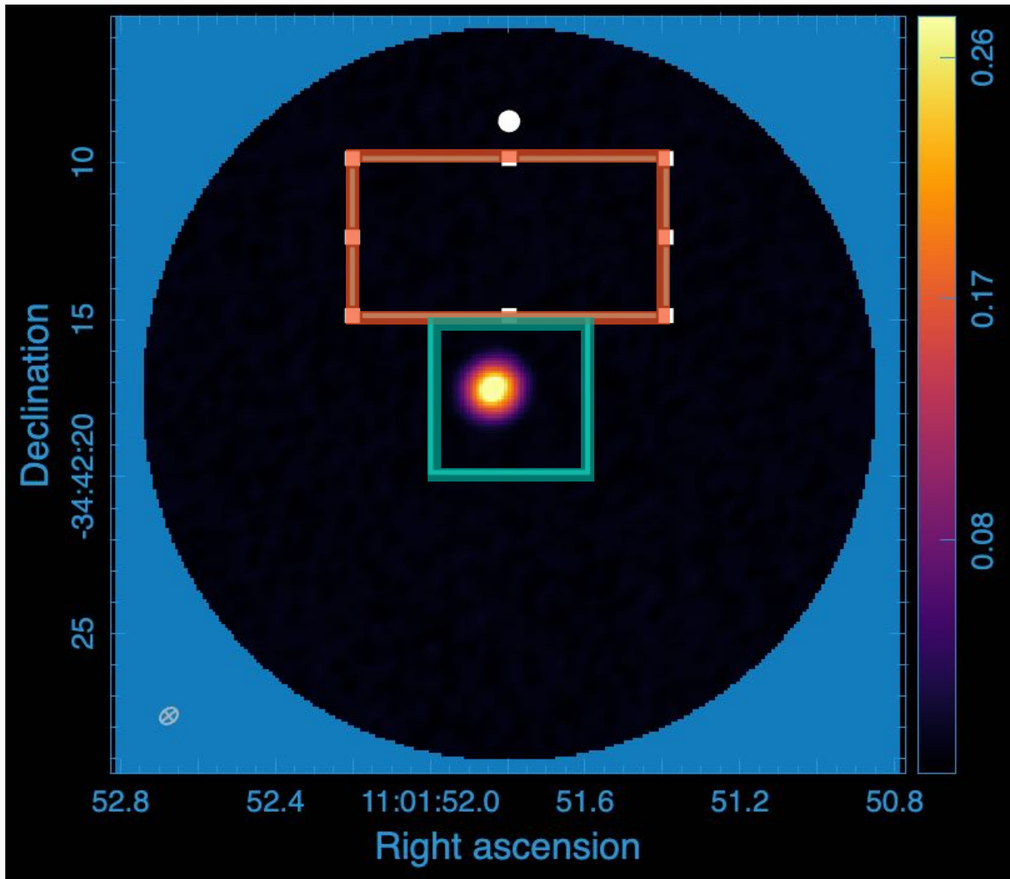
- Examples—selecting box region

```
CASA <6>: # statisitcs of off-source region
     ...: stat = imstat(
     ...:      imagename = 'twhya_cont.image',
     ...:      box = '75, 150, 175, 200',
     ...:      )
     ...: print('RMS: ', stat['rms'])
RMS:  [0.00199016]
```

```
CASA <5>: stat = imstat(
     ...:      imagename = 'twhya_cont.image',
     ...:      box = '100, 100, 150, 150',
     ...:      )
     ...: print('Flux density: ', stat['flux'])
Flux density:  [1.99766084]
```

- Use imhead to know pixel coordinates of image center

Pixel coordinates of bottom left and top right corners



Getting Image Statistics

- Examples—selecting channels

```
CASA <11>: # line
       ...: # statisitcs of line free channels
       ...: stat = imstat(
       ...:     imagename = 'twhya_n2hp.image',
       ...:     chans = '0~3; 9~14'
       ...:     )
       ...: print('Line RMS: ', stat['rms'])
Line RMS:  [0.02783996]
```

Selecting line-free channels

- Use CARTA to find line-detected/free channels

Reformat Images

- **imsubimage:** enables the user to extract a sub-image from a larger cube,
- **imtrans:** changes the axis order in an image,
- **imregrid:** sets the image onto a different spatial coordinate system or spectral grid,
- **imreframe:** changes the velocity system of an image
- **imrebin:** rebins an image in a spatial or spectral dimension
- **imcollapse:** collapses an image along an axis.

From CASA docs

Extracting Sub-region of Image

Task-imsubimage

```
# region to extract
region = 'rotbox[[11h01m51.837s, -034d42m17.21s], [6arcsec, 6arcsec], 0deg]'
# Refer to https://casaguides.nrao.edu/index.php/CASA_Region_Format for regio

# continuum
imsubimage(
    imagename = 'twhya_cont.image',
    region = region,
    outfile = 'twhya_cont_sub.image'
    )
```

Extracting Sub-region of Image

Task-imsubimage

Rotated box Box center Box width and rotation angle

```
# region to extract
region = 'rotbox[[11h01m51.837s, -034d42m17.21s], [6arcsec, 6arcsec], 0deg]'
# Refer to https://casaguides.nrao.edu/index.php/CASA_Region_Format for regio

# continuum
imsubimage(
    imagename = 'twhya_cont.image',
    region = region,
    outfile = 'twhya_cont_sub.image'
    )
```

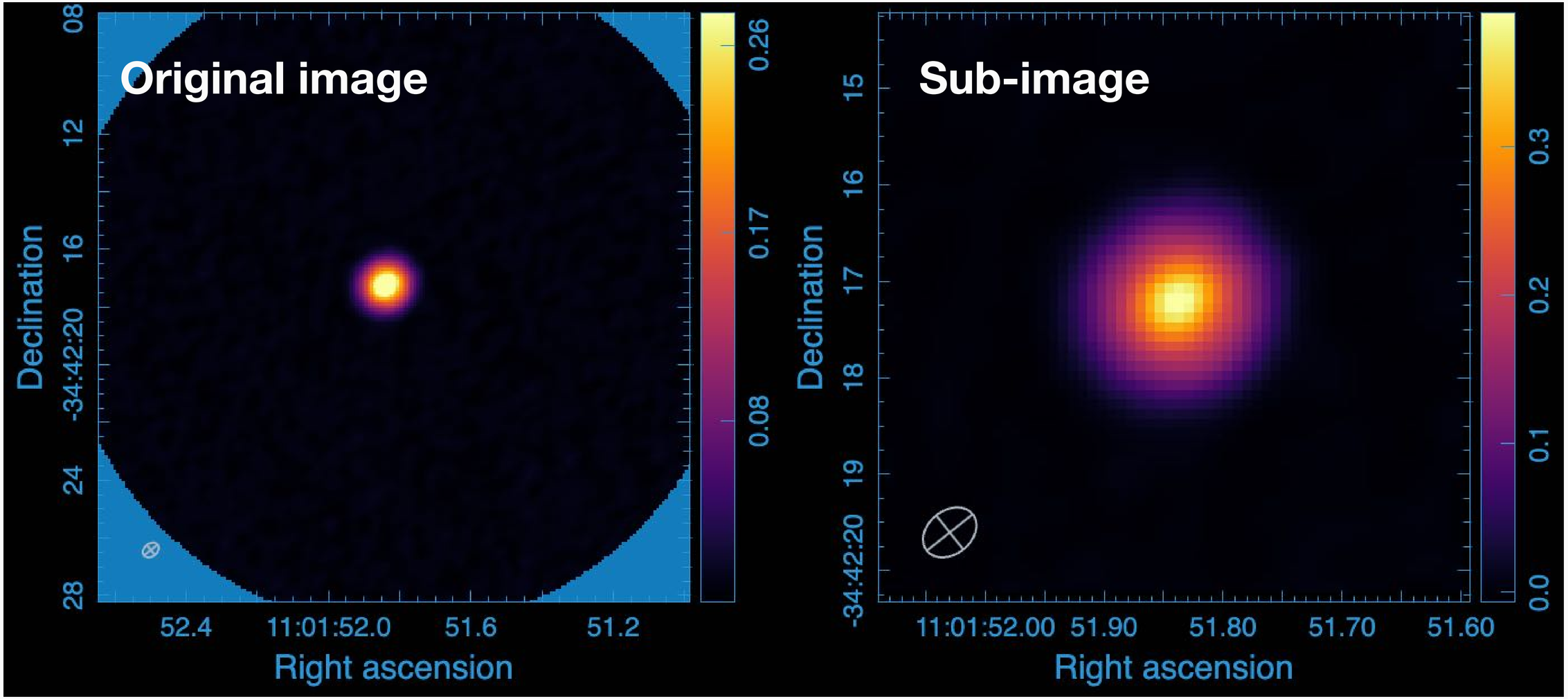
Extracting Sub-region of Image

Task–imsubimage

```
# region to extract
region = 'rotbox[[11h01m51.837s, -034d42m17.21s], [6arcsec, 6arcsec], 0deg]'
# Refer to https://casaguides.nrao.edu/index.php/CASA_Region_Format for regio

# continuum
imsubimage(
    imagename = 'twhya_cont.image',
    region = region,
    outfile = 'twhya_cont_sub.image'
    )
```

Extracting Sub-region of Image



CASA Region Format

- Region can be specified in flexible way
- `region` parameter is also used in many of other tasks introduced later
- Refer to https://casaguides.nrao.edu/index.php/CASA_Region_Format for more details

`region='box[[x1,y1], [x2,y2]]'`
is equivalent to `box='x1,y1, x2,y2'`

- **Rectangular box**; the two coordinates are two opposite corners:

```
box[[x1, y1], [x2, y2]]
```

- **Center box**; [x, y] define the center point of the box and [x_width, y_width] the width of the sides:

```
centerbox[[x, y], [x_width, y_width]]
```

- **Rotated box**; [x, y] define the center point of the box; [x_width, y_width] the width of the sides; rotang the rotation

```
rotbox[[x, y], [x_width, y_width], rotang]
```

- **Polygon**; there could be many [x, y] corners; note that the last point will connect with the first point to close the polygon:

```
poly[[x1, y1], [x2, y2], [x3, y3], ...]
```

- **Circle**; center of the circle [x,y], r is the radius:

```
circle[[x, y], r]
```

- **Annulus**; center of the circle is [x, y], [r1, r2] are inner and outer radii:

```
annulus[[x, y], [r1, r2]]
```

- **Ellipse**; center of the ellipse is [x, y]; semi-major and semi-minor axes are [bmaj, bmin]; position angle of the maj

```
ellipse[[x, y], [bmaj, bmin], pa]
```

Table of Contents

Two Dimensional Gaussian Fit

Task-imfit

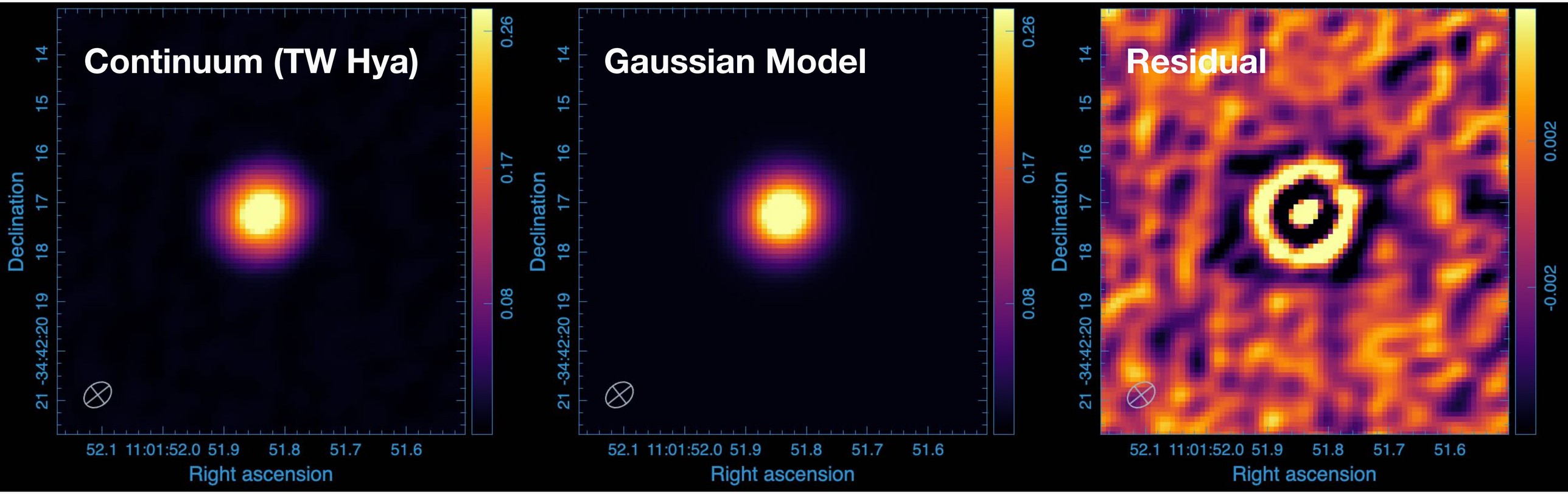
```
imfit(
    imagename = 'twhya_cont.image',
    box = "100, 100, 150, 150", # bottom-le
    logfile = 'Gaussian_fit.log',
    residual = 'Gaussian_fit.residual',
    model = 'Gaussian_fit.model'
    )
```

ESTIMATES

Initial estimates of fit parameters (peak intensity, peak x pixel coordinate, peak y pixel coordinate, major axis, minor axis, position angle) may be specified via an estimates text file. Each line of this file should contain a set of parameters for a single Gaussian. Optionally, some of these parameters can be fixed during the fit. The format of each line is: peak intensity, peak x-pixel value, peak y-pixel value, major axis, minor axis, position angle, fixed.

- box (or region)—Region included in fitting
  - For box, give pixel coordinates of bottom left and top right corners
- logfile—Name of log file. Recommended to give, otherwise no log will be saved.
- residual, model—Names of residual and model images if they need to be saved
- estimates—Initial guess of parameters. Must be provided if multi-Gaussian components fitting

Two Dimensional Gaussian Fit—Results



Two Dimensional Gaussian Fit—Results

Log file

```
****** Fit performed at Thu Feb 6 13:41:28 2025******

Input parameters ---
       --- imagename:           /almalustre/scratch/jsai/ImagingWS/working_image_imsize_1000/twhya
_cont.image
       --- region:
       --- channel:             0
       --- stokes:              I
       --- mask:
       --- include pixel range: []
       --- exclude pixel range: []
*** Details of fit for channel number 0
Number of pixels used in fit: 2601
Input and residual image statistics (to be used as a rough guide only as to goodness of fit)
       --- Standard deviation of input image: 0.0627873 Jy
       --- Standard deviation of residual image: 0.00424296 Jy
       --- RMS of input image: 0.0672026 Jy
       --- RMS of residual image: 0.00429986 Jy

Fit on twhya_cont.image component 0
Position ---
       --- ra:    11:01:51.83693 +/- 0.00048 s (0.00587 arcsec along great circle)
       --- dec: -034.42.17.20754 +/- 0.00579 arcsec
       --- ra:  119.953 +/- 0.059 pixels
       --- dec: 126.585 +/- 0.058 pixels

Image component size (convolved with beam) ---
       --- major axis FWHM:     1.281 +/- 0.014 arcsec
       --- minor axis FWHM:     1.214 +/- 0.013 arcsec
       --- position angle: 129.5 +/- 8.2 deg

Clean beam size ---
       --- major axis FWHM: 0.61 arcsec
       --- minor axis FWHM: 0.45 arcsec
       --- position angle: -51.58 deg
Image component size (deconvolved from beam) ---
       --- major axis FWHM:     1.128 +/- 0.026 arcsec
       --- minor axis FWHM:     1.126 +/- 0.026 arcsec
       --- position angle: 167 +/- 89 deg

Flux ---
       --- Integrated:   2.056 +/- 0.026 Jy
       --- Peak:         364.3 +/- 4.0 mJy/beam
       --- Polarization: I

Spectrum ---
     --- frequency:       372.6369 GHz (804.5162 um)

(END)
```

```
Fit on twhya_cont.image component 0
Position ---
       --- ra:    11:01:51.83693 +/- 0.00048 s (0.00587 ar
       --- dec: -034.42.17.20754 +/- 0.00579 arcsec
       --- ra:  119.953 +/- 0.059 pixels
       --- dec: 126.585 +/- 0.058 pixels

Image component size (convolved with beam) ---
       --- major axis FWHM:     1.281 +/- 0.014 arcsec
       --- minor axis FWHM:     1.214 +/- 0.013 arcsec
       --- position angle: 129.5 +/- 8.2 deg

Clean beam size ---
       --- major axis FWHM: 0.61 arcsec
       --- minor axis FWHM: 0.45 arcsec
       --- position angle: -51.58 deg
Image component size (deconvolved from beam) ---
       --- major axis FWHM:     1.128 +/- 0.026 arcsec
       --- minor axis FWHM:     1.126 +/- 0.026 arcsec
       --- position angle: 167 +/- 89 deg

Flux ---
       --- Integrated:   2.056 +/- 0.026 Jy
       --- Peak:         364.3 +/- 4.0 mJy/beam
       --- Polarization: I
```

Table of Contents

Moment Maps

- 0th moment: $M_0 = \sum_i I_i$ **Integrated intensity**
- *N*th moment: $M_N = \frac{\sum_i w_i I_i}{M_0}$
  - Usually integration is along velocity axis
  - 1st moment: $M_1 = \frac{\sum_i v_i I_i}{M_0}$ **Intensity-weighted mean velocity**
  - 2nd moment: $M_2 = \frac{\sum_i (v_i - M_1)^2 I_i}{M_0}$ **Intensity-weighted velocity dispersion**

Moment Maps

`Task-immoments`

Examples—moment 0 (integrated intensity)

```
CASA <16>: # Calculate moment maps
      ...: # moment 0 (integrated intensity)
      ...: immoments(
      ...:     imagename = 'twhya_n2hp.image',
      ...:     moments = [0],
      ...:     chans = '5~7',
      ...:     outfile = 'twhya_n2hp_mom0.image')
```

Or

```
chans = ('range=[2.5 km/s, 3.5 km/s], restfreq=372.672490 GHz'),
```

Velocity range **Rest frequency**; use imhead to know

- `imagename`—Input image
- `moments`—*i* of *i*-th moment to be computed. Multiple moments can be calculated at once: `moments=[0,1]`.
- `chans`—Channels over which moment maps are computed
- `outfile`—Output image name

Moment Maps

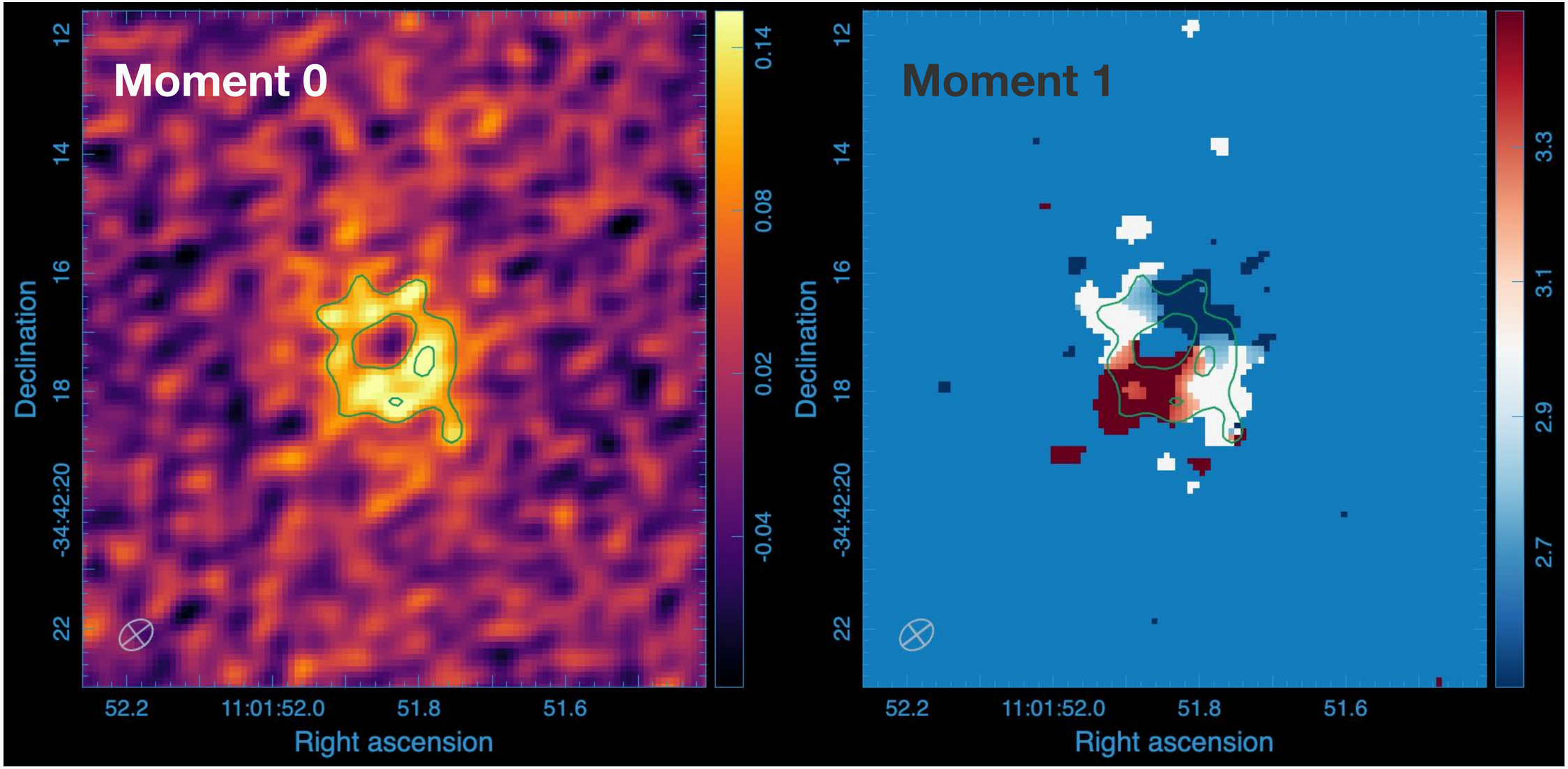
`Task-immoments`

Examples—moment 1 (mean velocity)

```
CASA <18>: # moment 1 (mean velocity)
       ...: rms = 27.8e-3 # Jy/beam
       ...: immoments(
       ...:     imagename = 'twhya_n2hp.image',
       ...:     moments = [1],
       ...:     includepix = [3*rms, 1000], # cutoff noise
       ...:     chans = ('range=[2.5 km/s, 3.5 km/s], rest
       ...:     outfile = 'twhya_n2hp_mom1.image',
       ...:     )
```

- `includepix/excludepix` —Pixel values included in/excluded from computing of moment maps

Moment Maps


Moment 0
Declination
12
14
16
18
-34:42:20
22
52.2
11:01:52.0
51.8
51.6
Right ascension
0.14
0.08
0.02
-0.04
Moment 1
Declination
12
14
16
18
-34:42:20
22
52.2
11:01:52.0
51.8
51.6
Right ascension
3.3
3.1
2.9
2.7

More Moments...

- moments = -1 - mean value of the spectrum
- moments = 0 - integrated value of the spectrum
- moments = 1 - intensity weighted coordinate; traditionally used to get “velocity fields”
- moments = 2 - intensity weighted dispersion of the coordinate; traditionally used to get “velocity dispersion”
- moments = 3 - median value of the spectrum
- moments = 4 - median coordinate
- moments = 5 - standard deviation about the mean of the spectrum
- moments = 6 - root mean square of the spectrum
- moments = 7 - absolute mean deviation of the spectrum
- moments = 8 - maximum value of the spectrum
- moments = 9 - coordinate of the maximum value of the spectrum
- moments = 10 - minimum value of the spectrum
- moments = 11 - coordinate of the minimum value of the spectrum

Refer to CASA docs for more details

From CASA docs

Position-velocity (PV) Diagrams

Task-impv

```
CASA <49>: impv(
      ...:     imagename = 'twhya_co.image',
      ...:     mode = 'length',
      ...:     center = '11h01m51.837s, -034d42m17.21s',
      ...:     length = '8arcsec',
      ...:     pa = '167deg',
      ...:     width = 1,
      ...:     outfile = 'twhya_co_pa167deg.pv',
      ...:     )
```

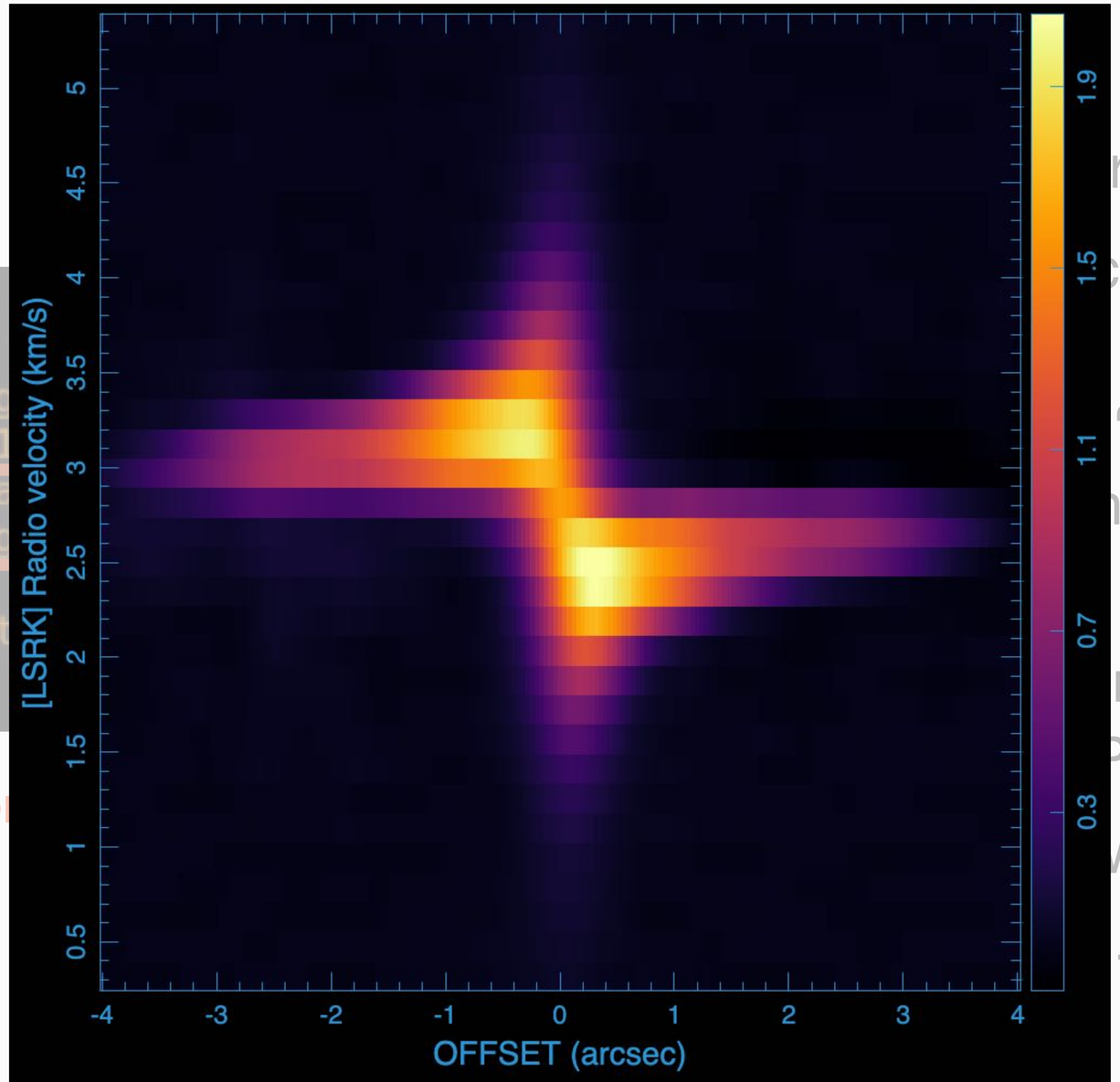
From imfit to continuum

- `imagename`—Input image
- `mode`—“coords” or “length”. If “coords”, use start and end values. If “length”, use center, length, and pa values.
- `center, length, pa`—Center, length and position angle of PV cut
- `width`—Width of PV cut
- `outfile`—Output image name

Position-velocity (PV) Diagrams

```
CASA <49>: impv(
     ...:       imagename =
     ...:       mode = 'leng
     ...:       center = '11
     ...:       length = '8a
     ...:       pa = '167dec
     ...:       width = 1,
     ...:       outfile = 't
     ...:       )
```

Fro



me—Input image

coords" or "length". If
use start and end
"length", use center,
nd pa values.

length, pa—
ngth and position
PV cut

Width of PV cut

—Output image name

Exporting/Importing

`Task-exportfits/importfits`

```
CASA <1>: # Export images to fits files
     ...: exportfits(
     ...:     imagename = 'twhya_cont.image',
     ...:     fitsimage = 'twhya_cont.fits')
     ...:
```

```
imporfits(
    imagename = 'output_casaimage.image',
    fitsimage = 'input_fits.fits')
```

Some more optional and useful bool parameters

- `velocity`—If true, output fits file has velocity axis instead of frequency axis
- `dropdeg`—If true, drop degenerated axes (stokes and/or frequency)
- `overwrite`—If true, overwrite output file when same file already exists

Any Questions?

Thank you!