Combining Single Dish and Interferometric Data

Peter Teuben (UMD)

and for the tutorial:

Sandra Burkatean (IRA-INAF)

Thomas Stanke (ESO)

Goals

- Single Dish vs. Interferometry, and why combine?
- Learn techniques to combine
- How to measure success
  - Total Flux, of Flux(channel) “spectrum” plot
  - Noise
  - Image Fidelity (like a S/N map)
    - F = Model / (Model - Observed)
  - Image Properties (science driven?)
    - PDS
    - Mom0 (flux), Mom1 (mass), Mom2 (turbulence)
- Group Photo
- Lunch
- Tutorial
- Bier & B

Tutorial Session

2h + 1.5h or 1.5h + 2h

?

- 13:00 – 13:05 : git on github **“why git?”**
- 13:05 – 15:00 : common exercises **“repeat after me”**
  - Feather and tp2vis combination
- 15:30 – 17:00 : group exercices **“hack hour”**
  - What-IF Exploratory exercises
    - Scale factors
    - Weight factors
    - Influence of choice or arrays and dish sizes
    - Current Algorithms (feather, tp2vis, sd2vis, ssc)
    - New Algorithms (stani)

And then there was this....

- HDD:
  - **Data** for tutorials 1,3,4/5
    - Directories: CASA_SD,CLASS, CASA_QAC
  - **Papers** : ~40 papers in ~ 140MB
    - Note these are not on wiki, only on the HDD
- **Software** via SD2018 github
  - Instructions for CASA,QAC,tp2vis etc.
  - Python3 miniconda + radio-astro-tools / astropy / APLpy / ....
  - Fits viewers: ds9, casaviewer, QfitsView
- GIT?
  - Intro to git for beginners
  - “git pull request” (*I’ll just send you a PR*)

I'm going to skip....

van Cittert - Zernicke relation

$$\mathcal{V}(u,v) = \iint I(l,m) e^{-2\pi i(ul+vm)} dl dm$$

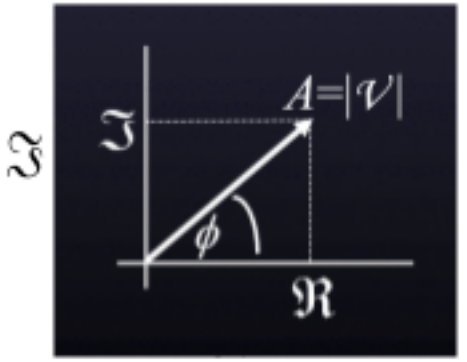
$$I(l,m) = \iint \mathcal{V}(u,v) e^{2\pi i(ul+vm)} du dv$$

$I(l,m)$ can be recovered from $V(u,v)$ via Fourier Transform

$V(u,v)$ expressed as (real, imaginary) or (amplitude, phase)

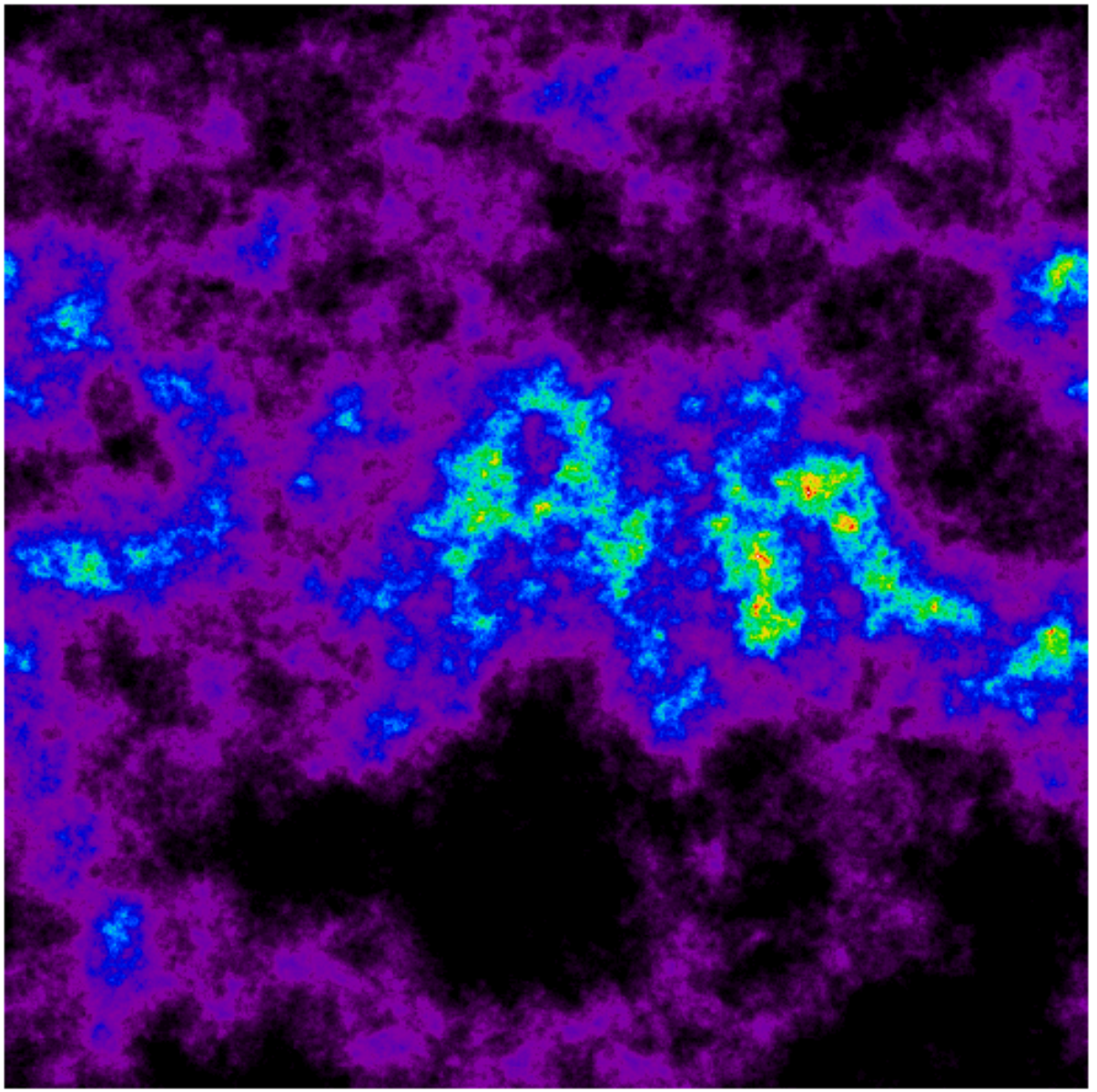
$$A = \sqrt{\Re^2 + \Im^2}$$

$$\phi = \tan^{-1}\left(\frac{\Im}{\Re}\right)$$

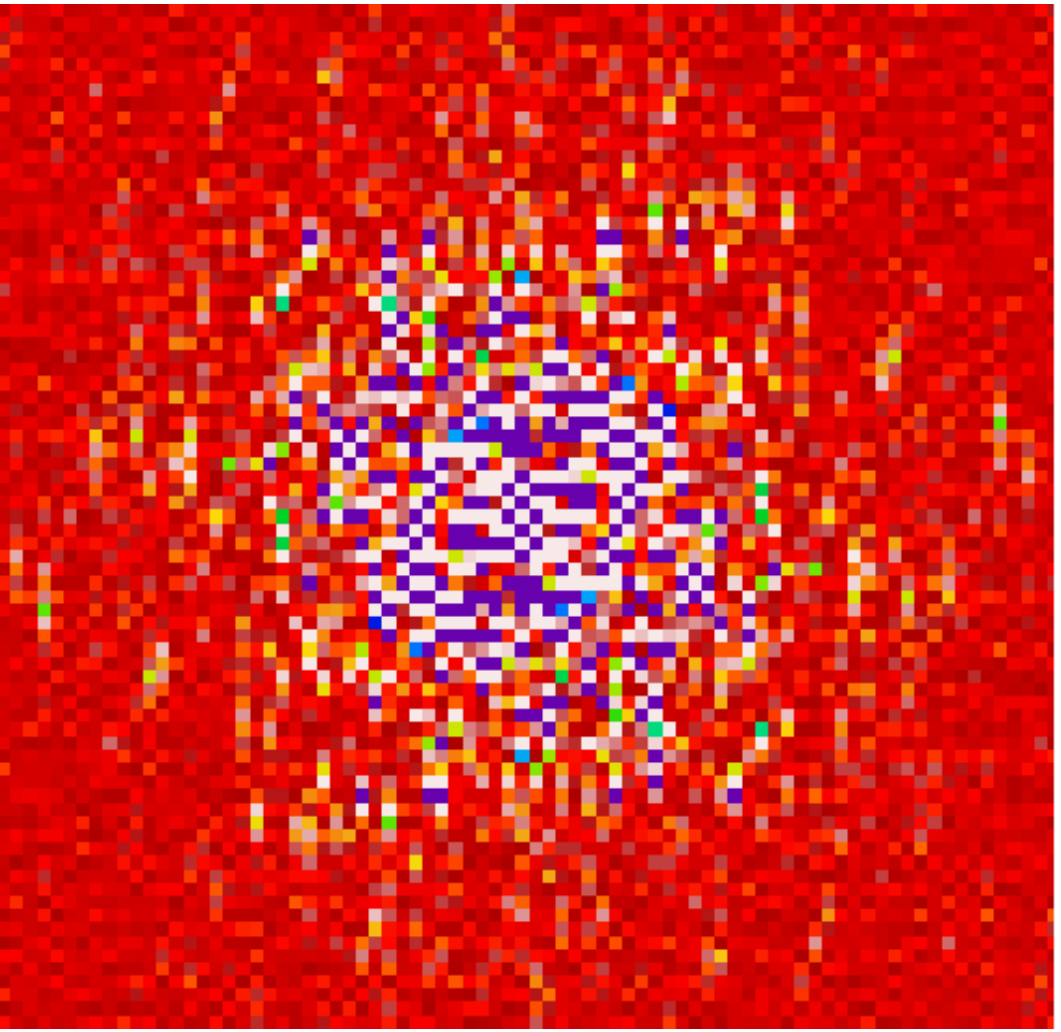


Carpenter – JAO lecture series (see your Papers/)

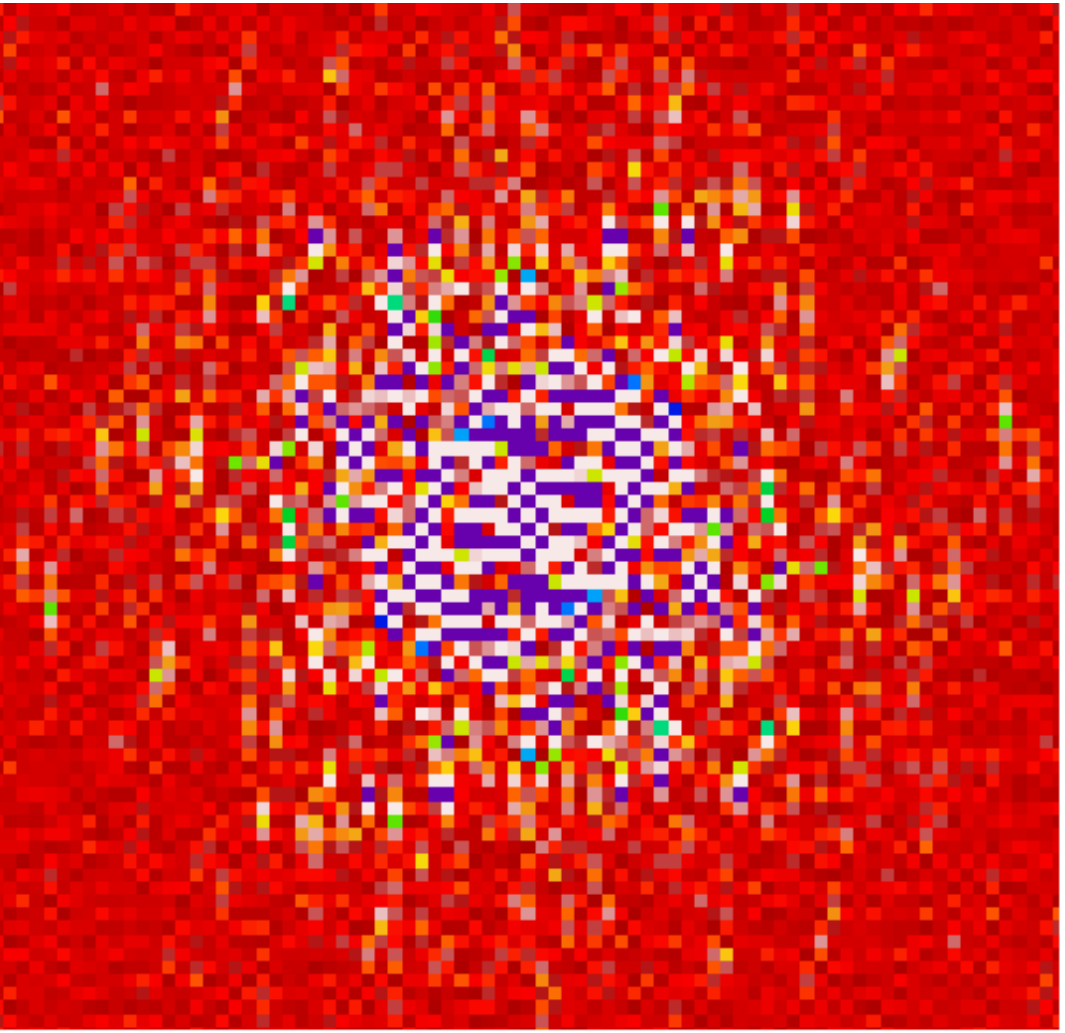
Let’s say we have a sky image

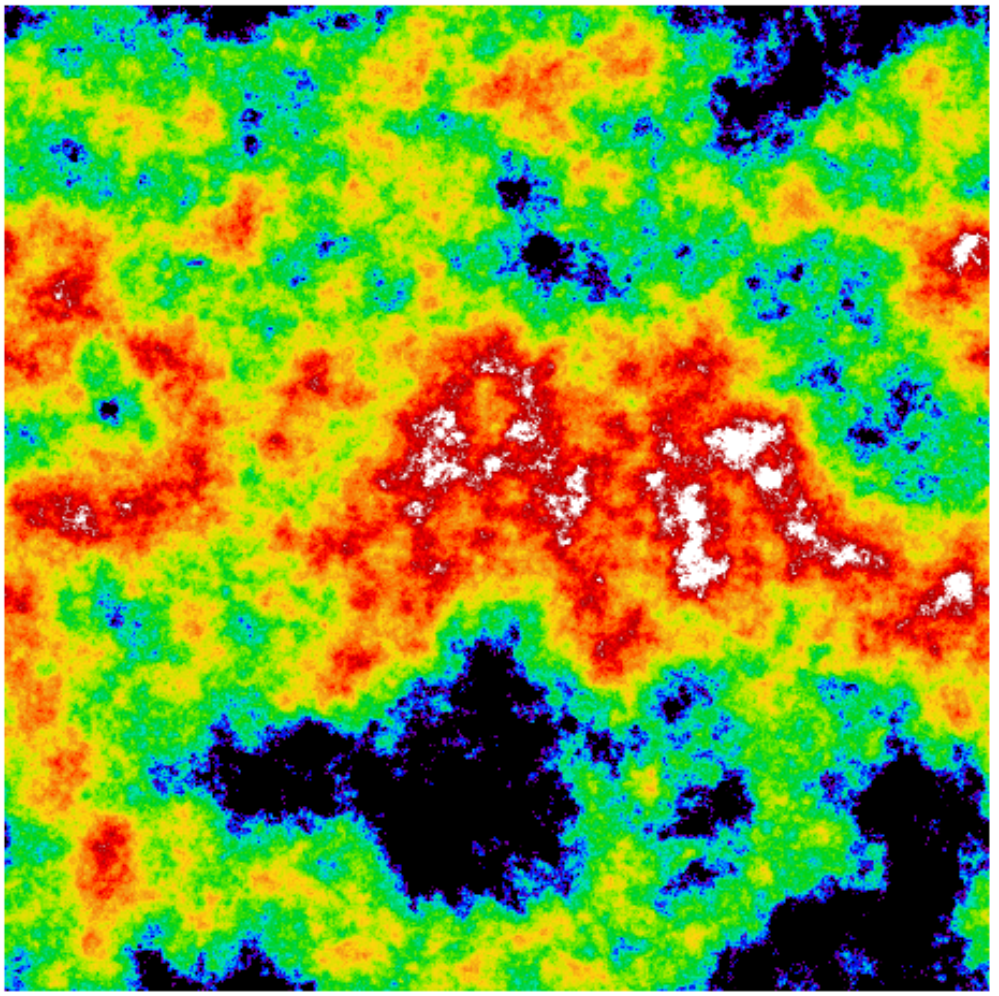


If only we had a regular UV grid of Amp and Phase

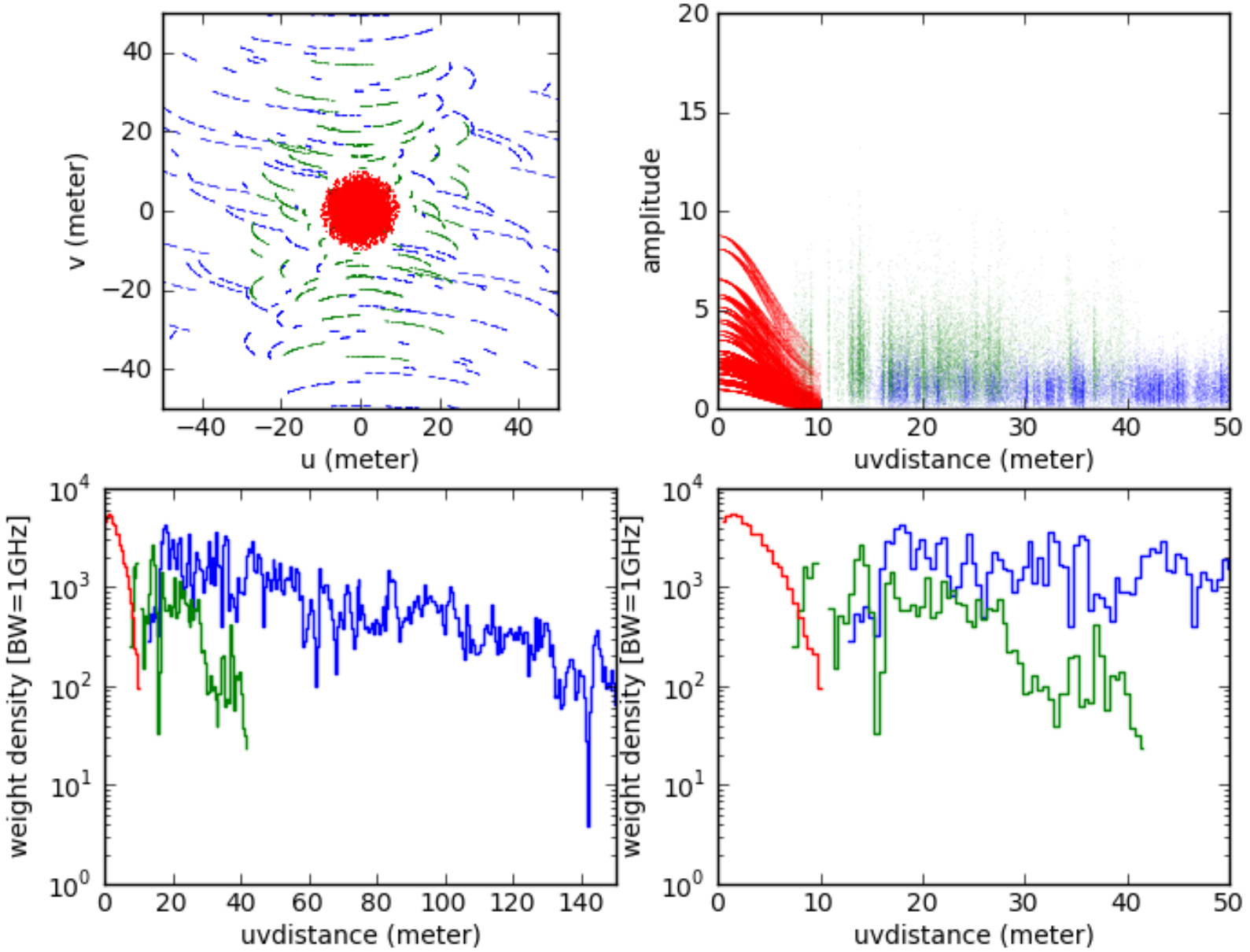


If only we had a regular UV grid of Amp and Phase

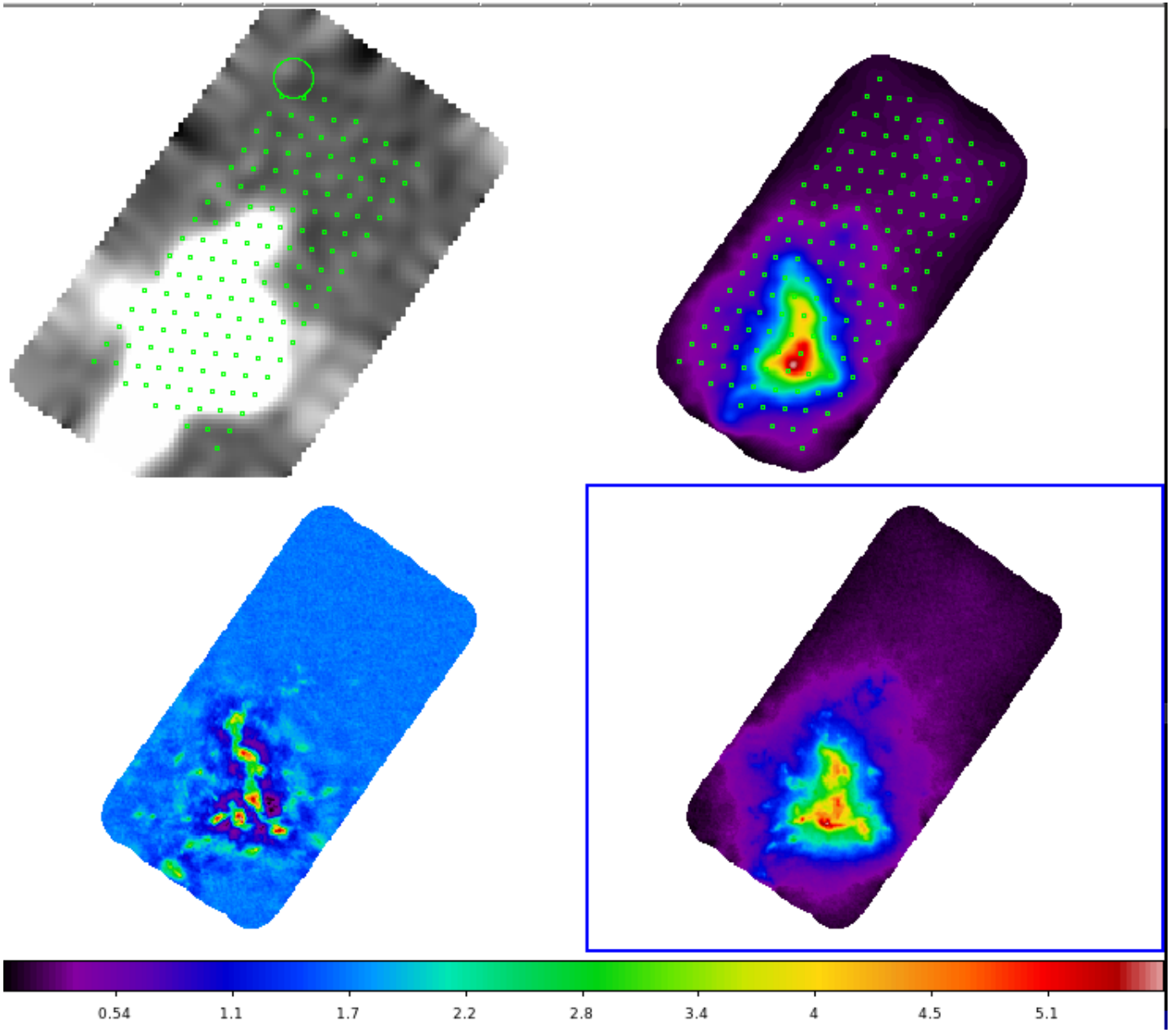




Sadly we have a “poorly” sampled UV plane



Sadly we have a “poorly” sampled UV plane



Missing Short Spacings

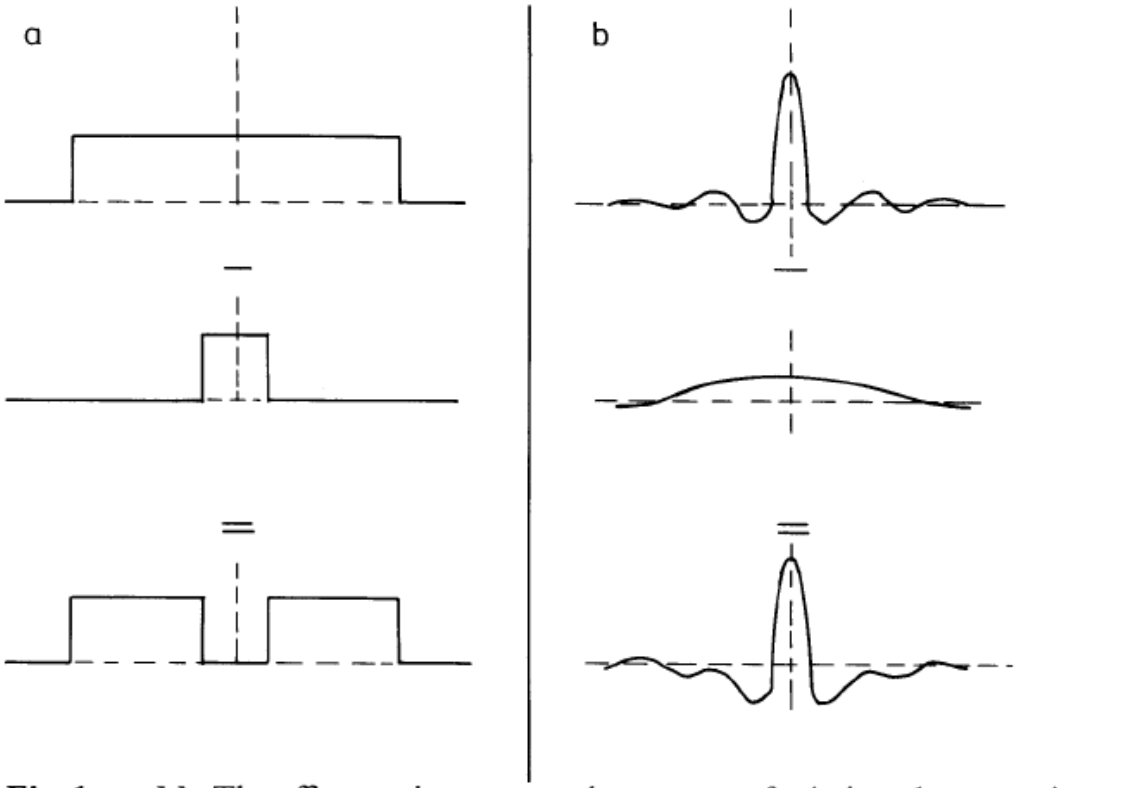
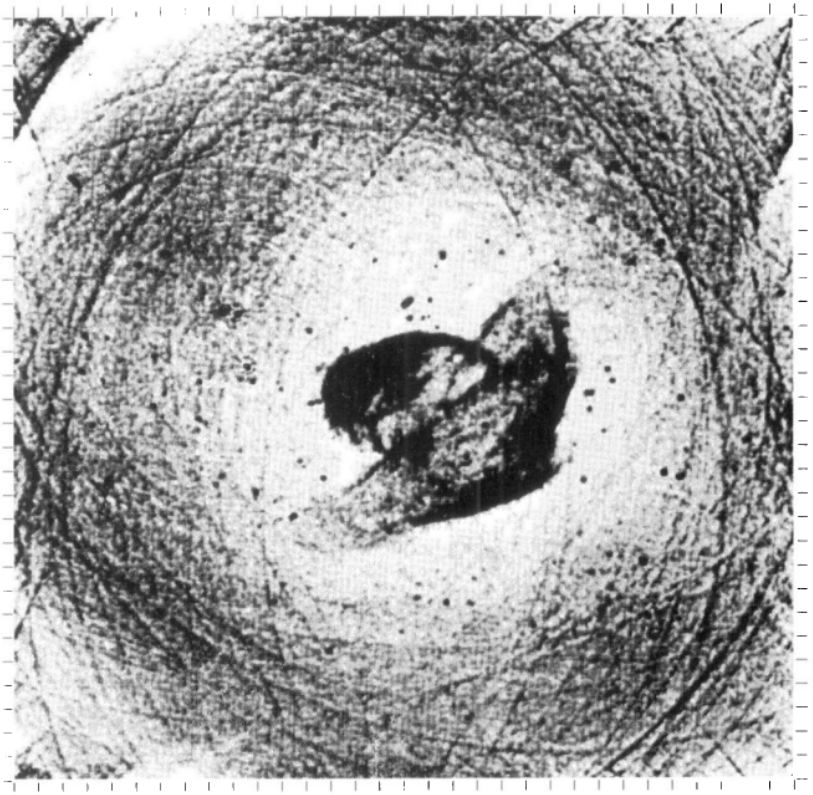


Fig. 1a and b. The effect on instrumental response of missing short spacings. **a** Observed spatial frequencies and **b** the corresponding instrumental response



Braun & Walterbos (1985)

See also: Ekers & Rots **1979**, Vogel et al. **1985**, Stanimirovic **1999**, Koda et al **2011**

Deconvolution

- CLEAN: Subtracting Dirty Beam components
  - Hogbom (1974)
  - Clark (1980)
    - Cotton-Schwab iterations : major and minor cycles
    - Minor cycle => deconvolution
    - Major cycle => imaging
  - **Schwarz** (1978) – 1D clean mathematical proof
  - Wakker & Schwarz (1988) – mrc – short spacings
  - Cornwell (2008) – multiscale clean
- MEM: Maximum Entropy
  - Cornwell & Evans (1985)

scales=[0,5,15] smallscalebias=0.5

Astron. Astrophys. 65, 345—356 (1978)

Mathematical-statistical Description of the Iterative Beam Removing Technique (Method CLEAN)

U. J. Schwarz

Kapteyn Laboratory, University of Groningen, Postbus 800, NL-8002 Groningen, T

Received October 13, 1976; revised July 15, 1977

Summary. The CLEAN method (Högbom, 1974), which is a deconvolution method, is analysed mathematically for the 1-dimensional case. It is shown that the method is equivalent to solving a system of linear equations by an iterative method (Temple, 1938). A criterion of convergence is given. In typical applications of the method the solution of the system of equations is *not* unique and the consequences for the CLEAN solution are discussed.

By applying the analysis to maps which are obtained from Fourier transformed data the convergence criterion is shown to be equivalent to the condition that all weights used for the Fourier transform have to be non-negative. It is proven that the method is in fact a statistically correct least-squares fit of sine functions to the observed data (the visibility function). The choice of clean beam and the effects of adding residuals are analysed.

An error analysis is given which allows the errors in interpolated and extrapolated information to be determined. Some numerical examples of error calculations are given.

The extension of the present analysis to 2 dimensional distributions is briefly discussed in an Appendix.

Key words: CLEAN — data-processing — Fourier transform — statistics — radio astronomy

Cheat Sheet
CASA-MIRIAD-GILDAS

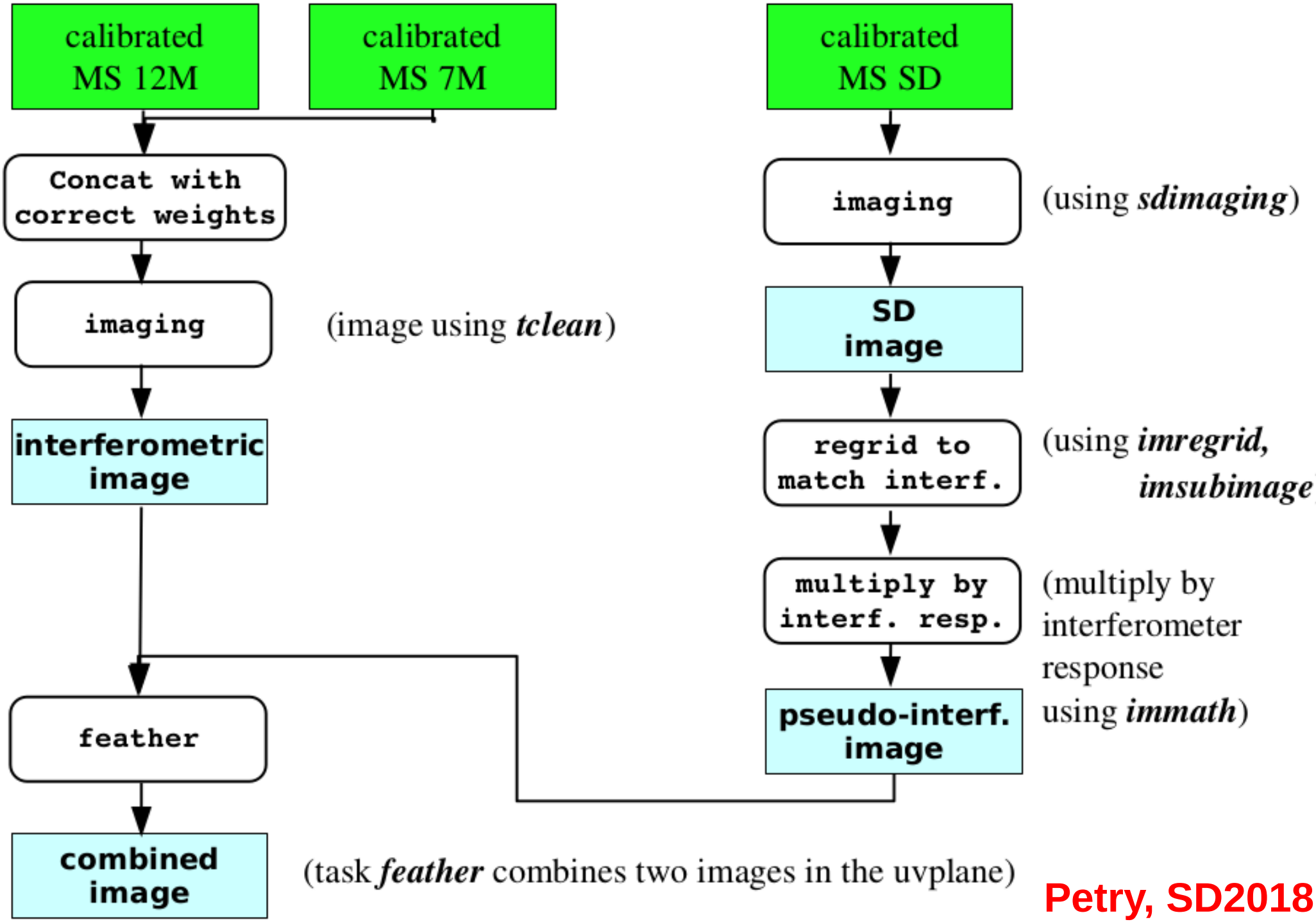
- tclean (*)
- feather
- tp2vis **NEW**
- tclean(scales=[])

- invert+clean+restor
- immerge
- [Koda's scripts]
- "mrc" [Wakker]

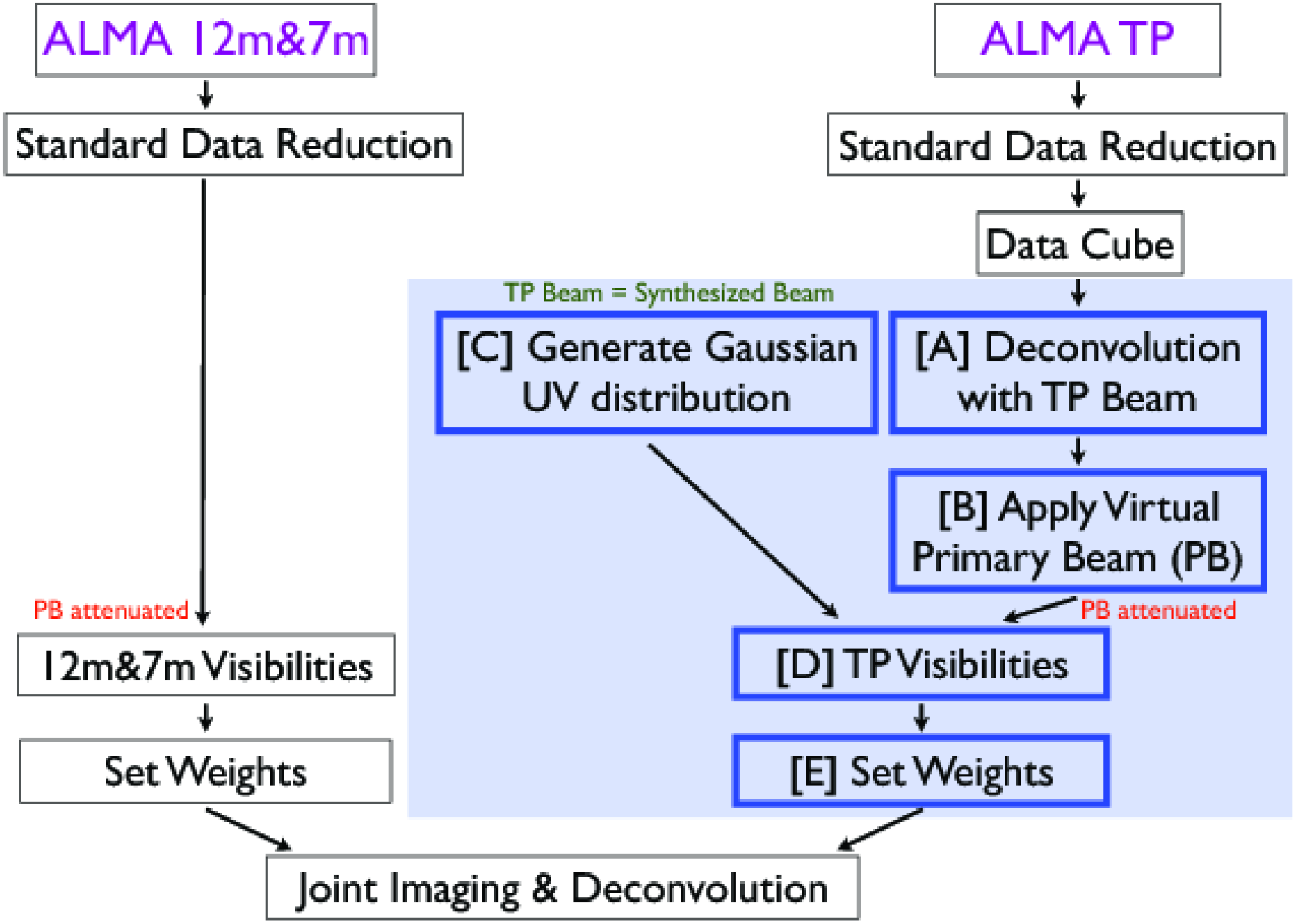
*** tclean see also:**

```
deconvolve()
im.restore()
```

Combination of interf. and SD data workflow with CASA



Petry, SD2018


ALMA 12m&7m
Standard Data Reduction
ALMA TP
Standard Data Reduction
Data Cube
TP Beam = Synthesized Beam
[C] Generate Gaussian UV distribution
[A] Deconvolution with TP Beam
[B] Apply Virtual Primary Beam (PB)
PB attenuated
PB attenuated
12m&7m Visibilities
[D] TP Visibilities
Set Weights
[E] Set Weights
Joint Imaging & Deconvolution

Array Combination Methods
[that (can) work in CASA]

- Feather: combine image via fourier domain
- Pure JD in combination with tclean()
  - TP2VIS
  - SD2VIS (single pointing)
- Maximum Entropy “JD” of Hires and Lowres
- CBD – Linear Combination (Stanimirovic et al 1999)
- CDD tclean + startmodel=
  - TP map (scaled to Jy/pixel)
  - TP map deconvolved
  - Feather (scaled to Jy/pixel)
- CAD – SSCIM (Faridani et al. 2017)

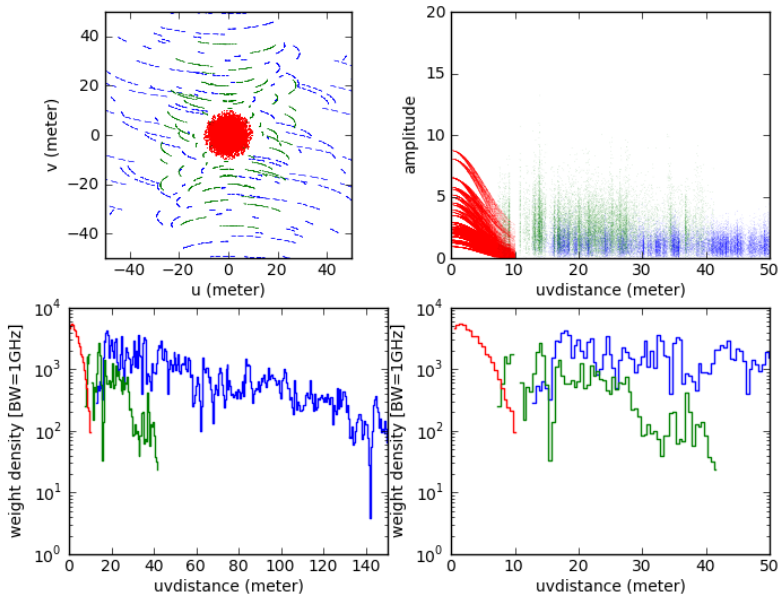
See also: Stanimirovic 2002

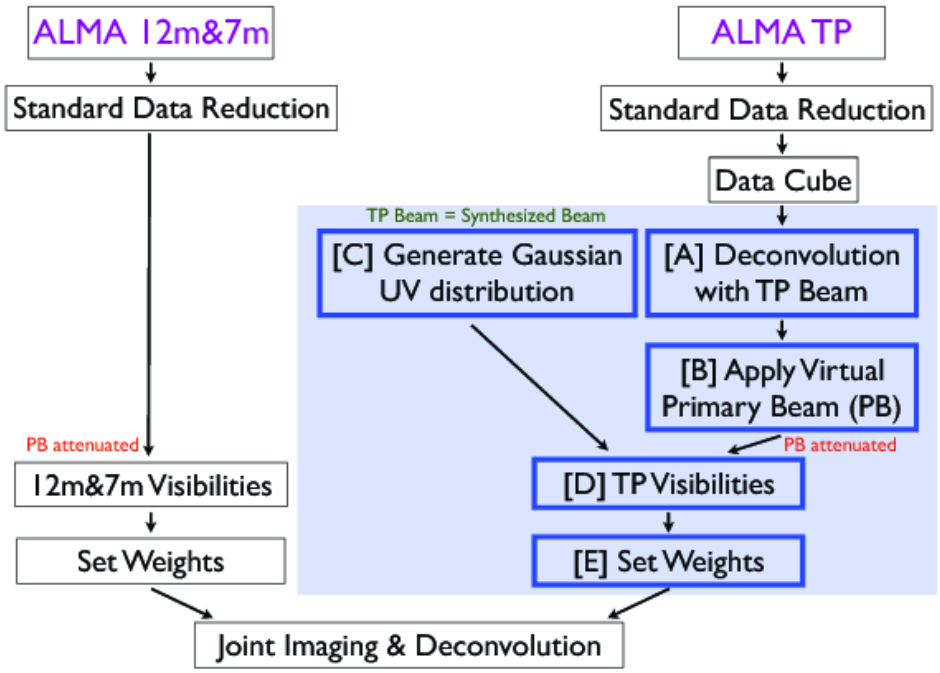
Feather (immerge)

- Get LoRes and HiRes image on same grid (imregrid)
- Ignore edges where PB < 0.2 (imsubimage)
- Use noise flat images:
  - LoRes = TP * .PB
  - HiRes = .image
- Run feather()
- Moments on this noise flat result
- MOM0/PB for fluxes, MOM1, MOM2

tp2vis
(Koda et al. , in prep.)

1) tp2vis(tpim, tpms, ptg, rms=0.15)
2) tp2viswt(tpms, mode='mult', value='0.1')
3) tp2vispl(tpms)
4) tclean([tpms,ms07,ms12],...)





SSCIM ("CAD")
(Faridani et al. 2017)

- Convolve INT map to SD resolution
- Subtract this from SD map => "missing flux"
- Scale missing flux by beam area ratio
- Add this back into the INT map => combined map

$$I_{combine} = I_{int} + \varepsilon * (I_{SD} - I_{int}^{conv})$$

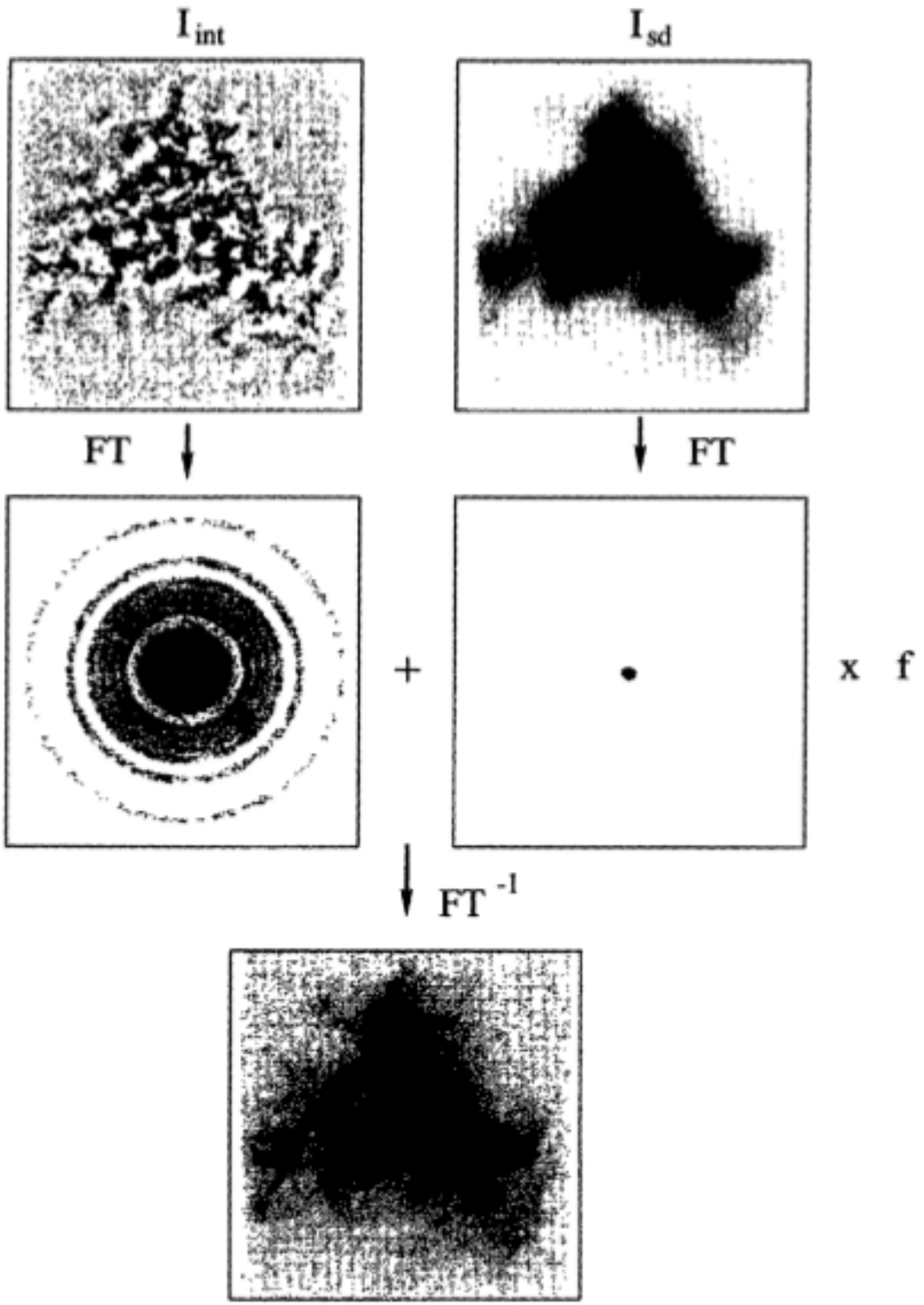
$$\varepsilon = \Omega_{int} / \Omega_{SD}$$

Linear Combination “CBD”
(Stanimirovic et al. 1999)

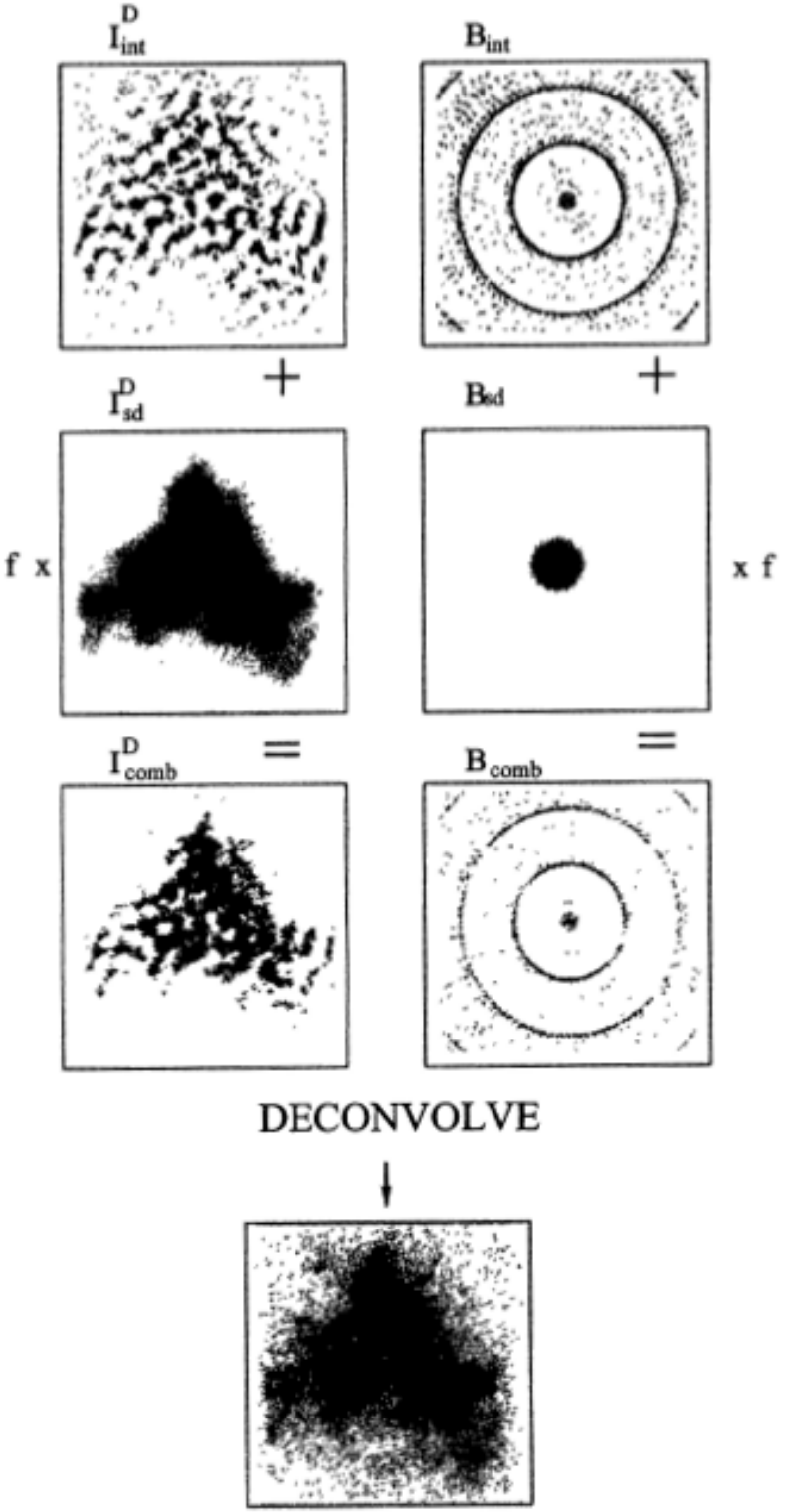
- Create dirty image (tclean niter=0)
- Define weights and flux factor
- Combine dirty image and beam
- Deconvolve

$$I_{comb} = I_{int} + f\,\varepsilon\, I_{SD}$$
$$B_{comb} = B_{int} + f\,\varepsilon\, B_{SD}$$

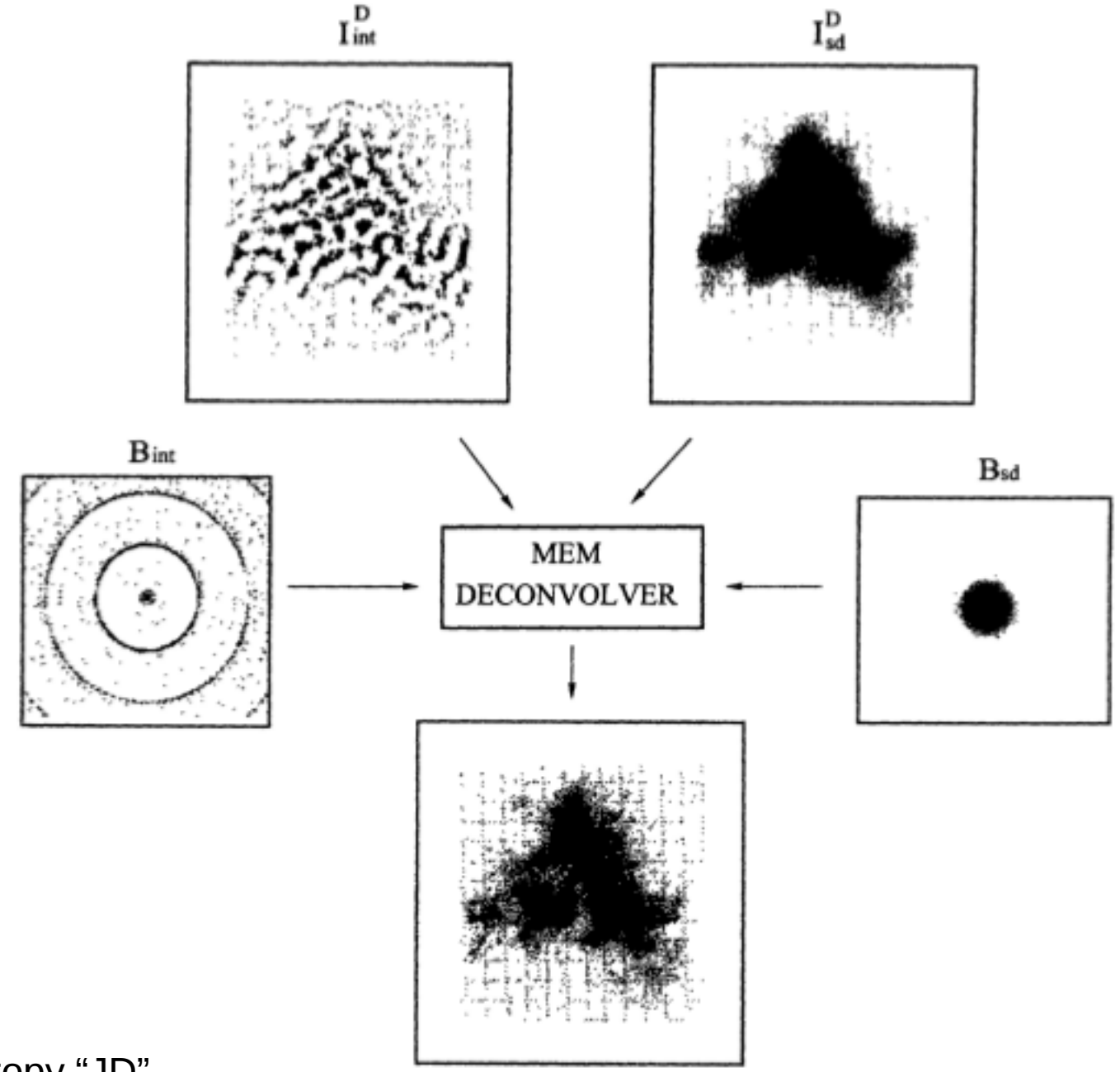
$$\varepsilon = \Omega_{int} / \Omega_{SD}$$



IMMERGE



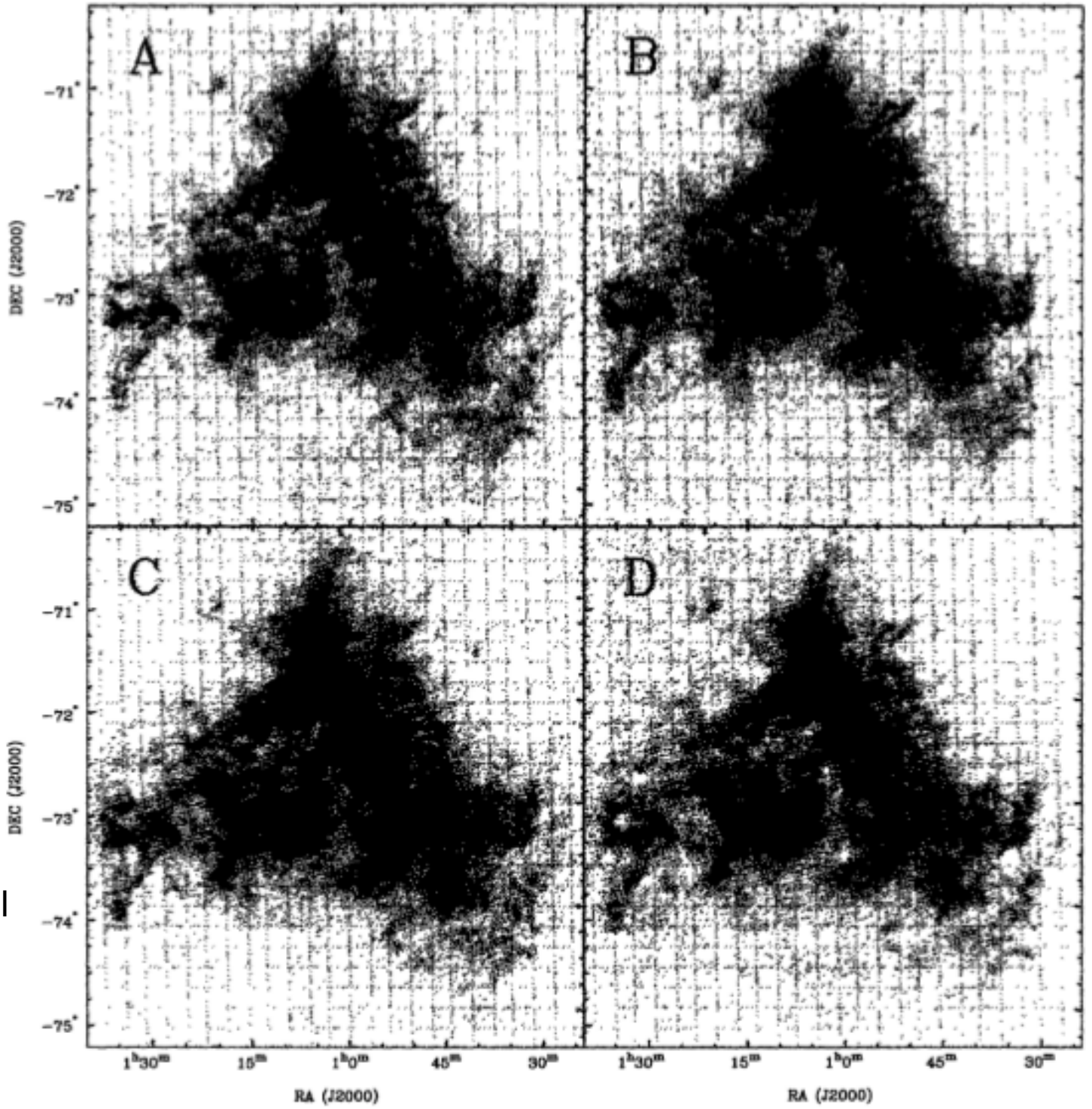
Linear Combination
(Stanimirovic 1999)



Maximum Entropy “JD”

Immerge

stani99



SD startmodel

JDmem

Performance

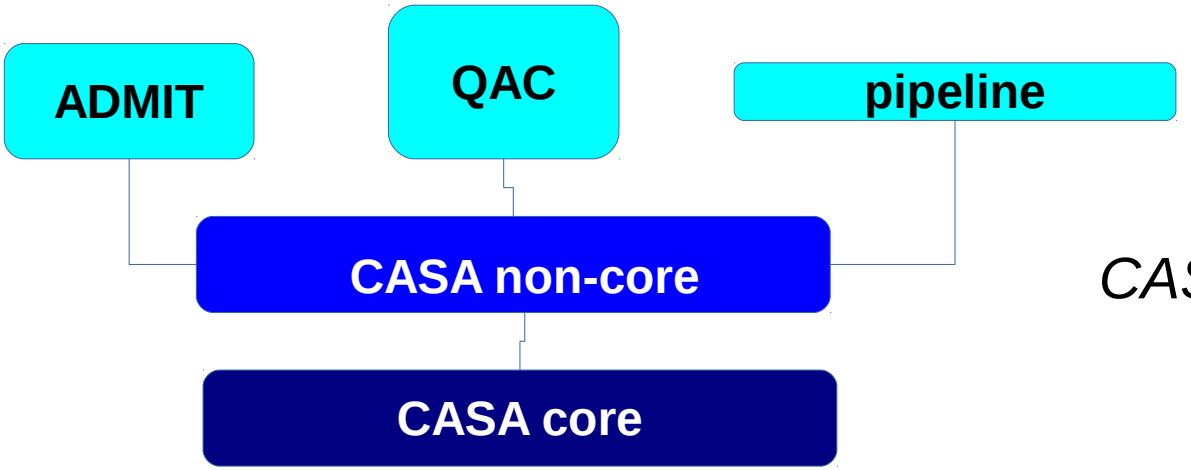
- CASA::tclean - lots of FFT
- CASA::feather – needs one FFT
- SSC – needs smooth (=FFT?)
- tp2vis – needs np.fft.fft2() and sm.predict()
-

Extending CASA
[could change in future CASA revisions]

- CASA uses the **~/.casa/init.py** and **prelude.py** much like your **.bashrc**
- **This is where you can automatically add new functionality**
- Several methods are available, each with caveats
  - Python's sys.path.append() [e.g. au]
    - this is not part of CASA yet
  - CASA's buildmytask() procedure [e.g. SD2VIS]
    - The manual claims this might become deprecated, but is widely used in Nordic Tools
  - Python's execfile() [e.g. tp2vis]
    - The **execfile(filename)** command does not exist in python3
    - It would be **exec(compile(open(filename, "rb").read(), filename, 'exec'))**
  - ADMIT's "casarun" and $PATH/$PYTHONPATH shell approach that allows for "import admit"

QAC
(Quick Array Combination)

- Why QAC when you have CASA?
  - Simple layer to CASA, but....
  - Write shorter scripts, but...
  - Easy to use in unix shell, but...
  - Organize Data, Comparison & Regression, but...



CASA Tasks & Tools

CASA Tasks and Tools

TASK:

```
CASA> imhead('ngc1234.fits')
```

TOOL:

```
CASA> ia.open('ngc1234.fits')
CASA> ia.summary()
CASA> ia.close()
```

CASA tasks/tools vs. QAC functions

```
CASA:
 immath([a,b],'evalexpr',c,'IM0+IM1')


QAC:
 qac_math(c,a,'+',b)
```

QAC

scripting: calling as a unix command

```
% casa -c sky1.py plot=1 maxcfg=4


% casa -c sky1.py test='"test2"'
      alma=1 plot=1
      niter='[0,100,300,1000,3000]'
      maxcfg=8 grid=10.0 dish=30.0
      > sky1a.log 2>&1
```

QAC – Creating data

- qac_alma(P, PIM, cfg, ptg)
- qac_tp_vis(P, PIM, ptg)
- qac_sd_vis(P,...)
- qac_tp_otf(P, PIM)

CASA::simobserve()

P = Project Directory
PIM = pixel image (jy/pixel)
cfg = array configuration
ptg = pointing grid (mosaic)

QAC – Cleaning Data

- qac_clean1(P, [ms],niter=[1,10,100],**line)
- qac_clean(P, tpms, [ms],niter=[1,10,100],**line)

P = Project Directory
PIM = pixel image (jy/pixel)
cfg = array configuration
ptg = pointing grid (mosaic)
[ms] = MS list
tpms = TP ms from tp2vis

QAC – Combining Data

- qac_combine(P, TPdata, INTdata)
- qac_clean(P, tpms, [ms], niter=[1,10,100],**line)
- qac_feather(P, hires, lores)
- qac_ssc(P, hires, lores)

QAC – Plotting Data

- qac_plot(cim)
- qac_plot_grid([cim], diff=10.0)
- qac_beam(psf)
- qac_psd(cim)
- qac_flux(cim)
- qac_stats(cim, test=)

P = Project Directory
PIM = pixel image (jy/pixel)
cfg = array configuration
ptg = pointing grid (mosaic)
[ms] = MS list
tpms = TP ms from tp2vis
cim = image

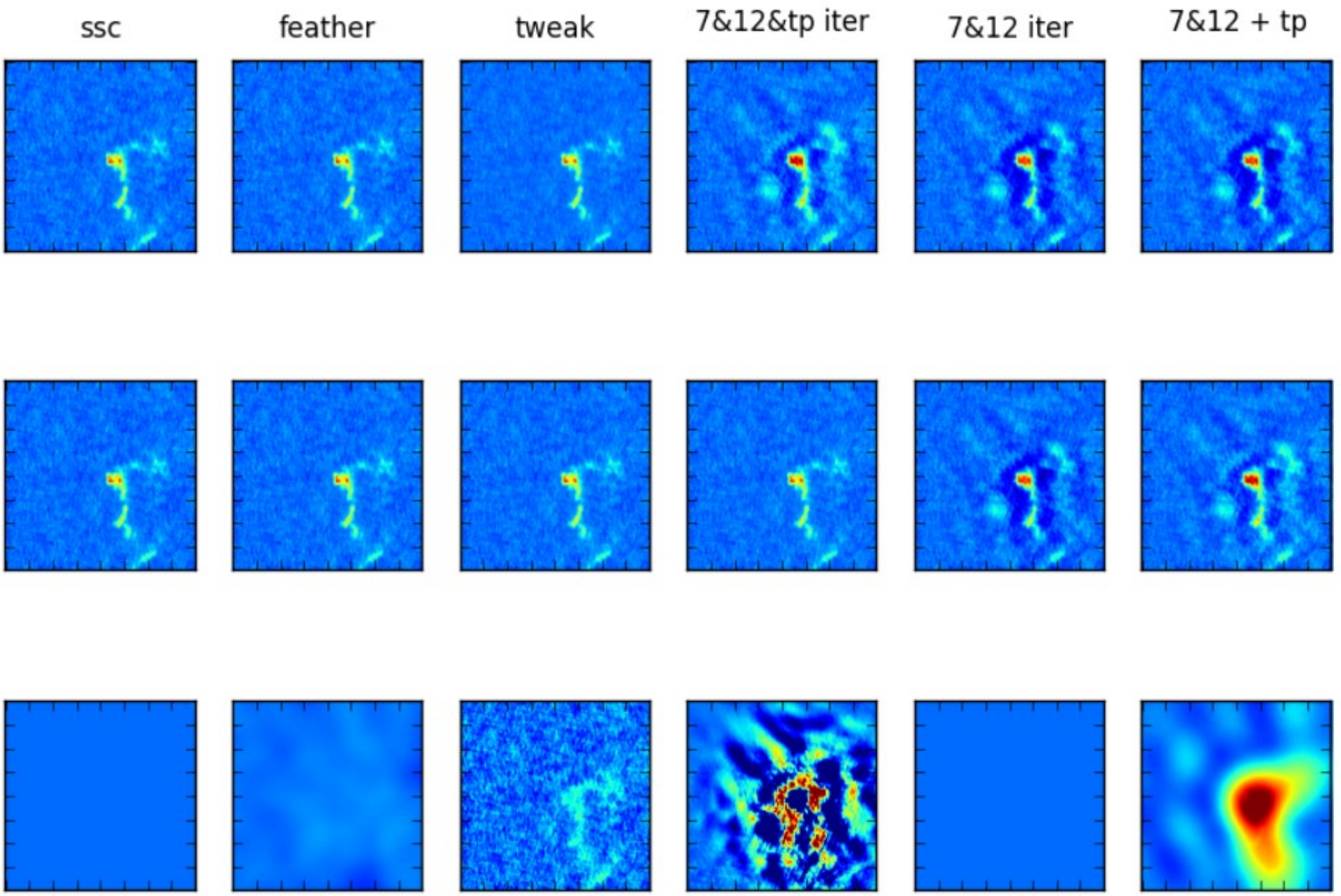
QAC – Analysis/Misc

- qac_mom(cim)
- qac_smooth(P, ...)
- qac_math(out,in1,oper,in2)
- qac_stats(cim, test=)

Scripts

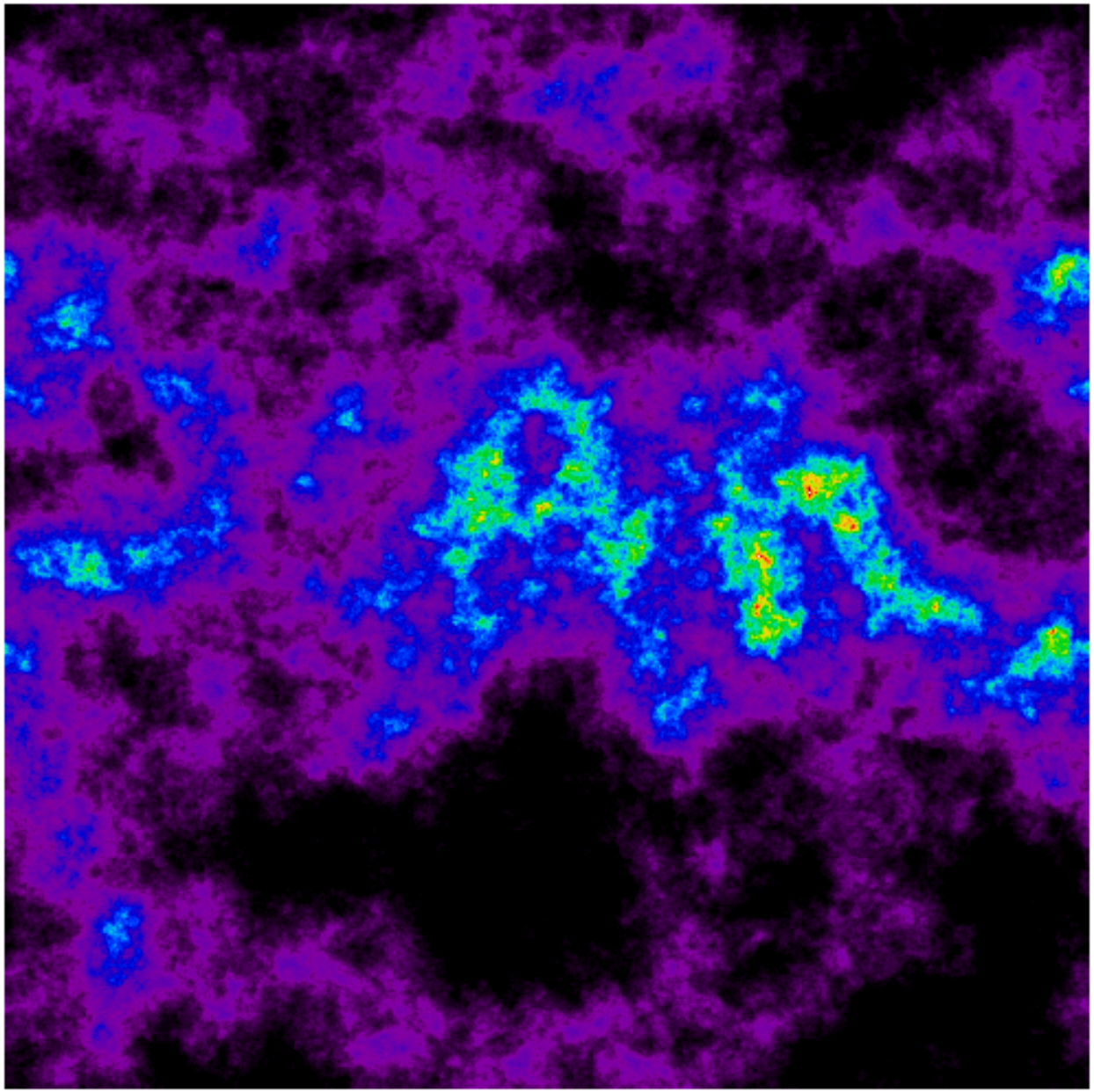
- SD2018/casa
  - `M100Band3Combine5.1.py` [NOT WORKING YET]
- QAC/test
  - `bench.py` - std M100 bench w/ 5 channels – talkes ~ 3 mins
  - **`bench0.py`** - tinker toy M100 w/ 1 channel
  - `sky1.py` – tinker toy skymodel.fits
- QAC/workflows [used for tp2vis project, most use QAC]
  - `workflow4.py` ~30 min
  - `workflow6a.py` (CASA Guide fixed for casa5) : ~60 min
  - `workflow6.py` ~70 min

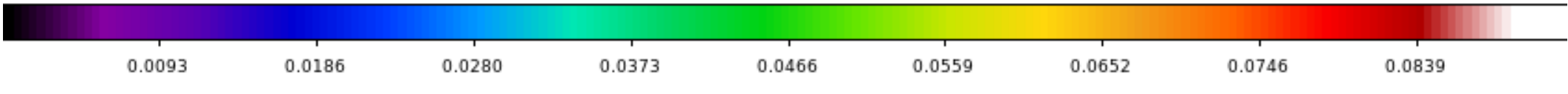
  `example1.py` (no QAC, vanilla CASA)
-


ssc
feather
tweak
7&12&tp iter
7&12 iter
7&12 + tp

Preparations (prevent data avalanche, YMMV)

- Combining data means
  - Selecting spatially
    - split(ms1,ms2,field='M100',spw='3,5',datacolumn=)
  - Spatial match
    - imregrid(input, template, output)
  - Spectral match
    - mstransform(spw=, outframe='LSRK',**line)




0.0093
0.0186
0.0280
0.0373
0.0466
0.0559
0.0652
0.0746
0.0839

Tutorial Session

2h + 1.5h or 1.5h + 2h

?

- 13:00 – 13:05 : git on github **“why git?”**
- 13:05 – 15:00 : common exercises **“repeat after me”**
  - Feather and tp2vis combination
- 15:30 – 17:00 : group exercices **“hack hour”**
  - What-IF Exploratory exercises
    - Scale factors
    - Weight factors
    - Influence of choice or arrays and dish sizes
    - Current Algorithms (feather, tp2vis, sd2vis, ssc)
    - New Algorithms (stani)

Common Exercises

- First: bench0.md (i.e. bench0.py)
  -
- Workflow6a (M100 casaguide on feather)
  -
- Workflow6 (M100 with tp2vis joint deconvolution)
  -
- Workflow4 (skymodel jy/pixel)
  -

Possible Exercises

- Workflow6a:
  - Convert clean to tclean
  - Use tclean's new automasking
  - Use tclean's multi-scale
  - Play with Briggs' robust weighting
  - Use qac_mom() and study mom0, mom1 and mom2
    - Mom0=flux Mom1=dynmass Mom2=turbulence
- Workflow6 (Example1)
  - Play with weights and watch different beams, influence on total flux
  - Are tweak maps between different niters' similar?

Possible Exercises

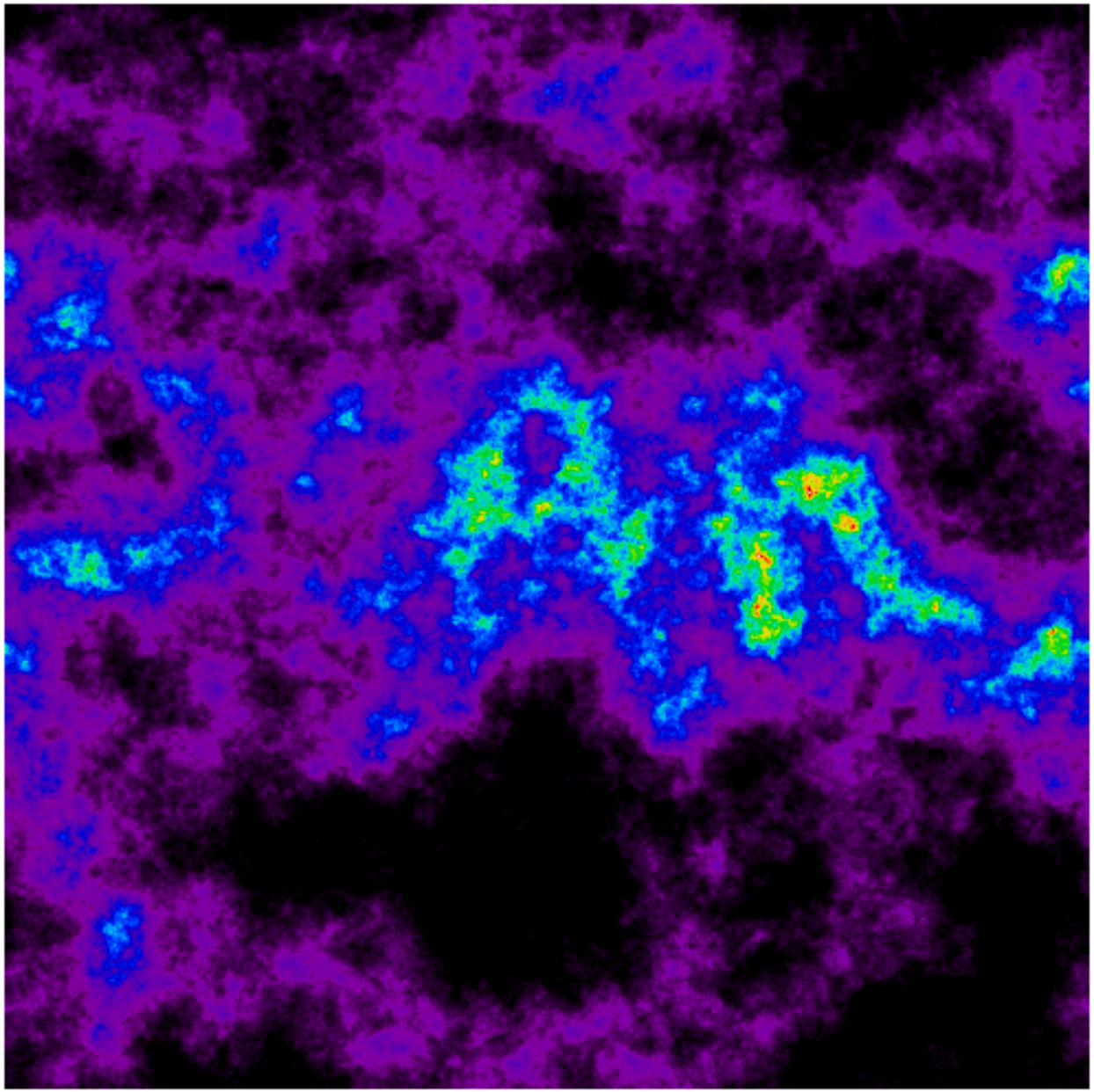
- Workflow4:
  - Compare AMPS on different arrays (tpvispl), what happens if we scale the TP by 10 or 20 %
    - The parameter **tp_scale** can be played with
- Compare qac_feather() with qac_ssc()
- Add bigger SD? (cf. ATLAST)
- Can we combine without the 7m?
  - cf. GILDAS
- Can we scale down the 7m by factor 2 ?
  - It “seems” the 7m amplitudes too high?

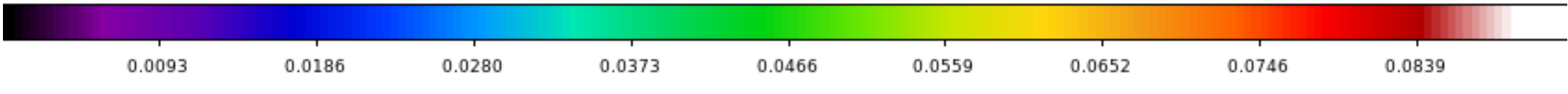
Possible Exercises

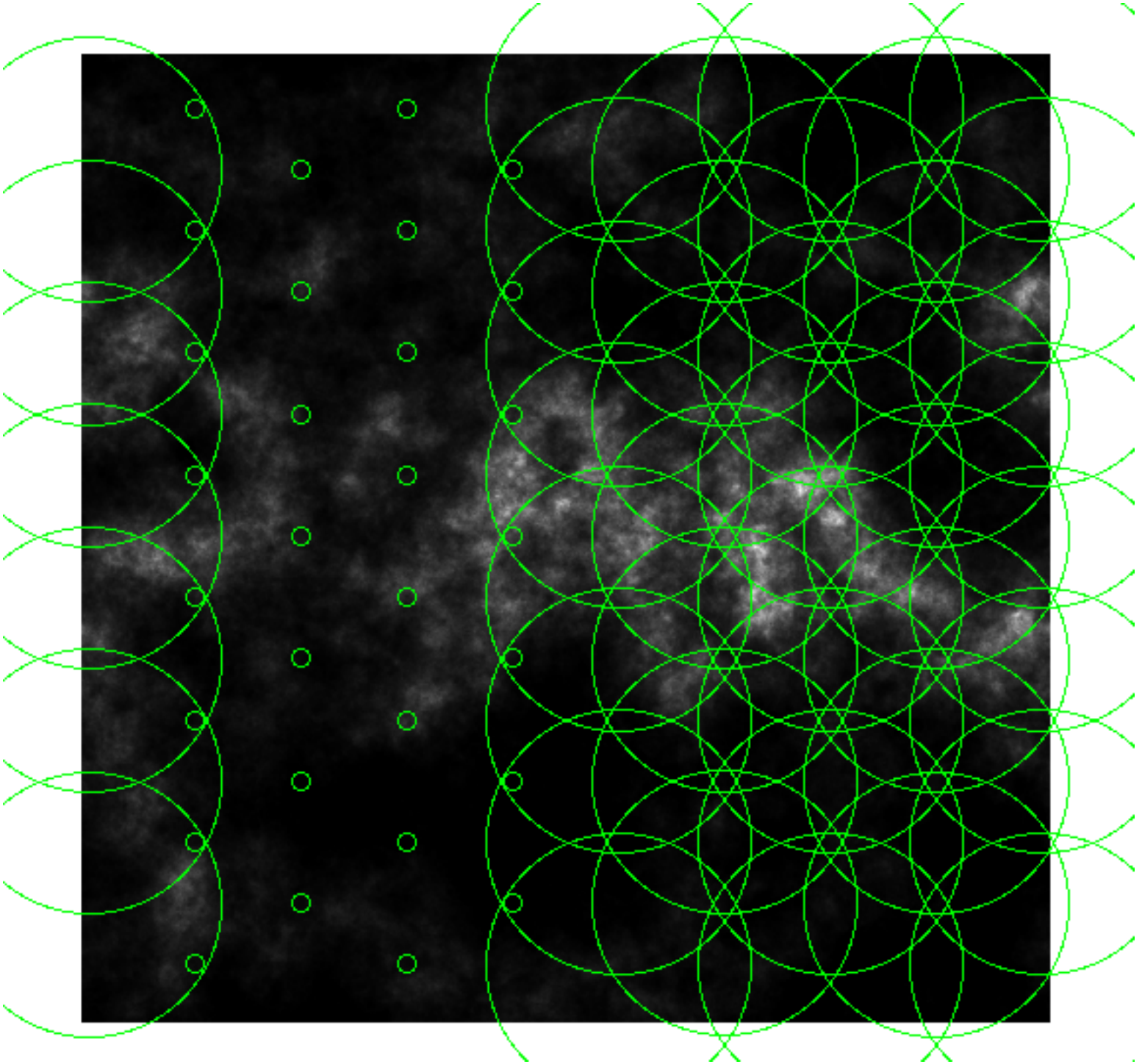
- QAC @todo
  - Add a mom2 option to qac_mom()
  - Use radialProfile in qac_beam() [cf. qac_psd()]
  - Implement qac_stani()

Neat Tricks

- If you have enough memory on linux, use /dev/smem/$USER
  - See sd2018/casa/qac_bench.txt
  - How to do this on mac?




0.0093
0.0186
0.0280
0.0373
0.0466
0.0559
0.0652
0.0746
0.0839



The missing short spacing Problem

- Visibilities → (fft) → Image
- Missing short spacings (large scale structures)
  - Merge two different resolution images (using fft)
    - MIRIAD::immerge
    - AIPS::imerge
    - **CASA**::feather
  - Virtual interferometer + Joint Deconvolution (using clean)
    - MIRIAD::uvrandom+demos+uvmodel
    - **CASA**::tp2vis →this work; ALMA development study (PI: Koda)
- Fluxes (“Jy/beam”)

Comparing (1/2)
Apples and Oranges?

- Array Combination methods:
  - Image Feathering – use only images
  - Joint Deconvolution – use only visibilities
  - Hybrids (e.g. give TP as startmodel= for tclean) – e.g. Kauffman
- Clean options (weighting, robust, gridder, automask, ...)
- Flux Conservation (e.g. Jörsäter & van Moorsel 1995)

Comparing (2/2)
Apples and Oranges?

- Mosaicing options:
  - CASA::clean(imagermode=,ftmachine=)
    - Clean each field, then use “lm”
    - Clean each field, combined deconvolution (mosaic/ft)
    - Combine in UV, fft, clean the mosaic (mosaic/mosaic)
  - CASA::tclean(gridder=)
    - gridder=mosaic
    - gridder=imagemosaic (uses lm?)

Comparing
Apples and Oranges?

- Array Combination methods:
  - Image Feathering – use only images
  - Joint Deconvolution – use only visibilities
  - Hybrids (e.g. give TP as startmodel= for tclean) – e.g. Kauffman
- Clean options (weighting, robust, gridder, automask, ...)
- Flux Conservation (e.g. Jörsäter & van Moorsel 1995)
- Mosaicing options:
  - CASA::clean(imagermode=,ftmachine=)
    - Clean each field, then use “lm”
    - Clean each field, combined deconvolution (mosaic/ft)
    - Combine in UV, fft, clean the mosaic (mosaic/mosaic)
  - CASA::tclean(gridder=)
    - gridder=mosaic
    - gridder=imagemosaic (uses lm?)

Interferometry Data
only exists between **UV_min** and **UV_max**

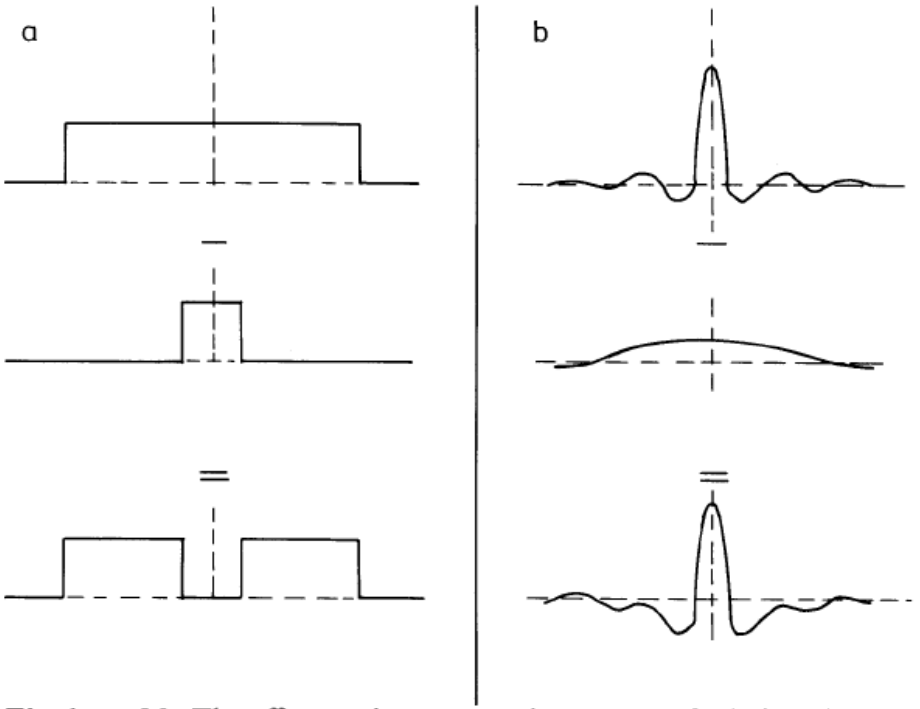
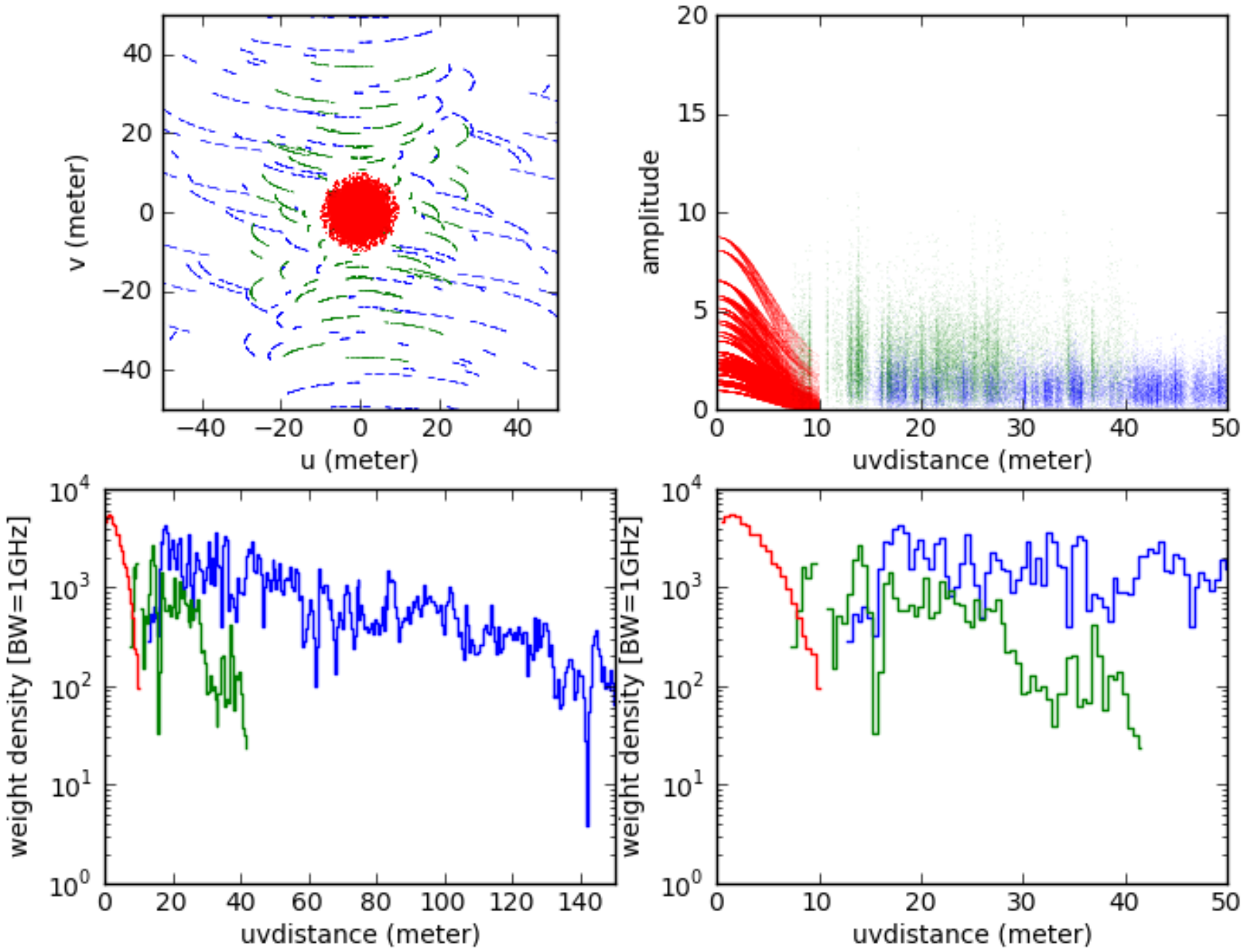


Fig. 1a and b. The effect on instrumental response of missing short spacings. **a** Observed spatial frequencies and **b** the corresponding instrumental response

UV LM

Braun & Walterbos (1985)


v (meter)
u (meter)
amplitude
uvdistance (meter)
weight density [BW=1GHz]
uvdistance (meter)
weight density [BW=1GHz]
uvdistance (meter)

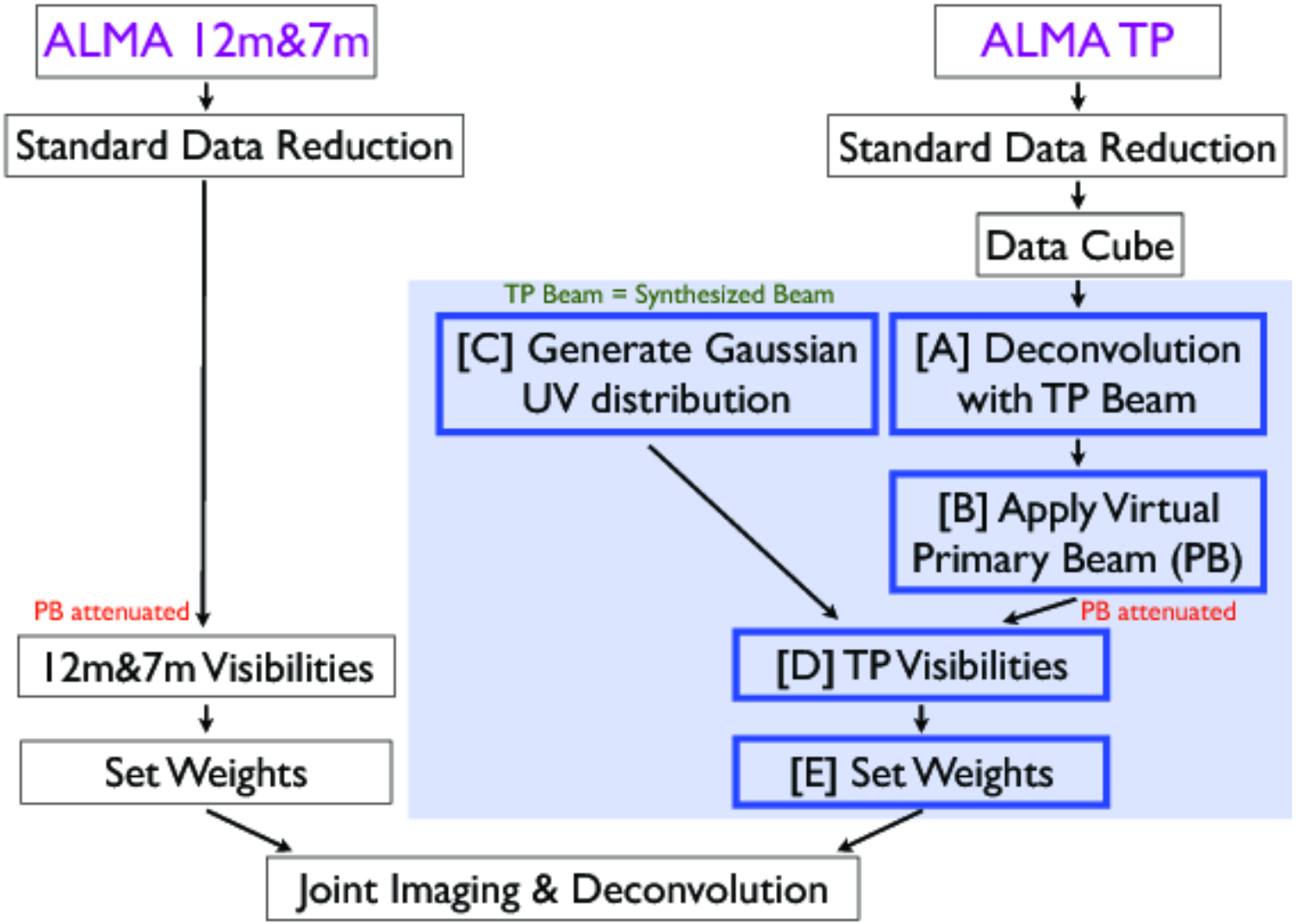
Abbreviations

- SPP = Short Spacing Problem
- SSC = Short Spacing Correction
  - SD = Single Dish
  - TP = Total Power
  - OTF = On The Fly map (“gaussian smoothing”)
  - VP = Voltage Pattern
  - PB = Primary Beam (PB = VP**2)
  - MS = Measurement Set
    - Pseudo-Visibilities
    - Virtual Visibilities
  - JD = Joint Deconvolution

ALMA dishes

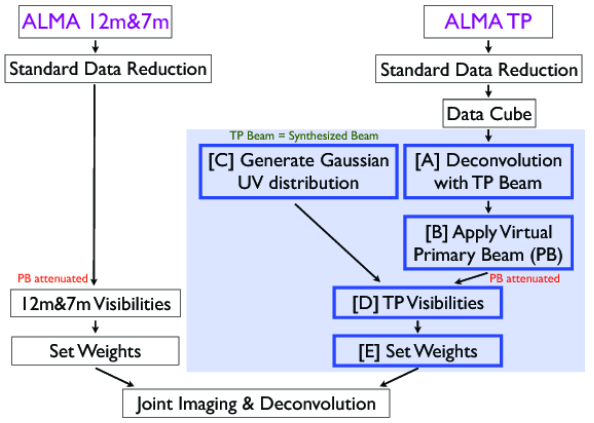
- 12m: **10.7m** Airy dish w/ 0.75m blockage
- 7m: **6.25m** Airy dish w/ 0.75m blockage

Flow Chart of the **TP2VIS** procedure



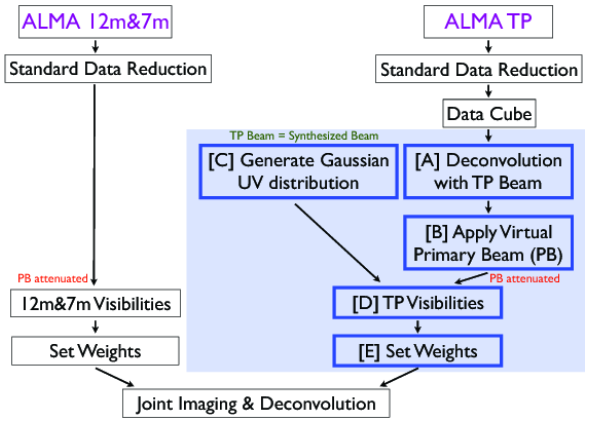
Step A:
Deconvolution with TP beam

- Input TP map is sky convolved with TP beam →
- Map needs to be deconvolved with TP beam for input to the virtual interferometer
  - ALMA: gaussian 56.6” at 115.2 GHz
    - To the rescue: VP manager?
  - Non-ALMA (e.g. GBT, VLA) can also be used



Step B:
Apply Virtual Primary Beam

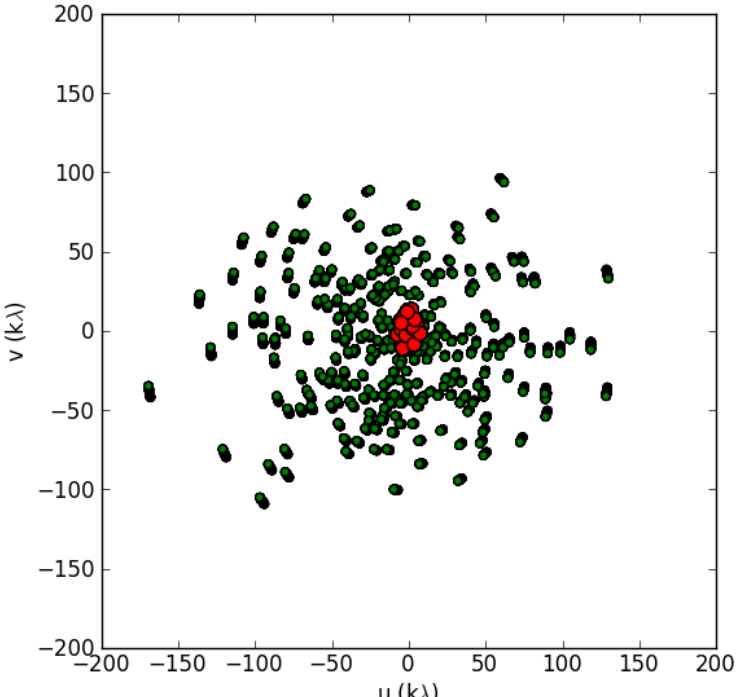
- A selected virtual primary beam is applied to all pointings selected for “observing”
- Pointings can be picked from:
  - Another (interferometric) data set (MS)
  - An ascii file with RA,DEC positions
  - An auto-filled set of pointings e.g. using hex-pattern nyquist sampling

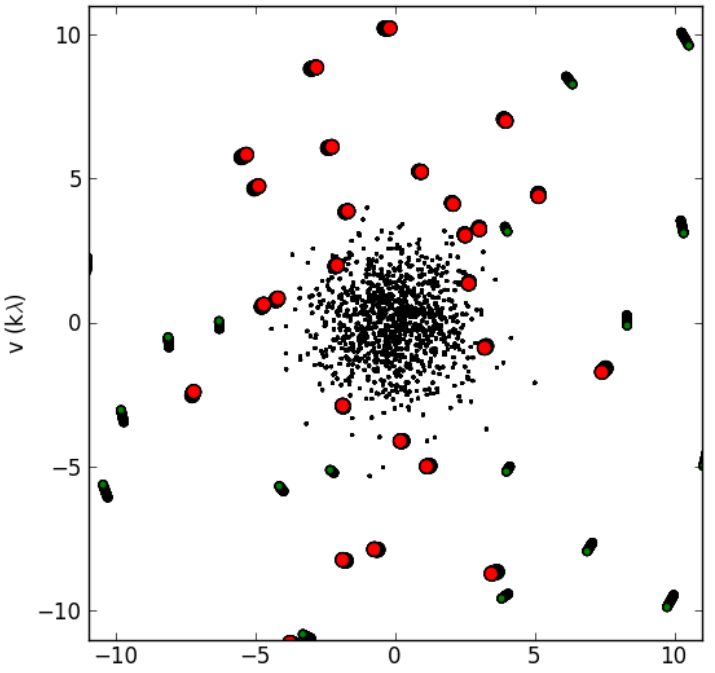


Step C:
Generate Gaussian UV Distribution

- Random gaussian distribution UV points, including (0,0) for the total flux
- Fourier transform of these points should represent the TP beam

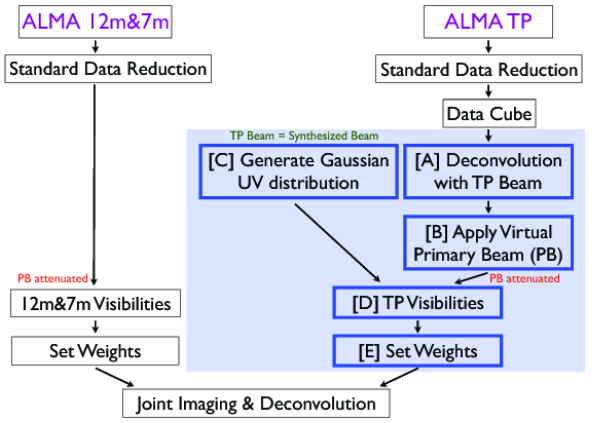
7m (red) + 12m (green) + TP (black)





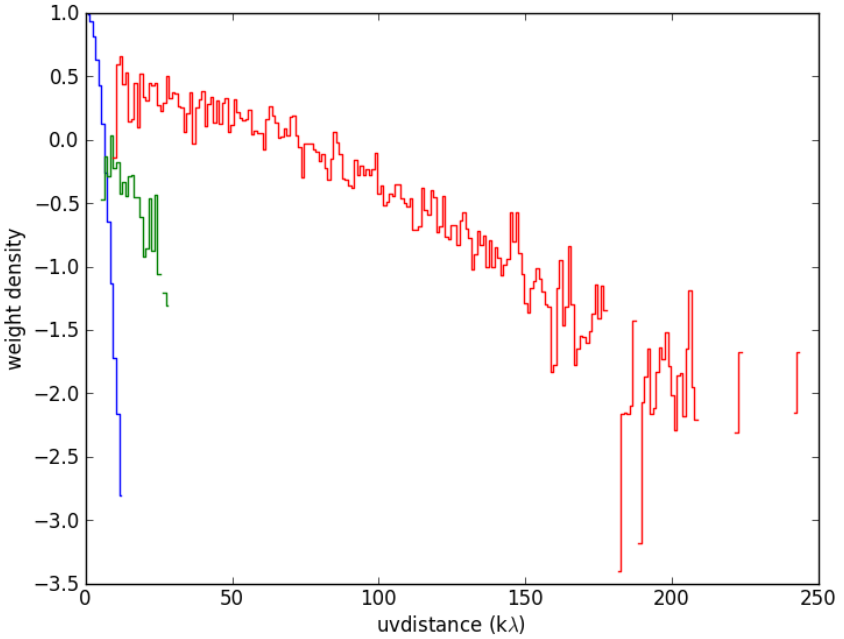
Step D:
Fill Visibility Amplitudes and Phases

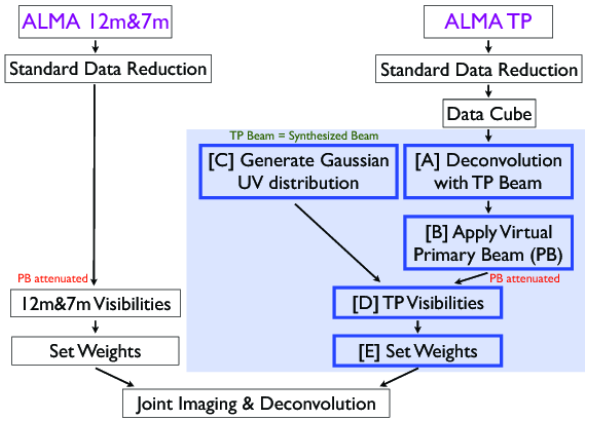
- Sky brightness from step B and spacings from step C can now predict what visibilities are “observed”
- Cleaning these visibilities with a natural weighting scheme should now result in reproducing the input image
- Weights are needed to combine with 7+12m



Step E:
Set Weights of TP Visibilities

- By default weights will be set consistent with the RMS noise in the TP cube
- "**tp2viswt**" can set different weights, e.g. match dirty beam area to the beam solid angle





Schematic Code in CASA

```
sm.open("ms file")

sm.setconfig()                # observatory, antenna pos.
sm.setspwindow()              # spectral window
sm.setfeed()                  # polarization
for "loop over pointings":
    sm.setfield()             # pointing centers
sm.setlimits()                # shadowing, el. limit
sm.setauto()                  # weight of auto-corr. = 0
sm.settimes()                 # integ. time per vis.
sm.observemany()              # generate "box" of vis.

tb.open("ms file")            # open MS file as table
uvw  = tb.getcol('UVW')       # get uvw
# Generate gaussian distributions for (u,v),
# including (u,v)=(0,0). w=0.
tb.putcol('UVW',uvw)          # put uvw back
tb.close()

sm.setdata()                  # all pointings
sm.setvp()                    # primary beam
sm.predict("image name")      # replace amp/pha

sm.done()
```

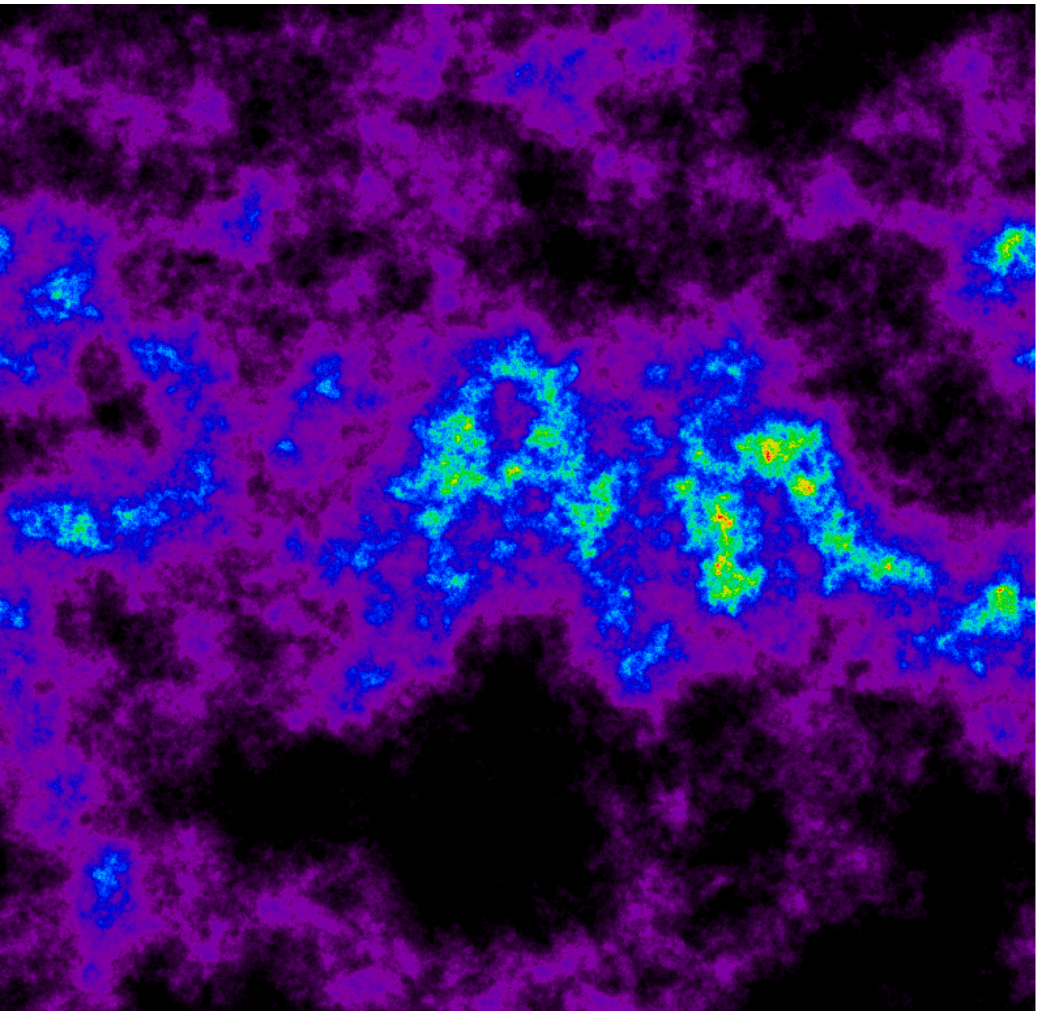
TP2VIS for CASA users is now simple!

```
#  load TP2VIS  (can also use the ~/.casa/init.py trick)
execfile('tp2vis.py')

#  convert Image to Visibilities
tp2vis('alma_tp.im','alma_tp.ms', ptg='alma_12m.ms', rms=0.7)

#  joint deconvolution using standard CASA task
tclean(vis=['alma_7m.ms','alma_12m.ms','alma_tp.ms'],imagename='alma_all',
     niter=1000, imsize=[512,512], cell=['0.5arcsec'])
```

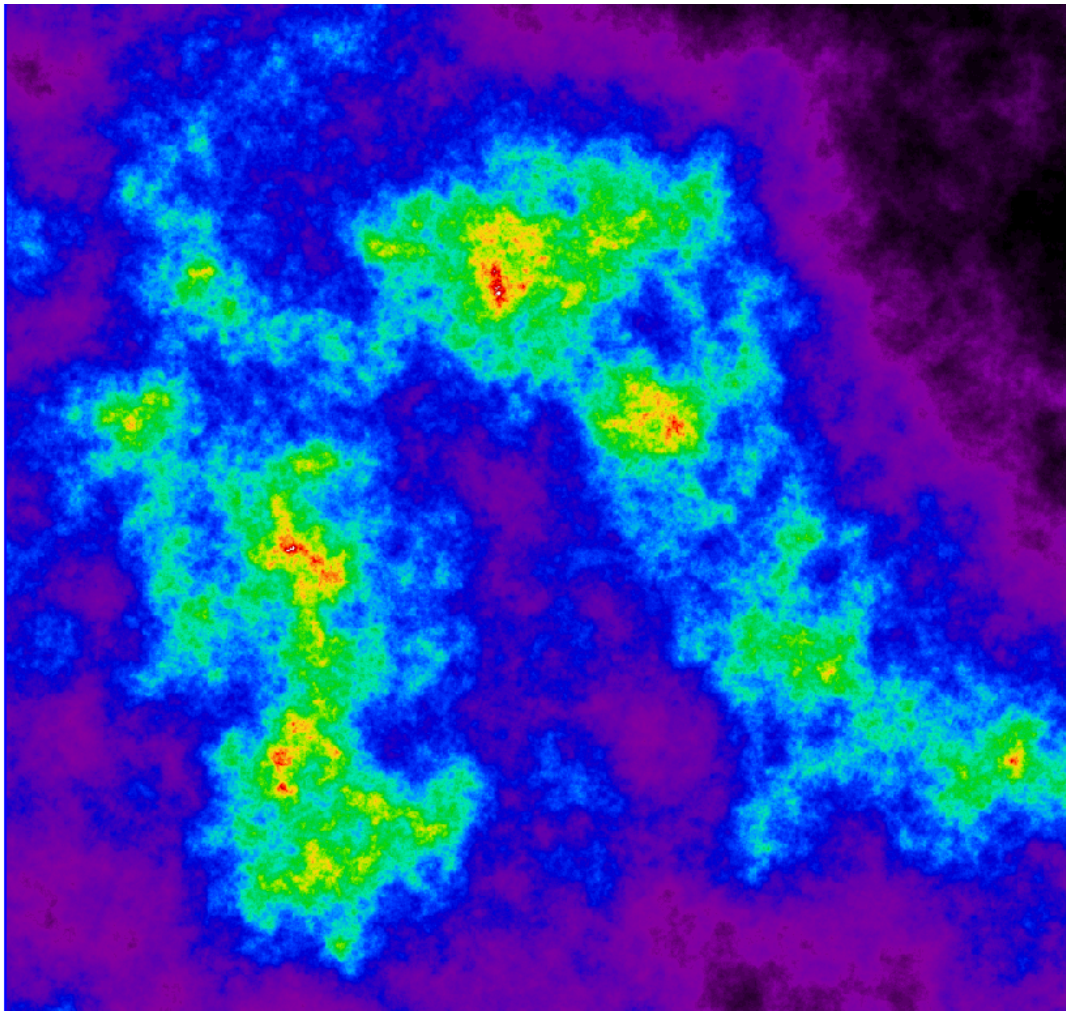
feather in CASA is also easy to use!

```
#  get two identically gridded image, lowres and highres

feather(imagename = 'final.image',
        highres   = 'highres.image',
        lowres    = 'lowres.image')
```

Simulated Data
(power spectrum of sources)

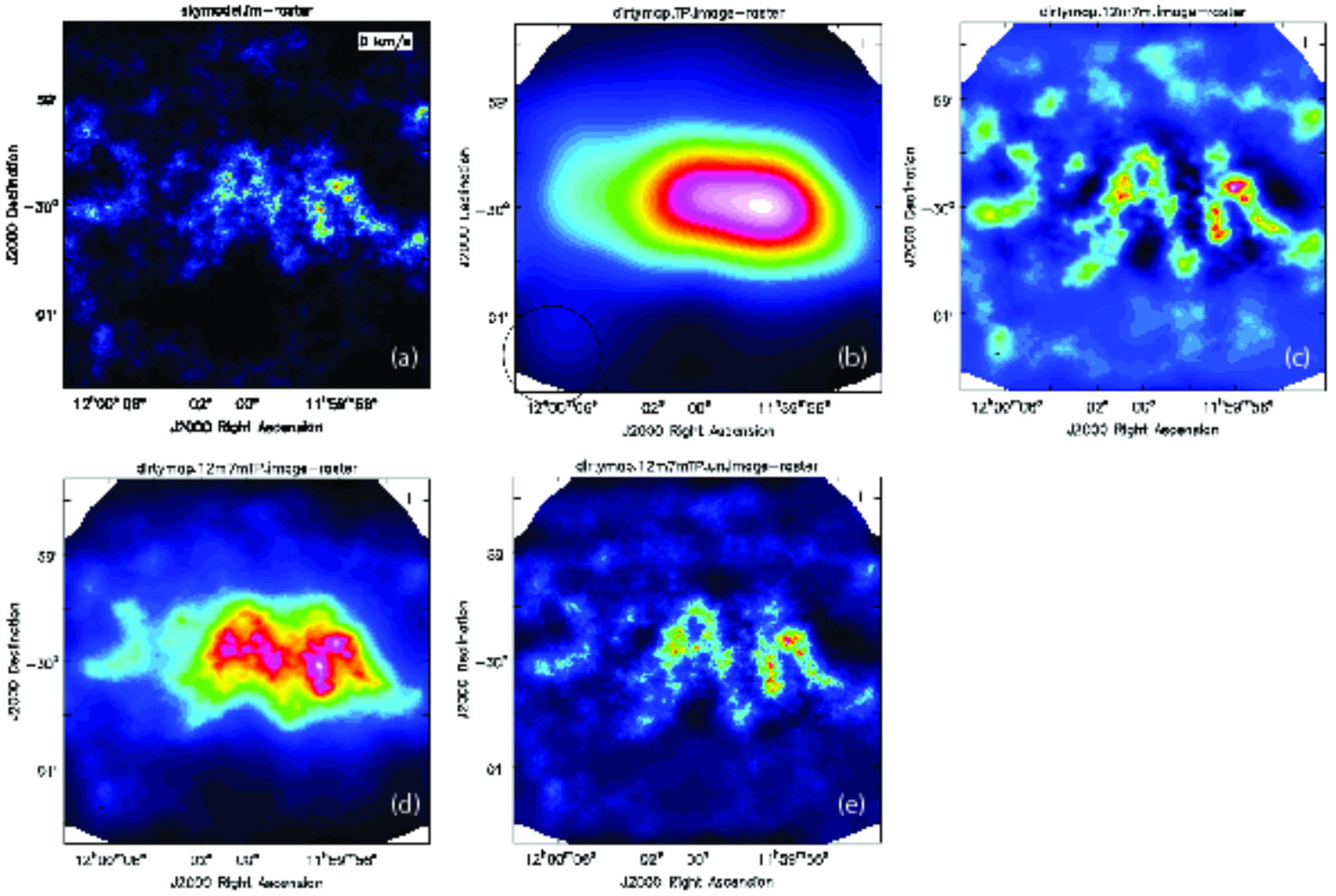


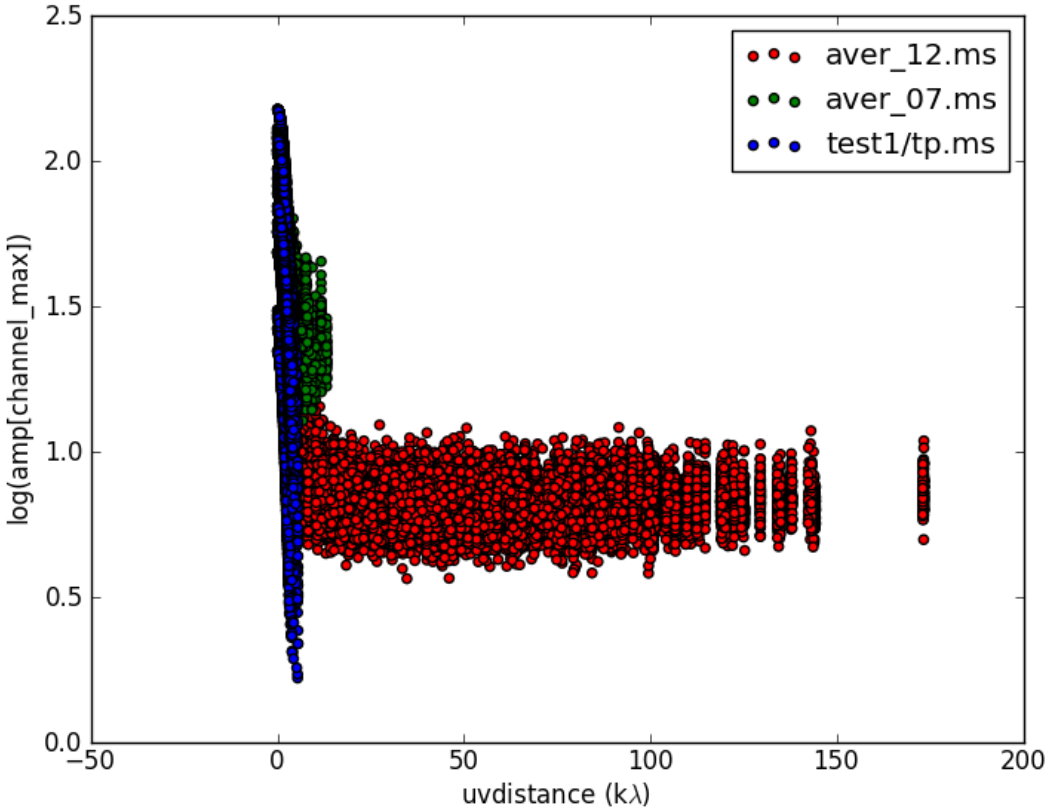
4096 x 4096



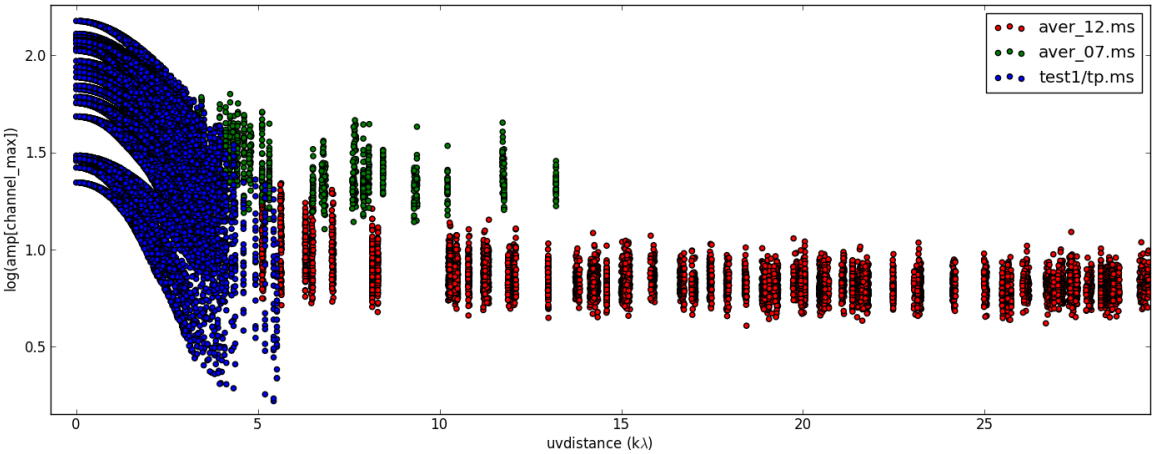
Zoom
(middle right)

Examples: simulated GMC





Visibility Amplitudes as function of UV distance

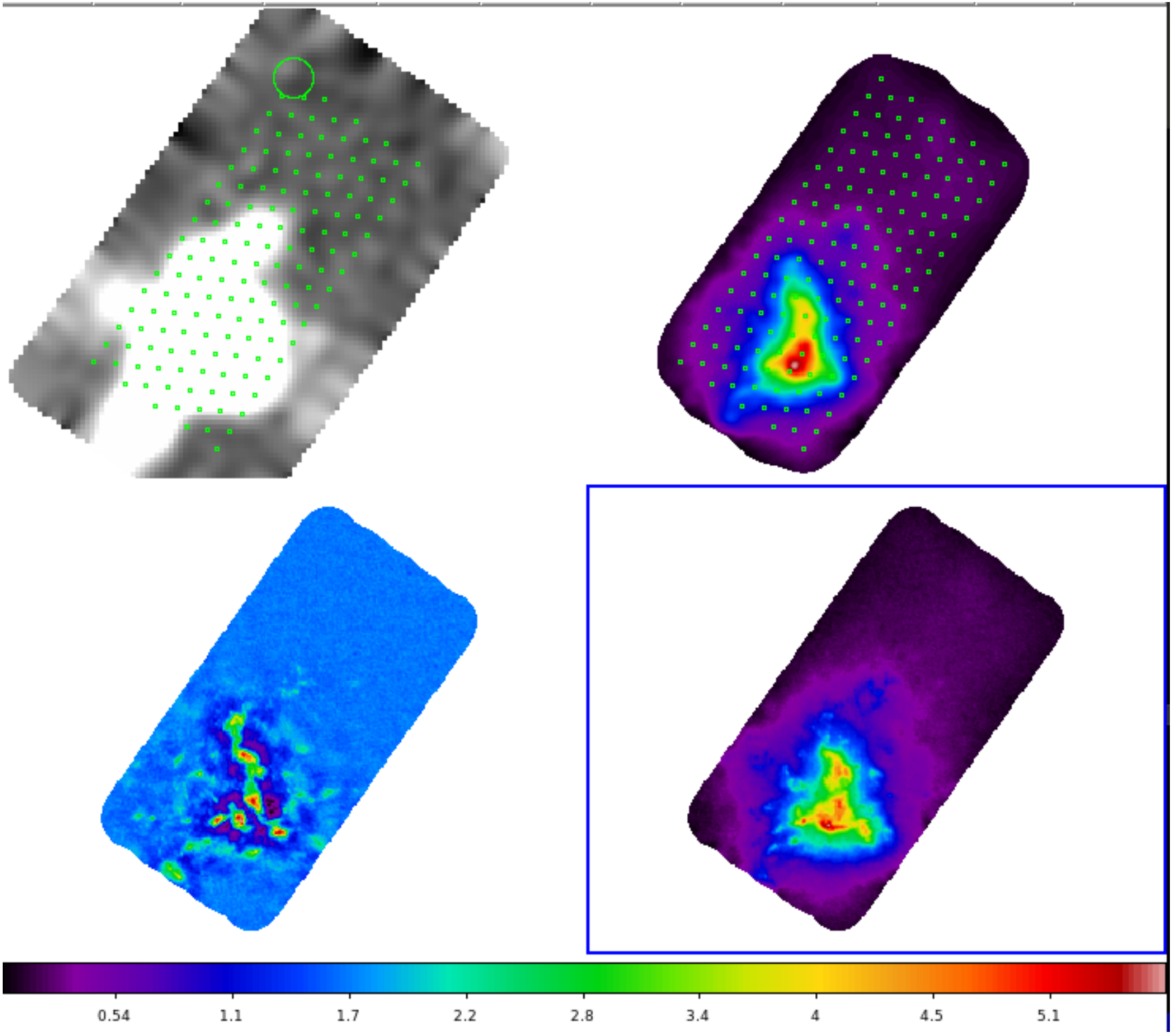


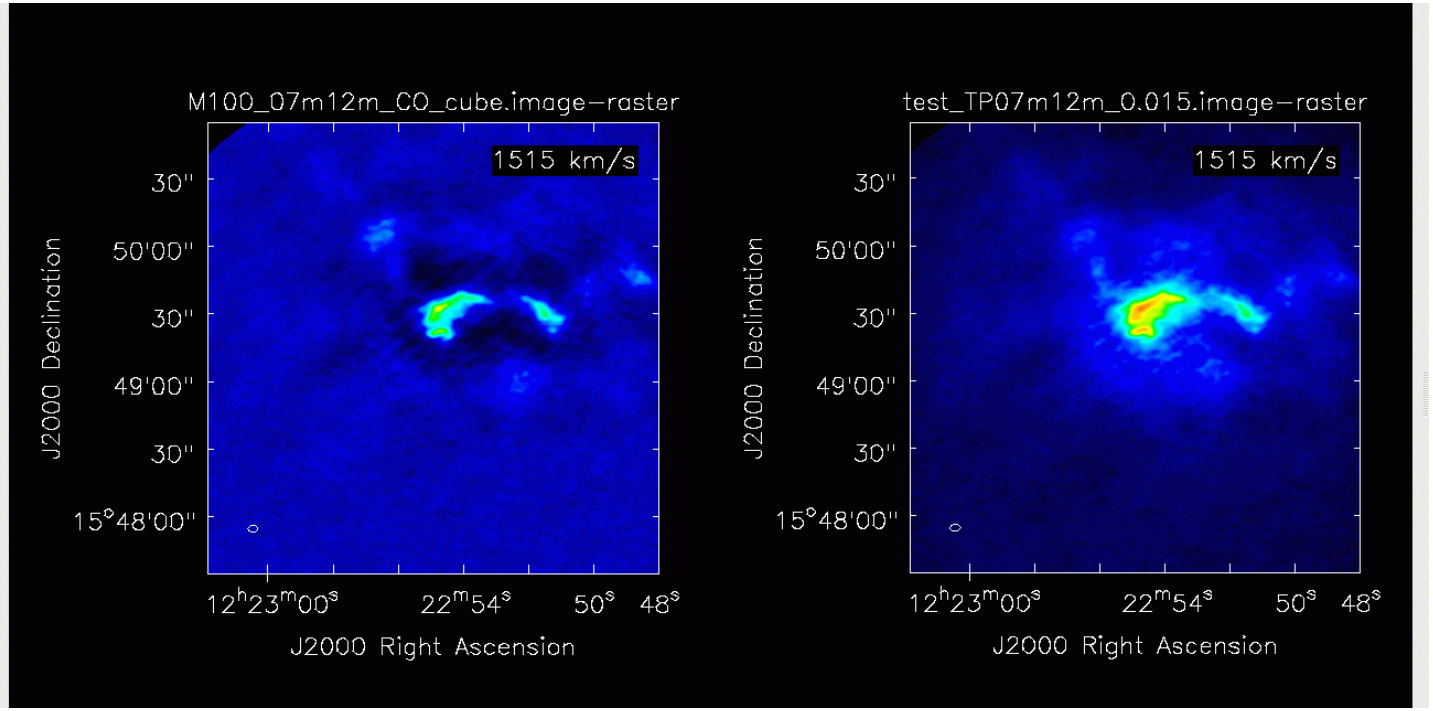
Weights

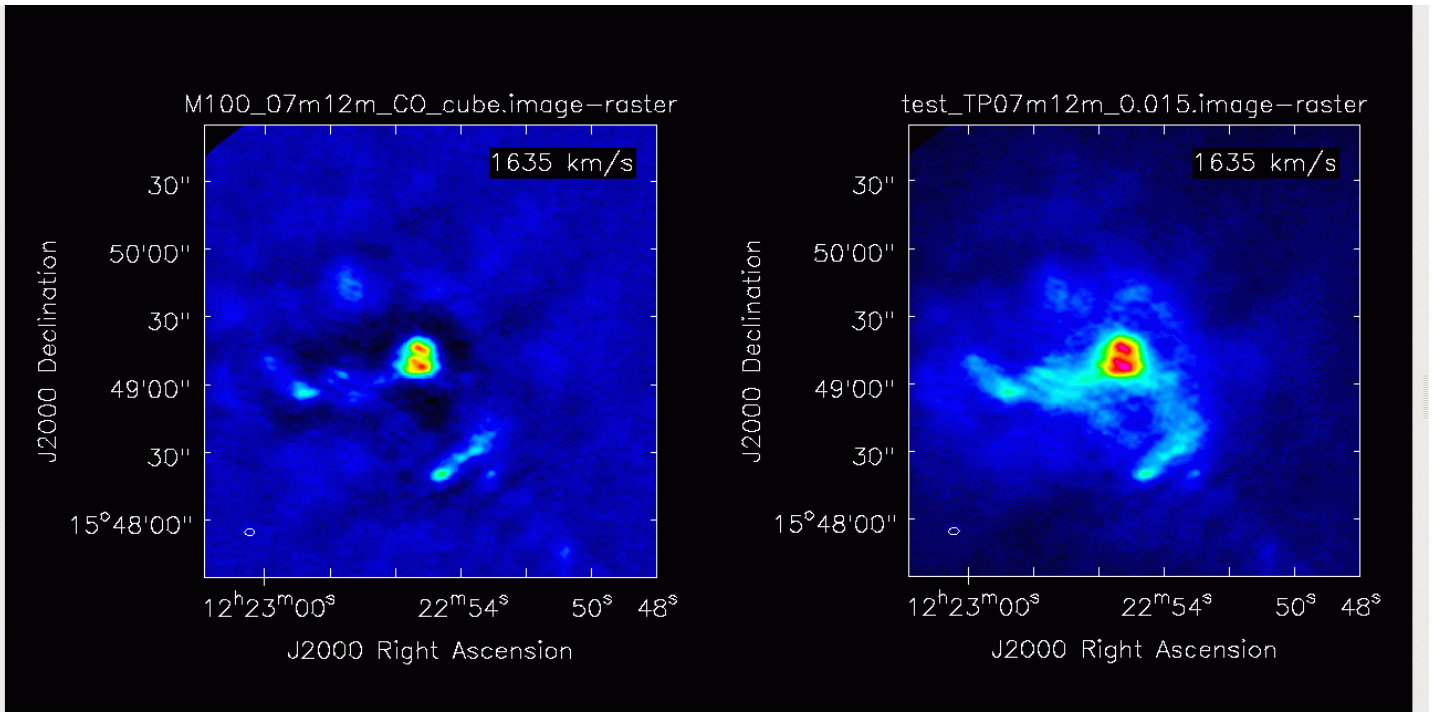
- CLEAN uses weights to give preference to certain data.
  - “natural” vs. “uniform” vs. “Briggs robust”
- We experiment with three different weights for the TP visibilities:
  - RMS from cube based (default in “tp2vis”)
  - Match synthesized beam to beam solid angle
  - Manually adjusting weights

$$\sigma_{ij}(Jy) = \frac{2k}{\eta_q \eta_c A_{eff}} \sqrt{\frac{T_{sys,i} T_{sys,j}}{2\Delta\nu_{ch} t_{ij}}} \times 10^{26}$$

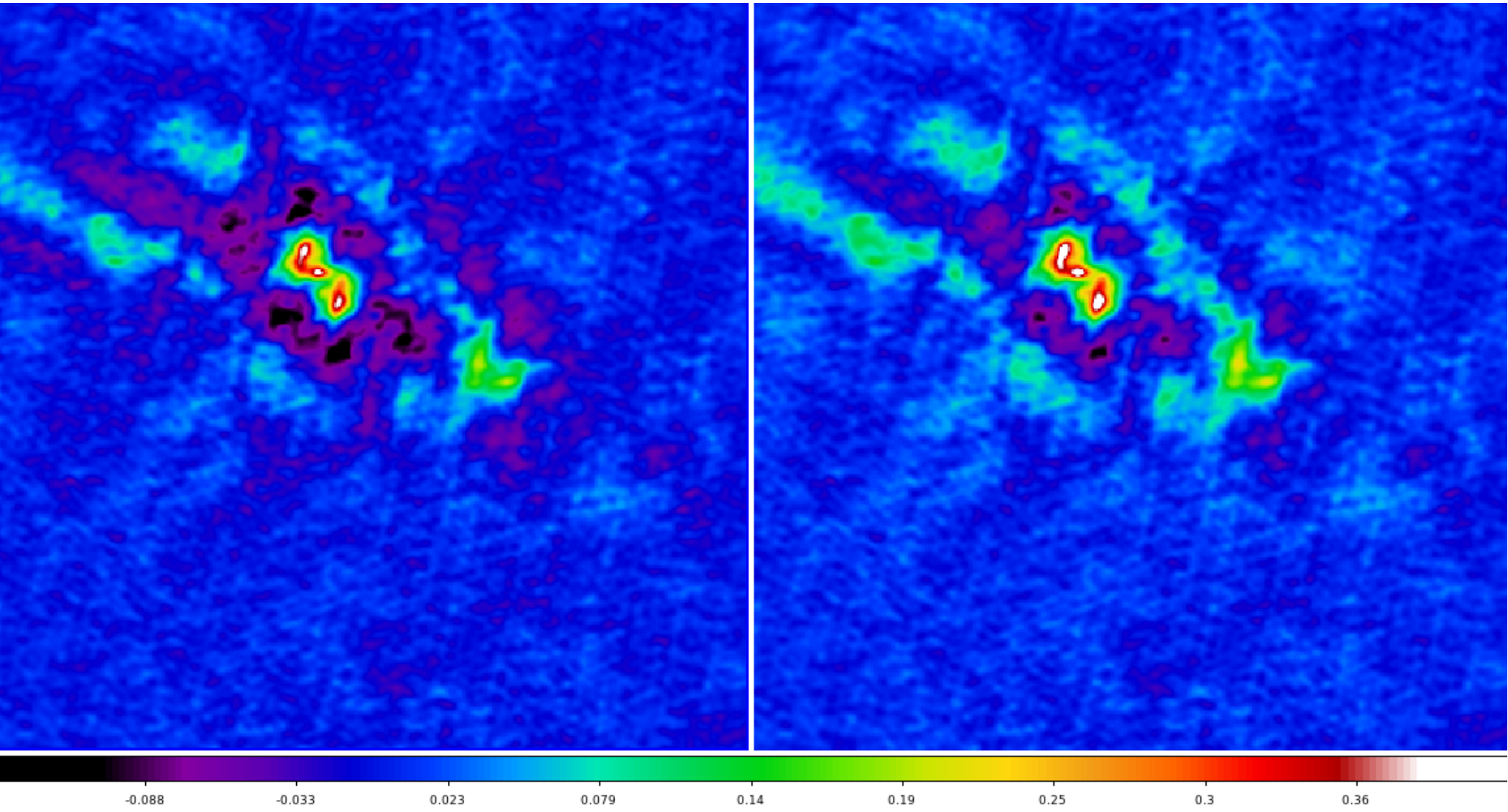
Sadly we have a “poorly” sampled UV plane


0.54
1.1
1.7
2.2
2.8
3.4
4
4.5
5.1


M100_07m12m_CO_cube.image-raster
1515 km/s
30''
50'00''
30''
49'00''
30''
15°48'00''
J2000 Declination
12h23m00s
22m54s
50s
48s
J2000 Right Ascension
test_TP07m12m_0.015.image-raster
1515 km/s
30''
50'00''
30''
49'00''
30''
15°48'00''
J2000 Declination
12h23m00s
22m54s
50s
48s
J2000 Right Ascension

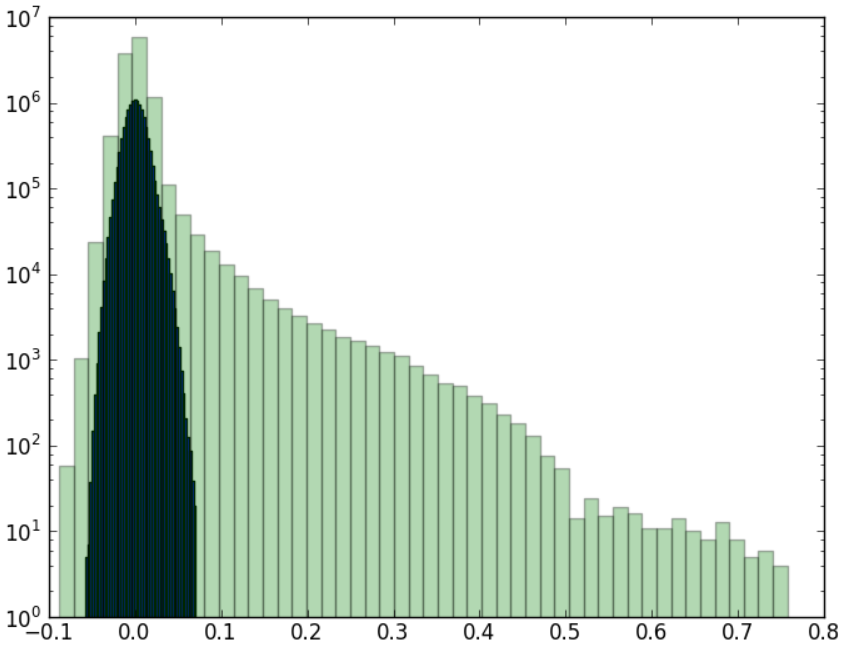

M100_07m12m_CO_cube.image-raster
1635 km/s
30''
50'00''
30''
49'00''
30''
15°48'00''
J2000 Declination
12h23m00s
22m54s
50s
48s
J2000 Right Ascension
test_TP07m12m_0.015.image-raster
1635 km/s
30''
50'00''
30''
49'00''
30''
15°48'00''
J2000 Declination
12h23m00s
22m54s
50s
48s
J2000 Right Ascension

ALMA vs. TPALMA – channel 36 – no deconvolution

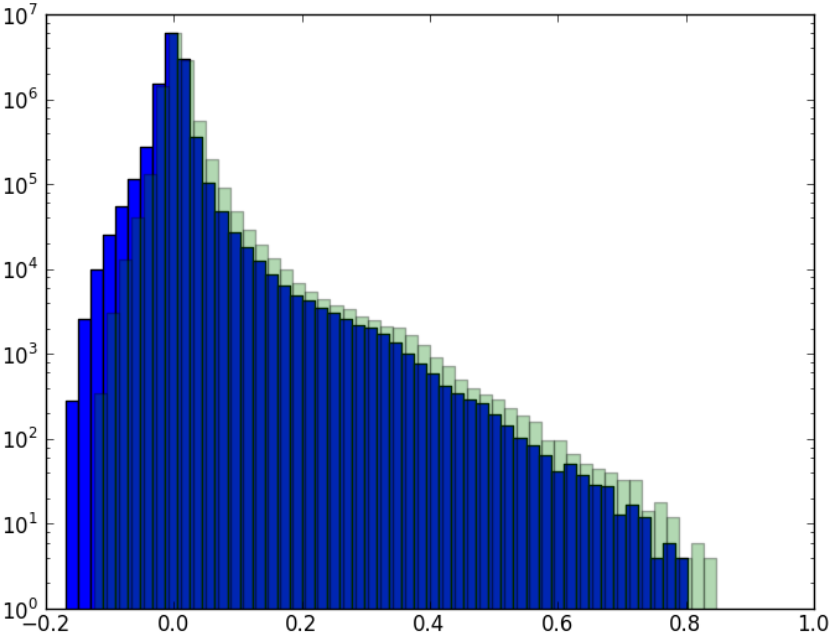


References

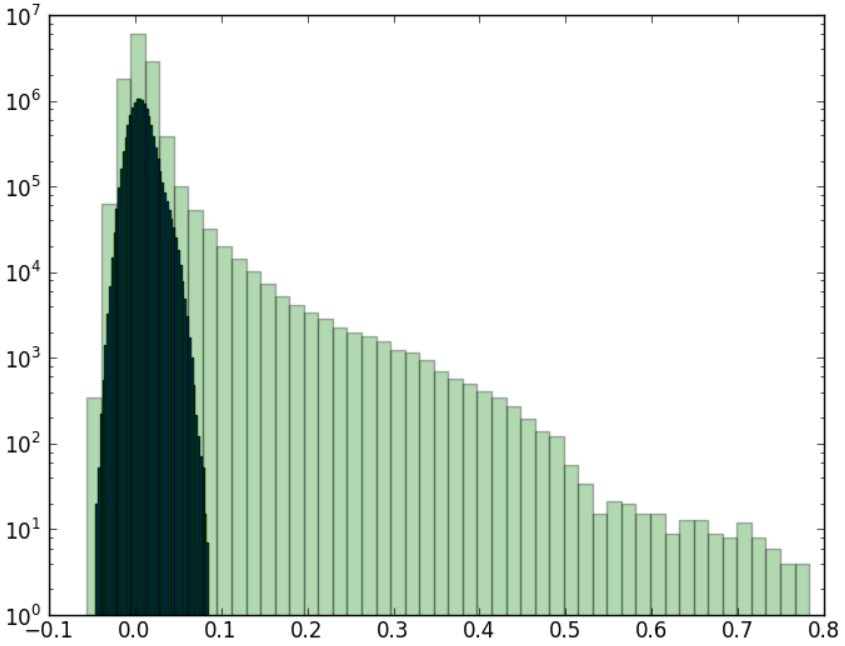
- Too many.....



ALMA: image vs. residuals



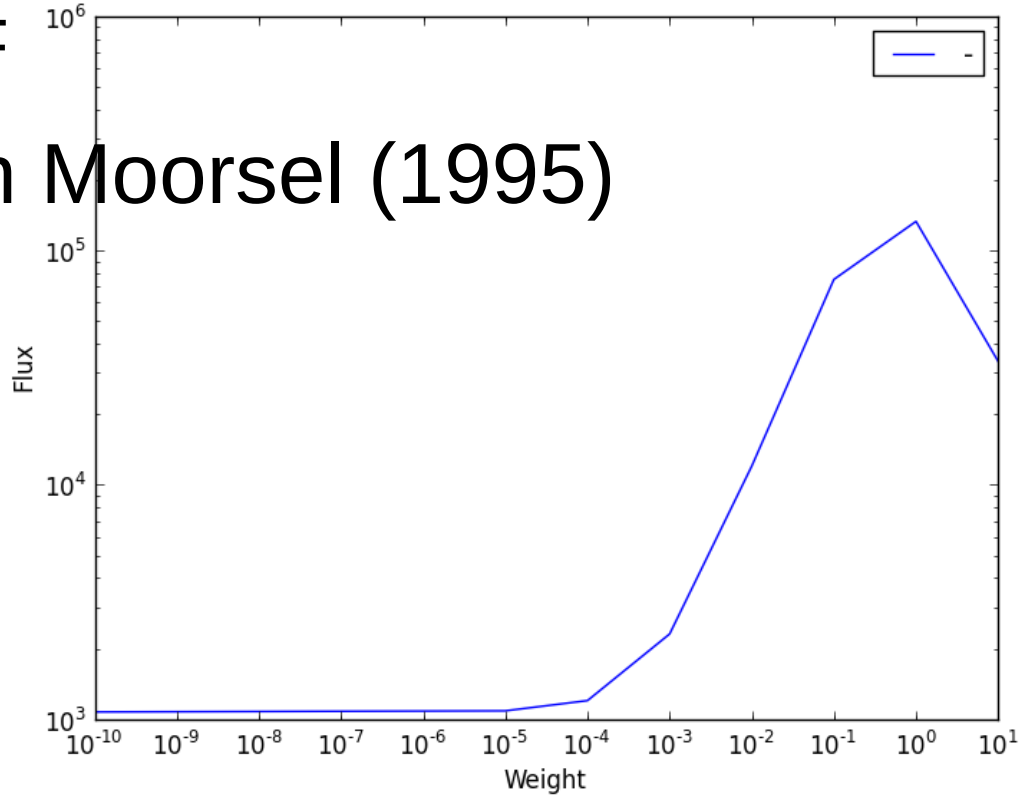
ALMA vs. TPALMA image



TP+ALMA: image vs. residuals

Issues – Conserving Flux

- imstat()['flux'] = imstat()['sum'] * Δv / Nppb
- Nppb = 1.1331 * (Bx*By)/(px*py)
- But what are really Bx and By?
  - Gauss fit to peak of PSF
- See also Jörsäter & van Moorsel (1995)



Issues – CASA

- mstransform() seems to loose first and/or last channel sometimes
- vpmanager: can’t set a private VIRTUAL?
- concat() needed before tclean([ms1,ms2]) to prevent a crash out of CASA
-

Timeline

- TP2VIS 0.6 “friendly-beta release” 12-dec-2017
  - URL: https://github.com/tp2vis/distribute
  - There is a separate developer github repo
- TP2VIS 1.0 31-Jan-2018
  - Formal NRAO delivery
  - Developer repo available as well
- Comparison study
  - See also: pjt.py: pjt_combine()

References

- Too many.....

DEMO time

loosely following example1
from the beta release

see also
https://github.com/tp2vis/distribution/example1.md