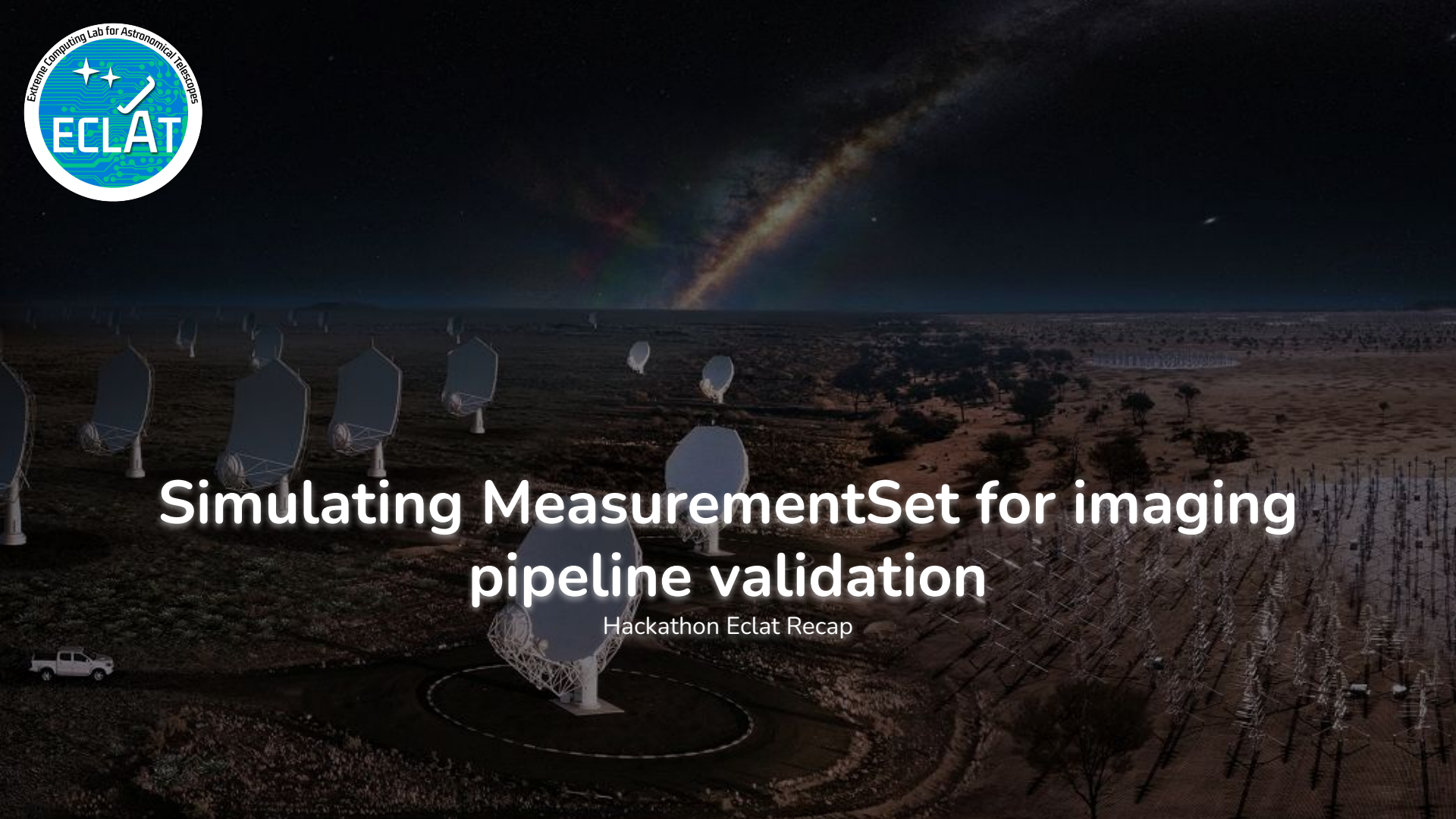
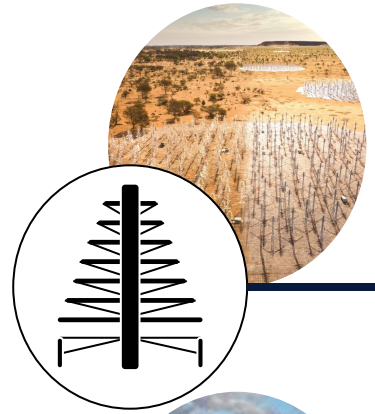


Simulating MeasurementSet for imaging pipeline validation

Hackathon Eclat Recap

Square Kilometre Array (SKA)



CSP

CSP

SDP

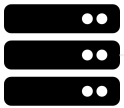
SDP

Visibilities

Output image

CSP: Central Signal Processor
SDP: Science Data Processor

Near Real-time requirement



dataAcquisitionTime ≅ pipelineProcessingTime

Latency(pipeline)

Note

Data = `N_ANT` x `N_CHANNEL` x `N_POL` x `BANDWIDTH` x `OBS_TIME` x `N_BYTE`

(10min)=130000 × 4000 × 2 × 10^6 × 600 × 4 = 2 Po
(12h)= 130000 × 4000 × 2 × 10^6 × 43200 × 4 = 179 Po

LOFAR (LOw Frequency ARray) comparison:
(10min)=4992 × 256 × 2 × 195000 × 600 × 2 = 300 To
(12h)= 4992 × 256 × 2 × 195000 × 43200 × 2 = 21 Po

NenuFAR (New Extension in Nançay Upgrading LOFAR) comparison:
(10min)=1936 × 768 × 2 × 195000 × 600 × 2 = 348 To
(12h)= 1936 × 768 × 2 × 195000 × 43200 × 2 = 25 Po

Final goal

Generating **MeasurementSet**:

👉 To benchmark pipeline implementations (latency, memory requirement, energy, output quality ...)

👉 To control the “true sky”

👉 To simulate SKA data size (without requiring instrument)

Imaging pipeline (SEP, ddfacet, generic imaging pipeline ...)

DS9 ...

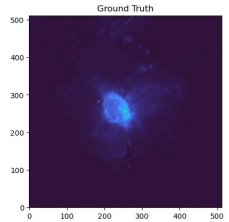
MeasurementSet.ms

[Vis, instrument config]

Vis.csv

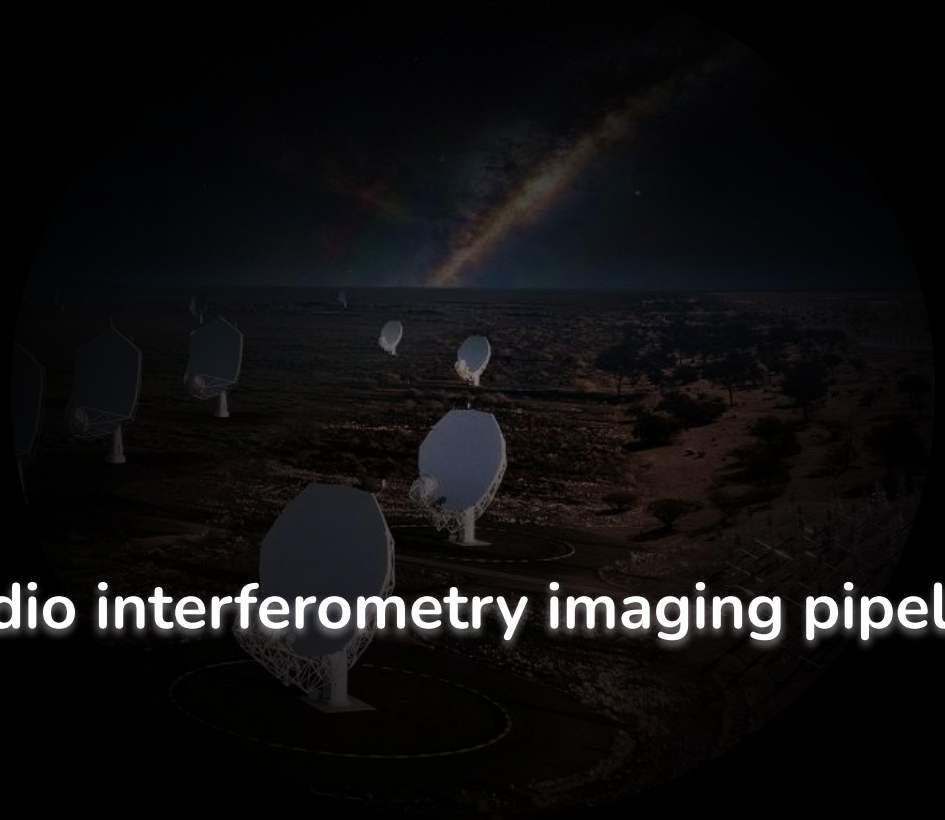
deconvolved.csv

deconvolved.fits



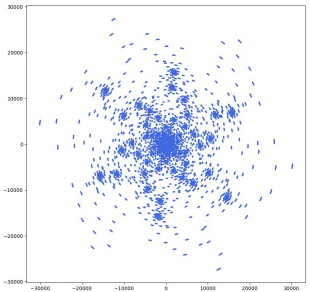
Common pipeline direction

This project



Radio interferometry imaging pipeline

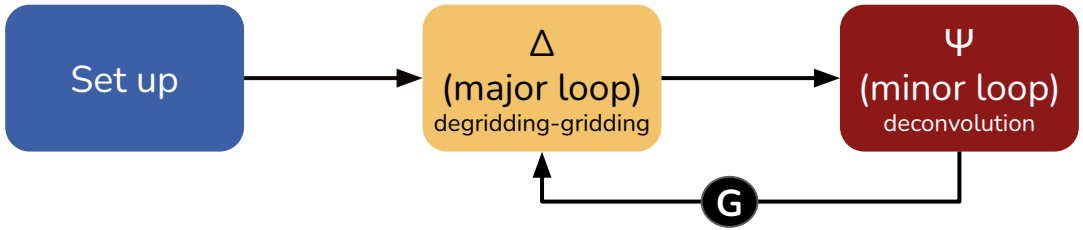
Generic radio-interferometric imaging pipeline



CSP visibilities

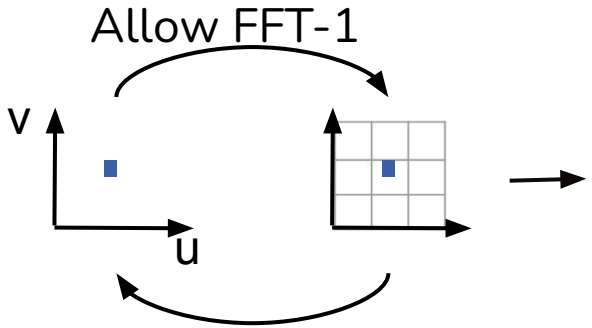
↑

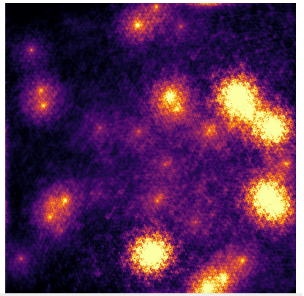
correlation point of a pair of antenna



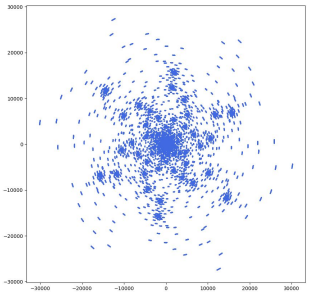
Output image

Δ: v → dirty

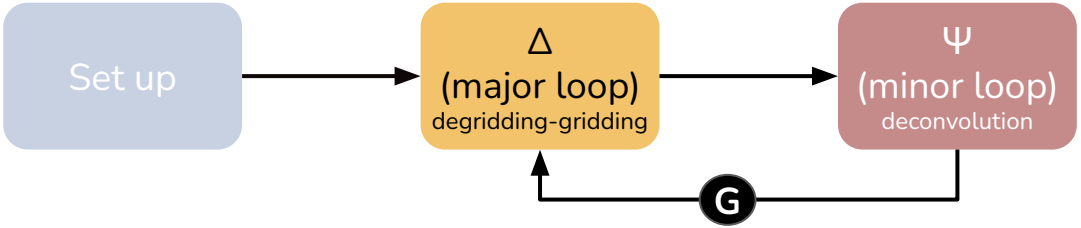




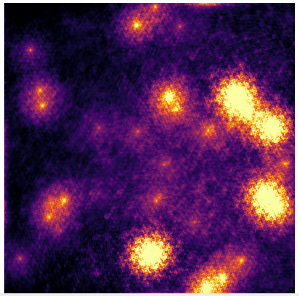
Dirty image



CSP visibilities

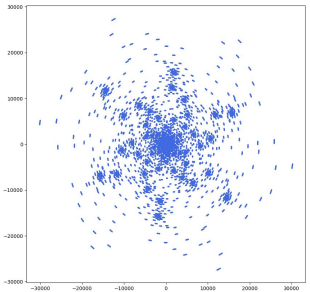


Ψ: clean dirty

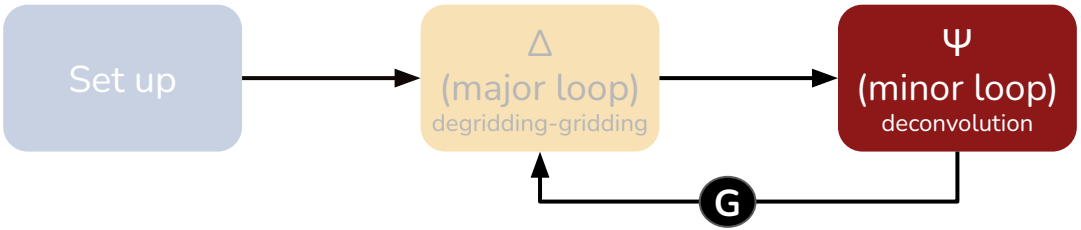


Dirty image

= Sky Image * PSF



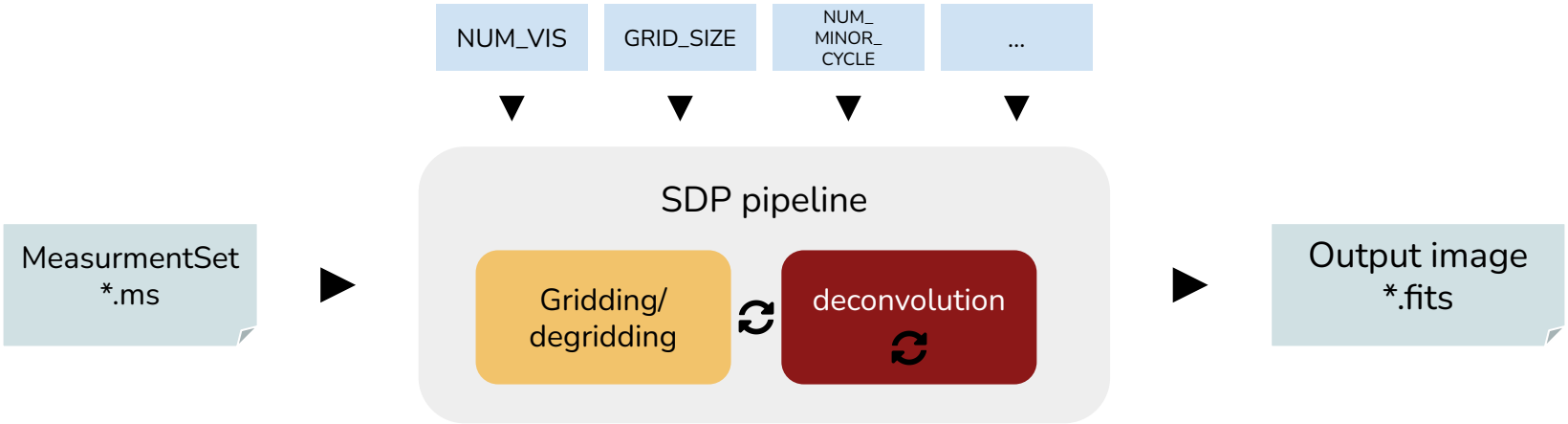
CSP visibilities

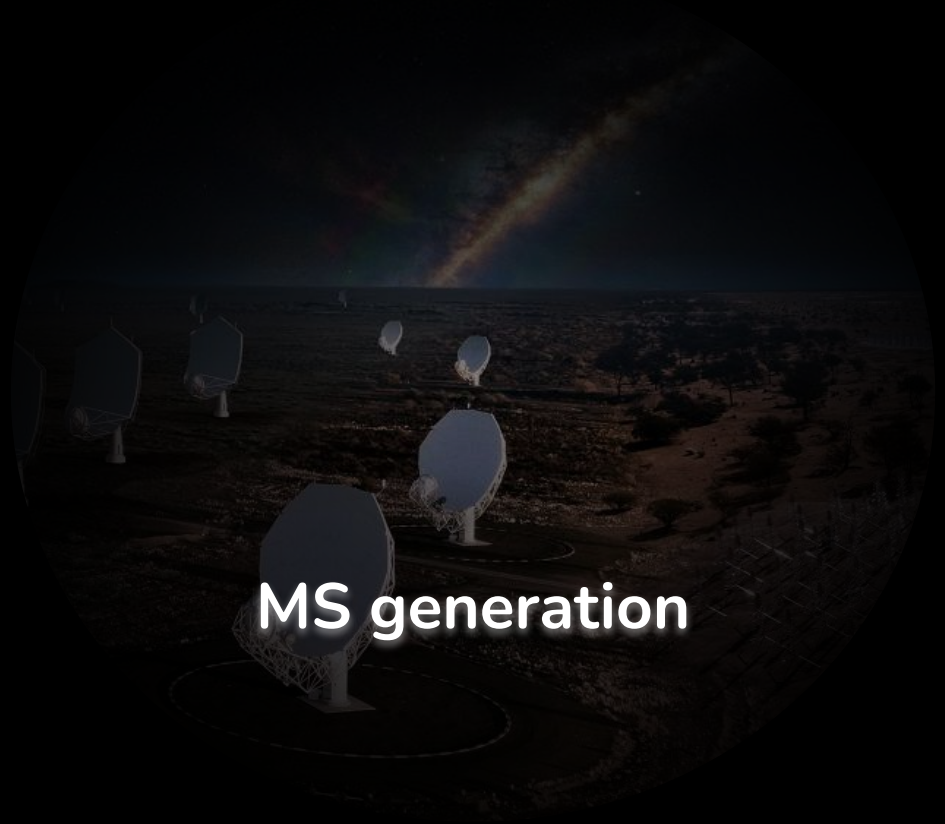


Existing imaging pipelines

	Dataflow implem	Deployment
SDP Evolutionary Pipeline (SEP) (insa gitlab here)	✅	out of date
Generic Imaging Pipeline - **DFT + Hogbom clean** - **FFT + Hogbom clean** - **G2G + Hogbom clean** (I m working here)	✅	Under discussion for SKA
DDFacet (Cyril github here)	❌ (python)	LOFAR
RASCILL (SKAO github here)	❌ (python)	??
...		

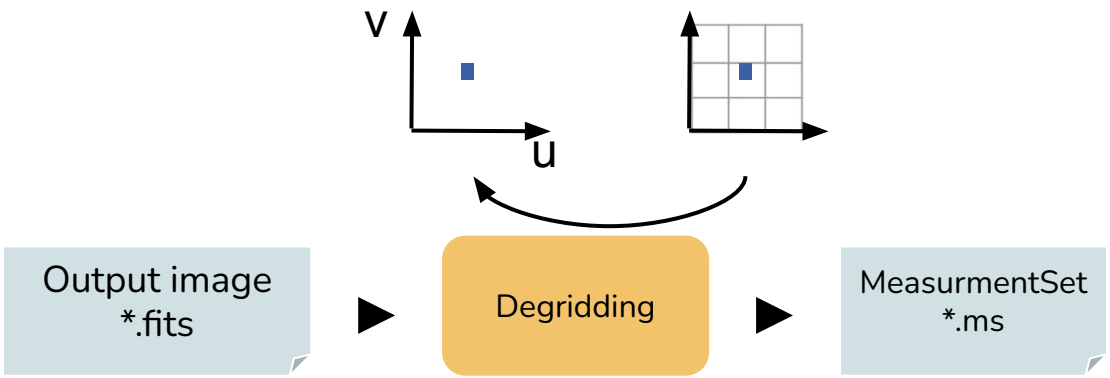
Pipelines operation





MS generation

The process



For the Hackathon, **RASCIL** degridder is used in order to avoid circular validation

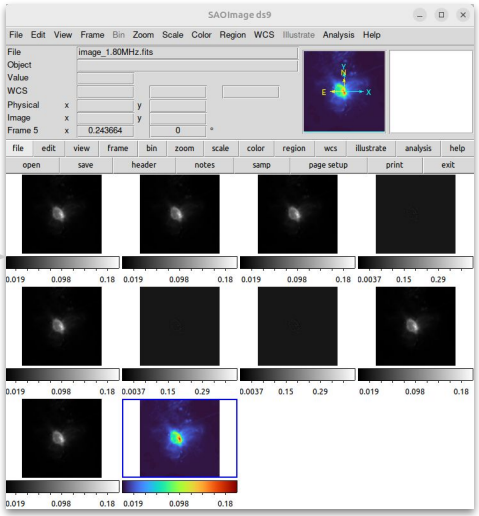
Radio Astronomy Simulation, Calibration and Imaging Library (RASCIL)

Advantages	Limitations
✅ integrate ska sdp library	❌ python
✅ facilitate existing radio-telescope simulation	❌ sequential (super long for big files)
	❌ ...

Visualize .fits

Install ds9: `sudo apt install saods9`

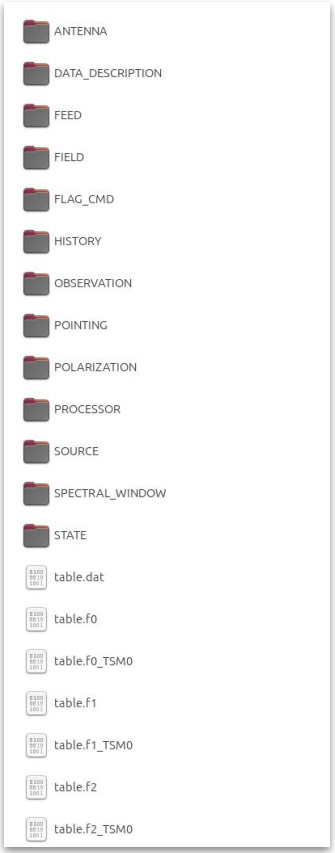
Run:`ds9 *.fits -lock frame wcs -zoom to fit`



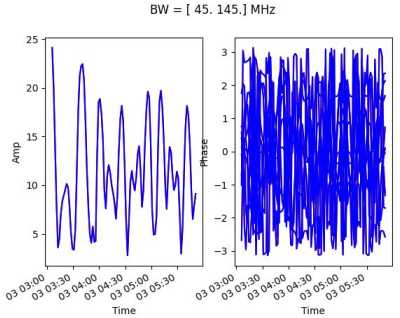
To reveal the contrasts:

- Color > Matplotlib > turbo (recommended by Sunrise)
- Color > Matplotlib > viridis / inferno (most popular in astro-papers)

Validating .ms



casacore based script



- Distributed MS simulation
  - ✅ Jupiter notebook to simplify the usage
  - ✅ Contain DDFacet tutorial
  - ✅ Contain MS validation (casacore)

  - 🌐 Project available on github: https://github.com/Ophelie-Renaud/vis-generator
  - 🌐 Official repos: https://framagit.org/eclat

- Project in progress since we completed 1/2 hackathon
  - 🚀 Configurable MS

SCAN ME