Representative Use Cases for Testing Data Reduction Pipelines

I. Prandoni (INAF-IRA), M. Rivi (INAF-IRA), M. Vaccari (IDIA)

Observational datasets

Datasets publicly available for testing pipelines optimised for different purposes and related to fields covered by different telescopes:

- COSMOS (2 sq. degrees, also targeted by MIGHTEE) www.mpia.de/COSMOS
  - Mosaic of 7 pointings in JVLA L-band, 75 MHz bandwidth, 240 hours observing time
  - mosaic of 192 pointings in JVLA S-band, 2048 MHz bandwidth, 384 hours total observing time (about 20 TBytes)
  - JVLA S-band + X-band ultra-deep pointings (partially overlapping with CHILES (HI) and CHILES CON-POL L-band), total observing time: 90 hours (S-band) + 100 hours (X-band)
- Lockman Hole (also observed with WSRT, GMRT, LOFAR)
- ELAIS N1 (also observed with DRAO, GMRT, LOFAR)

LOFAR HBA deep observation of Cluster Abell 2255 (also observed with VLA), 110-190 MHz, total observing time: up to 75 hours (about 100 TBytes)

IDIA pipeline - processMeerKAT

https://idia-pipelines.github.io/docs/processMeerKAT

- Written in Python 2.7
- Full Stokes calibration (continuum images, polarization cubes, spectral line cubes) using CASA 5.4.X tasks
- Uses a purpose-built *CASA Singularity container* for parallel processing at IDIA
- Uses multi-measurement sets (MMS) and *MPICASA* to run parallel processes over the cluster (ILIFU)
- Generates SBATCH files to submit jobs through SLURM scheduler

LOFAR DD calibration pipelines

Facet calibration for Direction Dependent effects, available on Singularity container

- **Factor** (https://github.com/lofar-astron/factor)
  - Python code exploiting "*Generic pipeline*" *framework* to define basic operations (self-calibration, model subtraction from data, imaging) as independent pipelines
  - Uses C/C++ codes of *DP3 package* for calibration and C++11 code *WSClean* for imaging (optimised for multi-threading)
  - Multi-node asynchronous parallelisation:
    - Submission of independent jobs per facet exploiting *multiprocessing* module
    - Splitting of MS per time chunks

LOFAR DD calibration pipelines

Facet calibration for Direction Dependent effects, available on Singularity container

- **DDFacet** (https://github.com/mhardcastle/ddf-pipeline)
  - Uses Python codes: *killMS* for calibration, *DDFacet* for imaging
  - Optimised parallelisation on shared memory (single node)
    - Submission of independent jobs exploiting *multiprocessing* and *SharedArray* modules for:
      - I/O and computation concurrent execution by sharing numpy arrays and splitting data in chunks
      - SSD deconvolution (default): asynchronous parallel jobs per facet
      - HMP deconvolution: vectorization of numpy expression using *numexpr* module
    - Computing intensive part (gridding/degridding) written in C++ and using OpenMP, independent jobs per facet
  - Not limited by the RAM size: data can be split in time chunks to be read and processed sequentially

Radio data simulations

- Visibilities simulations including observational effects (e.g. smearing, parallactic angle, ...)
- Available codes:
  - *MeqTrees* (within STIMELA package and KERN suite) http://meqtrees.net
  - *Montblanc* (tensorflow code exploiting both CPUs and GPUs) https://github.com/ska-sa/montblanc
  - *OSKAR* (MPI+CUDA exploiting GPU cluster) https://github.com/OxfordSKA/OSKAR
- Use cases:
  - *Replace degridding step* with model visibilities simulation using source catalog provided by the deconvolution step (implemented in DDFacet using Montblanc)
  - *Customised tests* of existing data reduction pipelines and/or new data analysis algorithms/schemes