Using massively parallel platforms for wide-field wide-band imaging

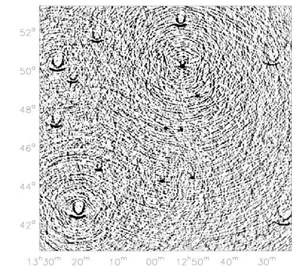
UNM, Albuquerque, April 7th 2016

S. Bhatnagar
NRAO

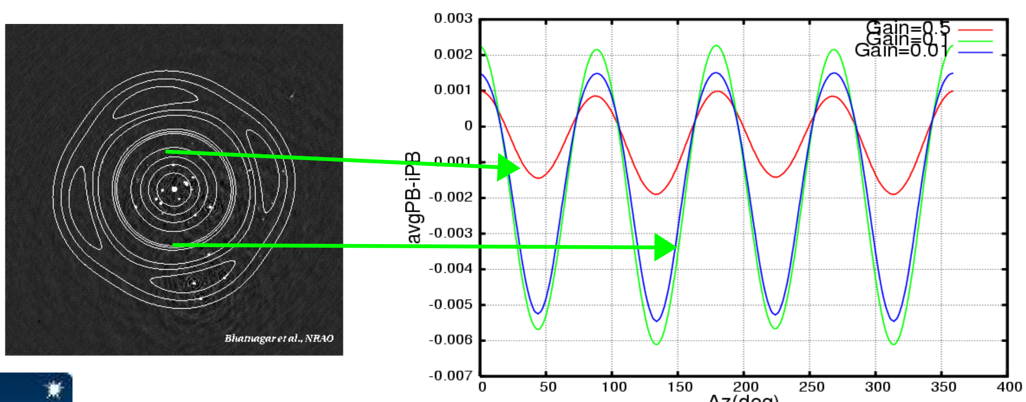
What do we call wide-field?

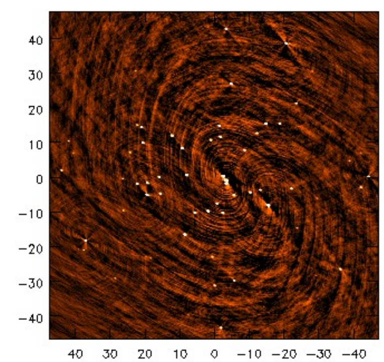
- Imaging that requires invoking any of the following:
  - Corrections for non co-planar baseline effects

$$\frac{\lambda}{B_{max}} \le \theta_f^2$$

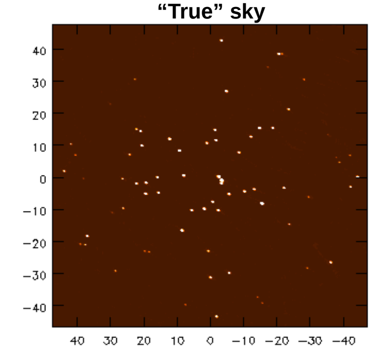


  - Corrections for the rotational asymmetry of the PB
    - Imaging beyond 50% point, mosaicking
  - Corrections for the frequency or polarization dependent effects
  - Noise limited imaging at 4-,P-,L-, S- (and probably C-Band)
    - Because of the radio brightness distribution
- Noise limited imaging of structure comparable to the PB beam-width
- Mosaicking: imaging on scales larger than the PB beam-width



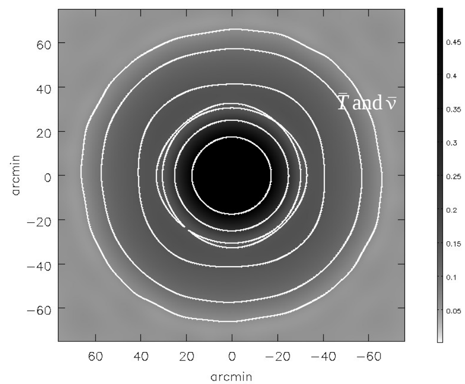


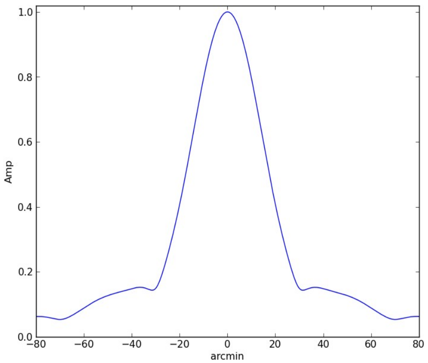
“True” sky



Why wide-field?

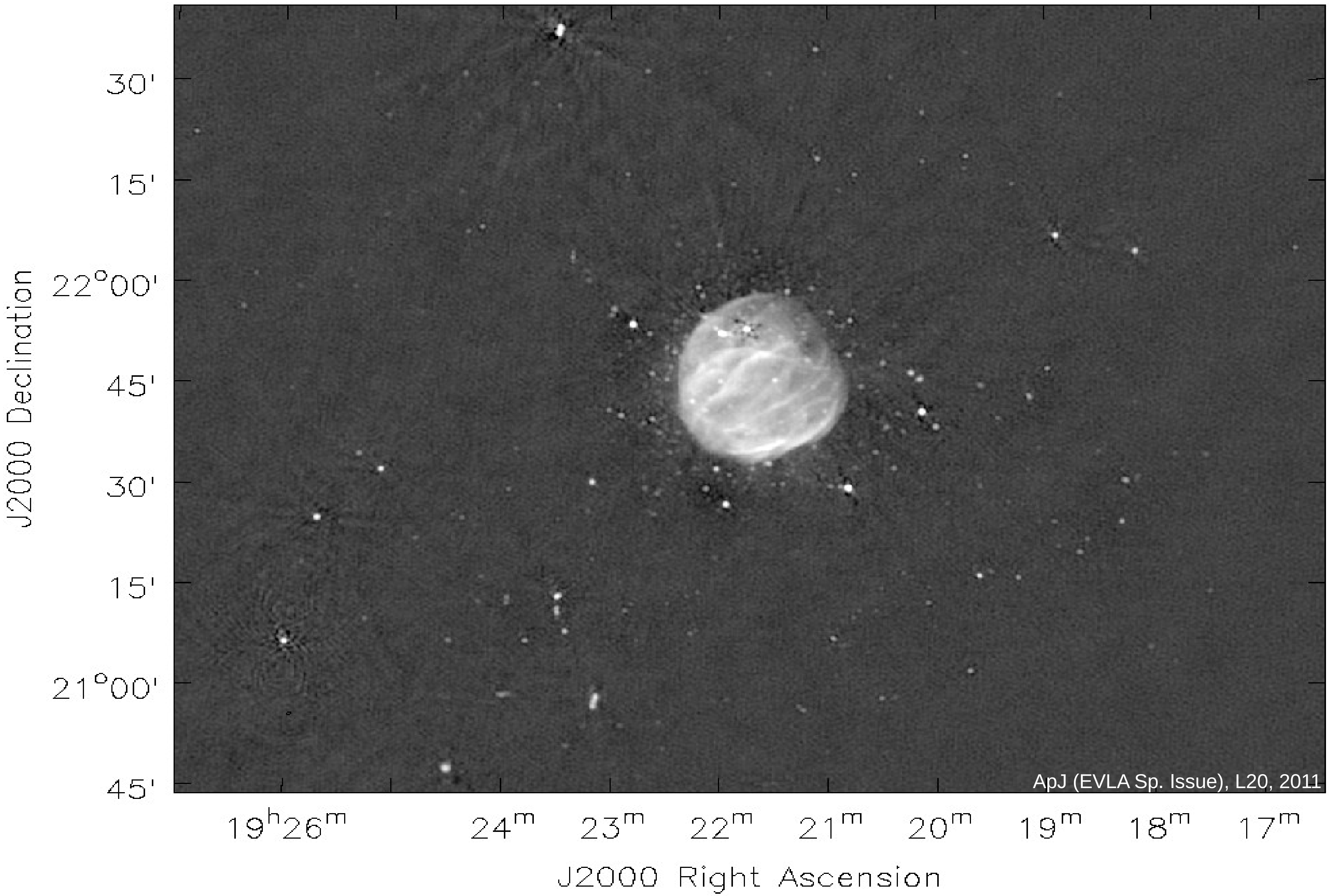
- Primarily due to improved continuum sensitivity
- @L-Band, PB gain ~1 deg. away can be up to 10%
  - In the EVLA sensitivity pattern, VLA sensitivity is achieved at the location of the VLA-null!
  - No null in the EVLA sensitivity pattern





- E.g. a 1% PSF side lobe due to a source away from the center is now significantly above continuum thermal noise limit
  - This is a largely independent of the total integration time

Wide-field Issues



Wide-field sensitivity because of wide-bandwidths

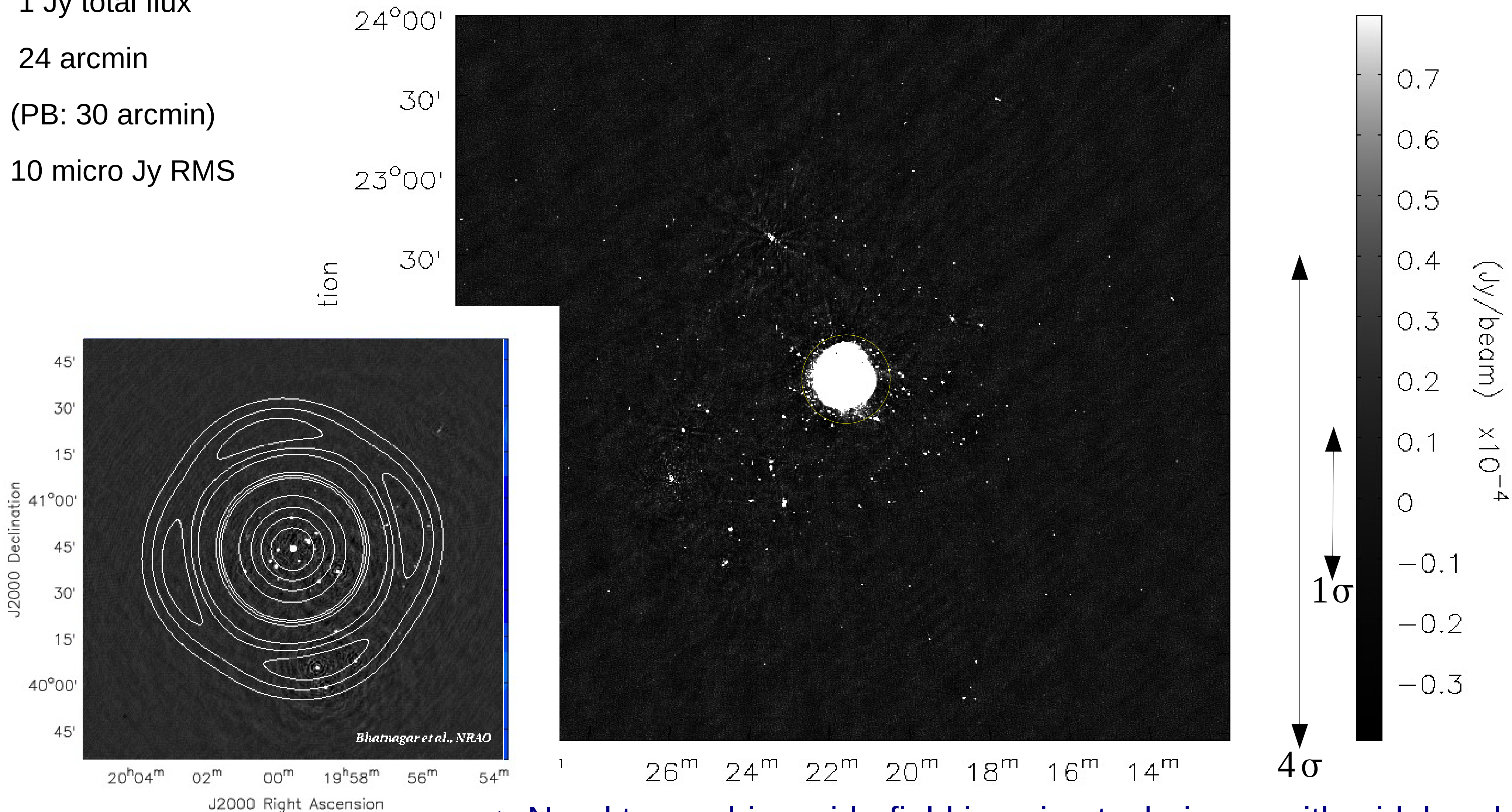
G55.7+3.4 : Galactic supernova remnant : 4 x 4 degree field-of-view from one EVLA pointing

1 Jy total flux

24 arcmin

(PB: 30 arcmin)

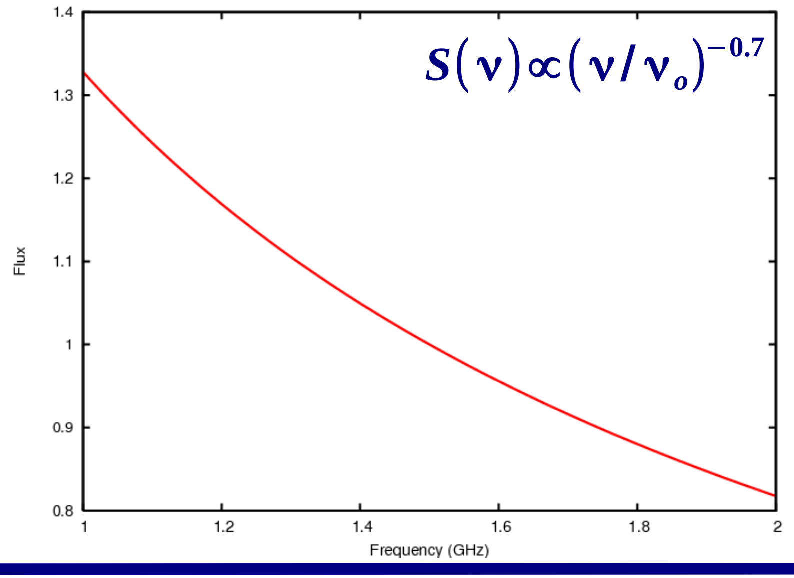
10 micro Jy RMS

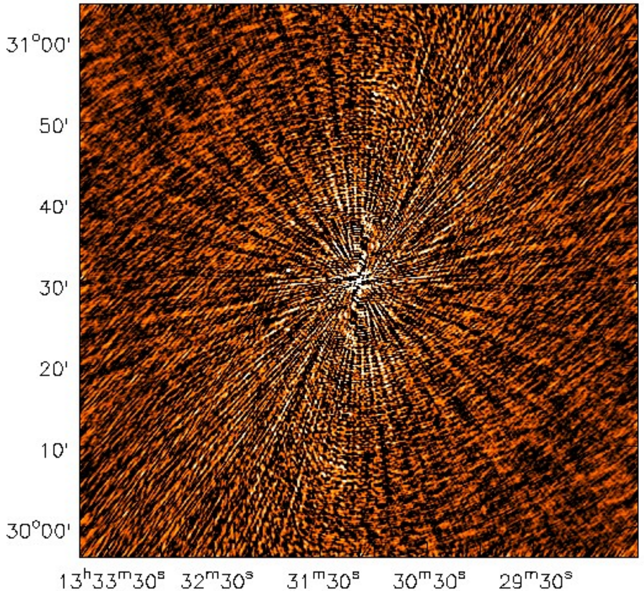


=> Need to combine wide-field imaging techniques with wideband..

What do we call wide-band?

- When fractional signal bandwidth used for imaging > ~20%
  - Plus source spectral index >= -1.0
  - Plus target dynamic range > 1000
- Spectral effects for higher source spectral index will become significant at lower bandwidth ratios
  - Empirical Dynamic range : $\frac{I\alpha}{100}$
  - Spectral line imaging, by definition, does not require wide-band imaging algorithms



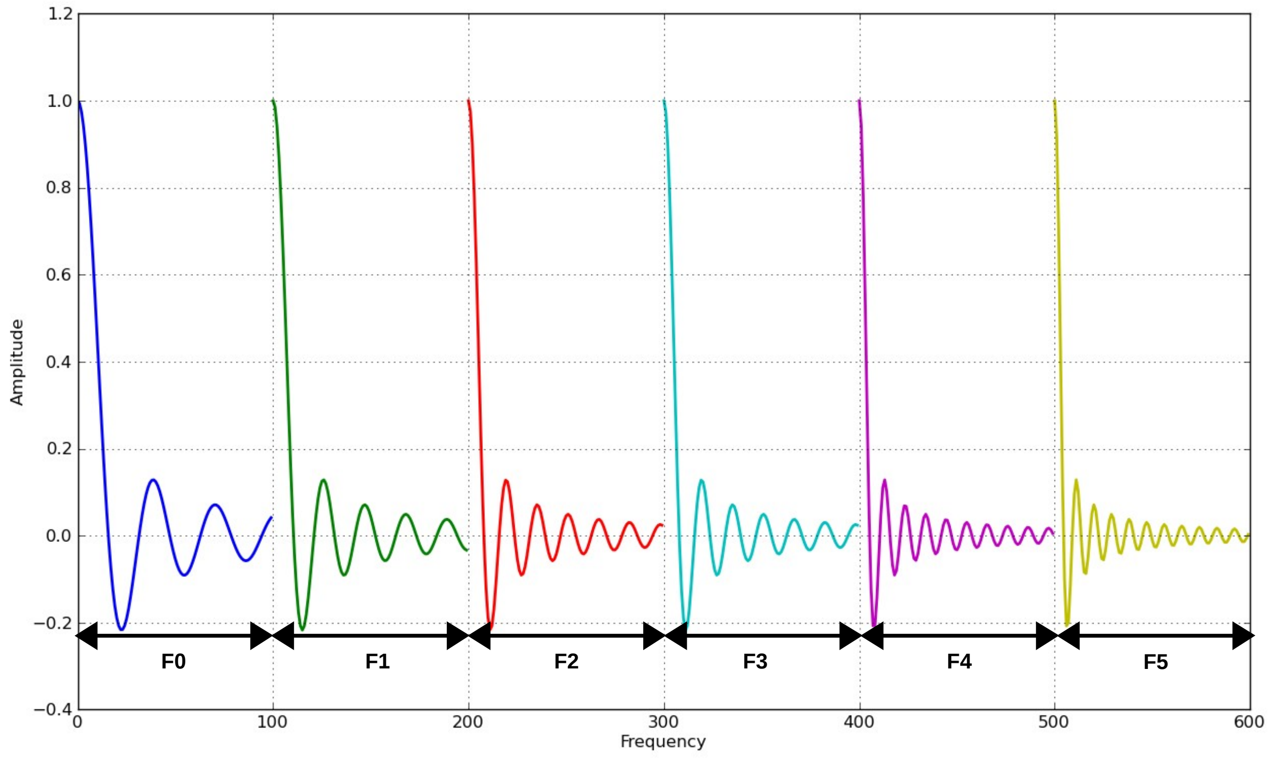


Frequency-dependent UV-coverages and PSFs

Spatial-frequency coverage and imaging properties change with frequency:

- PSF structure scales with frequency

$$S(u,v)_\nu = \frac{\vec{b}}{\lambda} = \frac{\vec{b}\,\nu}{c}$$



PSF sidelobes for sub-band imaging

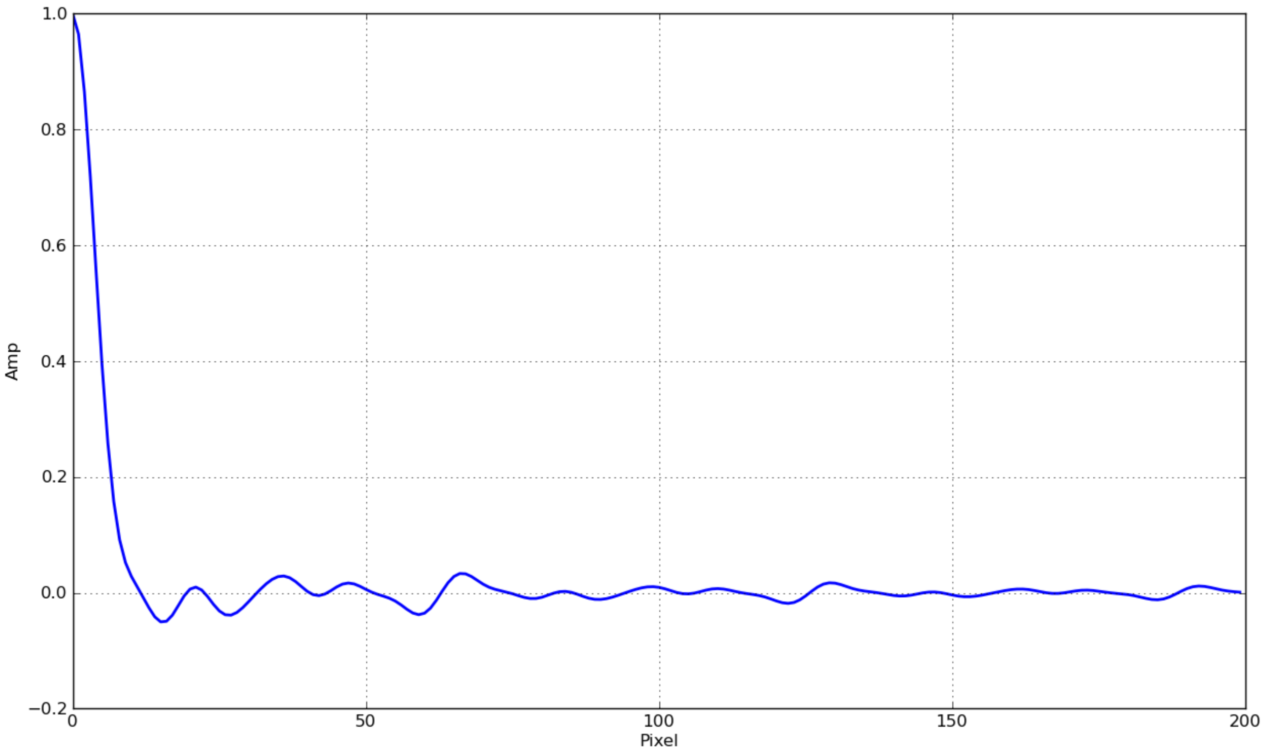
Frequency-dependent UV-coverages and PSFs

Spatial-frequency coverage and imaging properties change with frequency:

- PSF structure scales with frequency

$$S(u,v)_\nu = \frac{\vec{b}}{\lambda} = \frac{\vec{b}\,\nu}{c}$$

$$PSF_{Continuum} = \sum_\nu PSF(\nu)$$



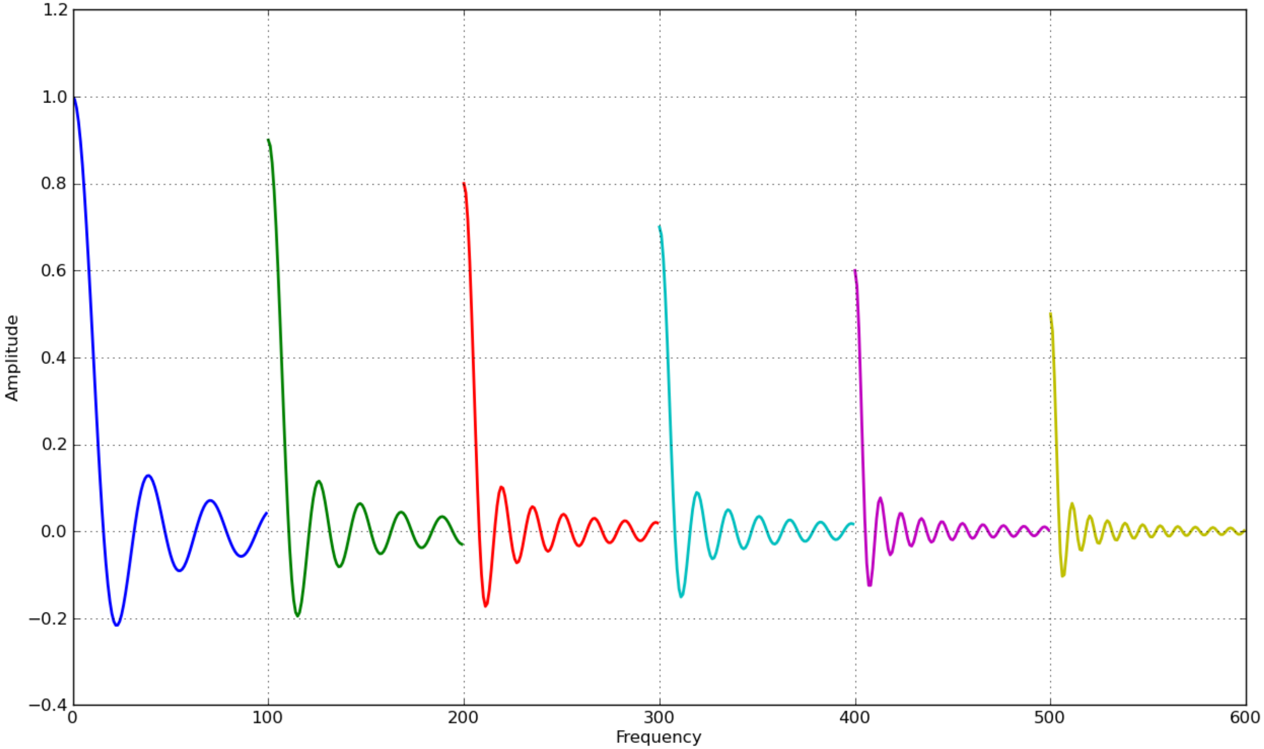
Lower PSF sidelobes compared to sub-band imaging

Frequency-dependent UV-coverages and PSFs

Spatial-frequency coverage and imaging properties change with frequency:

$$S(u,v)_\nu = \frac{\vec{b}}{\lambda} = \frac{\vec{b}\,\nu}{c}$$

- PSF structure scales with frequency
- PSF amplitude also changes with frequency at the location of the sources due to frequency dependent flux $\mathit{Flux}(\nu) \propto (\nu/\nu_o)^{-\alpha}$



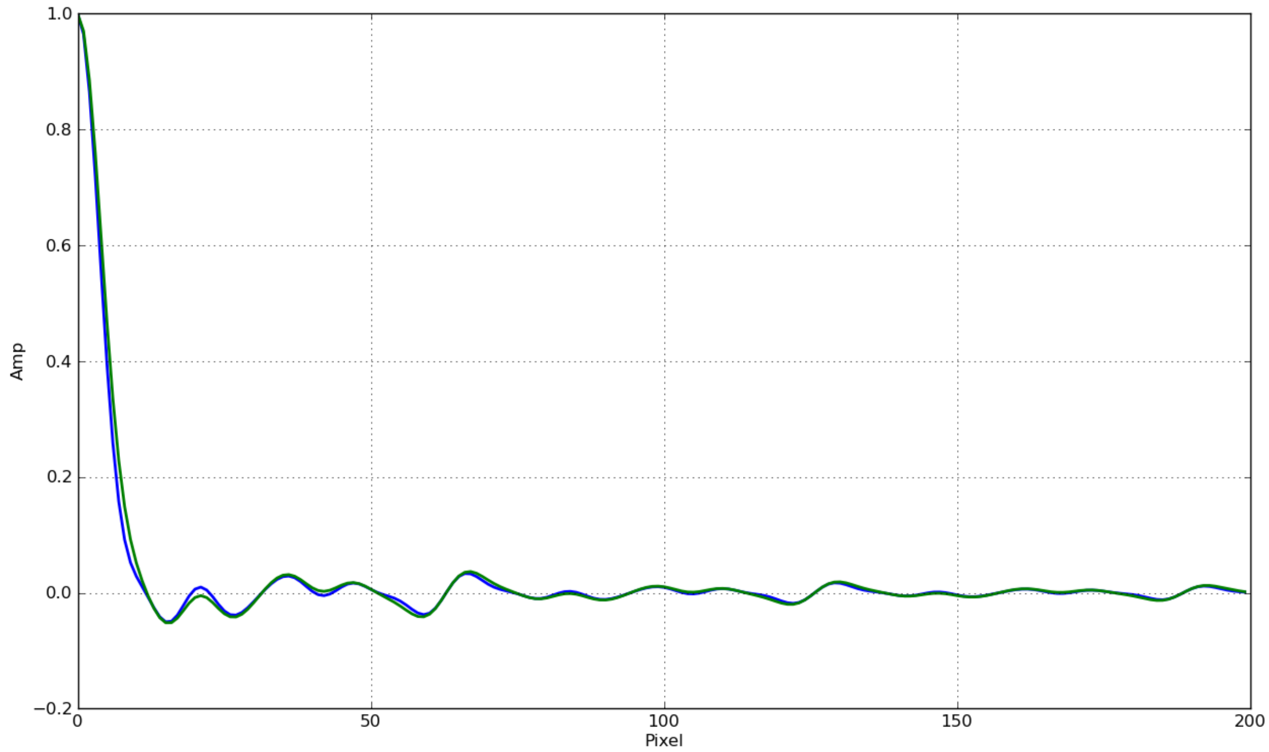
Frequency-dependent UV-coverages and PSFs

Spatial-frequency coverage and imaging properties change with frequency:

$$S(u,v)_\nu = \frac{\vec{b}}{\lambda} = \frac{\vec{b}\,\nu}{c}$$

- PSF structure scales with frequency
- Due to source Spectral Index, PSF amplitude also changes with frequency

$$PSF(x_o)_{Continuum} = \sum_\nu I(x_o,\nu)\, PSF(x - x_o, \nu)$$



Effective PSF for wide-band imaging becomes direction dependent

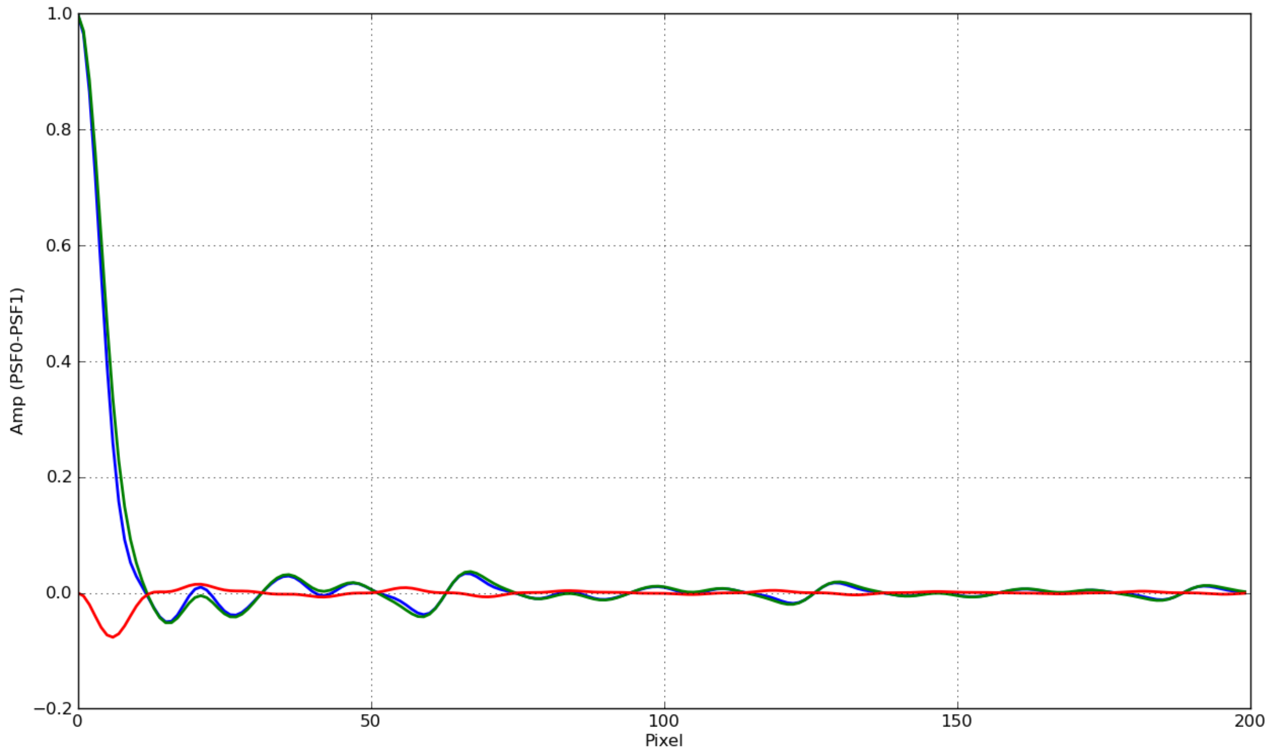
Frequency-dependent UV-coverages and PSFs

Spatial-frequency coverage and imaging properties change with frequency:

$$S(u,v)_\nu = \frac{\vec{b}}{\lambda} = \frac{\vec{b}\,\nu}{c}$$

- PSF structure scales with frequency
- Due to source Spectral Index, PSF amplitude also changes with frequency

$$Res(x_o)_{Continuum} = \sum_\nu PSF(x - x_o, \nu) - \sum_\nu I(x_o, \nu) PSF(x - x_o \nu)$$



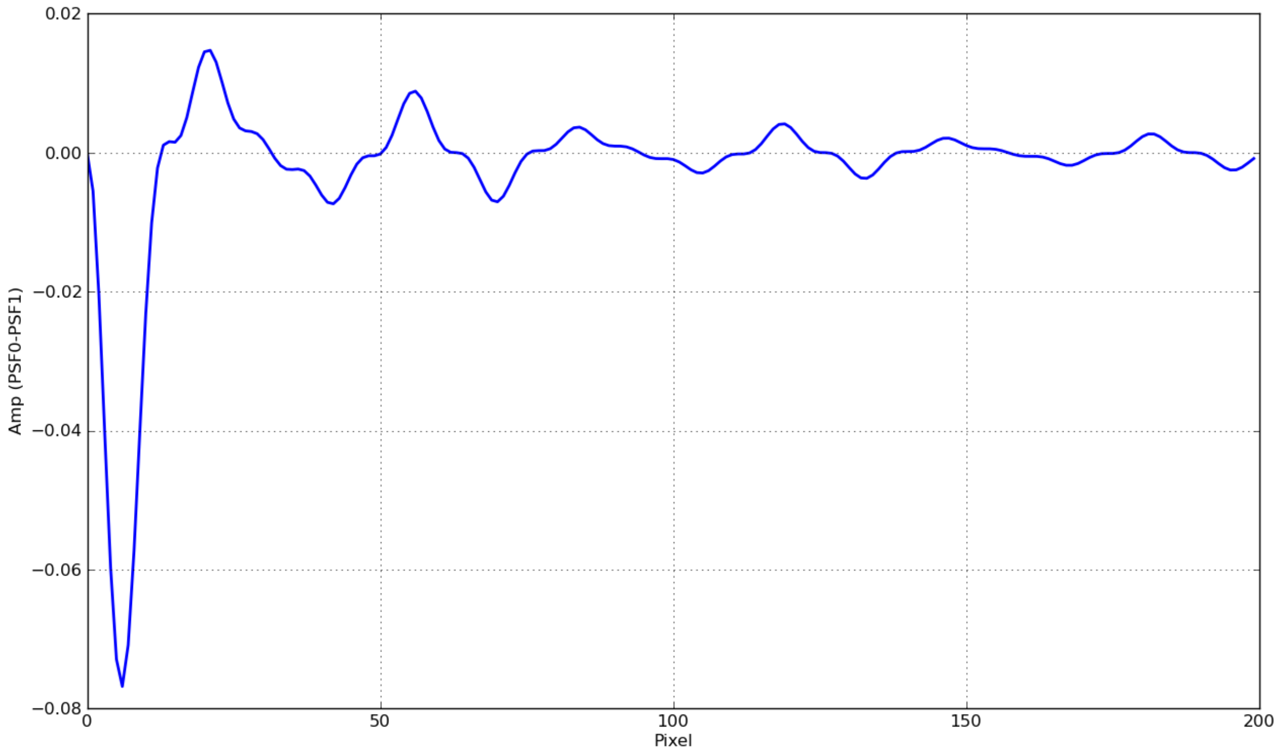
Frequency-dependent UV-coverages and PSFs

Spatial-frequency coverage and imaging properties change with frequency:

$$S(u,v)_\nu = \frac{\vec{b}}{\lambda} = \frac{\vec{b}\,\nu}{c}$$

- PSF structure scales with frequency
- Due to source Spectral Index, PSF amplitude also changes with frequency

$$Res(x_o)_{Continuum} = \sum_\nu PSF(x - x_o, \nu) - \sum_\nu I(x_o, \nu) PSF(\nu)$$

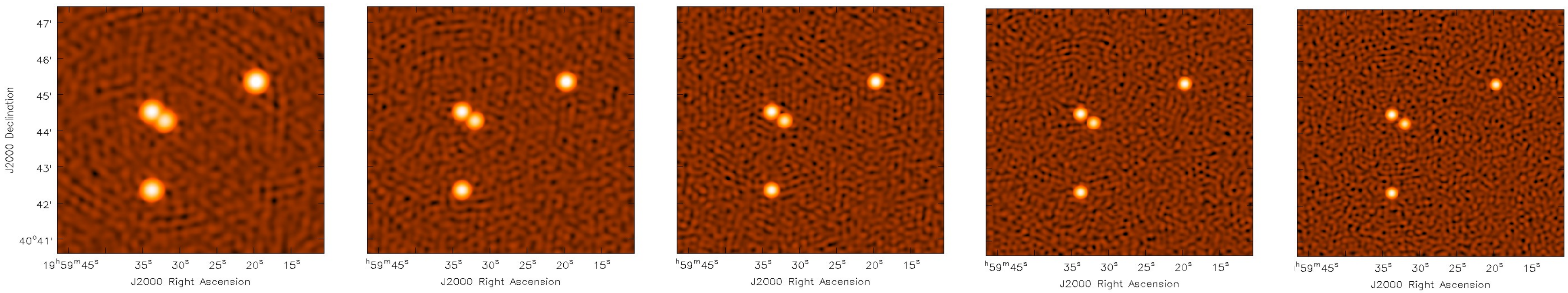


Deconvolution errors when frequency dependence of sky brightness is ignored

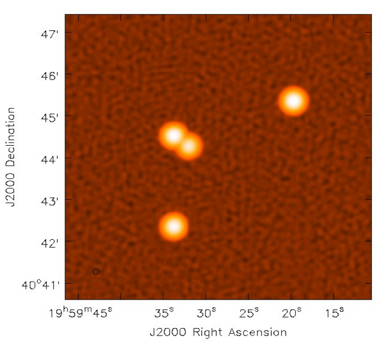
Single-channel vs MFS imaging – Angular Resolution

Simulated Example : 3 flat-spectrum sources + 1 steep-spectrum source (1-2 GHz)

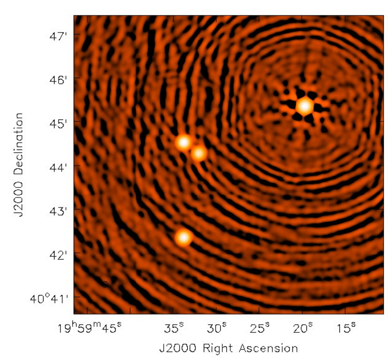
Images made separately at different frequencies between 1 and 2 GHz



Combine all single-frequency images (after smoothing)

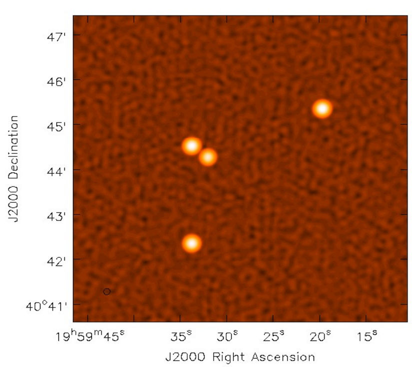


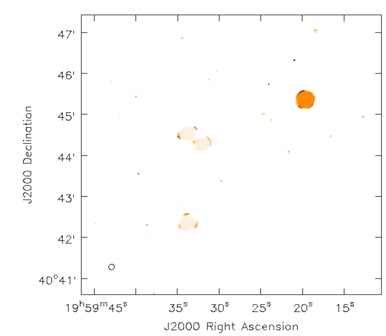
Use all UV-coverage together, but ignore spectra



Use all UV-coverage together
+ Model and fit for spectra too

Output : Intensity and Spectral-Index





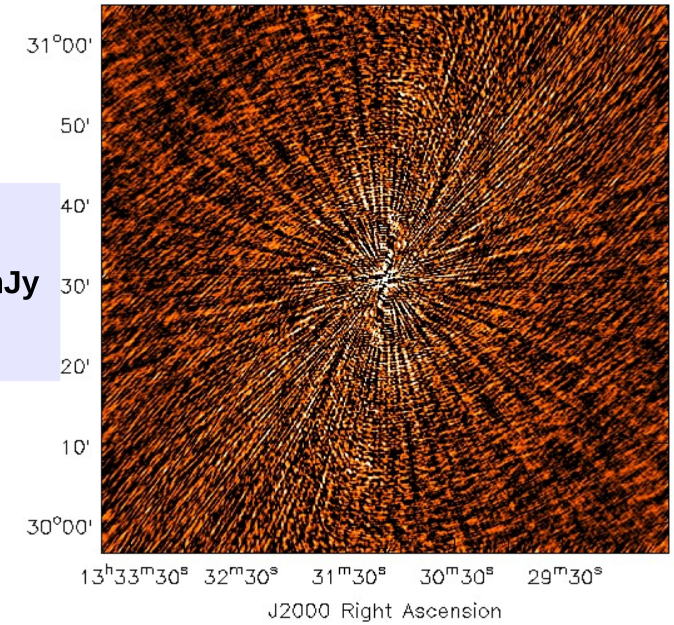
=> Imaging with a spectrum model : higher angular resolution + continuum sensitivity.

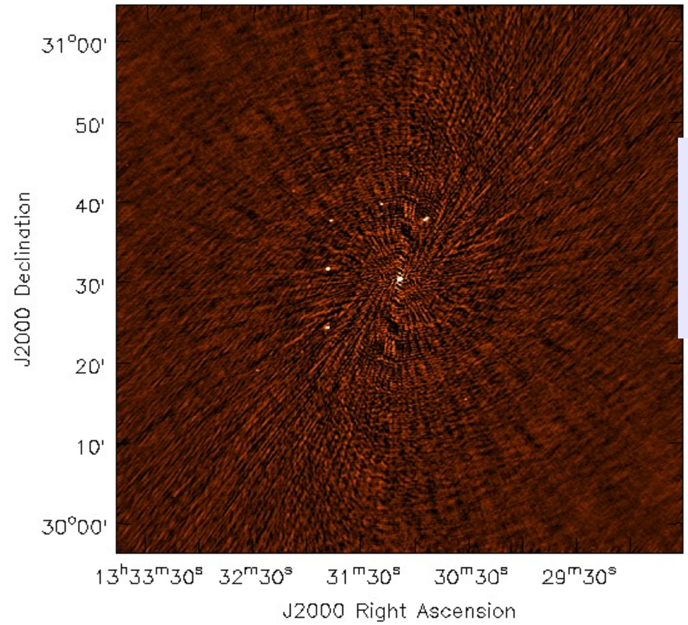
Dynamic-range with MS-MFS : 3C286 example : Nt=1,2,3,4

NTERMS = 1

Rms : 9 mJy -- 1 mJy

DR : 1600 -- 13000





NTERMS = 2

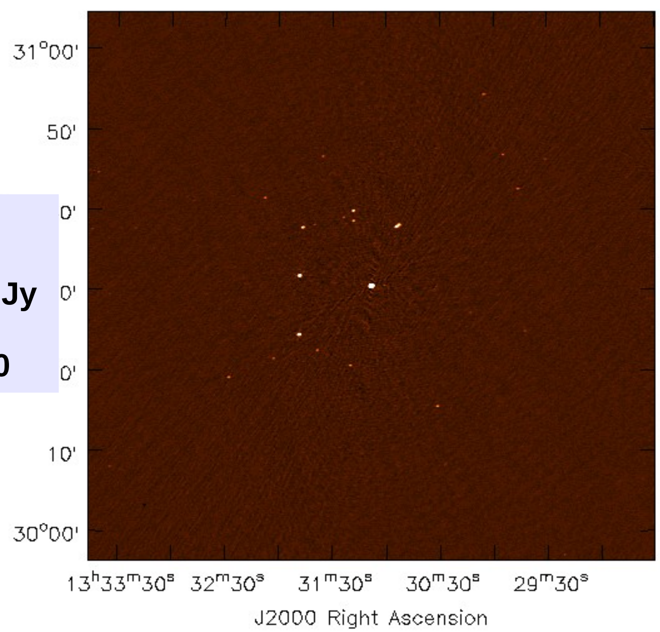
Rms :1 mJy -- 0.2 mJy

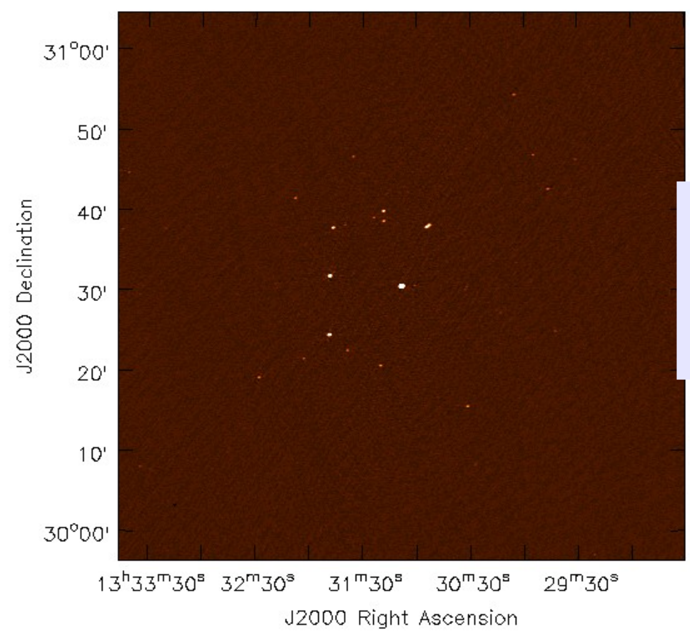
DR :10,000 -- 17,000

NTERMS = 3

Rms : 0.2 mJy -- 85 uJy

DR : 65,000 -- 170,000





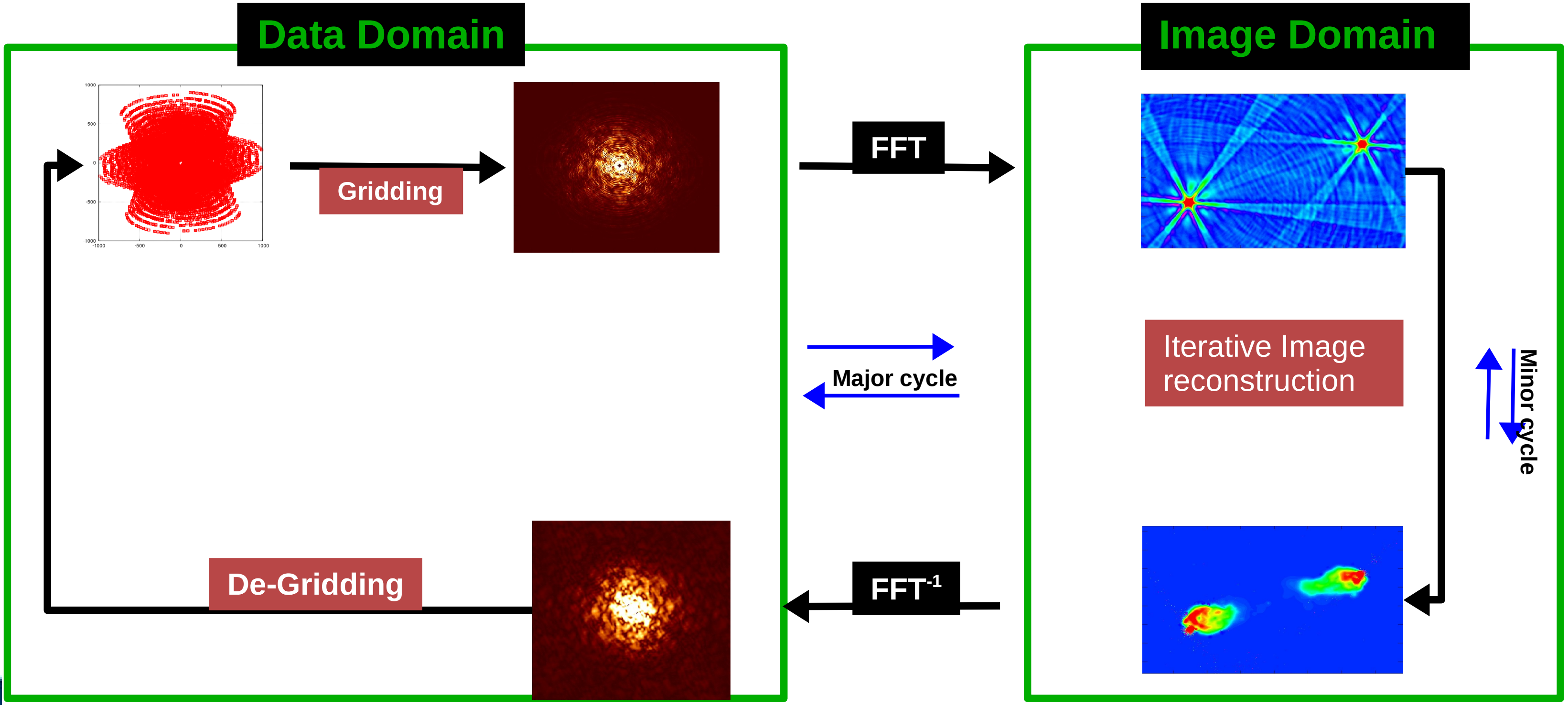
NTERMS = 4

Rms 0.14 mJy -- 80 uJy

DR :>110,000 -- 180,000

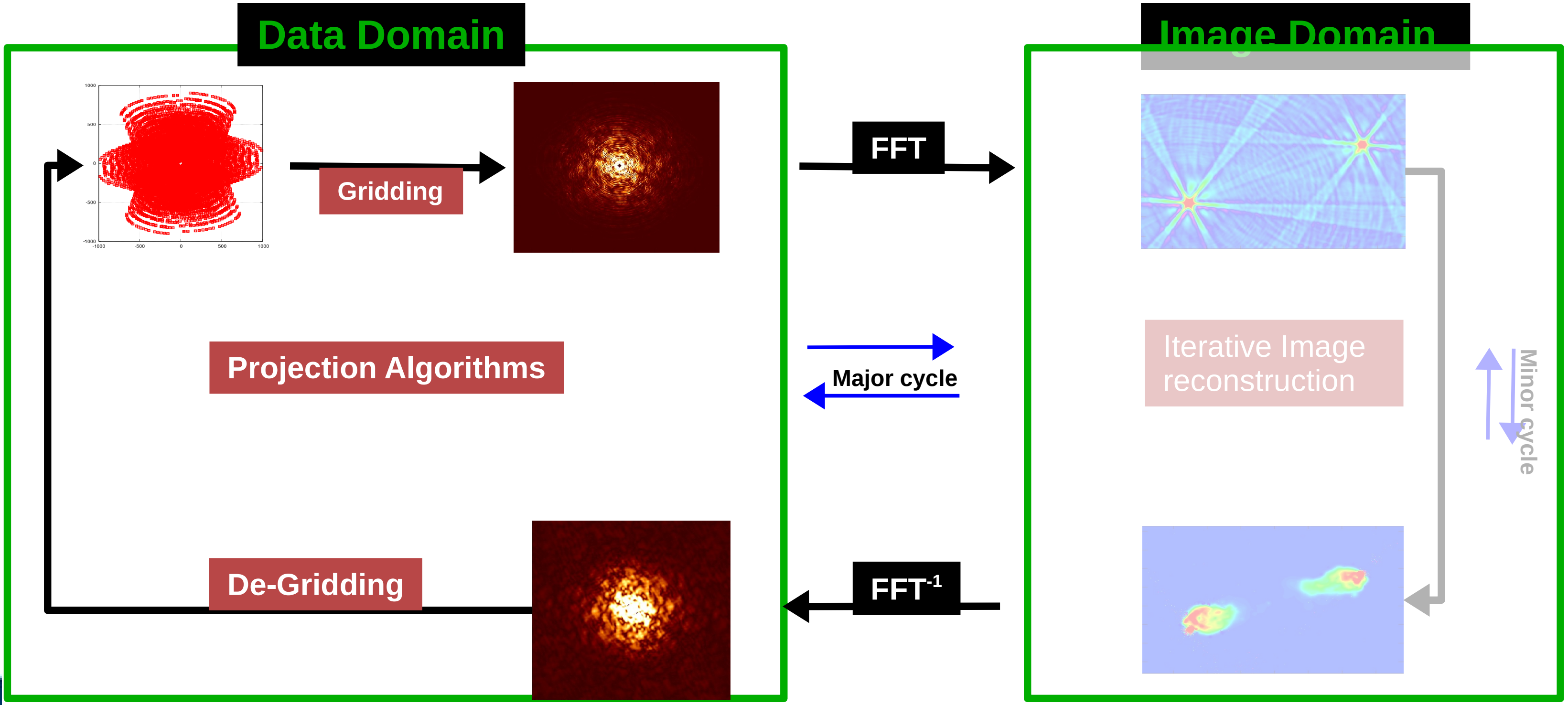
The Imaging Problem

- Basic computations
  - Use of FFT to transform data to image domain: Gridding + FFT
  - Image reconstruction (a.k.a. “deconvolution”): Search and subtract
  - Predict data given a model of the sky: FFT^{-1} + De-gridding



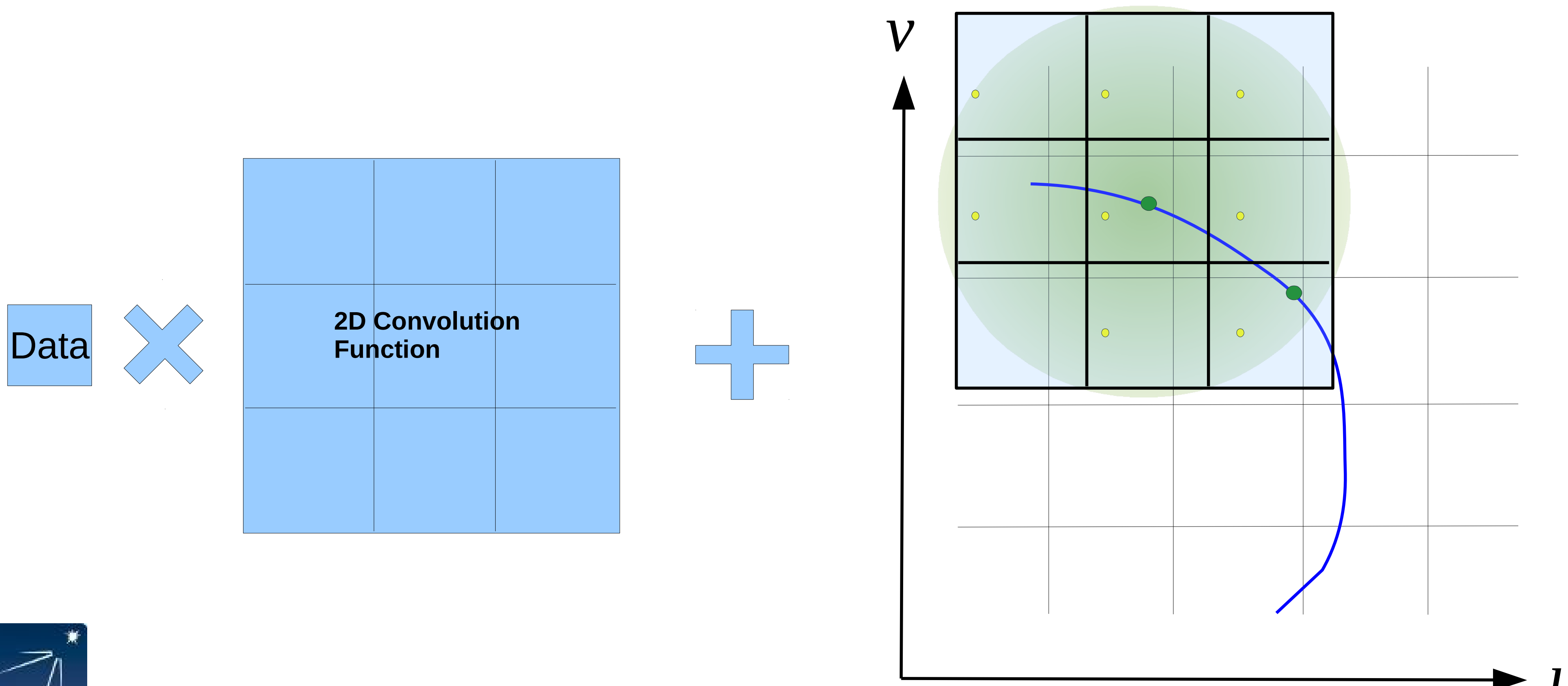
The Imaging Problem

- Basic computations
  - Use of FFT to transform data to image domain: Gridding + FFT
  - Image reconstruction (a.k.a. "deconvolution"): Search and subtract
  - Predict data given a model of the sky: FFT^{-1} + De-gridding



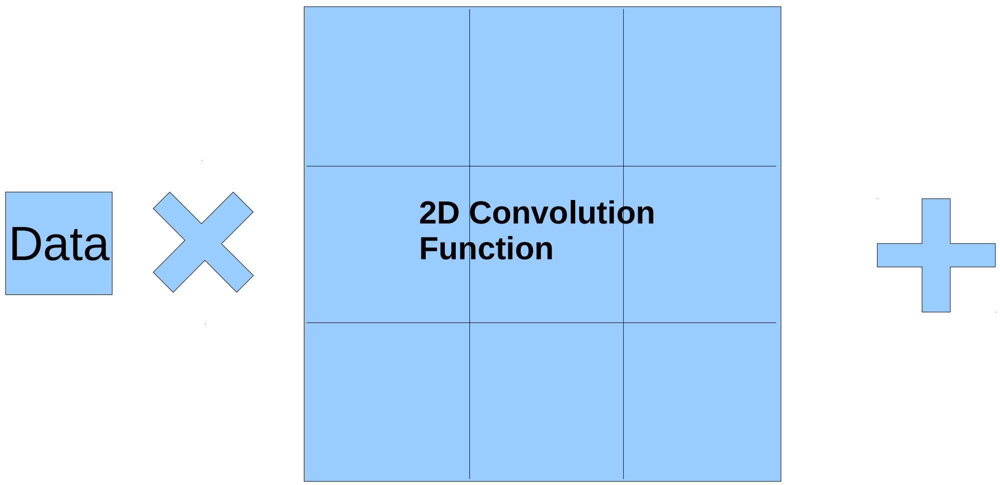
Gridding: Computations

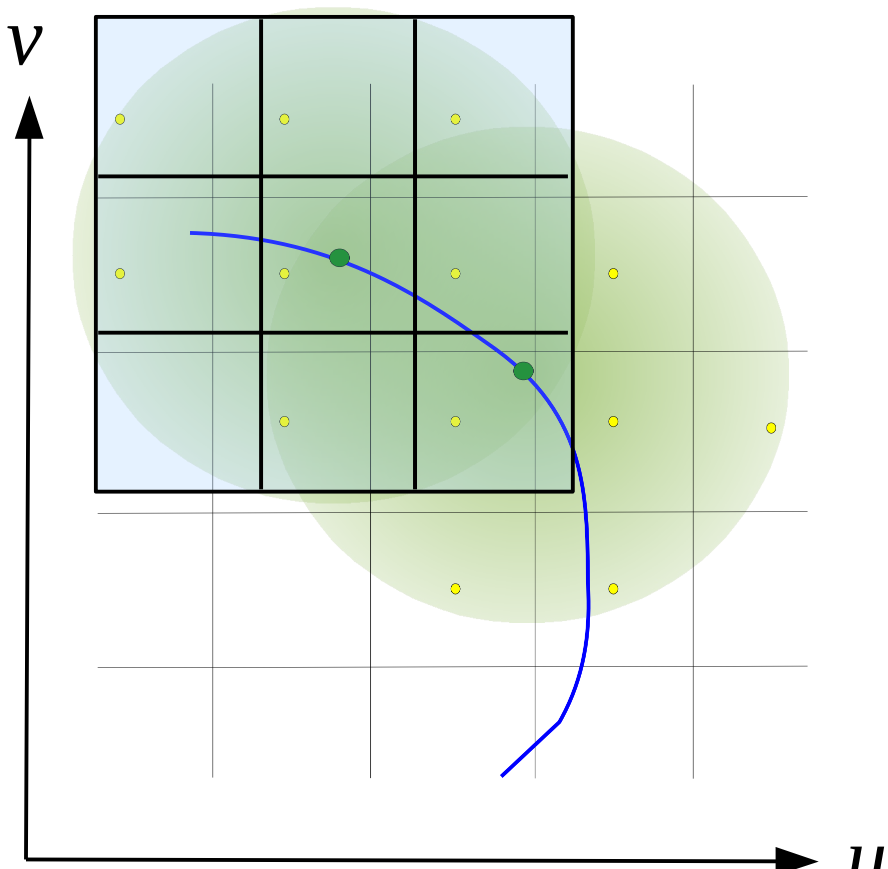
- Gridding/de-gridding: 2D interpolation via convolutional resampling
- 2D convolution functions ← → 2D weighting functions



Gridding: Computations

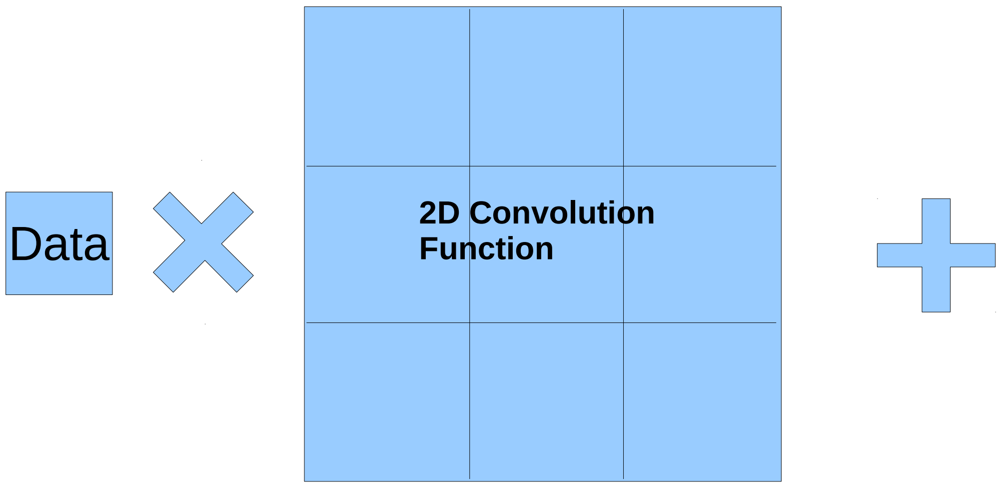
- Gridding/de-gridding: 2D interpolation via convolutional resampling
- 2D convolution functions ← → 2D weighting functions

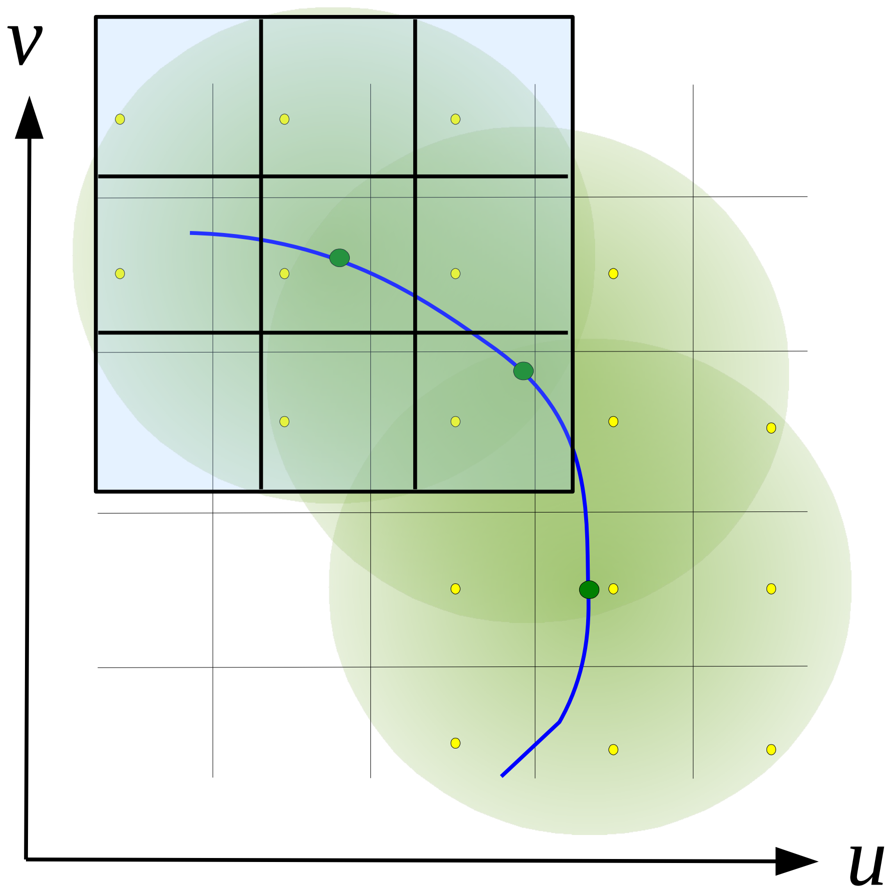




Gridding: Computations

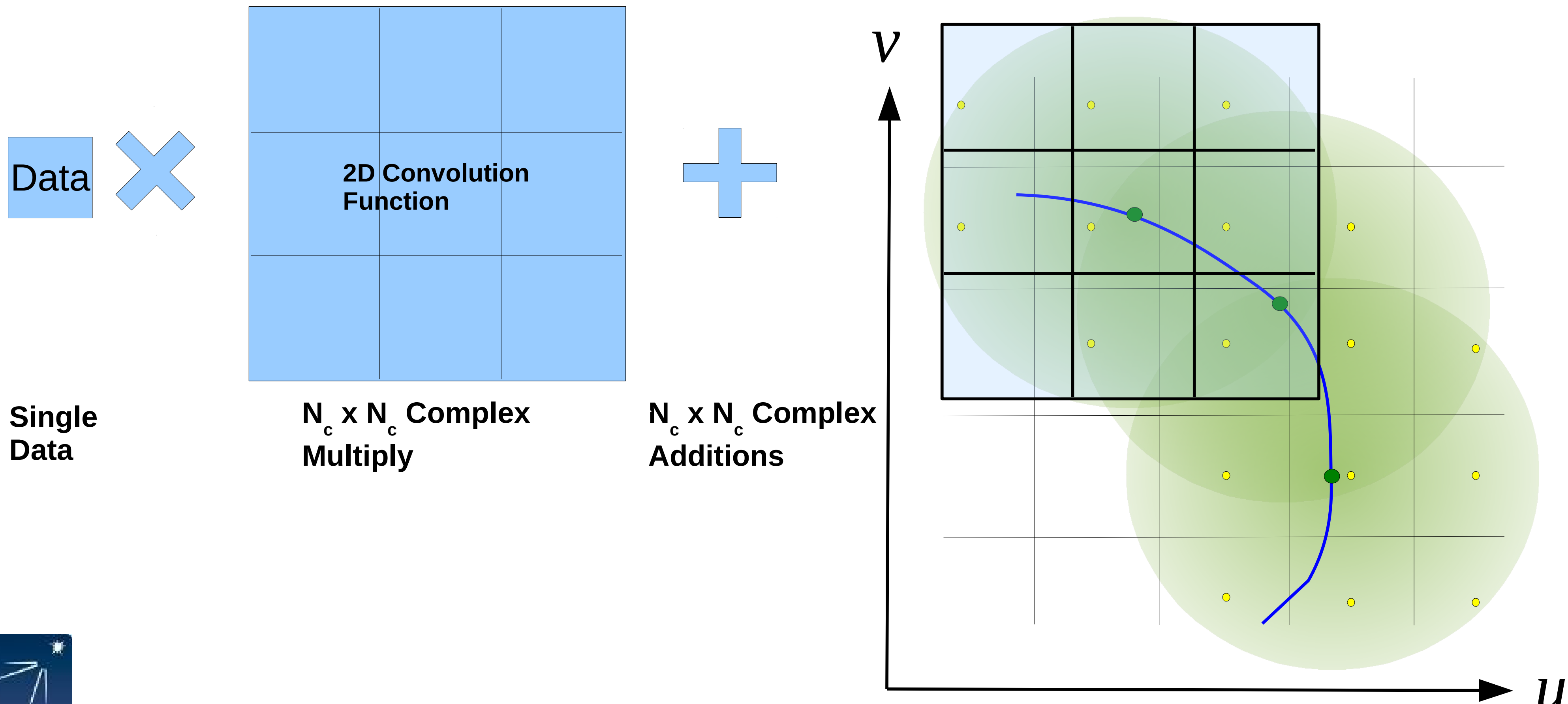
- Gridding/de-gridding: 2D interpolation via convolutional resampling
- 2D convolution functions ← → 2D weighting functions





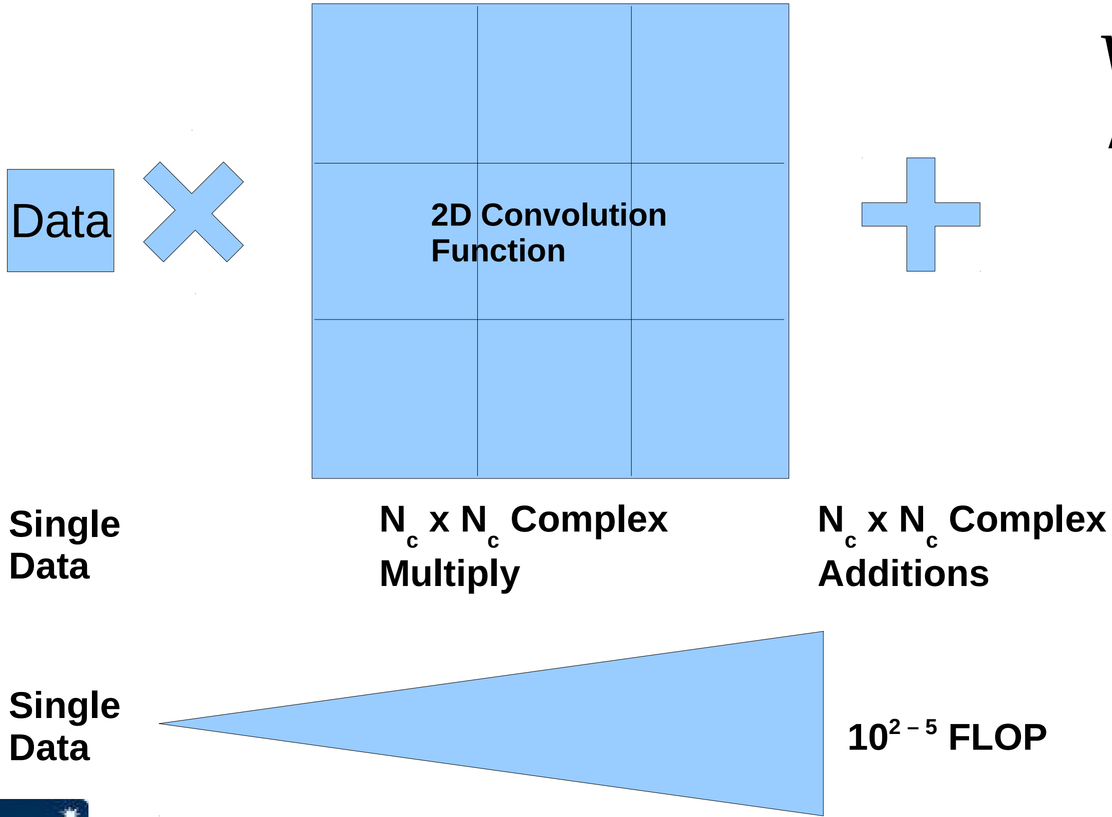
Gridding: Computations

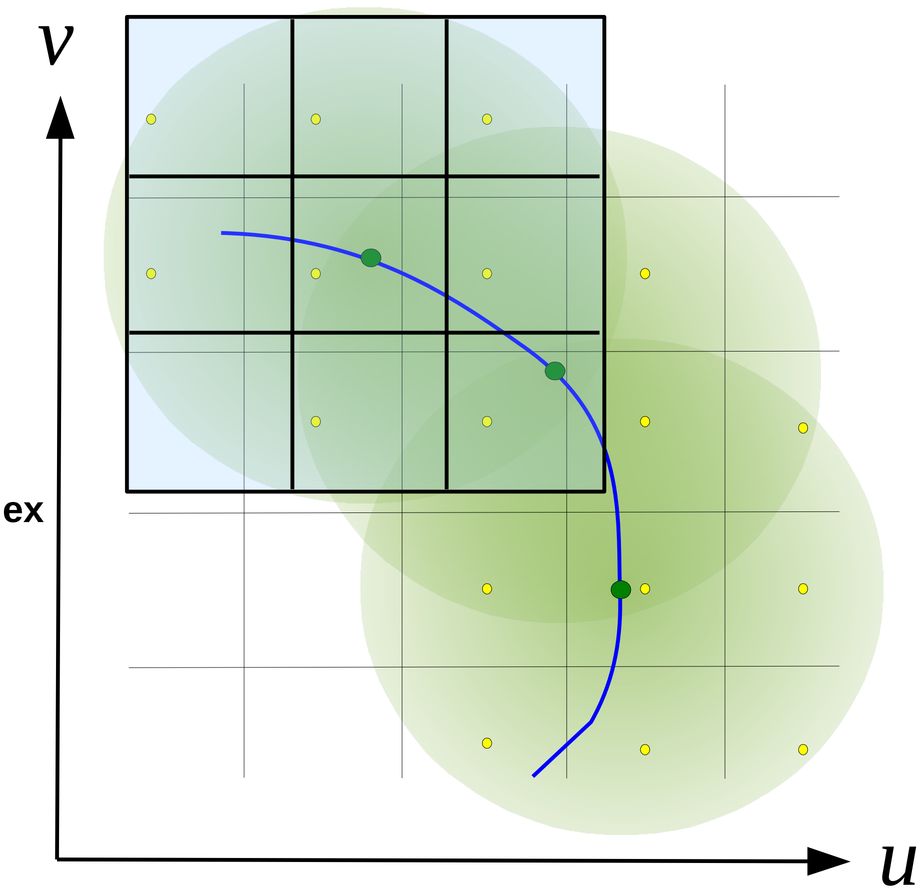
- Gridding/de-gridding: 2D interpolation via convolutional resampling
- 2D convolution functions ← → 2D weighting functions



Gridding: Computations

- Gridding/de-gridding: 2D interpolation via convolutional resampling
- 2D convolution functions ← → 2D weighting functions

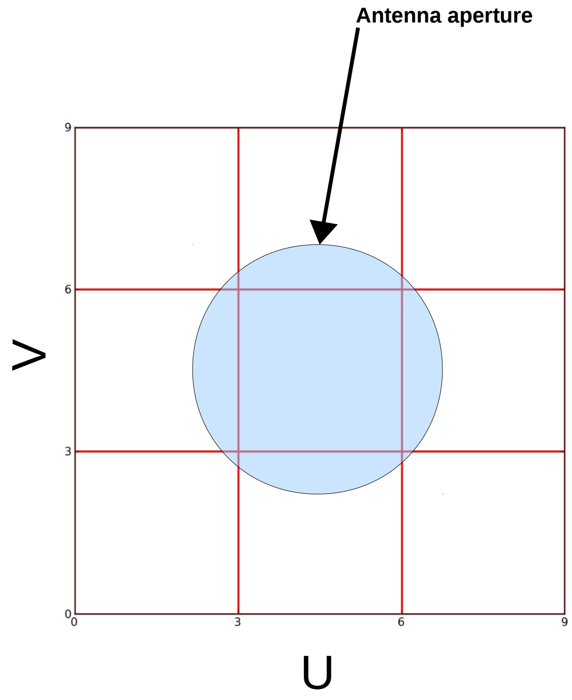




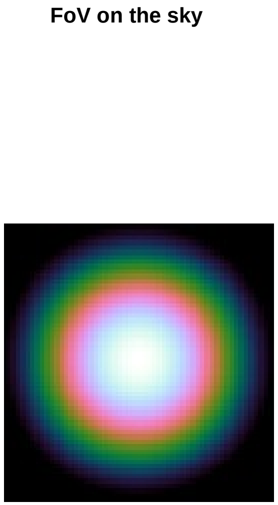
N_{data} x $12\, N^2_C$ $\quad N_{data} = O(10^{10-12})$

WF imaging: A-Projection

- WF imaging needs larger convolution functions (CF)



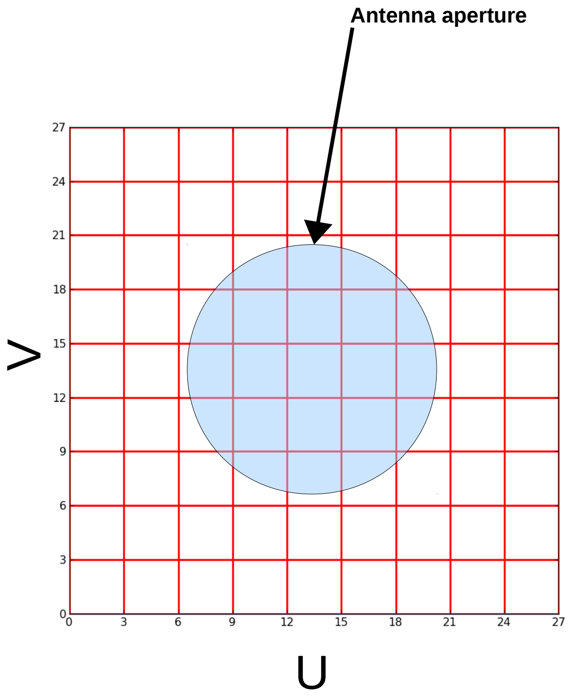
Number of uv-pixel
across antenna aperture



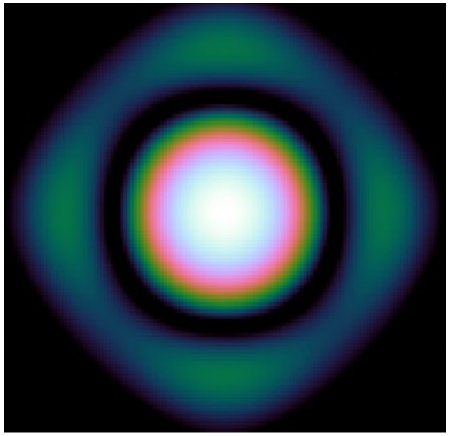
Just the main lobe (20% point)

WF imaging: A-Projection

- WF imaging needs larger convolution functions (CF)



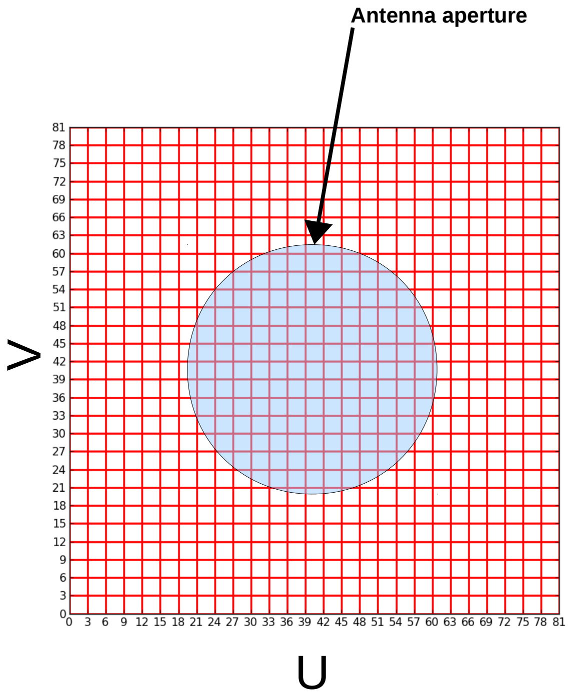
Number of uv-pixel
across antenna aperture

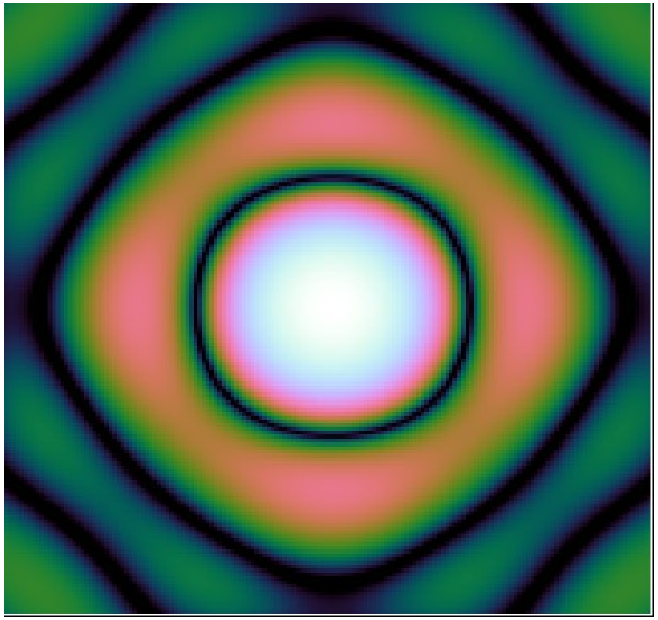


Include the first sidelobe (few%)

WF imaging: A-Projection

- WF imaging needs larger convolution functions (CF)





Number of uv-pixel
across antenna aperture

..beyond the first sidelobe

WF imaging: AW-Projection

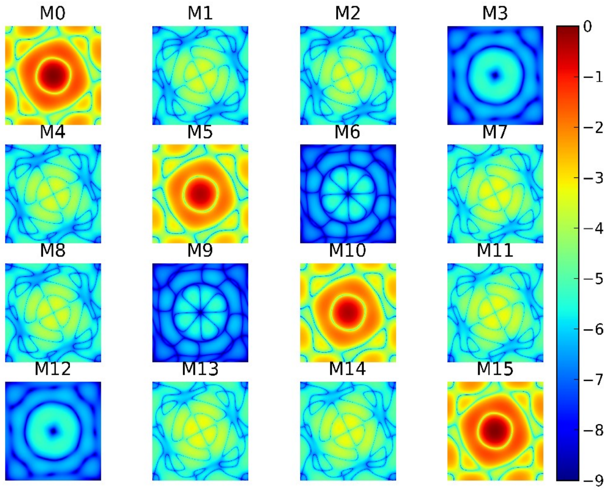
- WF imaging needs larger convolution functions (CF)

- Support size strongly affects gridding load $\left(\propto N_c^2\right)$
  - W-Projection: $N_c \rightarrow$ 10 – few x 100
  - A-Projection: $N_c \rightarrow$ 10 – 20

- CF computations for projection algorithms is expensive
  - Current approach: Compute once and cache
  - Increases memory footprint E.g. A-array imaging at L,S (and C?)-band

WF imaging: Full-pol A-Projection

- Number of CF determine the memory footprint
  - N_w : few x 100 – 1000 x (oversampling of 100)   [Complex]
  - N_a : 2 (Stokes-I only) – 8 (full Mueller) x (oversampling of 100)
  - AW-Projection: N_w x N_a:
    - [100s (Hi) – 1000s (Lo)] x [10 (diag) – 100s (Mueller)]
    - Up to 10s GB



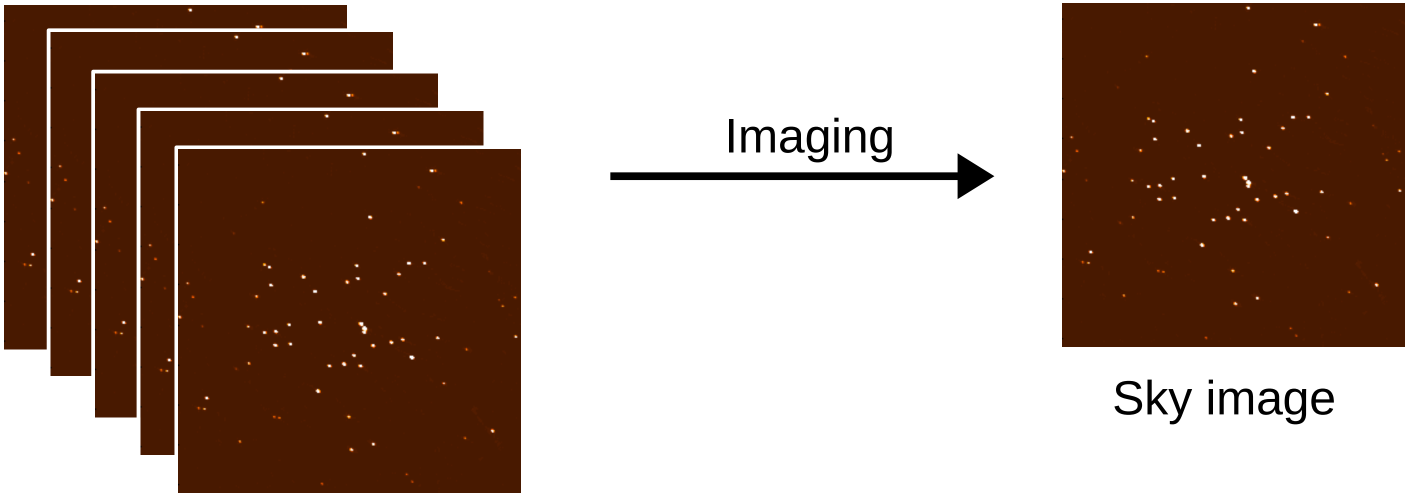
Imaging Memory footprint

- Each sky-image of size N_x x N_y requires
  - 2 x Complex x (N_x x N_y)+ (N_x x N_y) = **5 x (N_x x N_y) floats**

Imaging Memory footprint

- Each sky-image of size N_x x N_y requires
  - 2 x Complex x (N_x x N_y)+ (N_x x N_y) = **5 x (N_x x N_y) floats**

Mem. Buffers during gridding



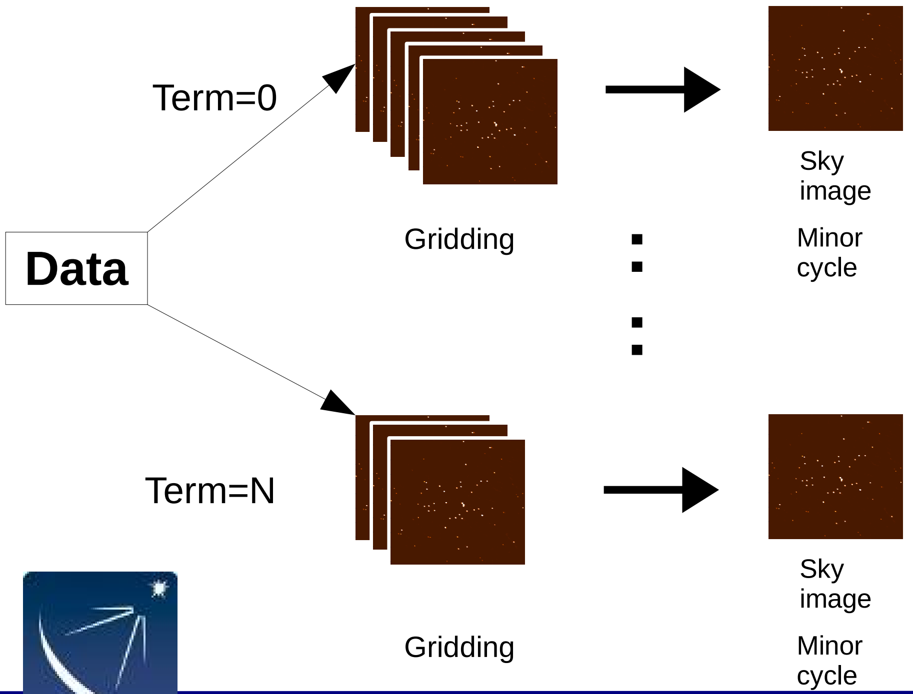
Sky image

Major cycle

Minor cycle

MT-MFS: Higher memory footprint

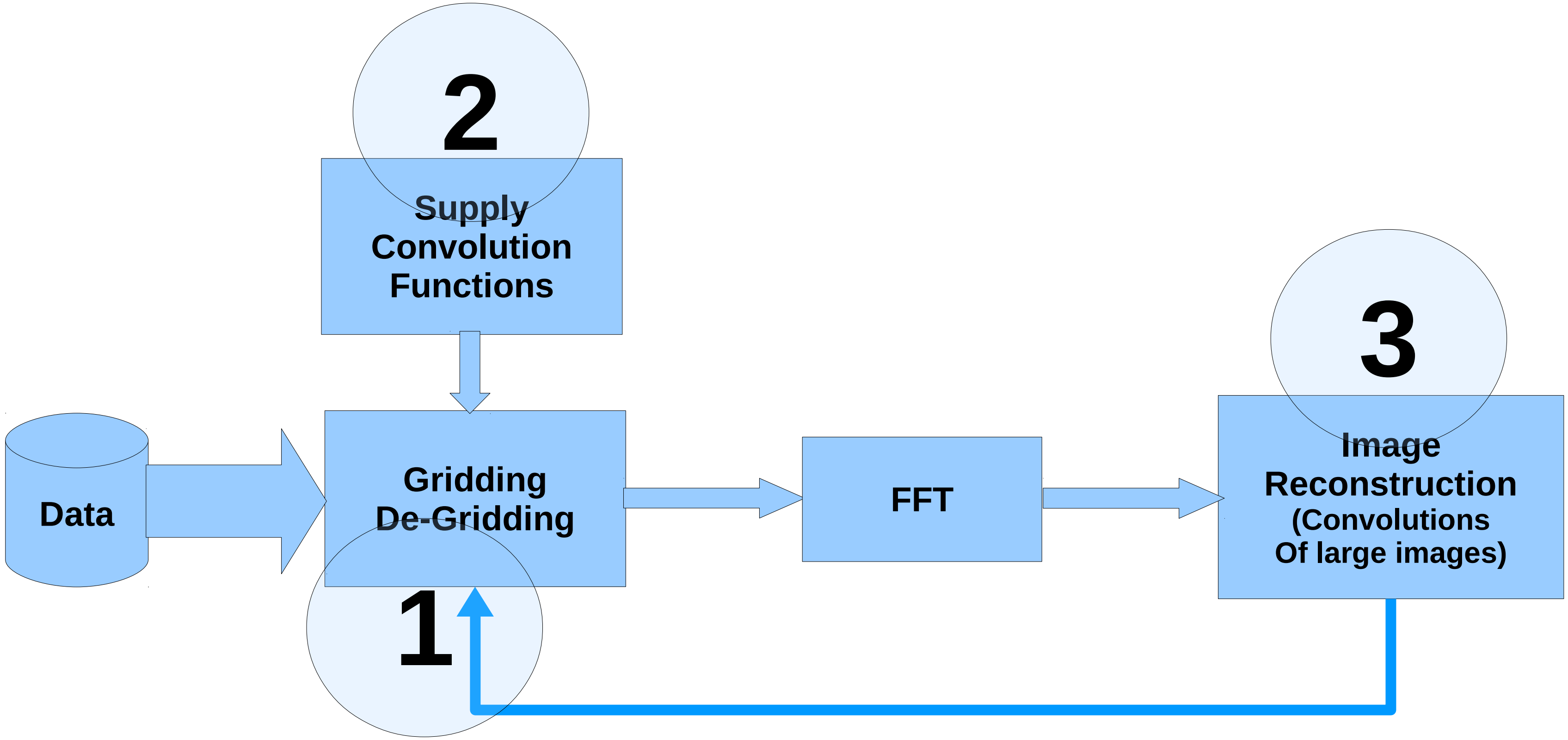
- WB A-Projection: N_A x N_{SPW} (order 10x increase in CF memory footprint)
- MS-MFS
  - Compute load: Gridding for N_{terms} images+ Convolution of large images
  - Memory: Multiple minor-cycle images (N_{scales})
  - Total images (each of size N_x x N_y) : N^2_{terms} x N^2_{scales}



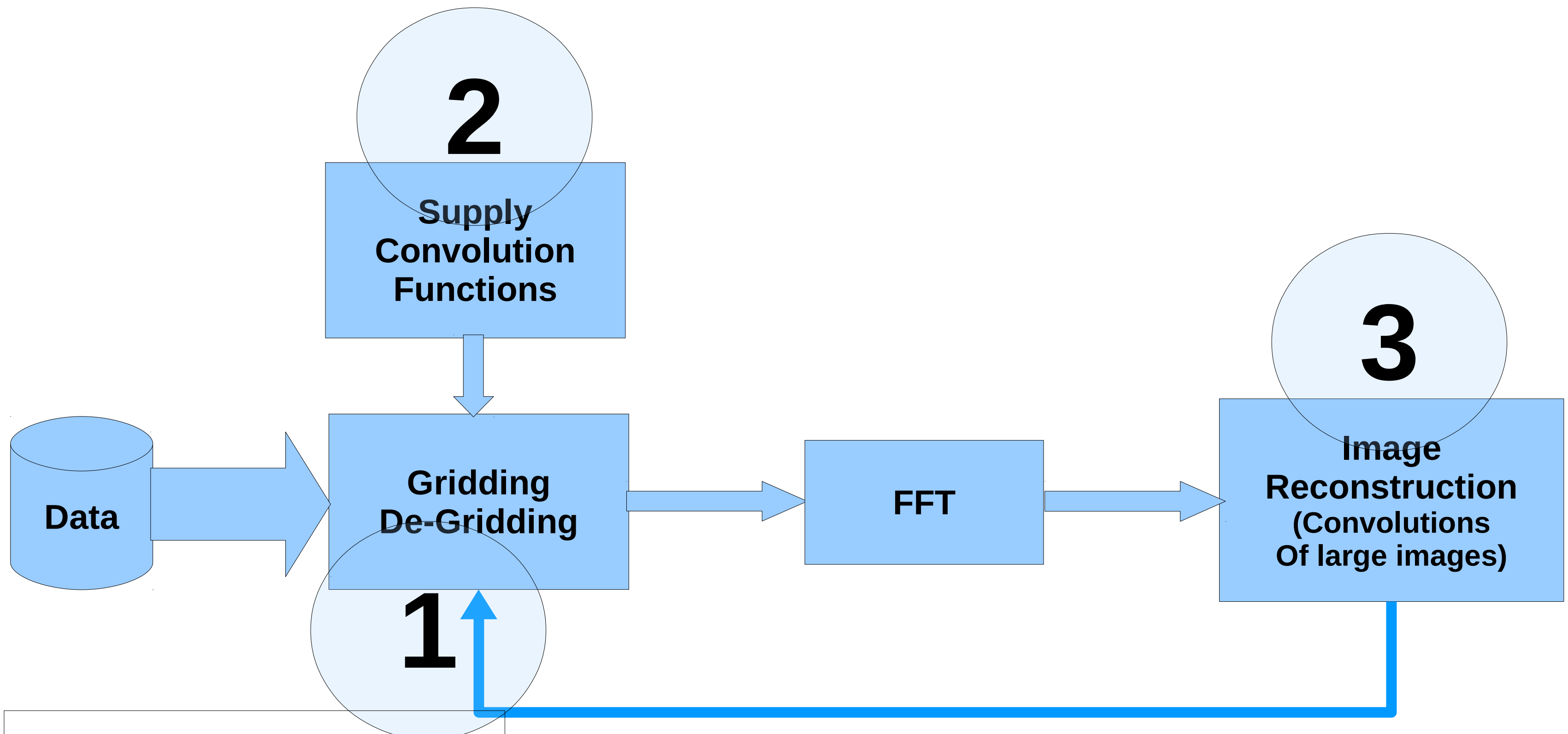
Memory storage for: N^2_{terms} x N^2_{scales}

Compute convolutions of images

The hot-spots



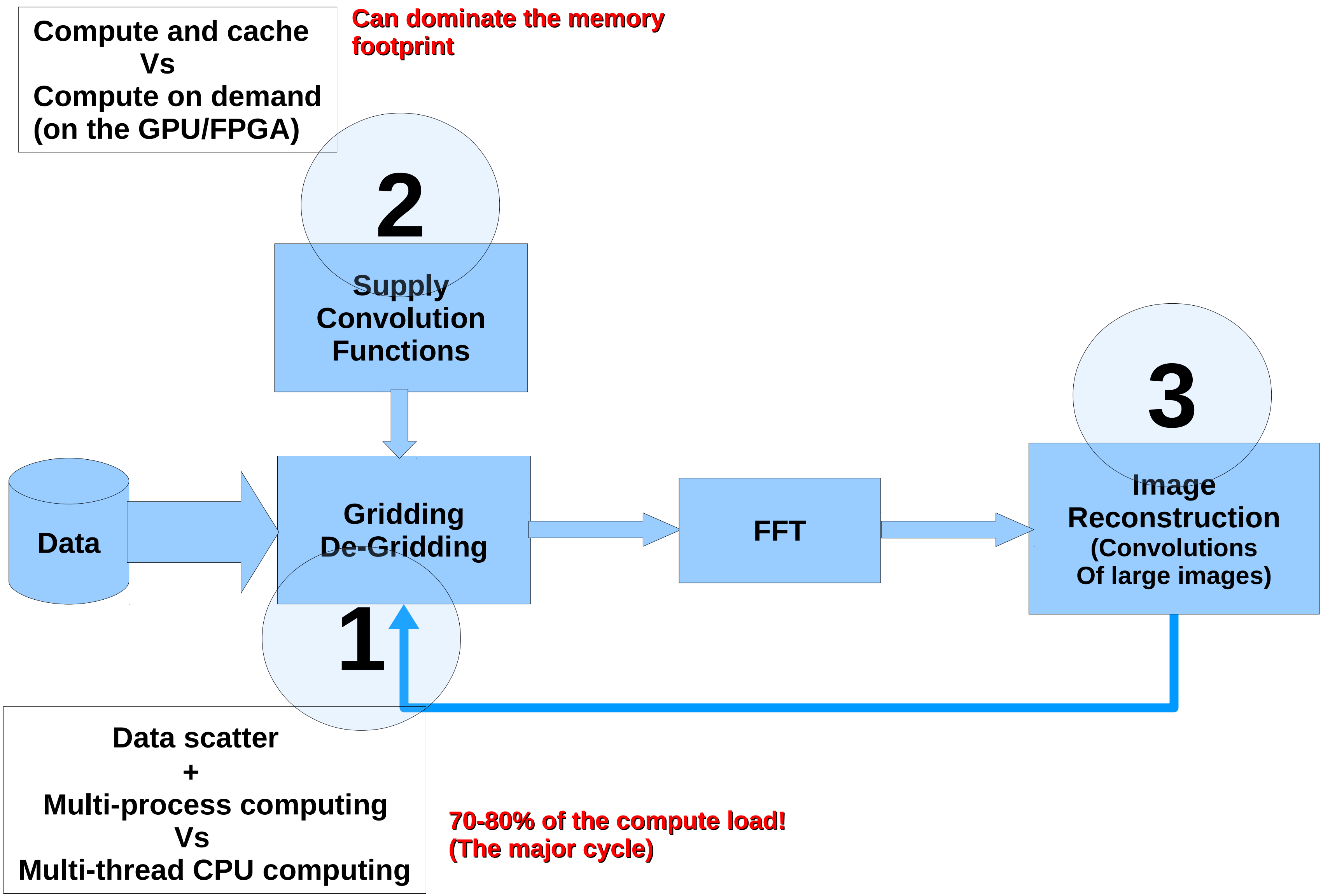
The hot-spots



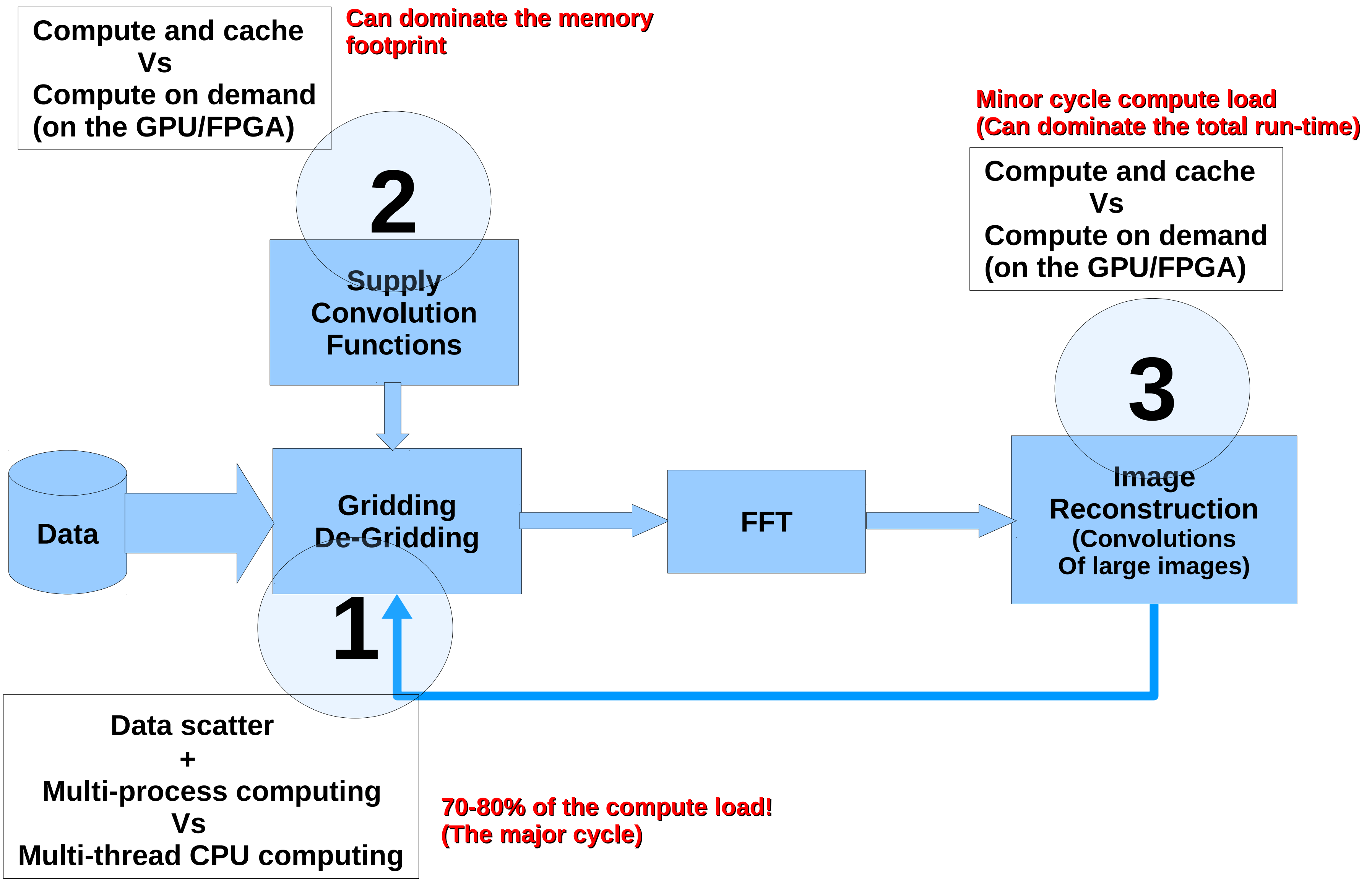
Data scatter
+
Multi-process computing
Vs
Multi-thread CPU computing

70-80% of the compute load!
(The major cycle)

The hot-spots



The hot-spots



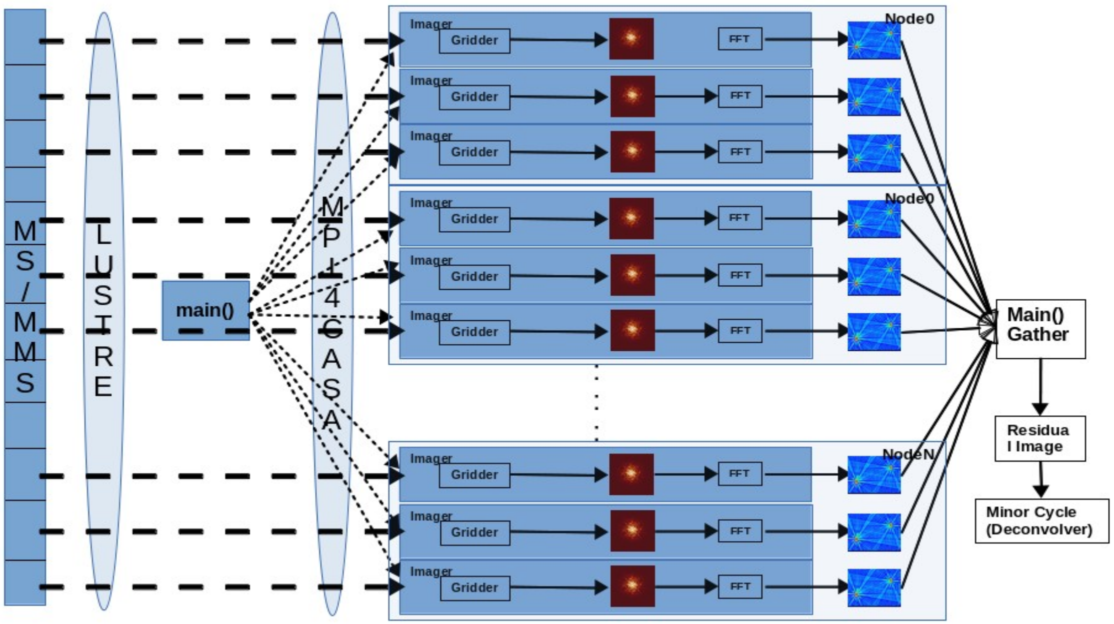
Gridding Parallelization (HS 1) - I

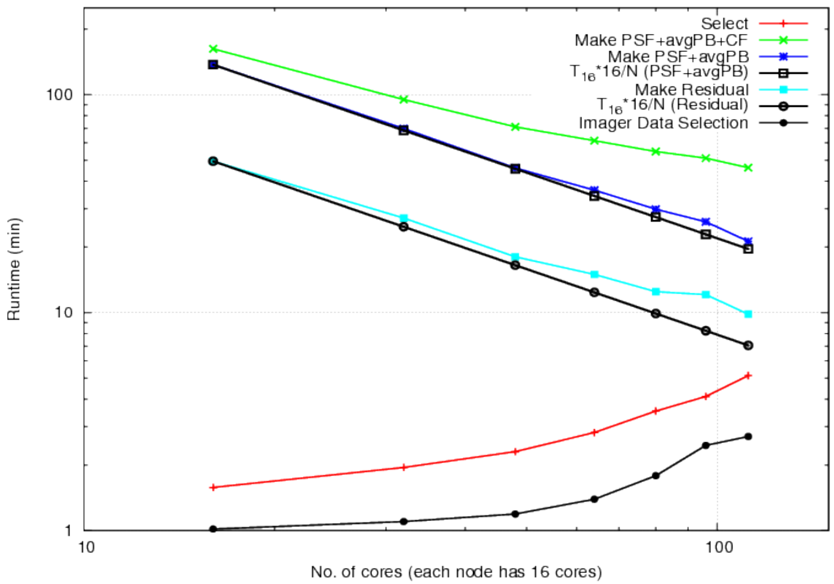
- Compute load: $N_v \times 12\ N_c^2$
- Scatter along data axis

Memory footprint increases
Linearly with no. of procs.

Too high for A-array imaging

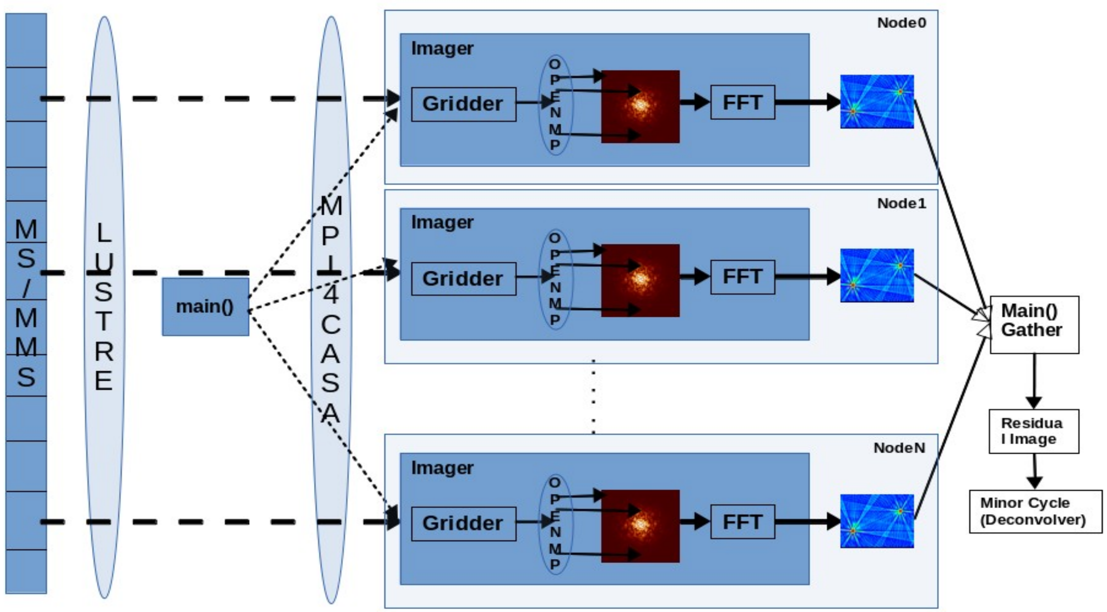
$N_{proc} \times 5 \times (N_x \times N_y)$

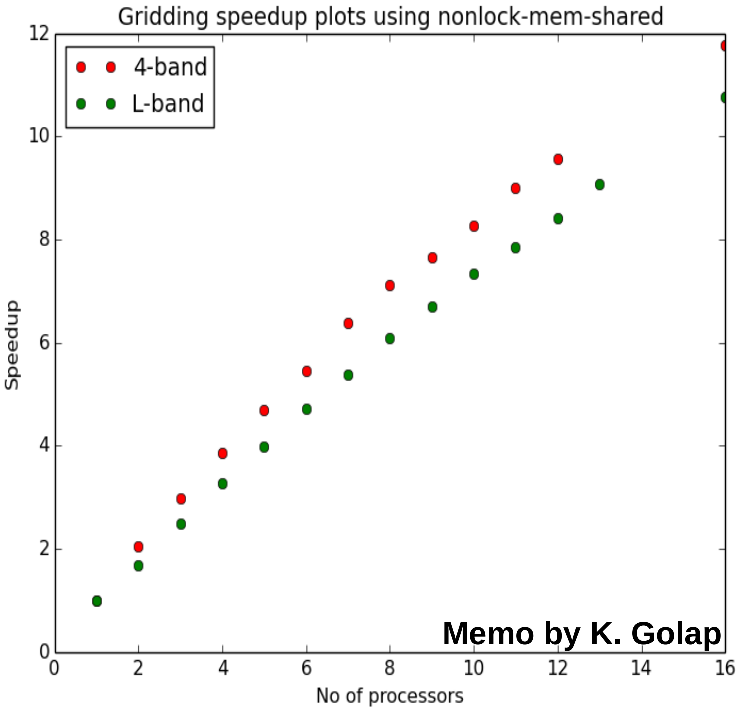




Gridder Parallelization (HS 1) - II

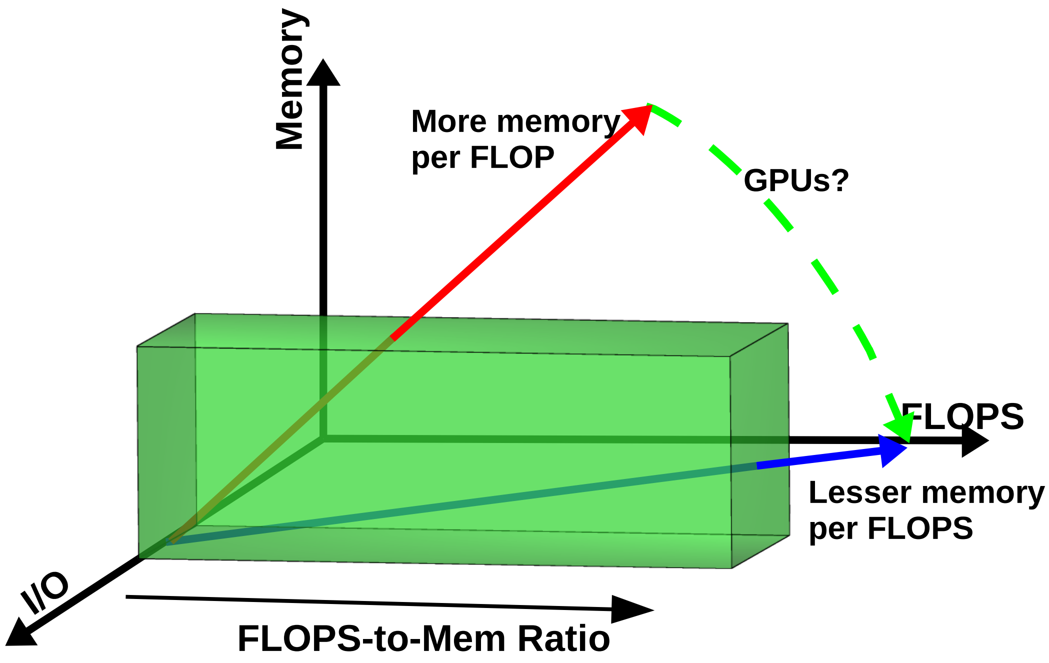
- Multi-threaded gridder – one per node
- Compute threads per core; memory footprint same as that of one gridder





Deployed in CASA

Algorithm architecture



CF Related calculations

- Compute and cache the CFs
  - Tadeoff: Extra mem. $N_{CF} = N_{Ant} x N_W$

    or
- Cache + OTF Convolutions
  - Tradeoff: $N_{CF} = N_{ant} + N_W$

    Lesser mem + Extra FLOPS

    or
- All convolutions OTF
  - Tradeoff:

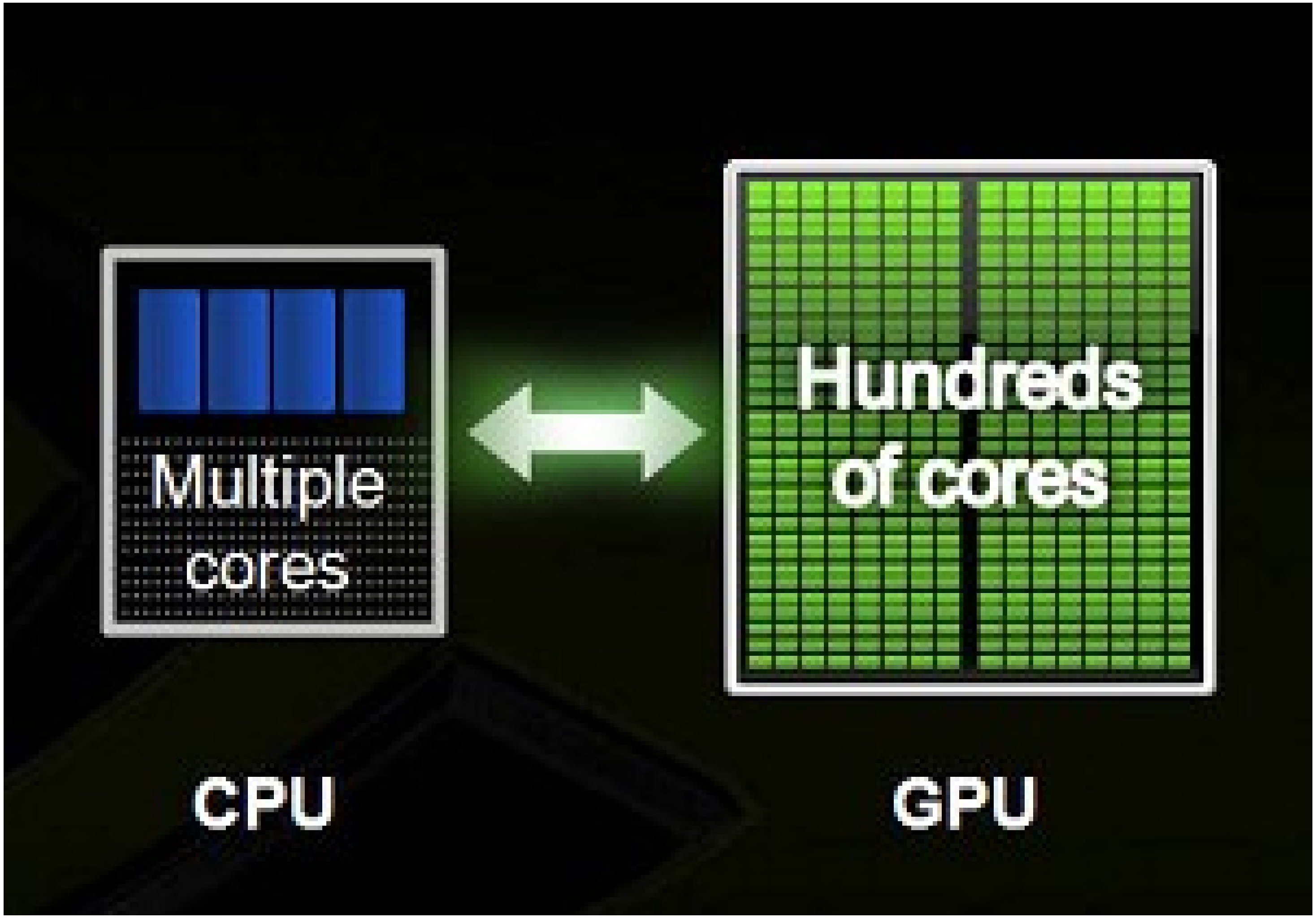
    $N_{CF} = 1$, Least mem.

    \+ More Extra FLOPS

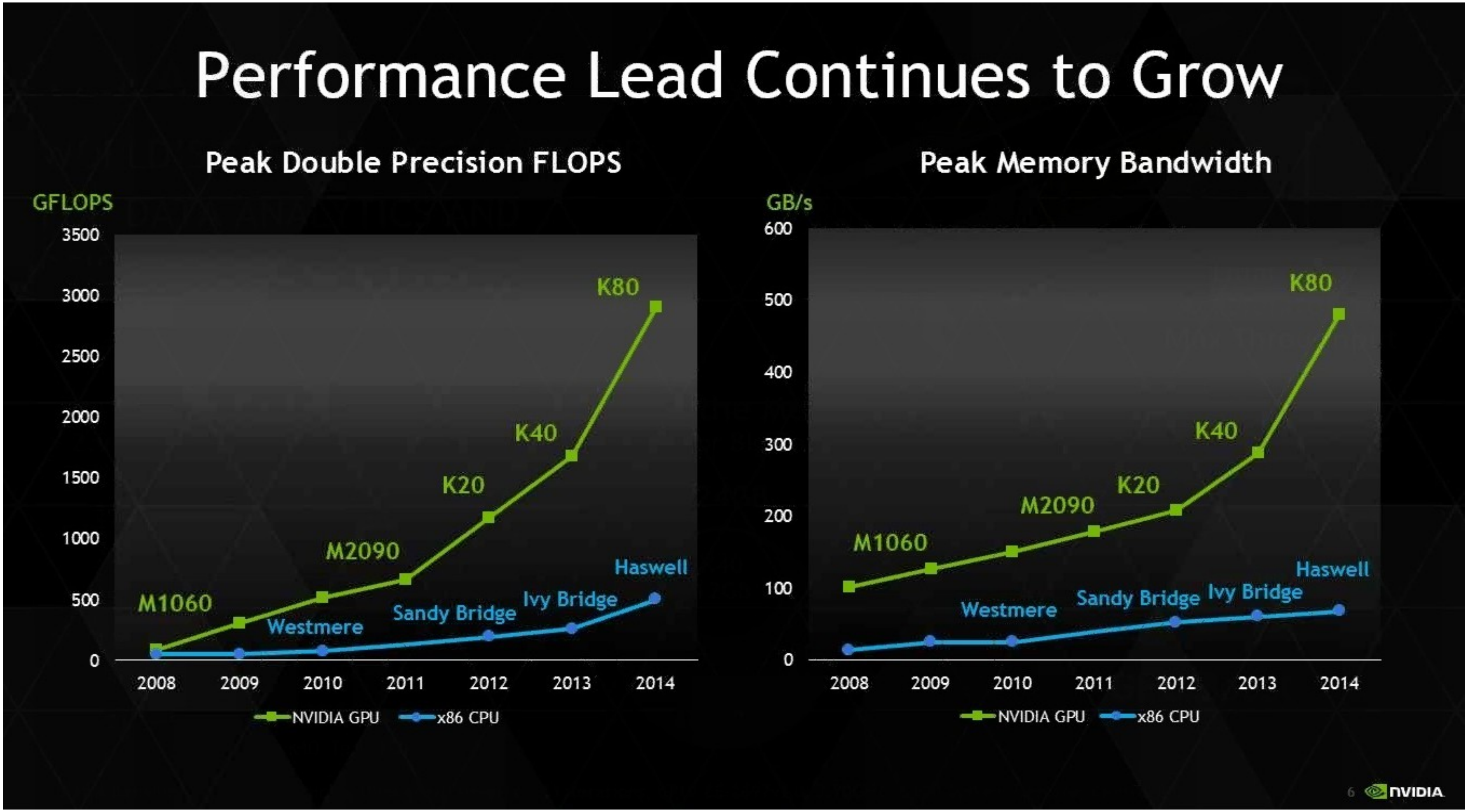
Algorithm design

- **Move towards algorithms with higher compute-to-I/O ratio**
- Reduce memory foot print, even if at the cost of higher compute-to-I/O ratio (remain inside the Green Box)
- The memory footprint of current algorithms (Red Arrow) may be reduced by trading off memory usage to higher computing via use of GPUs (Blue Arrow)

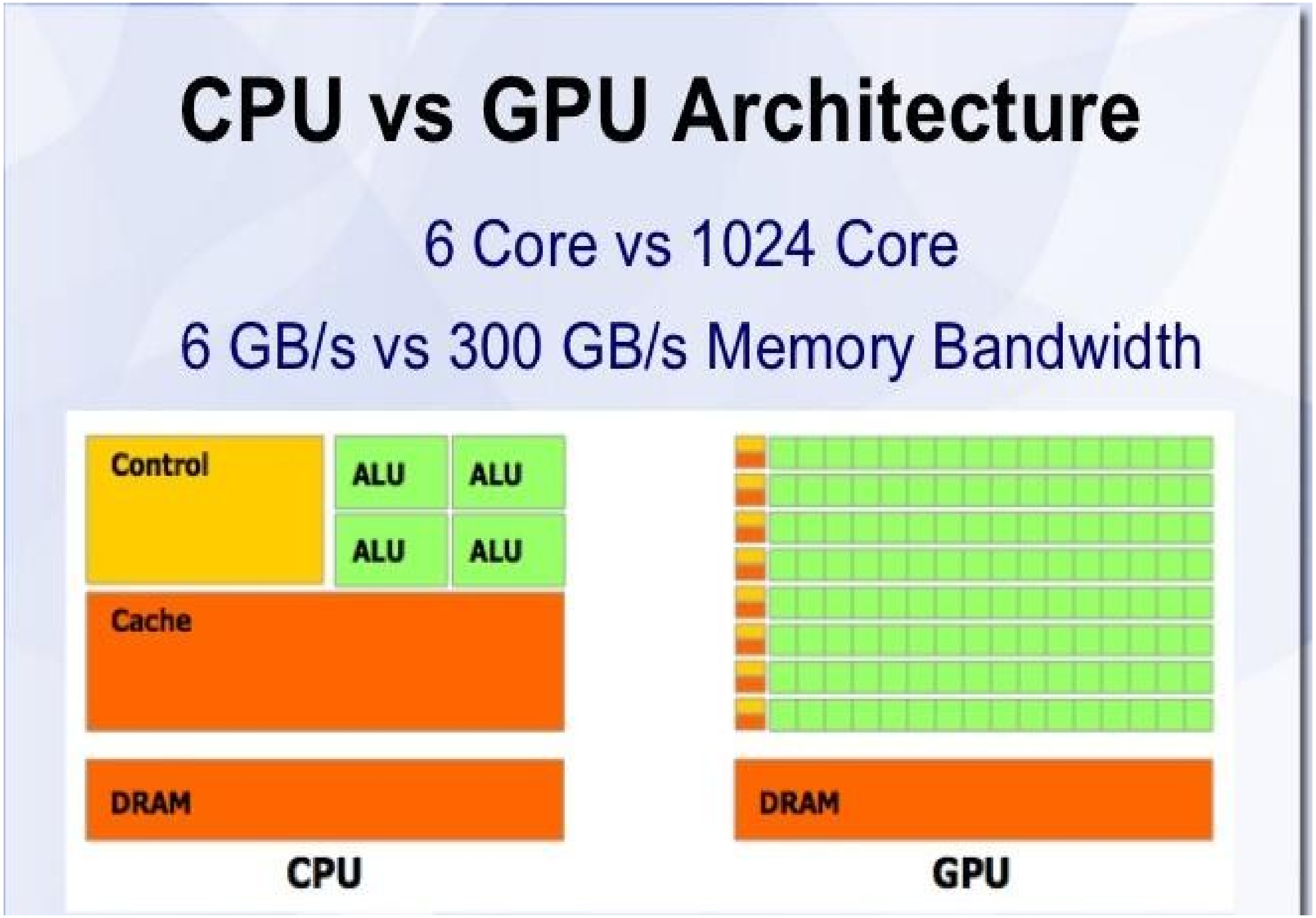
CPU vs MPH (GPU)



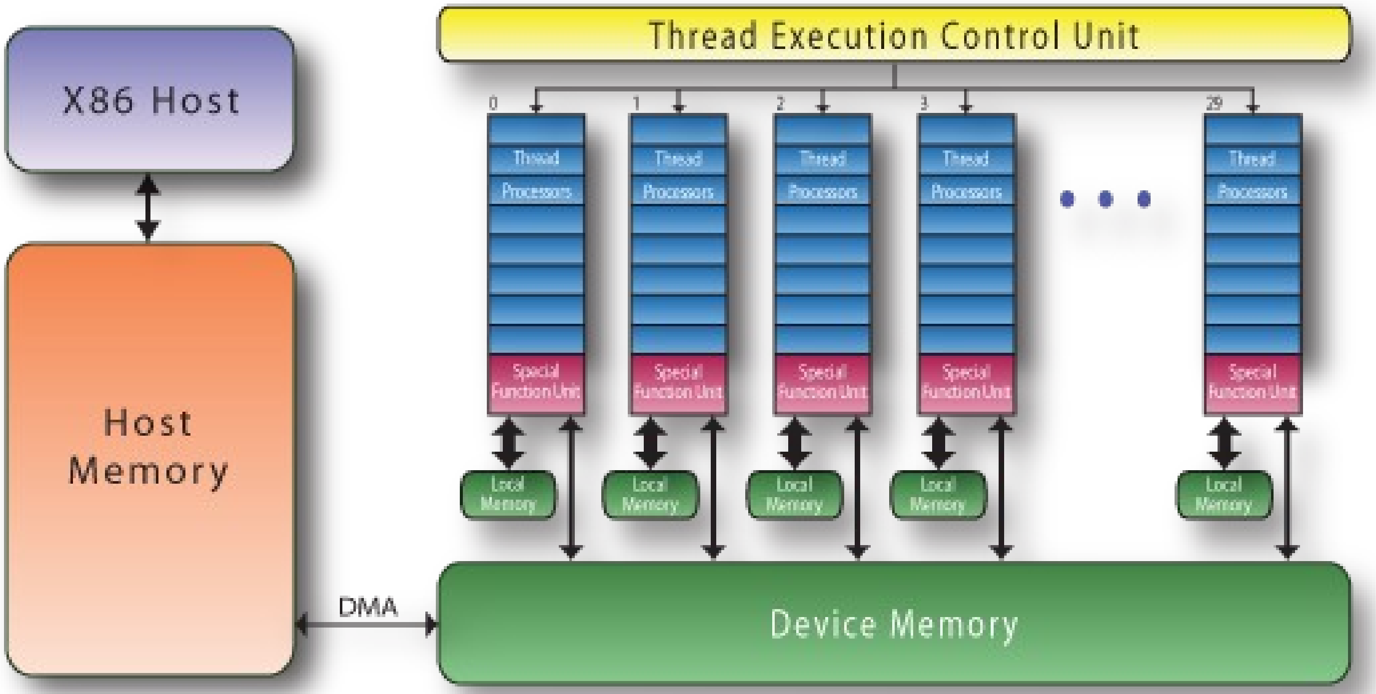
CPU vs MPH (GPU)



CPU vs MPH (GPU)

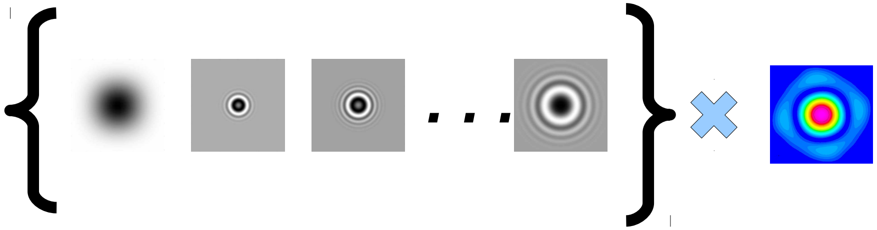


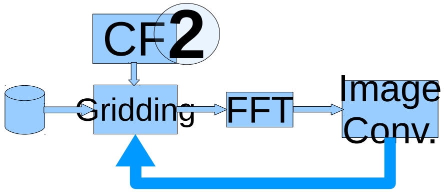
CPU vs MPH (GPU)



On-demand CF Computations (HS 2)

- Mostly computing (negligible I/o)

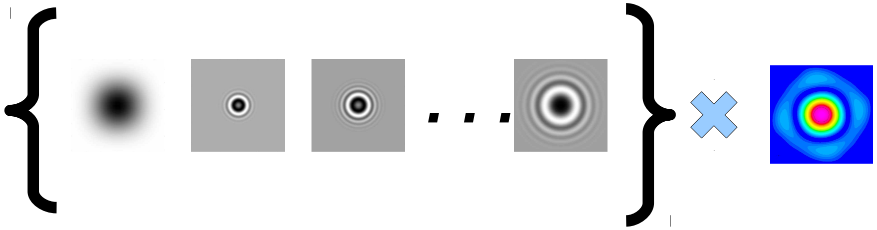


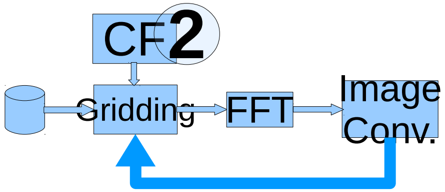


- Pre-compute A-term and cache on GPU
  - GPU/FPGA: Computer W-term OTF – one thread per pixel
  - GPU/FPGA: Multiply (W x A)
  - GPU/FPGA: FFT

On-demand CF Computations (HS 2)

- Mostly computing (negligible I/o)



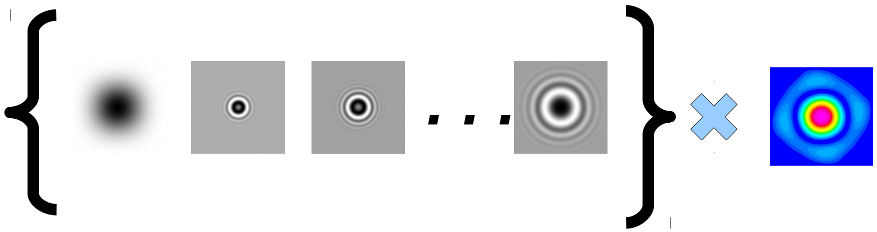


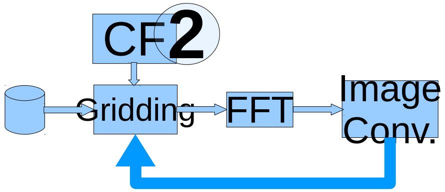
- Pre-compute A-term and cache on GPU
  - GPU/FPGA: Computer W-term OTF – one thread per pixel
  - GPU/FPGA: Multiply (W x A)
  - GPU/FPGA: FFT

- GPU measurements: 1K CF of size 2K x 2K in ~ 1ms
  - ~20x faster than CPU
  - Room for 2x – 3x improvement

On-demand CF Computations (HS 2)

- Mostly computing (negligible I/o)

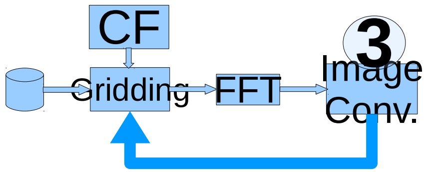




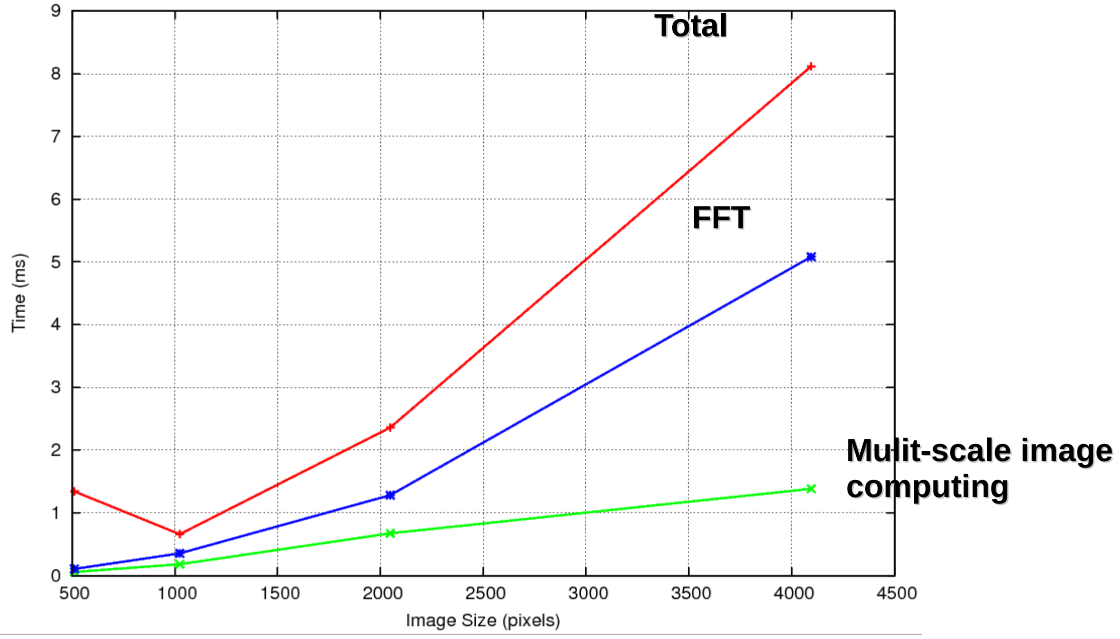
- Pre-compute A-term and cache on GPU
  - GPU/FPGA: Computer W-term OTF – one thread per pixel
  - GPU/FPGA: Multiply (W x A)
  - GPU/FPGA: FFT

- GPU measurements: 1K CF of size 2K x 2K in ~ 1ms
  - ~20x faster than CPU
  - Room for 2x – 3x improvement
- Collaboration with ALTERA Corp. for use of FPGAs in progress

Minor Cycle (HS 3)



- Compute Scale-images OTF
- Compute convolutions OTF
  - Replace memory footprint with massively parallel HPC



- To Do: Also do search and subtract on the GPU/FPGA
  - Both operations well suited for massively parallel h/w

Work from other groups

- ASTRON: Romain's implementation of gridding.
  - GPU algorithm similar/same as Golap's.
  - Speed-up 2-3x. Not quite clear compared to multi-core gridding or not (other packages do not have multi-threaded gridders for CPUs to compare with).

- University of Malta: PhD thesis of Daniel Muscat
  - Claims of 100x speed up compared to CASA
  - To be verified!
  - Measure it's performance in the useful part of the parameter space
- SKA: Various efforts with paid work from NVIDIA. But not openly available.

Work from other groups

- ASTRON: Romain’s implementation of gridding.
  - GPU algorithm similar/same as Golap’s.
  - Speed-up 2-3x. Not quite clear compared to multi-core gridding or not (other packages do not have multi-threaded gridders for CPUs to compare with).
- University of Malta: PhD thesis of Daniel Muscat
  - Claims of 100x speed up compared to CASA
  - To be verified!
  - Measure it’s performance in the useful part of the parameter space
- SKA: Various efforts with paid work from NVIDIA. But not openly available.
- None of these implementations processed realistic data. Integration with CASA or other s/w is hard and therefore haven’t been verified with realistic data set.

Take Away - 1

- GPU/FPGA very good match for computations that are independent for each pixel
  - 1000 – 5000 threads available! (Massively parallel)
  - Large on-board memory (though over slower bus)
  - Can deploy one thread per pixel (or per few pixels)
  - Good speed up with number of threads
    - OTF computations of CF
    - OTF computations of minor cycle scale images
    - Search and subtract operations of minor cycles

- Gridding is harder
  - CF support size couples pixels
    - Hierarchical memory: faster small memory , slower large memory
  - Lots of semi-random access of the image by multiple threads
  - Best speedup O(Few x) – not clear compared to what (multi-core CPU performance?)

Take Away - 2

- Basic architecture to exploit parallel computing
  - Scatter along data axis
  - Use multi-node cluster
    - Reduce total gridding time
    - Good utilization of resources at each node
  - Use one multi-threaded gridder per node
    - Keep gridding memory footprint low
    - ToDo: E2E characterization

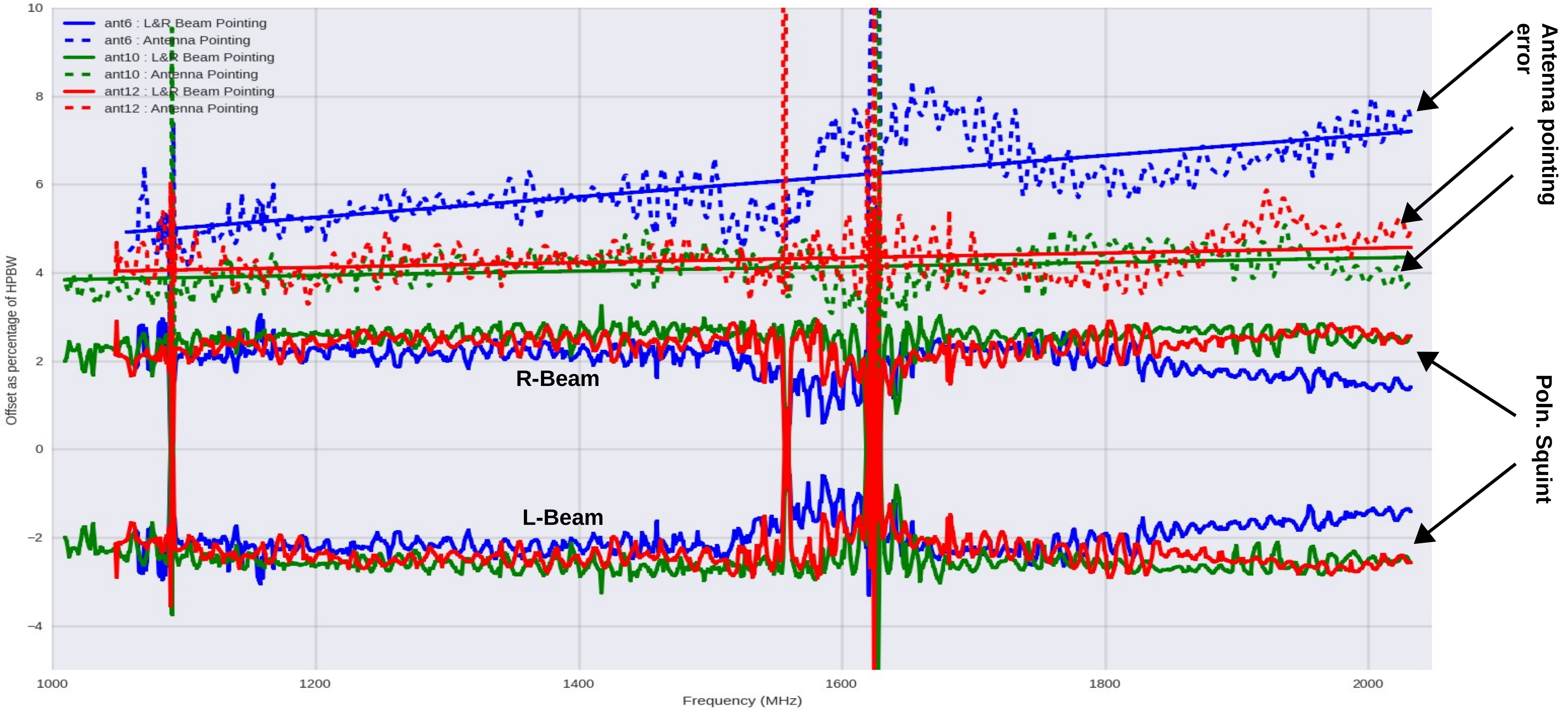
- Use GPUs/FPGAs as compute servers
  - OTF CF computations
  - Use component-based deconvolution (e.g. Asp-Clean)
      - OTF computations of the components
    - More efficient newer algorithms (Zhang et al., in prep)
  - OTF convolutions of scales

Wide-field Effects

- Pointing errors: E.g., squint: ~5.6% of PB (EVLA)

  Varies across the band. Does it matter? Limits Stokes-I DR ~10000:1

In the graph below: Optical effects should be independent of frequency (e.g. Poln. Squint)
Mechanical effects should show linear trends (e.g. Antenna pointing errors)



Data: R. Perley
Analysis: P.Jagannathan, S.Bhatnagar