Chapter 1

Software implementation

1.1 Data simulation

1.1.1 Design objectives

Our primary aim is to test and research mm-VLBI calibration, imaging and parameter estimation algorithms/strategies through the contruction of a synthetic data simulation framework. To address the many questions within the wide scope of this objective, one must be able to setup and run a diversity of experiments within the simulation framework. This places definite constraints on the software architechure. In particular, the framework should

- enable the implementation of all relevant classes of signal corruption in a consistent and rigorous manner,
- be compatible with time-variable GRMHD simulations,
- enable the construction and execution of arbitrary observations,
- create highly modular structure so as to incorporated/be incorporated by other interferometric algorithms.

1.1.2 Architechure and Workflow

In this section, we will review how the architechural design and workflow of the simulator architechure has been designed to meet the above objectives. To fulfill the first objective, we try to cast signal corruptions in the RIME (see section **??**) formalism, and where this is not possible, to fit those particular signal corruptions into the casually correct position in the pipeline. In the implementation of each signal corruption is described in the following subsections. The remaining objectives fall more into the realm of software design and will be discussed in this subsection. At the highest level, we have chosen to write the simulator using the PYTHON language. PYTHON is a general purpose language, is geared towards readability, and is well supported by a comprehensive library and wide user base (including astronomers). Specifically PYTHON interfaces well with the MEQTREES package, as well as our data formats of choice: FITS for image

cubes and the MEASUREMENT SET[1] (MS) for visibilities. Although the higher level functionality is written in PYTHON, the bulk of the computational load (MS and visibility generation) is called through other programs, which are written in the faster C++ language. The MEASUREMENT SET (MS) is data format of choice as it is directly accessible via the PYRAP library and is the format of choice for MEQTREES which is used for computing the RIME. Although in the mm-VLBI subfield other data formats are currently still more popular than the MS i.e. UVFITS and/or HOPS, with the completion of ALMA, the MS format will inevitably become the next generator data format and already is used at the Joint Institute for VLBI in Europe (JIVE).

A conceptual flow diagram of the simulator is shown in Fig. 1.1.

Highest level -¿ parameter dictionary and driver (azishe) script

parameter dictionary and standard driver script have set structure. Are highly mutable, especially the azishe scripts, e.g. the ISM calculation. Parameter dictionary contains pointers to other inputs : fits folder which details source model, station information which details weather and sefd, antenna table which details telescope positions, names and dish sizes..this is essential for the MS creation.

Input to the simulator is a sky model and configuration file. The former is typically a time-ordered list of FITS images, where each image represents the source total intensity[2] over a time interval $\Delta t_{\rm src} = t_{\rm obs}/N_{\rm src}$, where $t_{\rm obs}$ is the observation length and $N_{\rm src}$ is the number of source images. The configuration file specifies all parameters needed by the pipeline to determine the particular observation configuration (array, frequency, bandwidth, start time, etc) and which signal corruption implementation should be employed.

The MS is created using the SIMMS[3] tool.

The RIME is evaluated using the MeqTrees : turbo-sim script which visibilities are calculated through evaluation of the Fourier Transform at each UVW coordinate in the dataset, the time and frequency resolution of which is specified by the user.

The primary outputs of the pipeline are an interferometric dataset in MEASUREMENT SET format along with the closure phases and uncertainties and a dirty and/or deconvolved image (or spectral cube if desired).

Calculation is object oriented, we have should to create several classes containing Measurement Set and FITS file. The classes are separated as a way of modularisation, separating different level tasks. Makes it easier to debug.

First class is *SimpleMS* which contains only the original measurement set and derivative quantities e.g. calculated station elevations and closure phases. This class serves to make the often used attributes and functions associated with the basic MS easy to access. A key feature of SimpleMS is it specifies a function which is the only way that we have allowed data to be saved into the measurement set (aside from the meqtrees sim).

The second MS-related class is TropMS which handles all troposphere and thermal noise related corruptions. This class is a child of SimpleMS, hence it has all the attributes and functions of SimpleMS. It also has also the functions and attributes related to the tropospheric corruption - mostly taken from the station information input file.

[1] https://casa.nrao.edu/Memos/229.html

[2] Later versions of MEQSILHOUETTE will enable the full Stokes cubes as input.

[3] https://github.com/radio-astro/simms

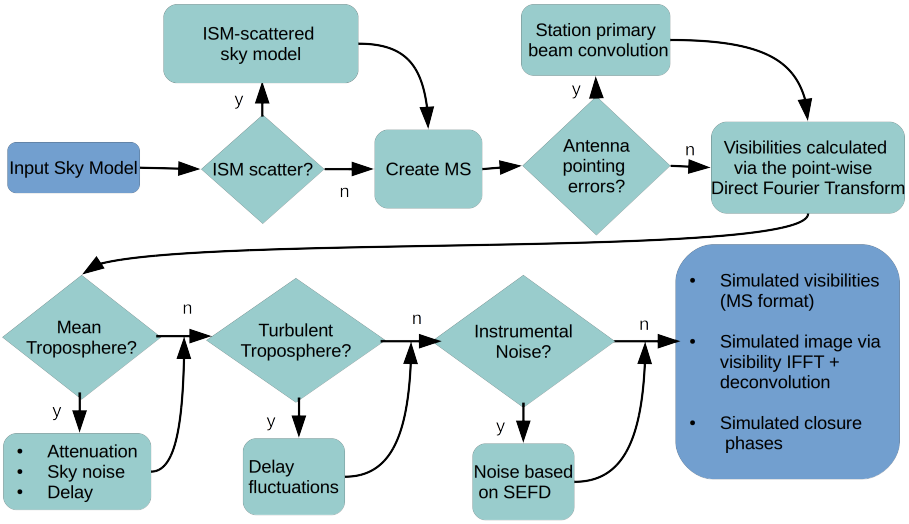

Figure 1.1: Flow diagram showing basic sequence of the MeqSilhouette simulation pipeline. The sky model could include (a) a time-ordered list of FITS images or (b) parametric source model consisting of Gaussians or point sources. The details of the station information, observation strategy, tropospheric and ISM conditions are specified in a user-defined input configuration file. The pipeline is flexible, allowing any additional, arbitrary Jones matrices to be incorporated. Further details in text.

The third MS-related class is the SimCoordinator class, a child of the TropMS class. SimCoordinator is designed to make various simulations very easy to construct and run. SimCoordinator is set-up and called in the driver script. Along with the functionality and attributes of its parents, SimCoordinator also knows about the input source model and has functionality of the MeqTrees and ISM scattering simulation scripts. Once constructed, SimCoordinator can run ISM scatter, interferometric simulations, tropospheric and instrumental corruptions, thereby presents high level functionality to the user.

The primary outputs of the pipeline are an interferometric dataset in MEASUREMENT SET format along with the closure phases and uncertainties and a dirty and/or deconvolved image cube. The modular structure of the pipeline allows for multiple imaging and deconvolution algorithms to be employed.

1.1.3 ScatterBrane

1.1.4 Atmospheric corruption simulator

1.1.5 Pointing error simulator

1.1.6 RODRIGUES interface

For community use, we host the online, RODRIGUES, interface, found at http://rodrigues.meqtrees.net/. Each of the components of the simulator run in Docker containers. **Looks like the infrustructure is going to change, re: discussions with Gijs and Sphe, so going to wait before writing this.

1.2 Parameter estimation

Bibliography