Manual for FARGO2RADMC3D

Clément Baruteau

1 What is FARGO2RADMC3D ?

It is a python program that post-processes the results of Dusty FARGO-ADSG hydrodynamical simulations with the 3D radiative transfer code RADMC3D (version 0.41, Dullemond et al. 2015). Dusty FARGO-ADSG is an extended version of the 2D hydrodynamical code FARGO-ADSG, which solves the gas equations on a polar grid and models dust as Lagrangian test particles. In its current implementation, FARGO2RADMC3D uses the dust's spatial distribution obtained in the simulations as input to compute synthetic maps of continuum emission, and the gas surface density as input to compute images of polarised scattered light. The code works with both the 2.X and 3.X versions of python.

The paragraphs below explain how FARGO2RADMC3D converts the outputs of Dusty FARGO-ADSG simulations into inputs for RADMC3D, and describe the various parameters of the configuration file `params.dat`. Since the code calls RADMC3D for the dust radiative transfer calculations, a basic knowledge of RADMC3D is definitely helpful. The reader is referred to RADMC3D's manual. The file FARGO2RADMC3D.py is heavily commented, so the main steps of the code described below should be easily identified in the program.

2 From Dusty FARGO-ADSG outputs to RADMC3D inputs

This section describes the different steps to convert the results of 2D gas+dust hydrodynamical simulations carried out with Dusty FARGO-ADSG into inputs for the 3D radiative transfer code RADMC3D. This description closely follows § 2.2 of Baruteau et al. (2019).

2.1 Spatial grid

The spatial grid used in RADMC3D is a 3D extension of the simulations grid in spherical coordinates. The structure of the simulations grid (size, number of cells in both the radial and azimuthal directions) is found by reading the simulations outputs. The grid's vertical extent on each side of the midplane is specified in units of the gas pressure scale height via `zmax_over_H` in params.dat, and the number of cells in colatitude by `ncol`. The gas pressure scale height is obtained via the .par parameter file used to run the Dusty FARGO-ADSG simulation, and which is always located in the simulations directory. The name of the simulations directory is set by `dir`. When calculating polarised light images, which requires to set `polarized_scat` to Yes, the cells in colatitude are evenly spaced, otherwise they are logarithmically spaced.

2.2 Dust's surface density

Dust is modelled as Lagrangian test particles in Dusty FARGO-ADSG, so dust drag on the gas is discarded. For this reason, the mass of the dust particles in the simulation has no dynamical role, and only the spatial distribution of the dust particles matters for the dust radiative transfer calculations. This is why the simulations use an arbitrary size distribution between a minimum and a maximum sizes (the number density of super-particles is usually taken inversely proportional to dust size, so that there is approximately the same number of particles per decade of size).

The dust's size distribution $n(a)$ adopted in the radiative transfer calculations is a power-law function of particles size, $n(a) \propto a^{-p}$, between a minimum and a maximum sizes, which are set by `amin` and `amax` in params.dat (both values in metres). The opposite of the power law exponent p is set by `pindex` (it should be positive). The total dust mass is specified via the dust-to-gas mass ratio, which is denoted by `ratio` in params.dat. The mass of the disc gas is that in the hydrodynamical simulation at the desired output number (the output number is set by `on`).

The dust's size range is decomposed into `nbin` logarithmically spaced size bins, and from the spatial distribution of the dust particles in the simulation we compute the dust's surface density for each size bin i, which we denote by $\sigma_{i,\rm dust}$. The quantity $\sigma_{i,\rm dust}$ can be expressed as

$$\sigma_{i,\rm dust}(r,\varphi) = \frac{N_i(r,\varphi)}{\mathcal{A}(r)} \times \frac{M_{i,\rm dust}}{\sum\limits_{r,\varphi} N_i(r,\varphi)}, \tag{1}$$

where N_i denotes the number of dust particles per bin size and in each grid cell of the simulation, $\mathcal{A}$ is the surface area of each grid cell, and $M_{i,\rm dust}$ is the dust mass per bin size, which takes the form

$$M_{i,\rm dust} = \xi M_{\rm gas} \times \frac{a_{i+1}^{4-p} - a_i^{4-p}}{a_{\rm max}^{4-p} - a_{\rm min}^{4-p}} = \xi M_{\rm gas} \times \frac{a_i^{4-p}}{\sum\limits_i a_i^{4-p}}, \tag{2}$$

where [a_i,a_{i+1}] is the size range of the $i^{\rm th}$ size bin, $a_{\rm min}$ and $s_{\rm max}$ are the minimum and maximum particle sizes, $-p$ is the power-law exponent of the dust's size distribution $n(a)$, $M_{\rm gas}$ the total mass of gas in the simulation, and ξ the dust-to-gas mass ratio.

When calculating polarised light images, the spatial distribution of the (small) dust particles is assumed to be the same as that of the gas in the simulation. In this case, the dust's surface density for each size bin simply reads

$$\sigma_{i,\rm dust}(r,\varphi) = \xi \Sigma_{\rm gas}(r,\varphi) \times \frac{a_{i+1}^{4-p} - a_i^{4-p}}{a_{\rm max}^{4-p} - a_{\rm min}^{4-p}} = \xi \Sigma_{\rm gas}(r,\varphi) \times \frac{a_i^{4-p}}{\sum\limits_i a_i^{4-p}}, \tag{3}$$

with $\Sigma_{\rm gas}$ the gas surface density in the simulations grid.

2.3 Dust's mass volume density

For the vertical distribution of the dust's mass volume density, hydrostatic equilibrium is assumed and for each size bin a Gaussian profile is adopted in which the dust's scale height $H_{i,\rm dust}$ of the $i^{\rm th}$ size bin is given by :

$$H_{i,\rm dust} = H \times \left(\frac{\alpha}{\alpha + {\rm St}_i}\right)^{1/2} \text{ if `z_expansion` is set to T,} \tag{4}$$

$$H \text{ if `z_expansion` is set to G,} \tag{5}$$

where H denotes the gas pressure scale height, St$_i$ the average Stokes number of the dust particles in the $i^{\rm th}$ size bin, and α the alpha turbulent viscosity in the hydrodynamical simulation (if a constant kinematic viscosity ν is used instead of an α-viscosity in the simulation, α is simply evaluated from ν). Note that `z_expansion` is automatically set to G if polarised images are computed. The dust's mass volume density is finally re-normalized such that the sum over the 3D grid's volume of the dust's mass volume density times the volume of each grid cell does give us the right total dust mass, equal to $\xi M_{\rm gas}$ with above notations.

The calculation of the dust's surface and volume densities requires to set `recalc_density` to Yes in params.dat. It produces a rather large file dust_density.inp which contains the dust's mass volume density for each size bin at every grid cell.

2.4 Dust opacities

Dust opacities are computed via the python scripts makedustopac.py and bhmie.py, written by C. Dullemond, and based on the original Mie code by Bruce Draine. For continuum emission maps, absorption opacities, scattering opacities for anisotropic scattering and the mean scattering angle are computed as a function of wavelength and stored in dustkappa*.inp files. For polarised light images, the scattering matrix is also computed in the dustkapscatmat*.inp files. The optical constants for various dust compositions can be found in the archive opac.tar.gz (files with a .lnk extension) :

- mix_2species_60silicates_40carbons.lnk : compact dust particles made of 60% astrosilicates and 40% amorphous carbons (internal density is 2.7 g cm^{-3}),
- mix_2species_ice70.lnk : compact dust particles comprised of 70% water ices and 30% astrosilicates (internal density is 1.3 g cm^{-3}),
- mix_2species_porous_ice70.lnk : porous dust particles with an internal density of 0.1 g cm^{-3}. The dust is actually a mixture of a silicate matrix, water ices and a vacuum inclusion. For the mix aggregate to have 70% of its solids in water ices and 30% in silicates, the level of porosity (volume fraction of vacuum) needed to produce grains with a density of 0.1 g cm^{-3} is $\sim$92%. We applied the Bruggeman rules to compute the optical constants of the mix.

The optical constants of water ices are obtained from the Jena database, those of astrosilicates are from Draine & Lee (1984), and those of amorphous carbons are from Li & Greenberg (1997).

The calculation of the dust opacities requires to activate `recalc_opac` in params.dat. The choice of the dust composition is set by `species`, and the directory that contains the optical constant files is specified via `opacity_dir`.

2.5 Call to RADMC3D

Before actually executing the dust radiative transfer, the code writes a number of input parameter files needed by RADMC3D, which requires to specify in params.dat :

- the wavelength at which the continuum or polarised light images are to be computed, via `wavelength` (in millimetres),
- the star's radius (`rstar`, in Solar radii) and effective temperature (`teff`, in Kelvin),
- the disc's inclination (`inclination`), position angle (`posangle`) and phi angle (`phiangle`), with all three angles in degrees (the phi angle corresponds to a rotation in the simulations plane),
- the number of photon packages (`nb_photons`) and that for scattering (`nb_photons_scat`),
- how RADMC3D will treat scattering, via the `scat_mode` parameter. When set to 0, dust scattering is discarded (only thermal absorption is included). When set to 2, anisotropic scattering is included via the Henyey-Greenstein approximation for the scattering phase function. When set to 5, a full treatment of polarisation is adopted using the dust's scattering matrix,
- the number of pixels in the image (`nbpixels`),
- the number of cores to be used by RADMC3D for the radiative transfer calculation (`nbcores`).

RADMC3D is then called in two steps (requires `recalc_radmc` to be set to Yes). The dust temperatures are computed first with a thermal Monte Carlo calculation. The continuum or polarised light images are then computed by ray-tracing.

2.6 Miscellaneous options

Further options are available in the params.dat parameter file, which are briefly listed below :

- `plot_opac` : option to plot the absorption and scattering opacities at the desired wavelength.

- `xaxisflip` : option to add 180 degrees to the disc inclination (can be useful if the disc that you model rotates clockwise in the observations, since the disc will always rotate counter-clockwise in the simulations – this is the case for instance of the MWC 758 disc).
- `secondorder` : use (or not) of second-order integration for ray tracing in RADMC3D.
- `recalc_rawfits` : option to recompute final image if the raw image produced by RADMC3D (image.out) has been calculated on a machine or a computing cluster with no python support (RADMC3D's execution can be done via the script `script_radmc`).
- `check_beam` : option to check that the beam has the expected shape and size by adding a point-like source at the origin of the image grid.
- `plot_temperature` : option to plot the azimuthally-averaged radial profile of the temperature of the largest dust particles at the surface and the midplane.
- `mask_radius` (arcseconds) : option to truncate the dust density within `mask_radius` of the star for polarised intensity images.
- `truncation_radius` (arcseconds) : option to reduce the dust density beyond `truncation_radius` for polarised intensity images (by a power-law function of r which can be changed).

3 Computation of final image

The computation of the final image requires to set `recalc_fluxmap` to Yes. This is automatically done when RADMC3D is executed (via `recalc_radmc`). The disc distance needs to be specified in parsecs via `distance`.

3.1 Continuum emission

When computing a continuum image, RADMC3D calculates the raw flux of emission at the desired wavelength, and FARGO2RADMC3D then convolves the resulting image by an elliptical beam. The beam's major axis (`bmaj`), minor axis (`bmin`) are set in arcseconds in params.dat, and the beam's position angle (`bpaangle`) in degrees. The convolved flux image is returned in units of mJy/beam or μJy/beam. It is saved as .pdf and .fits files.

Synthetic maps of continuum emission can include noise if `add_noise` is set to Yes in params.dat. This is done by adding white noise to the raw maps of continuum emission. More specifically, at each pixel of the raw maps a random number is added that follows a Gaussian probability distribution with zero mean and given standard deviation. The noise's standard deviation is set in Jy/beam by `noise_dev_std`.

Note that the optical depth of continuum emission can be computed instead of the flux. This can be done by setting `plot_tau` to Yes. If `scat_mode` is set to 0, the absorption optical depth will be calculated. If it set to 2, the total (absorption + scattering) optical depth will be calculated.

3.2 Polarised scattered light

When computing a polarised light image, RADMC3D calculates the raw Stokes maps I, Q, U and V, and FARGO2RADMC3D then convolves the Stokes maps Q and U, which represent linear polarised intensities, by an elliptical beam. Like for continuum emission maps, the beam's major axis (`bmaj`), minor axis (`bmin`) need to be set in arcseconds in params.dat, and the beam's position angle (`bpaangle`) in degrees. Next, the convolved Stokes maps are post-processed to obtain the local Stokes Q_ϕ following the procedure described in Avenhaus et al. (2017). Each pixel of the Q_ϕ synthetic image is finally scaled with the square of the deprojected distance from the central star, and normalised such that the intensity of the strongest pixel is 1. The final image is saved as .pdf and .fits files.

Like for continuum emission maps, white noise can be included via the `add_noise` parameter. If so, at each pixel of the Stokes maps Q and U, random numbers are added that have Gaussian probability distributions with zero mean and a standard deviation of 0.4% the maximum value of each map (this value should be changed to reflect the desired rms noise level in the final synthetic map).

3.3 Deprojected images

By default, the synthetic images of continuum emission and/or of polarised light are displayed in the sky plane, and the x- and y-axes show the offset from the stellar position in the right ascension (RA) and declination (Dec) in arcseconds (i.e., north is up and east is to the left). The maximum value for x and y is set by `minmaxaxis` in params.dat, in arcseconds. The synthetic images can also be deprojected on the disc plane, by setting `deproj_polar` to Yes. If so, the image will show orbital radius in arcseconds in $y-$axis, with a maximum value of `minmaxaxis`, and position angle in $x-$axis.

4 Analytical expression for the continuum emission flux

Instead of solving the full radiative transfer equation in 3D with RADMC3D, one can compute the analytical solution to the dust radiative transfer equation in 2D without scattering. This can be done with FARGO2RADMC3D by setting `calc_abs_map` to Yes in params.dat. In each cell of the simulations grid, the specific intensity $\mathcal{I}_\nu$ is calculated as

$$\mathcal{I}_\nu(r,\varphi) = \mathcal{B}_\nu(T_{\rm dust}) \times (1 - e^{-\tau}), \tag{6}$$

with τ the absorption optical depth, $\mathcal{B}_\nu$ the Planck function, and $T_{\rm dust}$ the dust's temperature, which is taken equal to the gas temperature in the simulation if `Tdust_eq_Tgas` is set to Yes. Otherwise, the code will look for the local temperature file "dust_temperature.bdat" computed by RADMC3D and take the midplane temperature of the largest dust particles. Upon discretization of the dust's size range, just like for the calculation of the dust's surface density in Sect. 2.2, τ may be written as

$$\tau(r,\varphi;\lambda) = \sum_i \sigma_{\rm i,dust}(r,\varphi)\kappa_i(\lambda)/\cos(i), \tag{7}$$

where κ_i and $\sigma_{\rm i,dust}$ denote the absorption opacity and the surface density of dust particles in the $i^{\rm th}$ size bin, respectively. The quantity $\sigma_{\rm i,dust}$ is given by Eq. 1, except if `dustdens_eq_gasdens` is set to Yes, in which case it is given by Eq. 3. The quantity κ_i is read from the local dustkappa*.inp files, except if `precalc_opac` is set to Yes, in which case the code will read the pre-calculated opacity files located in the `opacity_dir` directory.

The specific intensity is then projected on the sky plane, by performing first a rotation in the disc plane by `phiangle`, second a rotation along the line of sight by `inclination`, and third a rotation by `posangle` in the image plane. The specific intensity thus obtained is multiplied by the solid angle that subtends each pixel of the image plane. White noise can be added to the image. The resulting raw flux map is finally convolved by an elliptical beam (just like in Sect. 3). The final convolved image is saved as .pdf and .fits files, and can be further deprojected on the disc plane if `deproj_polar` is activated.

5 Contributing authors and acknowledgments

The code FARGO2RADMC3D has been written by Clément Baruteau, Sebastián Pérez and Marcelo Barraza, with substantial contributions from Simon Casassus and Gaylor Wafflard-Fernandez. If you use FARGO2RADMC3D in your publications, please cite Baruteau et al. (2019).

***