
Simulation of MWA Visibilities

Baijayanta Bhattacharyya

Feb 02, 2025

Contents:

1	GRF_Generate module	1
2	PB_Phase module	2
3	Params_MWA module	5
4	Vis_Gen module	6
5	Vis_calculate module	7
	Index	8

GRF_GENERATE MODULE

This can generate Gaussian Random Field For a given APS.

Functions:

- APS_func
- GRF_hmap_gen

GRF_Generate.**APS_func**(*l*)

This is the input model Angular Power Spectrum(APS) Function.

$$C_\ell = Amp \ell^\beta$$

Amp here defines the Amplitude of APS. β here is the power index.

Parameters

l (*float or array*) – The Angular Multipole value.

Returns

The Angular Power Spectrum Value for that multipole(s).

Return type

float or array

GRF_Generate.**GRF_hmap_gen**(*nside, l_max, nu, func, iseed=-1*)

Generates The GRF(Gaussian Random Field for all the pixels) for the input APS.

Parameters

- **nside** (*int*) – Healpix parameter Usually power of 2.
- **l_max** (*int*) – Maximum Value of Angular Multipole.
- **nu** (*float*) – The Observing frequency ν in MHz.
- **func** (*function*) – The input APS.
- **iseed** (*int or float*) – A seed value. In order to get different realizations for same APS.

Returns

Contains the GRF value for each pixels. Shape $1 \times 12 \text{ } nside^2$.

Return type

array

PB_PHASE MODULE

This can Primary Beam Pattern of MWA.

Functions:

- needful_pixels
- hat_n
- beam_mwa
- Primary_Beam_generate
- dot_cal_superfast
- calculate_phase

`PB_Phase.Primary_Beam_generate(nside, a0, d0, nu, beam_mwa)`

This is the main function which can generate the Primary Beam (for all the pixels which makes angle less than 90 deg by default w.r.t. the pixel in the direction given by *a0*,*d0*) for MWA given the RA,DEC and frequency.

Parameters

- **nside** (*int*) – Healpix parameter. Usually Power of 2.
- **a0** (*int or float*) – RA of the pointing direction. (In Degrees)
- **d0** (*int or float*) – DEC of the pointing direction. (In Degrees)
- **nu** (*float*) – Observing Frequency ν in MHz unit.
- **beam_mwa** (*function*) – The PB pattern of MWA.

Returns

PB – The value of Primary beam of The Telescope. Shape $6 \text{ } nside^2 \times 1$.

Return type

array

`PB_Phase.beam_mwa(nu, ne1, ne2)`

This is the primary beam function of the MWA Telescope.

$$A(\hat{n}, \nu) = \text{sinc}^2 \left(\frac{\pi b \nu \hat{n} \cdot \hat{e}_1(\alpha_p)}{c} \right) \text{sinc}^2 \left(\frac{\pi b \nu \hat{n} \cdot \hat{e}_2(\alpha_p)}{c} \right)$$

Parameters

- **nu** (*float*) – Observing Frequency ν in MHz unit.
- **ne1** (*float or array*) – The dot product by $\hat{n} \cdot \hat{e}_1$.
- **ne2** (*float or array*) – The dot product by $\hat{n} \cdot \hat{e}_2$.

Returns

The primary beam pattern value for square aperture(MWA).

Return type

float or array

PB_Phase.calculate_phase(*dot_product*, *bl*)

Given the baselines it can calculate the phase factor for for all the pixels which makes angle less than 90 deg by default w.r.t. the pixel in the direction given by *ra_ptg*,*dec_ptg* of the Telescope.

Parameters

- **dot_product** (*array*) – Contains the *xyz* comps for those pixels that makes less than 90 degree angle.
- **bl** (*array*) – The array conatins the components of the baselines.(defined by basis vectors $\hat{e}_1, \hat{e}_2, \hat{e}_3$). Shape $N_{Baselines} \times 3$

Returns

The phase factor . Shape $6 \text{ nside}^2 \times N_{Baselines}$. Excluding the factor $e^{-2\pi i \vec{U} \cdot \hat{p}}$ Which will be multiplied later on.

Return type

array

PB_Phase.dot_cal_superfast(*nside*, *ra_ptg*, *dec_ptg*)

This function can calculate the *xyz* component of \hat{n} for all the pixels which makes angle less than 90 deg by default w.r.t. the pixel in the direction given by *ra_ptg*,*dec_ptg* along the basis vectors $\hat{e}_1, \hat{e}_2, \hat{e}_3$ given by the RA,DEC.

Parameters

- **nside** (*int*) – Healpix parameter.Usually Power of 2.
- **ra_ptg** (*int or float*) – RA of the pointing direction.(In Degrees)
- **dec_ptg** (*int or float*) – DEC of the pointing direction.(In Degrees)

Returns

dot_products – Contains the *xyz* comps for those pixels.Shape $6 \text{ nside}^2 \times 3$.

Return type

array

PB_Phase.hat_n(*nside*, *ra_ptg*, *dec_ptg*)

This can calculate the *xyz* components of \hat{n} for all the pixels which makes angle less than 90 deg by default w.r.t. the pixel in the direction given by *ra_ptg*,*dec_ptg*.

Parameters

- **nside** (*int*) – Healpix parameter.Usually Power of 2.
- **ra_ptg** (*int or float*) – RA of the pointing direction.(In Degrees).
- **dec_ptg** (*int or float*) – DEC of the pointing direction.(In Degrees).

Returns

Containg the *xyz* comps of \hat{n} . $6 \text{ nside}^2 \times 1$

Return type

Array

PB_Phase.needful_pixels(*ra_ptg*, *dec_ptg*, *nside*, *radius=1.5707963267948966*)

This calculates the needful pixels which makes less than radius radian w.r.t the pixel in the direction given by *ra_ptg*,*dec_ptg*.

Parameters

- **ra_ptg** (*int or float*) – RA of the pointing direction.(In Degrees).
- **dec_ptg** (*int or float*) – DEC of the pointing direction.(In Degrees).

- **nside** (*int*) – Healpix parameter. Usually Power of 2.
- **radius** (*int or float*) – The maximum angle subtended by a pixel given the pixel center by *ra_ptg, dec_ptg*. (In Radians) By Default 90 degrees only upper hemisphere.

Returns

pix_mask – The pixels lie within the given disk radius. Shape $6 \text{ } nside^2 \times 1$.

Return type

array

PARAMS_MWA MODULE

This contains useful parameters of MWA.

The Important things are listed here:

- KB - Boltzmann Constant in $Jy\ m^2/mK$
- nside - Healpix parameter.
- Amp - Amplitude of APS
- beta - Power index.
- Nrea - No of Realizations.
- ra_ptg - Pointing direction of telescope.
- b - Dimensions of Square Aperture in m.
- nu_c - Central Frequency.
- dec_mwa - Declination of the telescope.

`Params_MWA.params(tele)`

`Params_MWA.paramsim()`

This Function contains the simulation parameters.

VIS_GEN MODULE

`Vis_Gen.visgen_mwa_multi(nside, in_sky_map, ra_ptg, dec_ptg, bl_file, nu)`

This is the main module which calculates the visibilities for all the baselines given the RA,DEC,Frequency,GRF map and the baseline files.(Also does Multiple Realizations)

Parameters

- **nside** (*int*) – Healpix parameter.Usually Power of 2.
- **in_sky_map** (*array*) – Contains the Sky Brightness Temperature simulated from model APS for all the pixels.(Can contain Multiple Realizations).Shape $N_{Realizations} \times 12 nside^2$.
- **ra_ptg** (*int or float*) – RA of the pointing direction.(In Degrees)
- **dec_ptg** (*int or float*) – DEC of the pointing direction.(In Degrees)
- **bl_file** (*array*) – The array conatins the components of the baselines.(defined by basis vectors $\hat{e}_1, \hat{e}_2, \hat{e}_3$).Shape $N_{Baselines} \times 3$
- **nu** (*float*) – Observing Frequency ν in MHz unit.

Returns

vis – Contains the simulated visibility for all the baselines given by bl_file.(Along axis=0 is the different Realizations,axis=1 is the baselines).Shape $N_{realizations} \times N_{Baselines}$

Return type

Complex array

VIS_CALCULATE MODULE

This is the file which calls the other modules This can simulate the visibilities for a given Baseline file. The Visibility is given by

$$\mathcal{V}(\vec{U}, \nu) = Q_\nu \int_{UH} d\Omega_{\hat{n}} T(\hat{n}, \nu) A(\Delta\hat{n}, \nu) e^{2\pi i \vec{U} \cdot \Delta\hat{n}}$$

In Healpix we discretize the sky and the integral becomes summation. But the integral is restricted to only upper hemisphere. So we find out those pixel indexing and sum only over them.

The discretized version is given as:

$$\mathcal{V}(\alpha_p, \vec{U}, \nu) = Q_\nu \Delta\Omega_{pix} \sum_q T(\hat{n}_q, \nu) A(\Delta\hat{n}_q, \nu) e^{2\pi i \vec{U} \cdot \Delta\hat{n}}$$

where

$$\begin{aligned} Q_\nu &= 2k_B/\lambda^2 \\ \Delta\hat{n} &= \hat{n} - \hat{p} \\ \vec{U} &= u \hat{e}_1(\alpha_p) + v \hat{e}_2(\alpha_p) + w \hat{e}_3(\alpha_p) \end{aligned}$$

Where $\Delta\Omega_{pix}$ refers to the solid angle subtended by each simulation pixel and \hat{p} refers to the pointing direction of the antenna. Here $\hat{p} = \hat{e}_3$.

Overview

- First we generate the **Gaussian Random Field** from given **Angular Power Spectrum**.
- Then We find out Which Pixels are in the Upper Hemisphere given by the pointing direction. Here Pointing Direction is vertically overhead.
- Calculate the **Primary Beam Pattern** for the Telescope.
- Calculate the components of \hat{n} along the basis vectors $\hat{e}_1, \hat{e}_2, \hat{e}_3$. This will help us to determine the dot product $\vec{U} \cdot \hat{n}$.
- Now We calculate the **Phase factor** $e^{2\pi i \vec{U} \cdot \hat{n}}$.
- Multiply the **GRF and PB** first and then multiply by phase factor, sum over the pixels. The whole thing is *done by matrix multiplication*.
- Finally multiply by $e^{-2\pi i \vec{U} \cdot \hat{n}} = e^{-2\pi i w}$, so that we get the correct Phase Factor.
- The Visibility is Simulated.

The **Basis** vectors $\hat{e}_1, \hat{e}_2, \hat{e}_3$ are dependent on the R.A. α and DEC δ and the components of them in the cartesian system xyz is given by:

$$\begin{aligned} \hat{e}_1 &= [-\sin \alpha, \cos \alpha, 0] \\ \hat{e}_2 &= [-\cos \alpha \sin \delta, -\sin \alpha \sin \delta, \cos \delta] \\ \hat{e}_3 &= [\cos \alpha \cos \delta, \sin \alpha \cos \delta, \sin \delta] \end{aligned}$$

INDEX

A

`APS_func()` (*in module GRF_Generate*), 1

B

`beam_mwa()` (*in module PB_Phase*), 2

C

`calculate_phase()` (*in module PB_Phase*), 3

D

`dot_cal_superfast()` (*in module PB_Phase*), 3

G

`GRF_Generate`
module, 1

`GRF_hpmap_gen()` (*in module GRF_Generate*), 1

H

`hat_n()` (*in module PB_Phase*), 3

M

module
 `GRF_Generate`, 1
 `Params_MWA`, 5
 `PB_Phase`, 2
 `Vis_calculate`, 7
 `Vis_Gen`, 6

N

`needful_pixels()` (*in module PB_Phase*), 3

P

`params()` (*in module Params_MWA*), 5

`Params_MWA`
module, 5

`paramsim()` (*in module Params_MWA*), 5

`PB_Phase`
module, 2

`Primary_Beam_generate()` (*in module PB_Phase*), 2

V

`Vis_calculate`
module, 7

`Vis_Gen`
module, 6

`visgen_mwa_multi()` (*in module Vis_Gen*), 6