Simple VLBI simulator documentation

Dr. Jack F. Radcliffe

1 Introduction

This simple simulator is designed to allow measurement sets to be created that can replicate VLBI arrays at low frequencies and can include custom arrays. It was borne out of initial simulations to look at the effect of primary beam attenuation on wide-field VLBI arrays but has been now developed as a tool to assist with the production of simple VLBI data sets and mosaics.

2 Installation instructions

2.1 Prerequisites

The majority of the work in getting this package installed is the various packages that are required for it to function. As a result, we recommend using `singularity` as the various packages require some shared environments which can clash with each other. These singularity images are available here.

If you decide not to use singularity, or want to build optimised images based on your compute architecture, then the packages you will need are the following:

- `simms` - https://github.com/ratt-ru/simms. This package is used to generate custom empty measurement sets. YOu can also get this installed as part of the `stimela` package.
- `CASA` - https://casa.nrao.edu. We recommend the modular version of CASA as you have more control over the packages that are installed with pip.
- `python v3.10+` - you will need the `pandas`, `astropy`, `numpy`, `scipy` and `matplotlib` packages which you can install using `pip` or `conda` (depending on how you installed python).
- `wsclean` with the `Image Domain Gridder` - https://wsclean.readthedocs.io/en/latest/. This is required if you are needing to simulate the primary beam effects and other direction-dependent effects.

2.2 Installation

Installation should be fairly simple and requires just a git clone of this repository (which you may have already done!). Nothing else is needed!

3 Quick start guide

The simulator is pretty simple to use and just needs to have the two input file (`simulator_inputs.txt` and `simulator_advanced_inputs.txt`) edited to select the array of your choice. You can copy the input files to whatever directory you want as long as they are specified when you run the simulator. The various inputs in this file and descriptions are described in Section 3.1. The simulator comprises three different steps that are controlled by a `bash` script per step. To make these scripts, we can use the following syntax on the command line,

```
singularity exec <path_to_casa_image> python <path_to_git_repository>/
    wf_vlbi_simulator.py <path_to_inputs>/simulator_inputs.txt <path_to_inputs>/
    simulator_advanced_inputs.txt <step_number>
```

The step number corresponds to the part of the simulation you want to perform and the number corresponds to the following,

1. **make ms** - generates a measurement set, inputs noise and optionally imports a model.
2. **single pointing** - s image of a single pointing including primary beam attenuation. If a mosaic is not needed, sensitivity maps are made.
3. **mosaic** - produces measurement sets corresponding to a mosaic defined in the input file. Makes images of each individual measurement set.
4. **make image** - combines the mosaic pointing into a single mosaic and generates sensitivity maps.

3.1 Input files

The simulator takes two input files, the `simulator_inputs.txt` file, which you will need to edit, while the `simulator_advanced_inputs.txt` file is probably ok being left as default (unless you want to make exact, large data size measurement sets). The `simulator_inputs.txt` file comprises a range of parameters that you need to set and the following subsections will describe these.

3.1.1 Software and paths

- `CASA_exec`, `wsclean_exec`, `stimela_exec`, and `rms_exec` - the full executable commands on how to open CASA, wsclean, stimela/simms and the rms-finding software. These must be the full command i.e., if you are using singularity then the CASA command in the input file may look something like,

```
CASA_exec = singularity exec /idia/software/containers/casa-6.3.simg python
```

- `output_path` - set this to put all outputs into this folder
- `repo_path` - path to the github repository and must include the `wf_vlbi_simulator` part.
- `prefix` - string to append to the start of all output file names. Will be set the 'sim' is left empty.

3.1.2 HPC options

The simulator is designed to be usable on both your local computer and any high-performance computing architecture which uses Slurm or PBS Pro as their job manager. For the following inputs, leaving a parameter blank means that the parameter will not be specified to the job manager (and may be set as the default value on the cluster).

- `job_manager` - defines the job manager software (options are `slurm` or `pbspro`). If this is set to `bash` then the codes can be run on your local machine.
- `partition` - selects the partition of the cluster to use.
- `walltime` - maximum time requested on the cluster to run the job.
- `nodes` - number of nodes requested.
- `cpus` - number of CPUs per node required.
- `mpiprocs` - number of tasks per node (normally good to set to the same as the number of CPUs).
- `nodetype` - defines the type of node to use (not used in Slurm)

- `max_jobs` - for the steps that use multiple jobs (e.g., the mosaic step), this specifies the maximum number of jobs to be submitted at once. Setting this to `-1` means that there is no maximum limit.

documentation in progress