protomidpy: Reconstruction of Protoplanetary Disk on Mid-plane in Python (Version 0.1.0)

Masataka Aizawa (`aizw.masa@gmail.com`)

Dec 9, 2024

Contents

Chapter 1

Overview

This code, *protomidpy* (Reconstruction of Protoplanetary Disk on Mid-plane in Python), implements an analytical framework for deriving surface brightness profile and geometry of a geometrically-thin axisymmetric disc from interferometric observation of continuum emission, as proposed in Aizawa et al. (2024). A unique feature of this code is that it allows posterior sampling for all parameters, including the brightness distribution $\boldsymbol{a}$, geometric parameters $\boldsymbol{g}$, and hyperparameters for the Gaussian Process $\boldsymbol{\theta}$ (see Fig. 1.1):

$$p(\boldsymbol{a}, \boldsymbol{g}, \boldsymbol{\theta} | \boldsymbol{d}) \tag{1.1}$$

With the precise determination of these parameters, we can discuss faint signals possibly embedded in proto-planetary disks. Additionally, there is no need to tune hyperparameters, making the inference highly objective. This work can be seen as a natural extension of the previous work by Jennings et al. (2020), which solves for $\boldsymbol{a}$ while fixing $(\boldsymbol{g}, \boldsymbol{\theta})$. Their code (*frank* or frankenstein) is available at `https://github.com/discsim/frank`.

The installation and requirements are summarized in Chapter 2. Parameters files, which are prepared in the `paras` folder, are detailed in Chapter 3. After setting the parameters, scripts can be run to realize the posterior distribution, as described in Chapter 4.

The author does not take any responsibility for any problems that the code may cause, so please use it at your own risk. If you have any questions or suggestions, please contact me (aizw.masa@gmail.com).

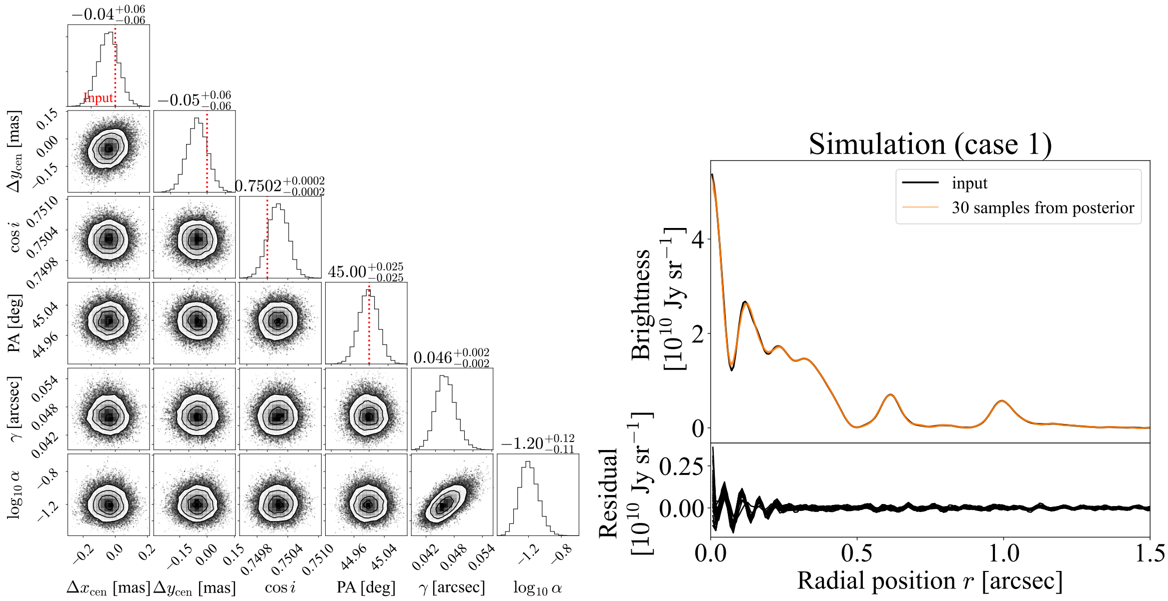

Figure 1.1: Example for posterior sampling for all of disc parameters. The figure is taken from Aizawa et al. (2024) (Fig. 2 in the paper)

Chapter 2

Install

After downloading the codes, you can install it by typing

```
pip install ./protomidpy
```

The following modules are required:

- astropy
- emcee
- corner
- matplotlib
- numpy
- scipy
- pandas
- Jupyter notebook
- tqdm

Chapter 3

Make configuration files

3.1 Format of input data

The data must be in form of ".npz" in numpy, and it needs to contain following items:

- **u_obs**: Spatial frequency "u" [lambda]
- **v_obs**: Spatial frequency "v" [lambda]
- **vis_obs**: Visibility
- **wgt_obs**: Weights

Download test data from `https://github.com/2ndmk2/dsharp_averaged_data` for reference.

3.2 Config files

We have three config files.

3.2.1 MCMC setting (example: `paras/mcmc_config.dat`)

Parameters for emcee and model

- **Nrad** (int)
  Number of radial points for model intensity
- **Nbin** (int)
  Determine grid size for log-binning. Grid size is (2*Nbin+1, 2*Nbin+1).
- **Dpix** (float)
  Radial spacing for model [arcsec].
  Outer disk radius is determined as $Rout = Nrad * Dpix$
- **Nwalker** (int)
  Number of walkers for emcee
- **Nchain (int)**
  Number of chains for emcee
- **Qmin** (float)
  Value determining lower boundary for log gridding.
- **Qmax** (float)
  Value determining upper boundary for log gridding.
- **out_folder** (str)
  Path to output folder

3.2.2 Initialize positions for walkers (example: `paras/AS209_paradic.dat`)

Parameters determining initial positions for mcmc. They are randomly generated with uniform distribution [value-scatter/2, value+scatter/2].

3.2.3 Prior distribution (example: `paras/prior.dat`)

Parameters for determining ranges of priors:

- **Regularization parameter** $\log \alpha$
  Uniform prior, $[\log_{10} \alpha_{\rm min}, \log_{10} \alpha_{\rm max}]$
- **Spatial paramter** γ
  Uniform prior, [min_scale [arcsec], max_scale [arcsec]]
- **Central position for disk**
  Uniform prior, [-delta_pos [arcsec], delta_pos [arcsec]]

Chapter 4

Run scripts

4.1 Run MCMC (example: `tests/run_sampling.py`)

4.1.1 Overview

With *emcee* (Foreman-Mackey et al., 2013), we take samples from posterior distribution for $p(\boldsymbol{g}, \boldsymbol{\theta}|\boldsymbol{d})$ given as:

$$p(\boldsymbol{g}, \boldsymbol{\theta}|\boldsymbol{d}) \quad \propto \quad \mathcal{N}(\boldsymbol{d} \mid \boldsymbol{0}, \bar{\boldsymbol{\Sigma}}_d + \bar{\boldsymbol{H}}\boldsymbol{\Sigma}_a\bar{\boldsymbol{H}}^T)p(\boldsymbol{g}, \boldsymbol{\theta}), \tag{4.1}$$

4.1.2 Example command

```
python run_sampling.py --n_process 4 --visfile ./vis_data/
AS209_continuum_averaged.vis.npz --config ./paras/mcmc_config.dat
--initial_para ./paras/AS209_paradic.dat --prior ./paras/prior.dat
```

The options are given as follows:

- **n_process** (int):
  Number of CPU cores to be used in emcee.
- **visfile** (str):
  Path to visibility file (3.1)
- **config** (str):
  path to mcmc config file (3.2.1).
- **initial para** (str):
  path to mcmc config file (3.2.2).
- **prior** (str):
  path to mcmc config file (3.2.3).

4.1.3 Output

The name for output file is "./result/***_mcmc.npz". The file includes the following item:

- **sample** (numpy.array, [1, Nwalker*Nchain, 6]):
  Sample array with size of taken from posterior distribution.
  The order for the parameters is $(\gamma[\mathrm{arcsec}], \log_{10}\alpha, \cos i, \mathrm{PA}[\mathrm{rad}], \delta_x\mathrm{cen}, \delta_y\mathrm{cen})$.
- **log_prior**:
  Path to visibility file (3.1)
- **log_prior** (numpy.array, [1,Nchain, Nwalker]):
  Prior probability.

- **log_likelihood** (numpy.array, [1,Nchain, Nwalker]):
  Likelihood.

- **nrad** (int):
  Number of radial points for model intensity. Same as Nrad.

- **dpix** (float):
  Radial spacing for model [arcsec]. Same as Dpix.

- **n_bin_log** (int)::
  Determine grid size for log gridding. Same as Nbin.

- **qmin** (float):
  Value determining lower boundary for log gridding.

- **qmax** (float):
  Value determining upper boundary for log gridding.

- **cov** (str): Convariance name (default to "RBF").

4.2 Postprocess Calculation (example: `tests/model_calc.py`)

4.2.1 Overview

Using samples from $p(\boldsymbol{g}, \boldsymbol{\theta} \mid \boldsymbol{d})$, we take sample from full posterior,

$$p(\boldsymbol{a}, \boldsymbol{g}, \boldsymbol{\theta}|\boldsymbol{d}) = p(\boldsymbol{a}|\boldsymbol{d}, \boldsymbol{g}, \boldsymbol{\theta})p(\boldsymbol{g}, \boldsymbol{\theta}|d), \tag{4.2}$$

and model visibilities:

$$\begin{pmatrix} \boldsymbol{V}_{\mathrm{real}} \\ \boldsymbol{V}_{\mathrm{imag}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{C}_{\mathrm{real}}\boldsymbol{H'} \\ \boldsymbol{C}_{\mathrm{imag}}\boldsymbol{H'} \end{pmatrix} \boldsymbol{a}, \tag{4.3}$$

where $\boldsymbol{H'}$, $\boldsymbol{C}_{\mathrm{real}}$, and $\boldsymbol{C}_{\mathrm{imag}}$ are defined as follows:

$$\{\boldsymbol{H'}\}_{j,k} = \frac{4\pi R_{\mathrm{out}}^2}{j_{0(N+1)}^2 J_1^2(j_{0k})} J_0\left(2\pi q_j r_k\right), \tag{4.4}$$

$$\{\boldsymbol{C}_{\mathrm{real}}\}_{j,k} = |\cos i| \cos(-2\pi(\Delta x_{\mathrm{cen}} u_j + \Delta y_{\mathrm{cen}} v_j))\delta_{j,k}, \tag{4.5}$$

$$\{\boldsymbol{C}_{\mathrm{imag}}\}_{j,k} = |\cos i| \sin(-2\pi(\Delta x_{\mathrm{cen}} u_j + \Delta y_{\mathrm{cen}} v_j))\delta_{j,k}. \tag{4.6}$$

4.2.2 Example command

```
python model_calc.py --n_sample_for_rad 20 --n_burnin 20000
--visfile ./vis_data/AS209_continuum_averaged.vis.npz
--mcmc_result_file ./result/AS209_continuum_averaged.vis_mcmc.npz
--initial_para ./paras/AS209_paradic.dat --prior ./paras/prior.dat
--out_file_for_model ./result/AS209_continuum_averagedmodel.npz
```

The options are given as follows:

- **n_sample_for_rad** (int): Number of samples for intensity profiles

- **n_burnin** (int): Number of Burnin samples

- **visfile** (str): Path to visibility file (3.1)

- **mcmc_result_file (str)**: Path to result file from `run_sampling.py` (4.1)

- **out_file_for_model** (str): Path to output file

4.2.3 Output

The name for output file is "./result/***model.npz". The file includes the following item:

- **r_n** (numpy.array, [Nrad]):
  Radial positions [arcsec] for model intensities
- **param_map** (numpy.array, [6]):
  MAP solution for parameters.
- **paras_random_selected** (numpy.array, [n_sample_for_rad, 6]):
  Randomly selected parameters with size of n_sample_for_rad.
- **flux_map_sample** (numpy.array, [200]):
  Model intensity $\boldsymbol{a}^{\dagger}$ [Jy arcsec^{-2}] taken from posterior assuming Maximum a posteriori estimation (MAP) solution for non-linear parameters $\boldsymbol{g}_{\rm MAP}, \boldsymbol{\theta}_{\rm MAP}$, which are taken from the result of MCMC.
- **flux_random_samples** (numpy.array, [n_sample_for_rad, Nrad]):
  Randomly selected model intensities [Jy arcsec^{-2}] with size of n_sample_for_rad.
- **vis_model_undeprojected** (numpy.array, [Nvis]):
  Undeprojected visibility models [Jy] with size of Nvis assuming $(\boldsymbol{a}^{\dagger}, \boldsymbol{g}_{\rm MAP}, \boldsymbol{\theta}_{\rm MAP})$.
- **residual_undeprojected** (numpy.array, [Nvis]):
  Undeprojected residual visibilties [Jy] with size of Nvis assuming $(\boldsymbol{a}^{\dagger}, \boldsymbol{g}_{\rm MAP}, \boldsymbol{\theta}_{\rm MAP})$.
- **qdist_deprojected** (numpy.array, [Nvis]):
  Deprojected spatial frequency distance q [lambda] with size of Nvis assuming $(\boldsymbol{a}^{\dagger}, \boldsymbol{g}_{\rm MAP}, \boldsymbol{\theta}_{\rm MAP})$
- **vis_model_deprojected** (numpy.array, [Nvis]):
  Deprojected visibility models [Jy] with size of Nvis assuming $(\boldsymbol{a}^{\dagger}, \boldsymbol{g}_{\rm MAP}, \boldsymbol{\theta}_{\rm MAP})$.
- **data_deprojected** (numpy.array, [Nvis]):
  Deprojected data [Jy] with size of Nvis assuming $(\boldsymbol{a}^{\dagger}, \boldsymbol{g}_{\rm MAP}, \boldsymbol{\theta}_{\rm MAP})$.
- **data_weights** (numpy.array, [Nvis]):
  Weights [Jy^{-2}] for the deprojected data with size of Nvis assuming $(\boldsymbol{a}^{\dagger}, \boldsymbol{g}_{\rm MAP}, \boldsymbol{\theta}_{\rm MAP})$.

4.3 See result (example: `tests/mcmc_plotter.ipynb`)

Run `mcmc_plotter.ipynb` with Jupyter. We use following variables to specity files to be loaded.

- **samplefile**: Path to file for posterior sampling output from `run_sampling.py`
- **modelfile**: Path to postprocess result from `model_calc.py`

Bibliography

Aizawa, M., Muto, T., & Momose, M. 2024, MNRAS, 532, 1361, doi: 10.1093/mnras/stae1549

Foreman-Mackey, D., Hogg, D. W., Lang, D., & Goodman, J. 2013, PASP, 125, 306, doi: 10.1086/670067

Jennings, J., Booth, R. A., Tazzari, M., Rosotti, G. P., & Clarke, C. J. 2020, MNRAS, 495, 3209, doi: 10.1093/mnras/staa1365