Millimeter Interferometry Simulation Cookbook

J. Pety[1], F. Gueth[1],S. Guilloteau[2]

31-May-2005

Version 1.0

(1) Institut de Radio Astronomie Millimétrique
300 Rue de la Piscine
F-38406 Saint Martin d'Hères

This document presents part of the tools used to simulate Millimeter Interferometry observations.

Related information is available in:

- SIC: Command line interpretor
- GREG: Graphical possibilities
- MAPPING: Imaging and deconvolution

Contents

1 List of Associated Tasks **3**
1.1 millimeter-simulation . . . 3
1.2 ARRAY_LAYOUT . . . 3
1.3 ARRAY_PROJECT . . . 4
1.4 AZIMUTH_AVERAGE . . . 4
1.5 IMAGE_SAMPLING . . . 4
1.6 SHORT_MODEL . . . 4
1.7 PHASE_SCREEN . . . 5
1.8 SORT_CLEAN . . . 6
1.9 UV_TABLE . . . 6
1.10 UV_MODEL . . . 6
1.11 UV_FMODEL . . . 6
1.12 UV_CCMODEL . . . 6
1.13 UV_CCT . . . 7
1.14 UV_TRACK_PHASE . . . 7
1.15 UV_OBSERVE . . . 8
1.16 UV_OBSERVE_NEW . . . 8
1.17 UV_POINTING . . . 9
1.18 UV_POINTING_NEW . . . 9
1.19 UV_SPLITFIELD . . . 10
1.20 UV_APPLYPHASE . . . 10
1.21 UV_SINUSPHASE . . . 10
1.22 UV_TIMEAVERAGE . . . 10
1.23 UV_TIMEBASE . . . 10
1.24 UV_HYBRID . . . 10
1.25 UV_FIDELITY . . . 11
1.26 UV_PLUGC . . . 11
1.27 UV_FCCT . . . 11
1.28 UV_APPLYPHASE . . . 11
1.29 UV_ADDNOISE . . . 11
1.30 UV_SOLVE . . . 11

1 List of Associated Tasks

1.1 millimeter-simulation

```
ARRAY_LAYOUT     Create an interferometer configuration file from input paramete
ARRAY_PROJECT    Undocumented...
AZIMUTH_AVERAGE  Compute the azimuthal sum and average of one plane of a data cu
IMAGE_SAMPLING   Define a sampling grid for use as input to SHORT_MODEL
SHORT_MODEL      Simulate single-dish observations
PHASE_SCREEN     Generate a 2-D phase screen to simulate atmospheric contributio
SORT_CLEAN       Sort a Clean Component Table by Intensity
UV_TABLE         Compute a Nyquist sampled UV table from a model data cube
UV_MODEL         Compute a UV table (using a template sampling) from a model dat
UV_FMODEL        Compute a UV table (using a template sampling) from a model dat
UV_CCMODEL       Create a model UV table from a Clean Component List.
UV_CCT            (Obsolete?, see UV_CCMODEL) Create a UV table from clean image
UV_TRACK_PHASE   Compute uv coverage + phase noise
UV_OBSERVE       Compute uv coverage + phase noise + amplitude errors
UV_OBSERVE_NEW   Compute uv coverage + phase/amplitude errors + anomalous refrac
UV_POINTING      Simulate pointing errors
UV_POINTING_NEW  Compute a UV table with pointing errors from a model data cube
UV_SPLITFIELD    Split a multi-field UV table into a series of single fields
UV_APPLYPHASE    Apply a gain solution to a UV table.
UV_SINUSPHASE    Create a model UV table with sinusoidal phase variation
UV_TIMEAVERAGE   Smooth a UV table according to time
UV_TIMEBASE       Sort a UV table into Time-Baseline order
UV_HYBRID        UV plane hybridization of two images
UV_FIDELITY      Compute Fidelity in the UV plane from two images
UV_PLUGC         Plug continuum averaged visibilities into a line UV table
UV_FCCT          Compute a UV table (using a template sampling) from a list of c
UV_APPLYPHASE    Apply a gain solution to a UV table.
UV_ADDNOISE      Add noise to a UV table,
UV_SOLVE         Compute a gain solution from a calibrator UV table
```

1.2 ARRAY_LAYOUT

ARRAY_LAYOUT

This task creates a configuration file from the input parameters (number of antennas, latitude,...) and the kind of array to be used:

1 = compact
2 = disk + random
3 = ring
4 = ring + random
5 = reuleaux triangles
6 = reuleaux triangles + random

7 = spiral

1.3 ARRAY_PROJECT

ARRAY_PROJECT

1.4 AZIMUTH_AVERAGE

AZIMUTH_AVERAGE

Compute the azimuthal sums and averages as a function of radius of one plane of a data cube. The azimuthal sums and averages are computed around the central pixel (i.e. nx/2+1, ny/2+1).

1.5 IMAGE_SAMPLING

IMAGE_SAMPLING

Define a sampling grid for use as input to SHORT_MODEL. The sampling grid is a regular rectangular grid large enough to cover the size of the input sky image. It is better than Nyquist sampled, i.e. the pixel size is the beam width divided by POINT_PER_BEAM$. One grid point either pass through the sky image center or the reference pixel available in the sky image.

To better mimic On-The-Fly observations the grid should have Nyquist sampling perpendicular to the scanning direction.

The grid nod positions are given in radian as offsets compared to the reference position of the sky image. The grid is duplicated by the number of Single-Dish antenna simultaneously observing. The output table has the following format: Number of position x Number of antenna x 2 (Lambda and Beta coordinates).

1.6 SHORT_MODEL

SHORT_MODEL

This task simulates observations performed with single-dish antennas, to be used later on as short spacings information. The model image is convolved by the antenna lobe (via Fourier Transform) and the intensity is then estimated at (i) the position of the mosaic fields, and (ii) on an externally defined grid (see IMAGE_SAMPLING documentation). If the required position does not coincide with a pixel center, a bilinear interpolation from the neighbor pixels is performed.

Pointing errors can be simulated by estimating the intensity at a slightly wrong position. This task can generate simple kind of pointing errors but it can also read an input pointing error table (enabling to simulate much more complex kind of pointing errors).

Thermal noise can then be added to the data, and the corresponding weight is stored. Finally, a calibration error can be simulated: the observed intensities are multiplied by a random factor whose mean value (different from 1 if a systematic error is present) and rms can be specified. In this case, we obtain: results = cal_rr*(model+thermal0ise).

The error on the amplitude gain is modeled as the sum of an offset and a drift with time. The offset and drift values are randomly reset at each calibration for each antennas. In addition, an offset common to all the antennas can also be added to the amplitude gain.

Intensity unit of input image is supposed to be Kelvin. The output table has the following format: 4 columns x Number of antenna x (NFIELDS$ + Number of observed positions). Column 1 is the X offset in radian, column 2 the Y offset in radian, column 3 the weight and column 4 the flux in Jy. The first NFIELDS$ lines of the model are the position of the mosaic for use by UV_ZERO and all the other ones are the nod of the externally defined grid (by IMAGE_SAMPLING) for use by UV_SINGLE. This complex line layout comes from historical reasons (as usual)...

Limitation: only image can be processed (i.e. *no* data cube) meaning that this task can not (yet) handle spectra cubes.

1.7 PHASE_SCREEN

PHASE_SCREEN

Warning: This tasks is encapsulated into the "phaseeen.map" procedure which enables an easy viewing of the result.

To compute the atmospheric phase errors, this tasks generate a 2-D phase screen on a grid with sufficient spatial resolution to sample the antenna diameter (typically 4-m pixels for ALMA, and 4-m pixels for Bure). The phase screen is generated in the Fourier plane with the constraint that its 2nd order structure function is a combination of 3 power laws in 3 different spatial ranges. The resulting phase screen is averaged over the effective dish diameter. Because of the size limitation imposed by the FFT, very long phase screen are built as a linear combination of independently generated screens. This is correct since atmospheric pathlength variations are completely uncorrelated at large distances.

Dynamic (anomalous) refraction is directly proportional to the phase gradient. This tasks thus computes the phase gradient associated to the phase screen, in order to obtain a coherent derivation of the dynamic refraction term. As the phase screen, the phase gradient is averaged over the effective dish diameter.

1.8 SORT_CLEAN

SORT_CLEAN

Sort a Clean Component Table in decreasing order of intensity.

1.9 UV_TABLE

UV_TABLE

Compute a UV table from an input image (3-D data cube are allowed). It differs from UV_MODEL in producing UV data on a critically sampled rectangular grid rather than along user specified tracks. It is typically used to analyse image plane observational data with UV tools.

1.10 UV_MODEL

UV_MODEL

Compute a UV table from an input image (3-D data cube are allowed) and a reference UV table used to specify the UV sampling. Task UV_FMODEL will do the same, but faster, by using an intermediate FFT plus resampling rather than the direct Sin and Cos.

Related task UV_TABLE will do the same for a regular output UV grid rather than on the sampling of a template table.

1.11 UV_FMODEL

UV_FMODEL

Compute a UV table from an input image (3-D data cube are allowed) and a reference UV table used to specify the UV sampling. This is the fast version of UV_MODEL, which uses an intermediate FFT plus resampling rather than the direct Sin and Cos.

1.12 UV_CCMODEL

UV_CCMODEL

Use a Clean Component Table produced by WRITE CCT (in Mapping) to compute a UV table from an input UV coverage. It can also be used to subtract the Clean Components from a UV table to compute the residuals.

1.13 UV_CCT

UV_CCT

Compute a UV table from a Clean Component Table. See also UV_CCMODEL.

1.14 UV_TRACK_PHASE

UV_TRACK_PHASE simulates the uv coverage of an observation specified by the source, observatory, and hour angle range. The output is a uv table that can be used as input for tasks such as UV_FMODEL. Caution: this uv table is not standard, as it includes two addition columns (#11 with elevation, #12 with integration time).

The weight column (#10) is filled with as realistic as possible estimates of the actual weights (derived from Trec, Tau, elevation).

The visibilities columns (#8 and #9) are filled with a point source (amplitude = 1, phase = 0). Random phase noise can be added, to simulate atmospheric phase noise.

If DO_SCREEN = YES, a phase screen is used to simulate phase noise: this file contains a statistically correct spatial distribution of the phase perturbation induced by the atmopshere. This screen moves above the array at the wind velocity. The correct phase value is computed each antenna at each time dump.

If DO_CALIB = YES, the observations of a calibrator are simulated, assuming a loop calibrator-source-calibrator-source. The corresponding uv table is created.

If DO_RADIOM = YES, a phase correction based on WVR measurements is simulated.

UV_TRACK family of tasks (in order of increasing complexity):

- UV_TRACK: uv coverage + random phase noise
- UV_TRACK_PHASE: phase noise from atmospheric screen + observations of calibrator + WVR correction
- UV_OBSERVE: also includes amplitude noise

- UV_OBSERVE_NEW: also includes anomalous refraction

1.15 UV_OBSERVE

UV_OBSERVE simulates the uv coverage of an observation specified by the source, observatory, and hour angle range. The output is a uv table that can be used as input for tasks such as UV_FMODEL. Caution: this uv table is not standard, as it includes two addition columns (#11 with elevation, #12 with integration time).

The weight column (#10) is filled with as realistic as possible estimates of the actual weights (derived from Trec, Tau, elevation).

The visibilities columns (#8 and #9) are filled with a point source (amplitude = 1, phase = 0). Amplitude calibration errors (offset + drifts) can be included. Random phase noise can be added, to simulate atmospheric phase noise.

If DO_SCREEN = YES, a phase screen is used to simulate phase noise: this file contains a statistically correct spatial distribution of the phase perturbation induced by the atmopshere. This screen moves above the array at the wind velocity. The correct phase value is computed each antenna at each time dump.

If DO_CALIB = YES, the observations of a calibrator are simulated, assuming a loop calibrator-source-calibrator-source. The corresponding uv table is created.

If DO_RADIOM = YES, a phase correction based on WVR measurements is simulated.

UV_TRACK family of tasks (in order of increasing complexity):

- UV_TRACK: uv coverage + random phase noise
- UV_TRACK_PHASE: phase noise from atmospheric screen + observations of calibrator + WVR correction
- UV_OBSERVE: also includes amplitude noise
- UV_OBSERVE_NEW: also includes anomalous refraction

1.16 UV_OBSERVE_NEW

UV_OBSERVE_NEW simulates the uv coverage of an observation specified by

the source, observatory, and hour angle range. The output is a uv table that can be used as input for tasks such as UV_FMODEL. Caution: this uv table is not standard, as it includes two addition columns (#11 with elevation, #12 with integration time).

The weight column (#10) is filled with as realistic as possible estimates of the actual weights (derived from Trec, Tau, elevation).

The visibilities columns (#8 and #9) are filled with a point source (amplitude = 1, phase = 0). Amplitude calibration errors (offset + drifts) can be included. Random phase noise can be added, to simulate atmospheric phase noise.

If DO_SCREEN = YES, a phase screen is used to simulate phase noise: this file contains a statistically correct spatial distribution of the phase perturbation induced by the atmopshere. This screen moves above the array at the wind velocity. The correct phase value is computed each antenna at each time dump.

If DO_CALIB = YES, the observations of a calibrator are simulated, assuming a loop calibrator-source-calibrator-source. The corresponding uv table is created.

If DO_RADIOM = YES, a phase correction based on WVR measurements is simulated.

UV_TRACK family of tasks (in order of increasing complexity):

- UV_TRACK: uv coverage + random phase noise
- UV_TRACK_PHASE: phase noise from atmospheric screen + observations of calibrator + WVR correction
- UV_OBSERVE: also includes amplitude noise
- UV_OBSERVE_NEW: also includes anomalous refraction

1.17 UV_POINTING

UV_POINTING

Fast pointing error simulation based on gridded interpolation from the Fourier Transform of an image

1.18 UV_POINTING_NEW

UV_POINTING_NEW

Compute a UV table from an input image (2-D images only) and a reference UV table used to specify the UV sampling. Pointing errors are added from an independant table. This task can be used to model a mosaic observation with or without pointing errors.

1.19 UV_SPLITFIELD

UV_SPLITFIELD

Use for ALMA simulation. Splits a special UV table containing data for overlapping fields into separate, single field UV tables.

1.20 UV_APPLYPHASE

UV_APPLYPHASE

Apply phase correction to a UV table. The first UV table contains the gain correction, the second the visibilities to be corrected. Works only for one channel for the time being...

1.21 UV_SINUSPHASE

Used in ALMA simulator during its debugging, and perhaps in some version still nowadays...
UV_SINUSPHASE

1.22 UV_TIMEAVERAGE

Smooth a UV table according to time.

1.23 UV_TIMEBASE

UV_TIMEBASE

1.24 UV_HYBRID

UV_HYBRID

Compute a synthetic images from two input ones which contain information on
different scales.

1.25 UV_FIDELITY

UV_FIDELITY

Compute Fidelity in the UV plane from two images

1.26 UV_PLUGC

UV_PLUGC puts a time average continuum UV table into the last channel of a spectral line one. It is intended for specific purposes wher having the continuum and line data in the same table is required. The weights of the continuum data is kept also.

1.27 UV_FCCT

UV_FCCT

Compute a UV table from a list of Clean Components (for several channels) and a reference UV table used to specify the UV sampling. This is the fast version of UV_CCT, which uses an intermediate FFT plus resampling rather than the direct Sin and Cos.

1.28 UV_APPLYPHASE

UV_APPLYPHASE

Apply phase correction to a UV table. The first UV table contains the gain correction, the second the visibilities to be corrected. Works only for one channel for the time being...

1.29 UV_ADDNOISE

UV_ADDNOISE

Add noise to a UV table, according to the weights of a specified channel, with an optional scale factor.

1.30 UV_SOLVE

UV_SOLVE

Compute a gain solution from a calibrator UV table

Index

ARRAY_LAYOUT, 3
ARRAY_PROJECT, 4
AZIMUTH_AVERAGE, 4

IMAGE_SAMPLING, 4

millimeter-simulation, 3

PHASE_SCREEN, 5

SHORT_MODEL, 4
SORT_CLEAN, 6

UV_ADDNOISE, 11
UV_APPLYPHASE, 10, 11
UV_CCMODEL, 6
UV_CCT, 7
UV_FCCT, 11
UV_FIDELITY, 11
UV_FMODEL, 6
UV_HYBRID, 10
UV_MODEL, 6
UV_OBSERVE, 8
UV_OBSERVE_NEW, 8
UV_PLUGC, 11
UV_POINTING, 9
UV_POINTING_NEW, 9
UV_SINUSPHASE, 10
UV_SOLVE, 11
UV_SPLITFIELD, 10
UV_TABLE, 6
UV_TIMEAVERAGE, 10
UV_TIMEBASE, 10
UV_TRACK_PHASE, 7