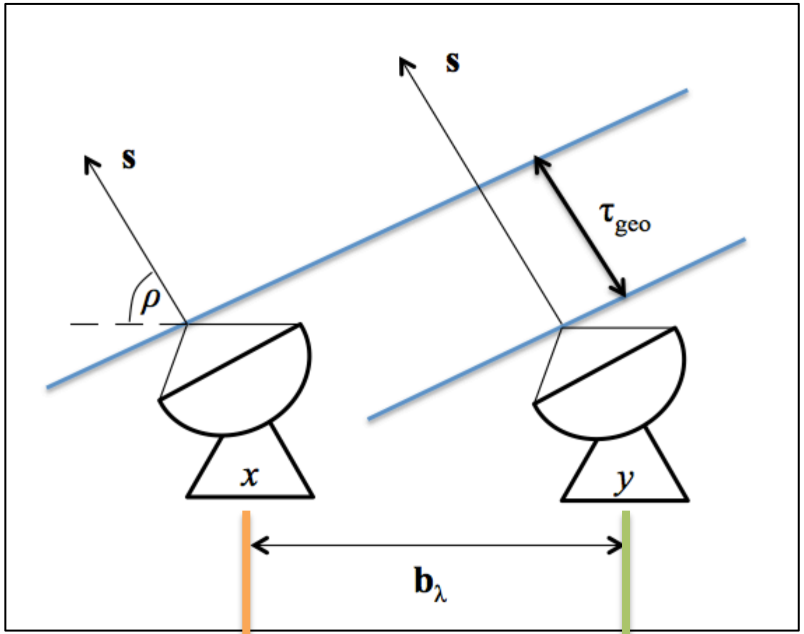
uv-coverage and imaging

The two element interferometer



Delay between wavefronts arriving at x then y:

$$\tau_{geo} = \frac{\mathbf{b}\cdot\mathbf{s}}{c} = \frac{bs\cos\rho}{c}$$

$$x(t) = \mathrm{v}_1 \cos 2\pi\nu t$$

$$y(t) = \mathrm{v}_2 \cos 2\nu\pi(t + \tau_{geo})$$

Receiver outputs

$$R_{x,y}(\tau_{geo}) = x \otimes y = X(\nu)Y^*(\nu)$$

Correlator's function

Do the maths

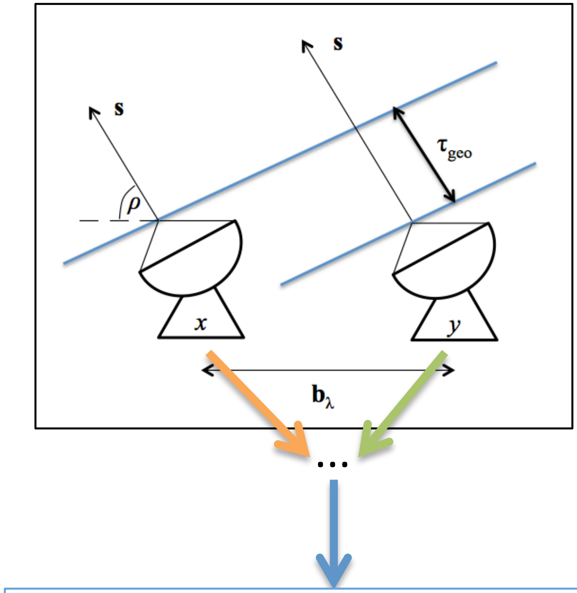
$$R_{x,y}(\tau_{geo}) = \mathrm{v}_1\mathrm{v}_2 \cos 2\pi\tau_{geo}$$

Correlator output

That’s all well and good but how does this

$$\boxed{R_{x,y}(\tau_{geo}) = \mathrm{v}_1 \mathrm{v}_2 \cos 2\pi\nu\tau_{geo}}$$

tell us anything about astronomical objects?



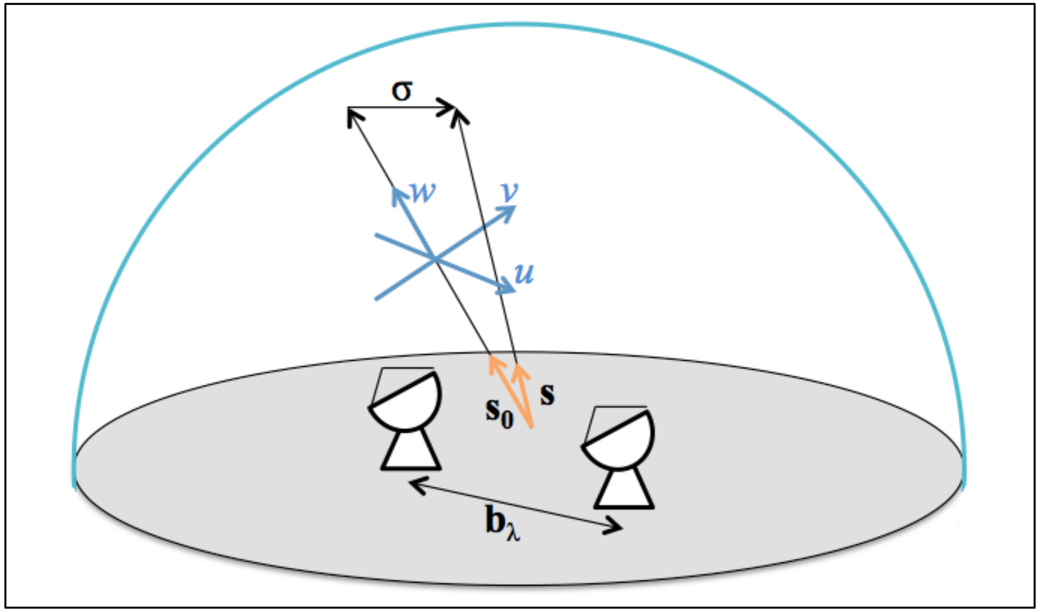
$$\boxed{R_{x,y}(\tau_{geo}) = \mathrm{v}_1 \mathrm{v}_2 \cos 2\pi\nu\tau_{geo}}$$

v_1 and v_2, the voltage outputs of x & y are directly related to:

- The brightness distribution, $I(\mathbf{s})$, of the astronomical object
- as seen over solid angle $\mathrm{d}\Omega$
- and $A(\mathbf{s})$ the area of the dish we use to observe it.

Leading to ...

$$\boxed{R_{x,y}(\tau_{geo}) = \Delta\nu \int A(\mathbf{s}) I(\mathbf{s}) \cos 2\pi \mathbf{b}_\lambda \cdot \mathbf{s} \ \mathrm{d}\Omega}$$



Adding in a bit more reality...

- The vector $\mathbf{s}$ is comprise of the addition of $\mathbf{s_0}$ and σ (so $\mathbf{s}=\mathbf{s_0}+\sigma$).
- We set τ_{geo} to zero with instrumental delays
- Meaning all delays in the data are from the vector σ

We then define the Complex Visibility as:

$$\boxed{V \equiv |V| e^{i\phi_V} = \int A(\sigma) I(\sigma)\, e^{-i2\pi \mathbf{b}_\lambda \cdot \sigma} \mathrm{d}\Omega}$$

which is rather nice as V is the Fourier transform of I.

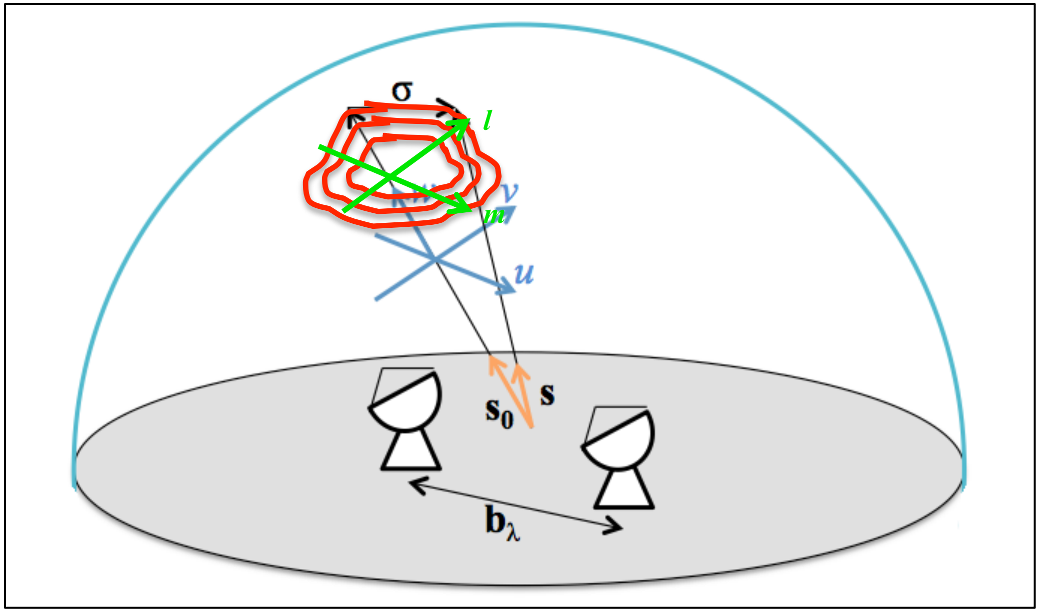
Relating the visibility equation to the correlator output gives

$$\boxed{R_{x,y} = \underset{\uparrow}{A_0}\left|V\right|\underset{\uparrow}{\Delta\nu}\cos(\underbrace{2\pi\nu\mathbf{b}_{\lambda}\cdot\mathbf{s}_0}_{\uparrow}-\phi_V)}$$

Known!*

* After proper calibration

A coordinate system for interferometry



We define u and v, as E-W and N-S positions w.r.t w axis which is parallel to $\mathbf{s}_{0}$.

l and m as direction cosines of $\mathbf{s}$ we can write the visibility equation as:

$$V(u,v) = \int A(l,m) I(l,m)\, e^{-i2\pi(ul+vm)} \frac{\mathrm{d}l\mathrm{d}m}{\sqrt{1-l^2-m^2}}$$

Given l and m are small the small angle approx applies and $V(u,v)$ becomes a direct Fourier transform of $I(x,y)$

Antenna spacing to *u, v, w*

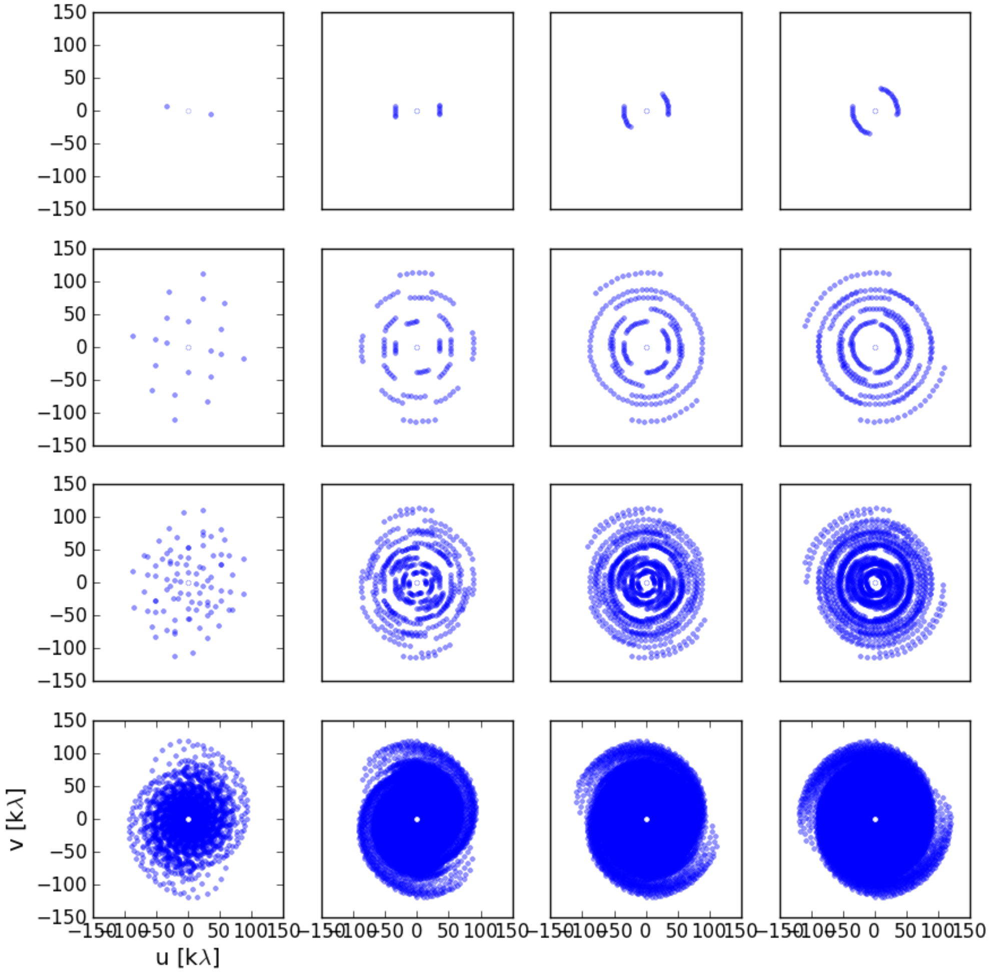
$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = \frac{1}{\lambda} \begin{pmatrix} \sin H_0 & \cos H_0 & 0 \\ -\sin\delta_0 \cos H_0 & \sin\delta_0 \sin H_0 & \cos\delta_0 \\ \cos\delta_0 \cos H_0 & -\cos\delta_0 \sin H_0 & \sin\delta_0 \end{pmatrix} \begin{pmatrix} L_X \\ L_Y \\ L_Z \end{pmatrix}$$

L_x, L_y, L_z = antenna coordinate differences
H_0, δ_0 = hour-angle and declination of the phase reference position
λ = central frequency of observation

For further reading see:

Thompson, Moran & Swenson *"Interferometry and Synthesis in Radio Astronomy"*
NRAO's: *"Synthesis Imaging in Radio Astronomy II"*

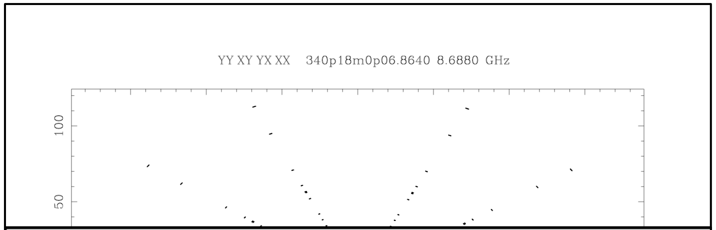
Filling the *uv*-plane



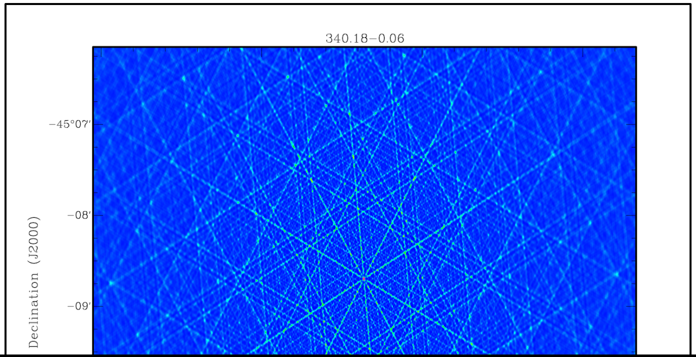
uv-coverage of an interferometer set out in a logarithmic spiral pattern comprised of two, five, ten and fifty antennas (top to bottom)
and observing for 10 s, 2, 4, and 6 h (left to right).

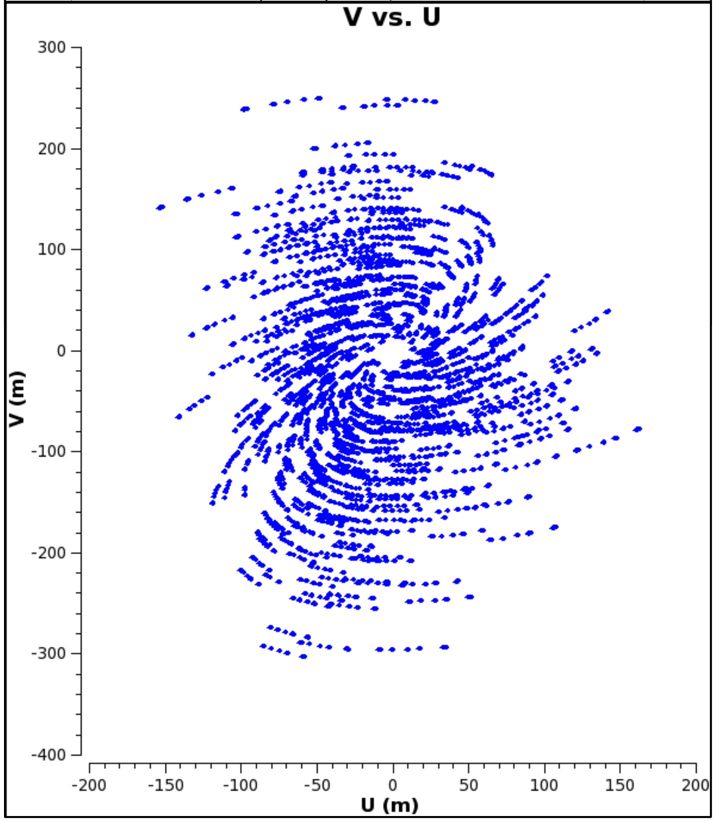
Filling the uv-plane

We want to fill the uv-plane because the uv-coverage is the FT of the synthesised beam:

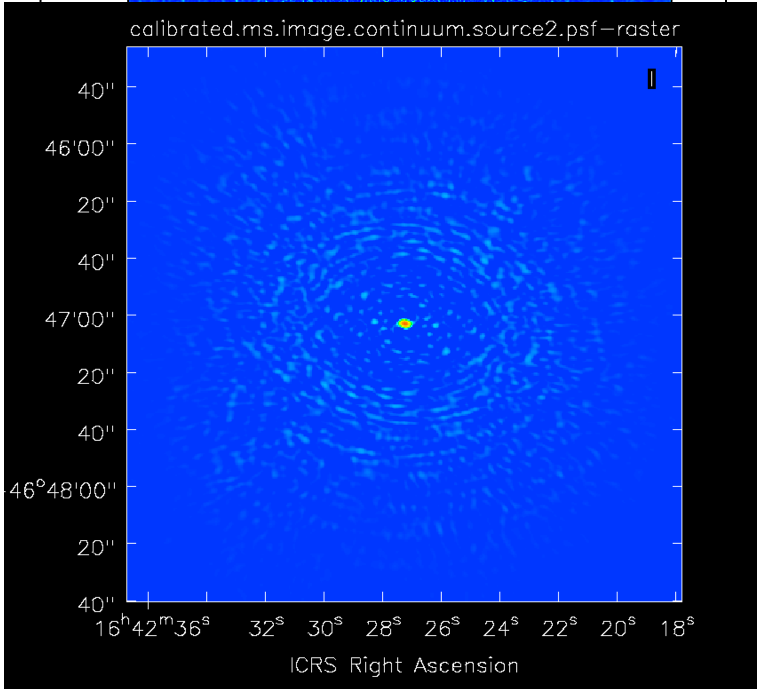


6 dishes
(~15 mins)

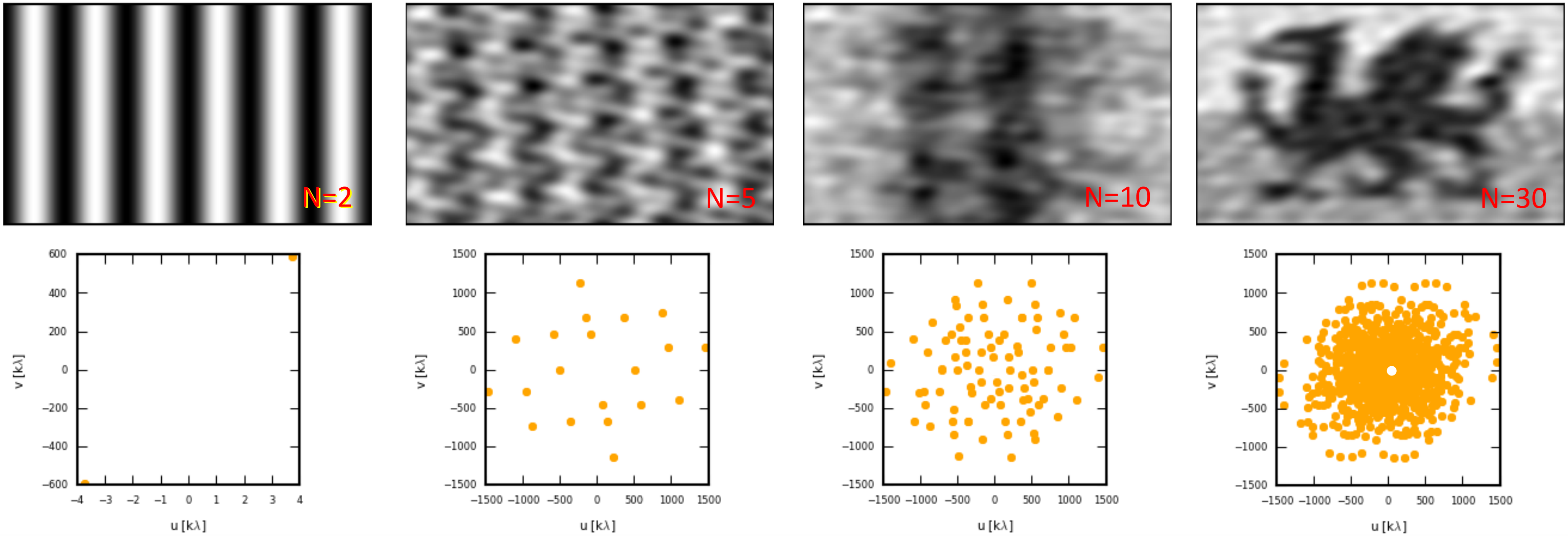




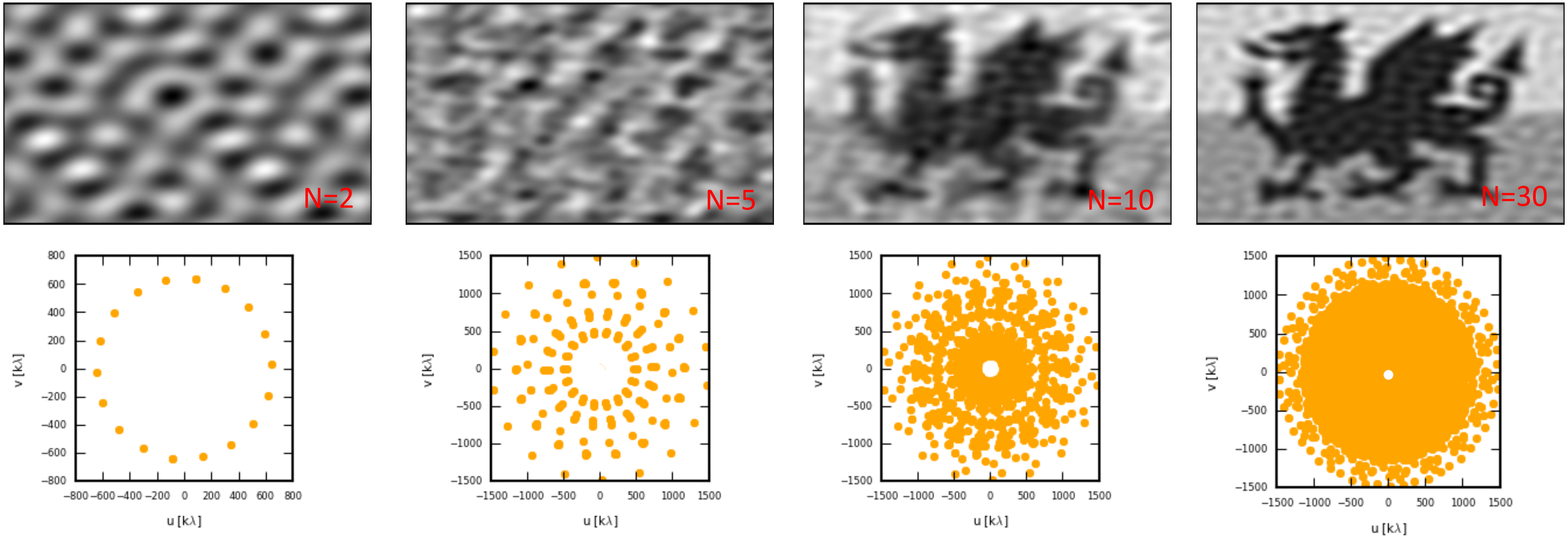
42 dishes
(~15 mins)

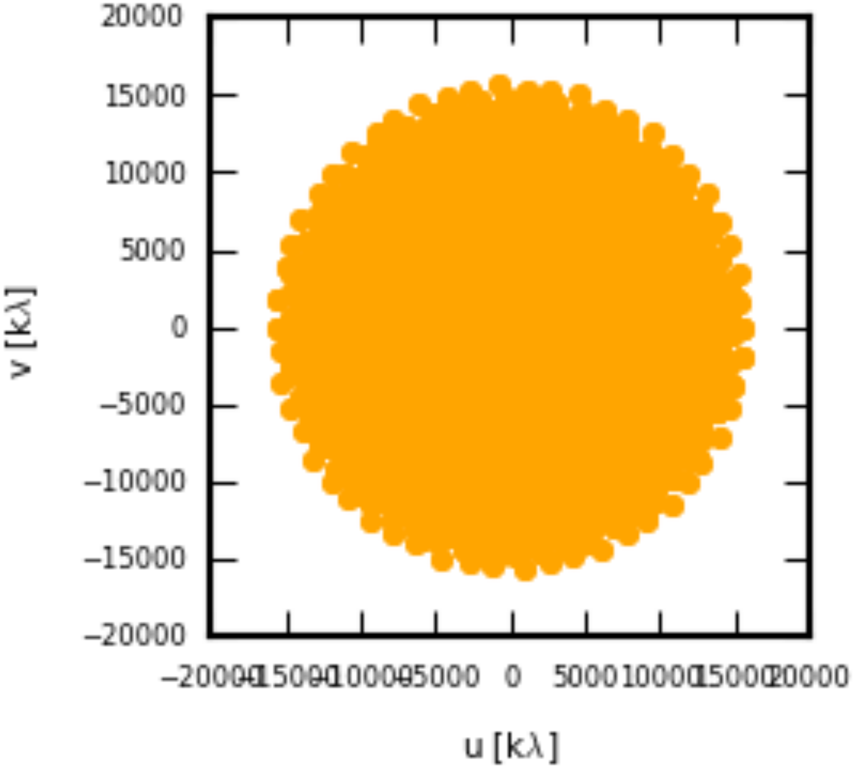


SNAPSHOT



EARTH ROTATION




20000
15000
10000
5000
0
−5000
−10000
−15000
−20000
v [kλ]
−20000
−15000
−10000
−5000
0
5000
10000
15000
20000
u [kλ]

Weighting

As stated previously the sampling of the *uv*-plane, *S*, is the FT of the synthesised beam, *B*.

$$S(u,v)=\sum_{k=1}^{M}\delta(u-u_k,v-v_k)$$

and *B*=FT(*S*)

Given this we can introduce weighting functions to control the shape of the synthesised beam.

$$W(u,v)=\sum_{k=1}^{M}R_kT_kD_k\delta(u-u_k,v-v_k)$$

R_k = Weights relating to data quality, i.e. down weight bad data. This is observation dependant and we have no post observation control over it (so ignore).

T_k = Tapering function. Apply a tapering function (i.e. Gaussian), to the uv-coverage to for example downweight the outer uv-points lowering resolution.

D_k = Density weighting... Next slide.

This is a truncated version of Chapter 7 (Briggs, Schwab & Sramek) of *"Synthesis Imaging in Radio Astronomy II"*

Density weighting

Due to the nature of how arrays are typically built it the uv-coverage density is typically higher toward the uv origin.

Two "extremes" of density weighting are typically used:

Natural weighting: $D_k = 1$. All visibilities are treated the same.
- This gives the highest signal to noise possible within the final image
- The poorest angular resolution and higher sidelobe effects.

Uniform weighting: $D_k = 1/N_s(k)$. Weight visibilities by the number of data points in a symmetric region, *s*.
- Downweights data in dense regions.
- Higher resolution, lower sidelobes, but worse SNR.

There are then super and sub-uniform (see e.g. CASA cookbook), and...

Briggs weighting: A 'sliding scale' between Natural and Uniform controlled by the 'robust' parameter. With (in CASA) +2 = nearly Natural and -2 = nearly Uniform. This is commonly used in ALMA QA2.

This is a truncated version of Chapter 7 (Briggs, Schwab & Sramek) of *"Synthesis Imaging in Radio Astronomy II"*

Further Reading

The slides from this talk are based on the fundamentals of interferometry which are explained in detail across:

- *“Interferometry and Synthesis in Radio Astronomy”* - Thompson, Moran & Swenson
- *“Synthesis Imaging in Radio Astronomy II”* – NRAO
- *“An introduction to Radio Astronomy”* – Burke and Graham-Smith
- *“Tools of Radio Astronomy”* – Wilson, Rohfls & Hüttemeister
- *“The CASA Cookbook”* – Ott & Kern et al.

Simulating observations

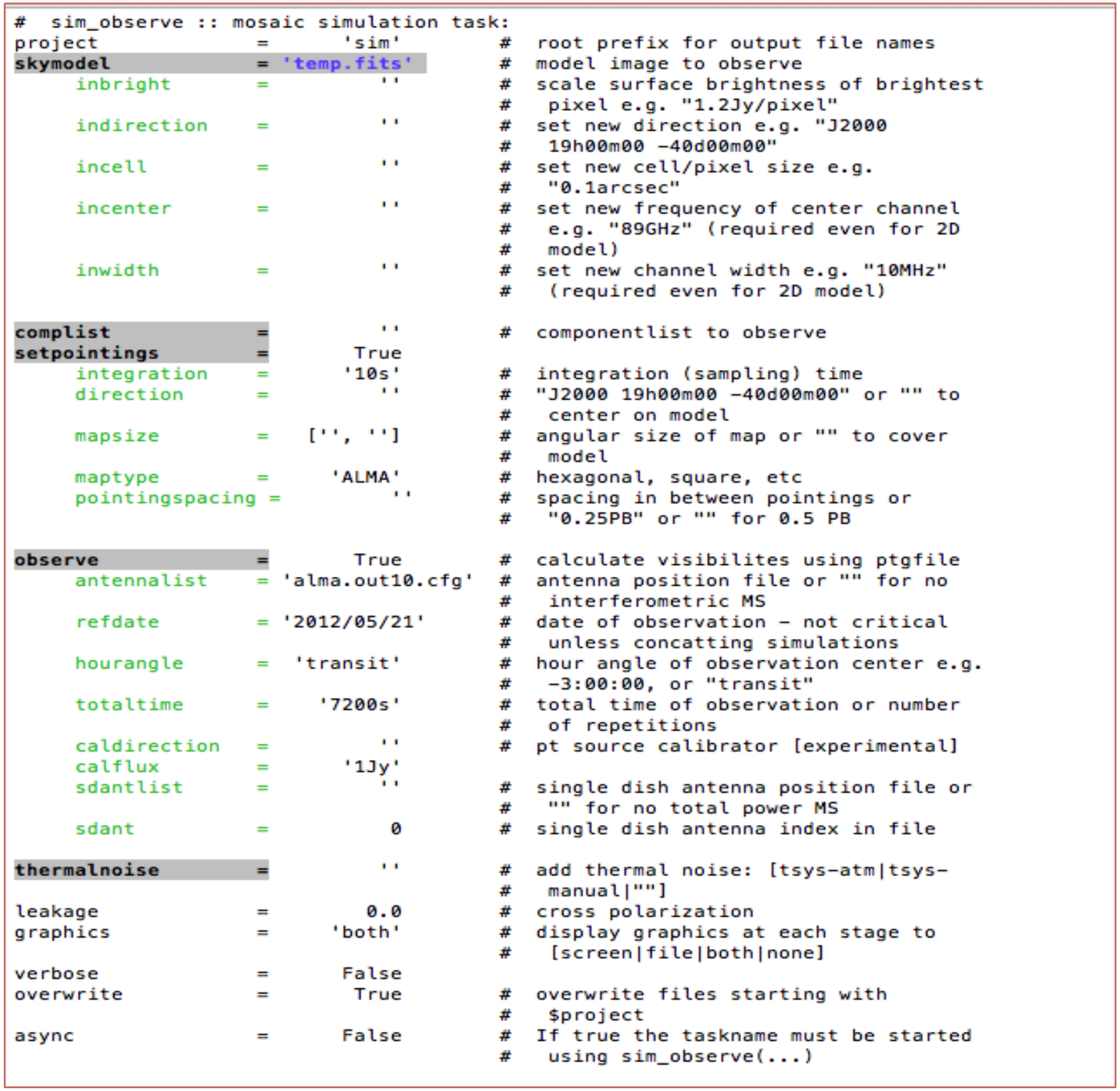
Simulating with CASA

Simobserve

simobserve is used to create the simulated interferometric observations within CASA and simanalyze is used to analyze the output of simobserve *(suprise suprise).*

Creating simulations in CASA, a recipe:

- First select an existing image of the region or type of object you want to simulate, for use as your skymodel.
- Your input sky model can then be rescaled in pixel size, brightness, sky position, central frequency and channel width with e.g. incell
- The setpointings and observe parameters are then used to define the simulated observation, with similar parameters to defining real observations.
- The corruption due to the atmosphere can then be set in thermalnoise.

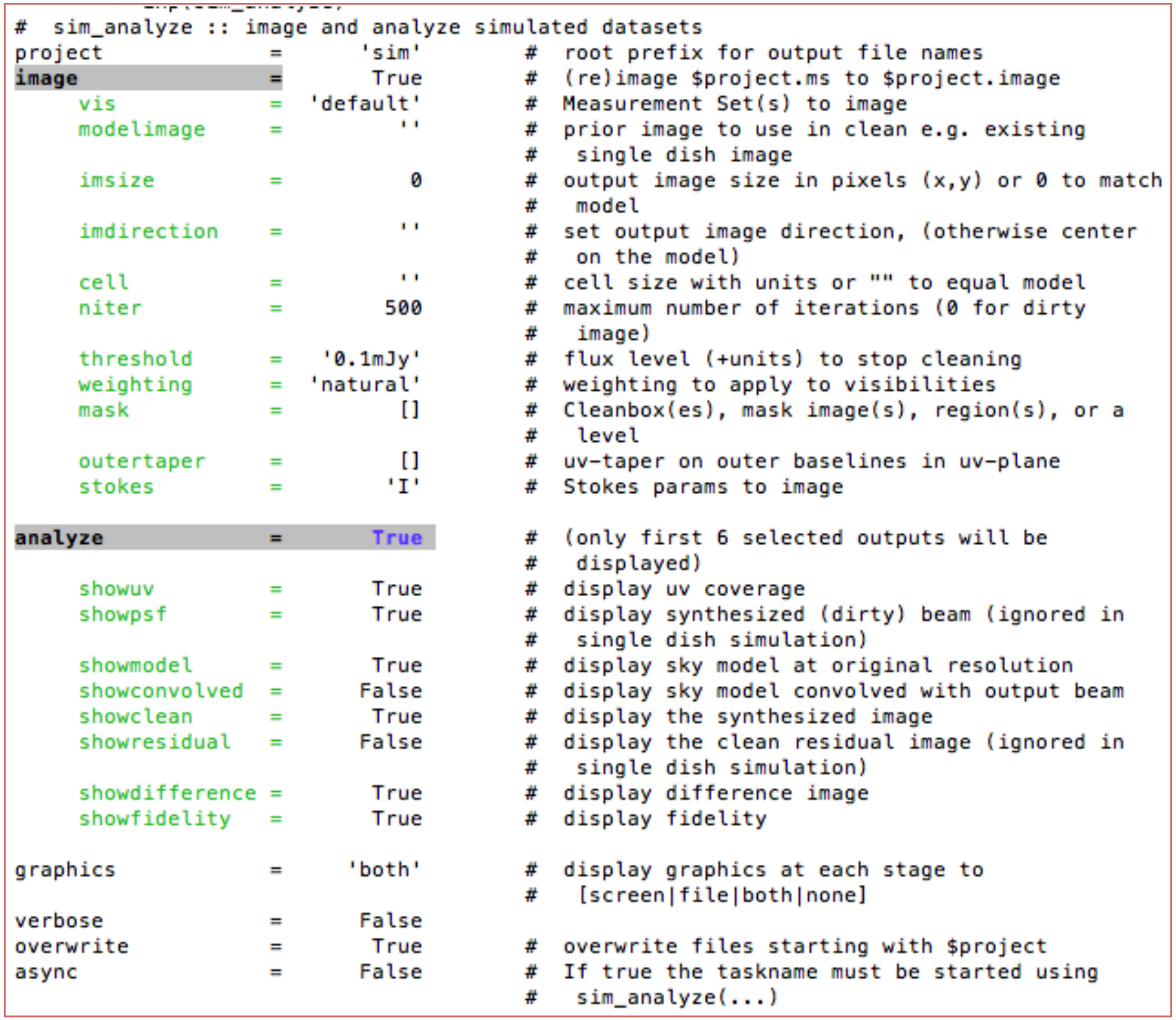


```
#  sim_observe :: mosaic simulation task:
project             =      'sim'        #  root prefix for output file names
skymodel            = 'temp.fits'       #  model image to observe
     inbright       =         ''        #  scale surface brightness of brightest
                                        #   pixel e.g. "1.2Jy/pixel"
     indirection    =         ''        #  set new direction e.g. "J2000
                                        #   19h00m00 -40d00m00"
     incell         =         ''        #  set new cell/pixel size e.g.
                                        #   "0.1arcsec"
     incenter       =         ''        #  set new frequency of center channel
                                        #   e.g. "89GHz" (required even for 2D
                                        #   model)
     inwidth        =         ''        #  set new channel width e.g. "10MHz"
                                        #   (required even for 2D model)

complist            =         ''        #  componentlist to observe
setpointings        =       True
     integration    =      '10s'        #  integration (sampling) time
     direction      =         ''        #  "J2000 19h00m00 -40d00m00" or "" to
                                        #   center on model
     mapsize        =   ['', '']        #  angular size of map or "" to cover
                                        #   model
     maptype        =     'ALMA'        #  hexagonal, square, etc
     pointingspacing =         ''       #  spacing in between pointings or
                                        #   "0.25PB" or "" for 0.5 PB

observe             =       True        #  calculate visibilites using ptgfile
     antennalist    = 'alma.out10.cfg'  #  antenna position file or "" for no
                                        #   interferometric MS
     refdate        = '2012/05/21'      #  date of observation - not critical
                                        #   unless concatting simulations
     hourangle      =  'transit'        #  hour angle of observation center e.g.
                                        #   -3:00:00, or "transit"
     totaltime      =    '7200s'        #  total time of observation or number
                                        #   of repetitions
     caldirection   =         ''        #  pt source calibrator [experimental]
     calflux        =      '1Jy'
     sdantlist      =         ''        #  single dish antenna position file or
                                        #   "" for no total power MS
     sdant          =          0        #  single dish antenna index in file

thermalnoise        =         ''        #  add thermal noise: [tsys-atm|tsys-
                                        #   manual|""]
leakage             =        0.0        #  cross polarization
graphics            =     'both'        #  display graphics at each stage to
                                        #   [screen|file|both|none]
verbose             =      False
overwrite           =       True        #  overwrite files starting with
                                        #   $project
async               =      False        #  If true the taskname must be started
                                        #   using sim_observe(...)
```

Simanalyze

Here we convert the CASA MS into an image file.

- The image parameter effectively acts like CLEANing a real dataset with iteration, weighting etc
- Next the analyze parameter defines which output images you would like from your analysis. Such as Clean image, UV coverage and image

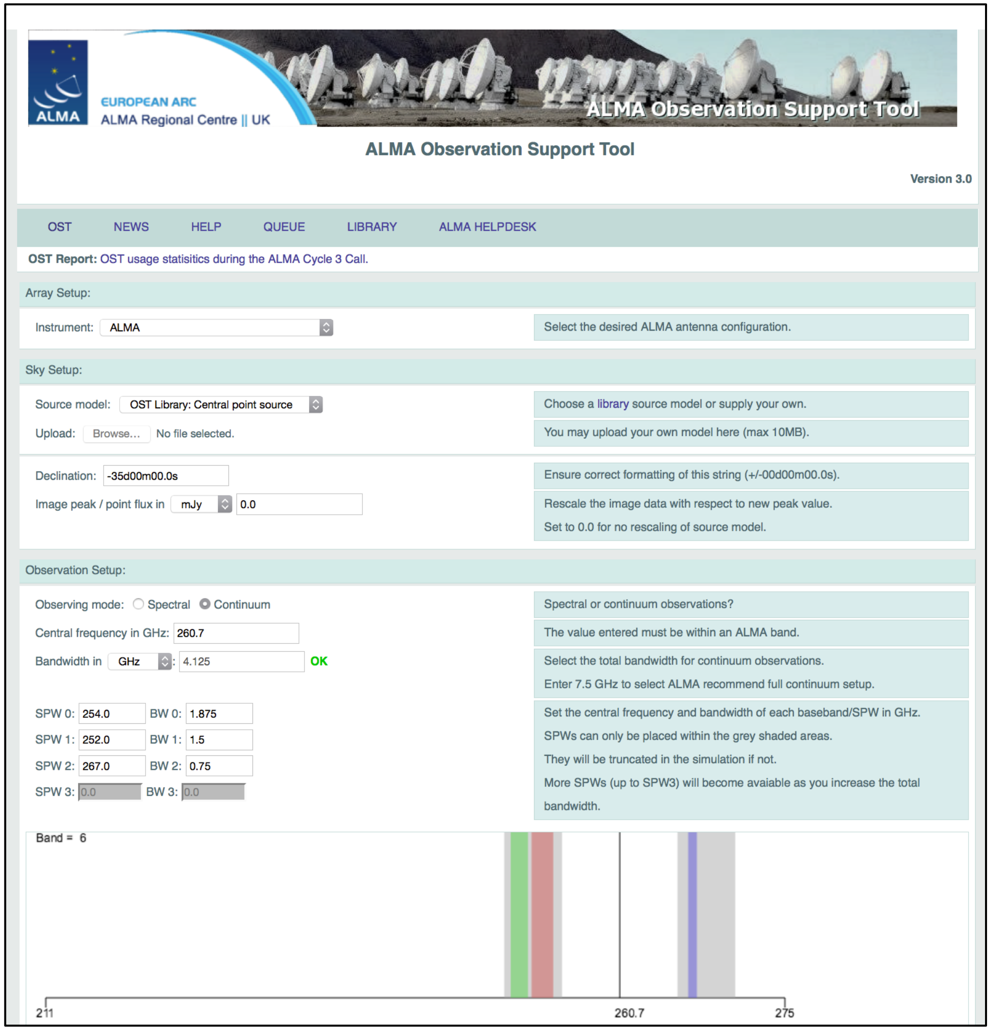


```
#  sim_analyze :: image and analyze simulated datasets
project             =      'sim'        #  root prefix for output file names
image               =       True        #  (re)image $project.ms to $project.image
     vis            =  'default'        #  Measurement Set(s) to image
     modelimage     =         ''        #  prior image to use in clean e.g. existing
                                        #   single dish image
     imsize         =          0        #  output image size in pixels (x,y) or 0 to match
                                        #   model
     imdirection    =         ''        #  set output image direction, (otherwise center
                                        #   on the model)
     cell           =         ''        #  cell size with units or "" to equal model
     niter          =        500        #  maximum number of iterations (0 for dirty
                                        #   image)
     threshold      =   '0.1mJy'        #  flux level (+units) to stop cleaning
     weighting      =  'natural'        #  weighting to apply to visibilities
     mask           =         []        #  Cleanbox(es), mask image(s), region(s), or a
                                        #   level
     outertaper     =         []        #  uv-taper on outer baselines in uv-plane
     stokes         =        'I'        #  Stokes params to image

analyze             =       True        #  (only first 6 selected outputs will be
                                        #   displayed)
     showuv         =       True        #  display uv coverage
     showpsf        =       True        #  display synthesized (dirty) beam (ignored in
                                        #   single dish simulation)
     showmodel      =       True        #  display sky model at original resolution
     showconvolved  =      False        #  display sky model convolved with output beam
     showclean      =       True        #  display the synthesized image
     showresidual   =      False        #  display the clean residual image (ignored in
                                        #   single dish simulation)
     showdifference =       True        #  display difference image
     showfidelity   =       True        #  display fidelity

graphics            =     'both'        #  display graphics at each stage to
                                        #   [screen|file|both|none]
verbose             =      False
overwrite           =       True        #  overwrite files starting with $project
async               =      False        #  If true the taskname must be started using
                                        #   sim_analyze(...)
```

Simalma

- A wrapper of simobserve and simanalyze which has some of these tasks parameters set to typical ALMA values.

The Observation Support Tool



ALMA Observation Support Tool

Version 3.0

OST Report: OST usage statisitics during the ALMA Cycle 3 Call.

Array Setup:

Instrument: ALMA — Select the desired ALMA antenna configuration.

Sky Setup:

Source model: OST Library: Central point source — Choose a library source model or supply your own.

Upload: Browse... No file selected. — You may upload your own model here (max 10MB).

Declination: -35d00m00.0s — Ensure correct formatting of this string (+/-00d00m00.0s).

Image peak / point flux in mJy 0.0 — Rescale the image data with respect to new peak value. Set to 0.0 for no rescaling of source model.

Observation Setup:

Observing mode: ○ Spectral ● Continuum — Spectral or continuum observations?

Central frequency in GHz: 260.7 — The value entered must be within an ALMA band.

Bandwidth in GHz: 4.125 OK — Select the total bandwidth for continuum observations. Enter 7.5 GHz to select ALMA recommend full continuum setup.

SPW 0: 254.0 BW 0: 1.875
SPW 1: 252.0 BW 1: 1.5
SPW 2: 267.0 BW 2: 0.75
SPW 3: 0.0 BW 3: 0.0

Set the central frequency and bandwidth of each baseband/SPW in GHz. SPWs can only be placed within the grey shaded areas. They will be truncated in the simulation if not. More SPWs (up to SPW3) will become avaiable as you increase the total bandwidth.

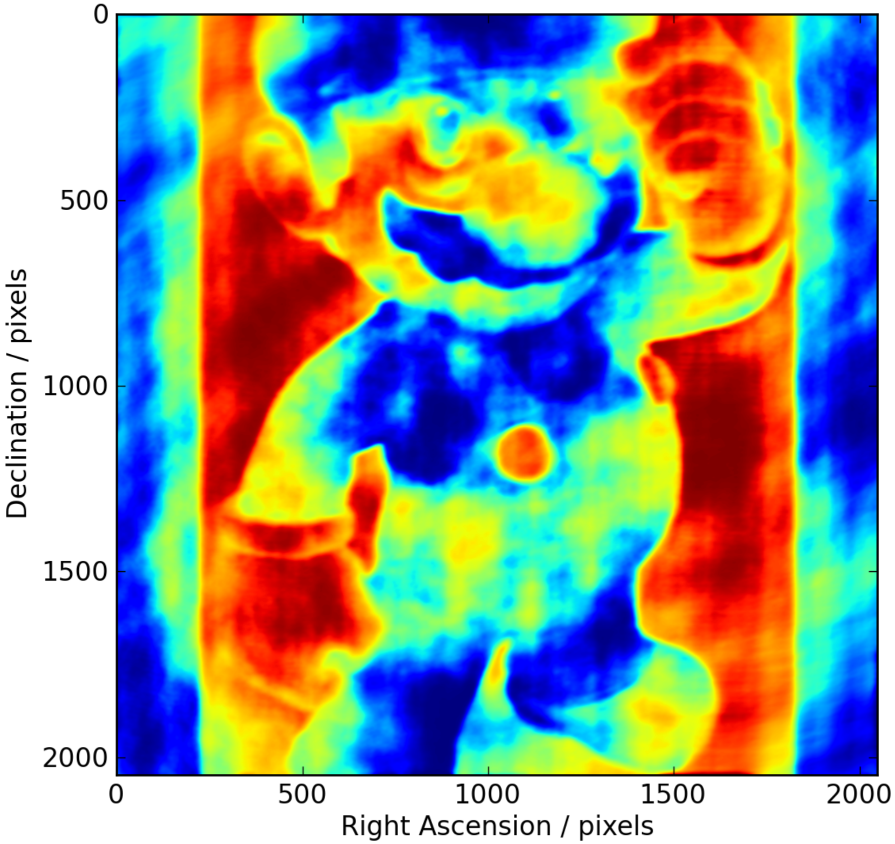
The ALMA Observation Support Tool (OST) is a web-based ALMA simulator aimed at the non-interferometry expert user.

Available since ALMA Cycle 0 CfP. Has been extensively used by the international community ALMA Cycle 0, 1, 2, 3 and 4 call for proposals.

Created in 2011 by Ian Heywood and since updated and developed by Adam Avison

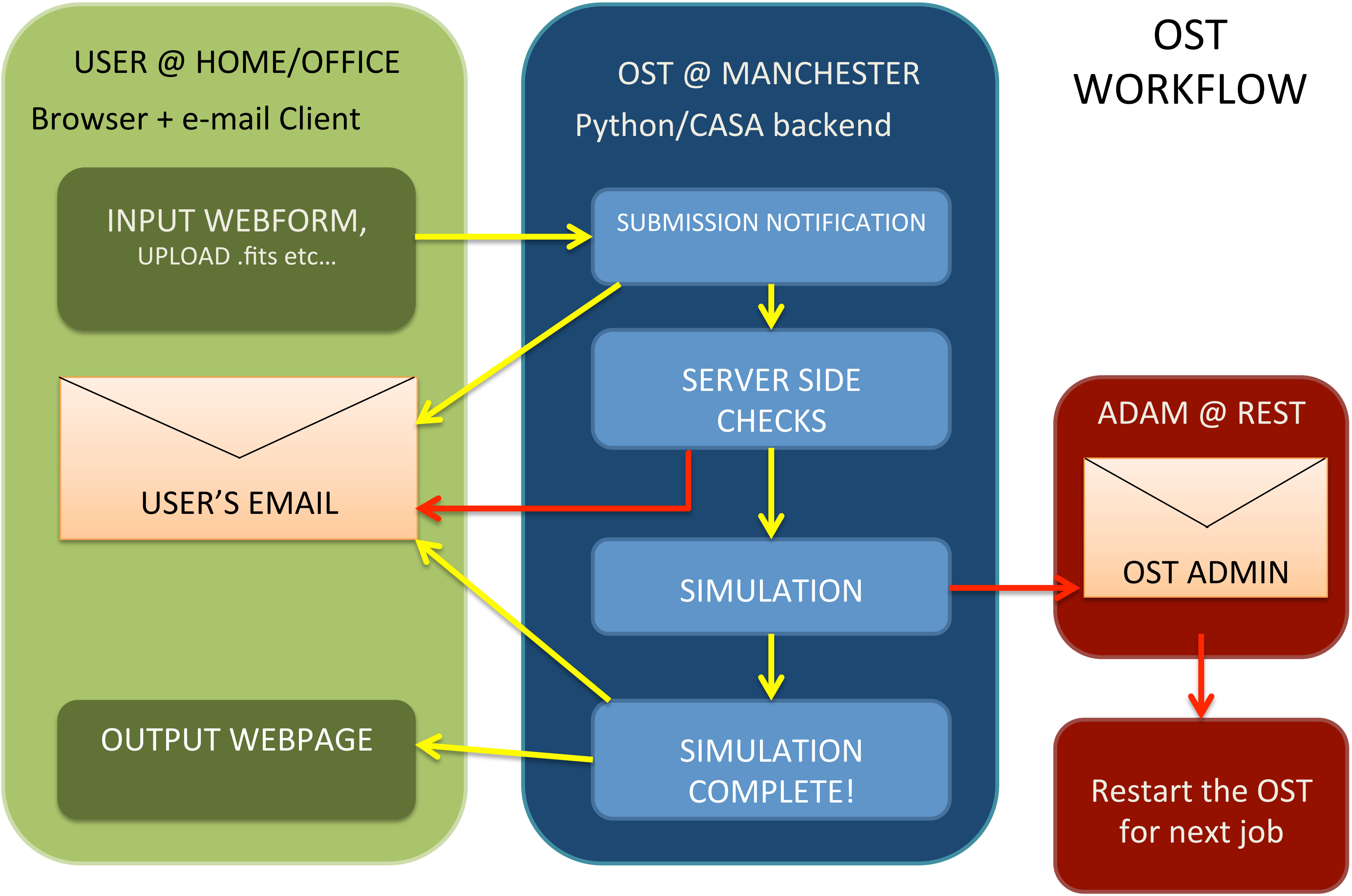
See *http://almaost.jb.man.ac.uk*

OST Walkthrough



- OST Simulation of the ‘Super’ M-4R10 Galaxy

OST WORKFLOW



USER @ HOME/OFFICE

Browser + e-mail Client

INPUT WEBFORM,
UPLOAD .fits etc…

USER'S EMAIL

OUTPUT WEBPAGE

OST @ MANCHESTER

Python/CASA backend

SUBMISSION NOTIFICATION

SERVER SIDE CHECKS

SIMULATION

SIMULATION COMPLETE!

ADAM @ REST

OST ADMIN

Restart the OST for next job

Back to OST output