radmc3dPy v0.24

Attila Juhász

module: setup

get_model_desc() - Returns the brief description of the model

get_model_names() - Returns the list of available models

problem_setup_dust() - Creats a dust continuum model setup

problem_setup_gas() - Creates a gas model setup

module: analyze

read_data() - Reads variables (e.g. dust density, gas velocity, etc)

read_grid() - Reads the spatial and frequency grid

readopac() - Reads the dust opacities

readparams() - Reads the parameter file (problem_params.inp)

write_default_parfile() - Writes the default parameters for a model

radmc3dData.get_sigmadust() - Calculates the dust surface density in g/cm^2

radmc3dData.get_sigmagas() - Calculates the gas surface density in molecule/cm^2

radmc3dData.get_tau() - Calculates the continuum optical depth

radmc3dData.write_vtk() - Writes variables to a VTK format for visualisation with e.g. Paraview

module: image

makeimage() - Calculates an image with RADMC-3D (both dust continuum and channel maps)

plotimage() - Plot the image / channel map

readimage() - Reads the image

radmc3dImage.imconv() - Convolve the image with a Gaussian beam

radmc3dImage.plot_momentmap() - Plots moment map for a 3d image cube

radmc3dImage.writefits() - Writes the image to a FITS file with CASA compatible header

Dust continuum model

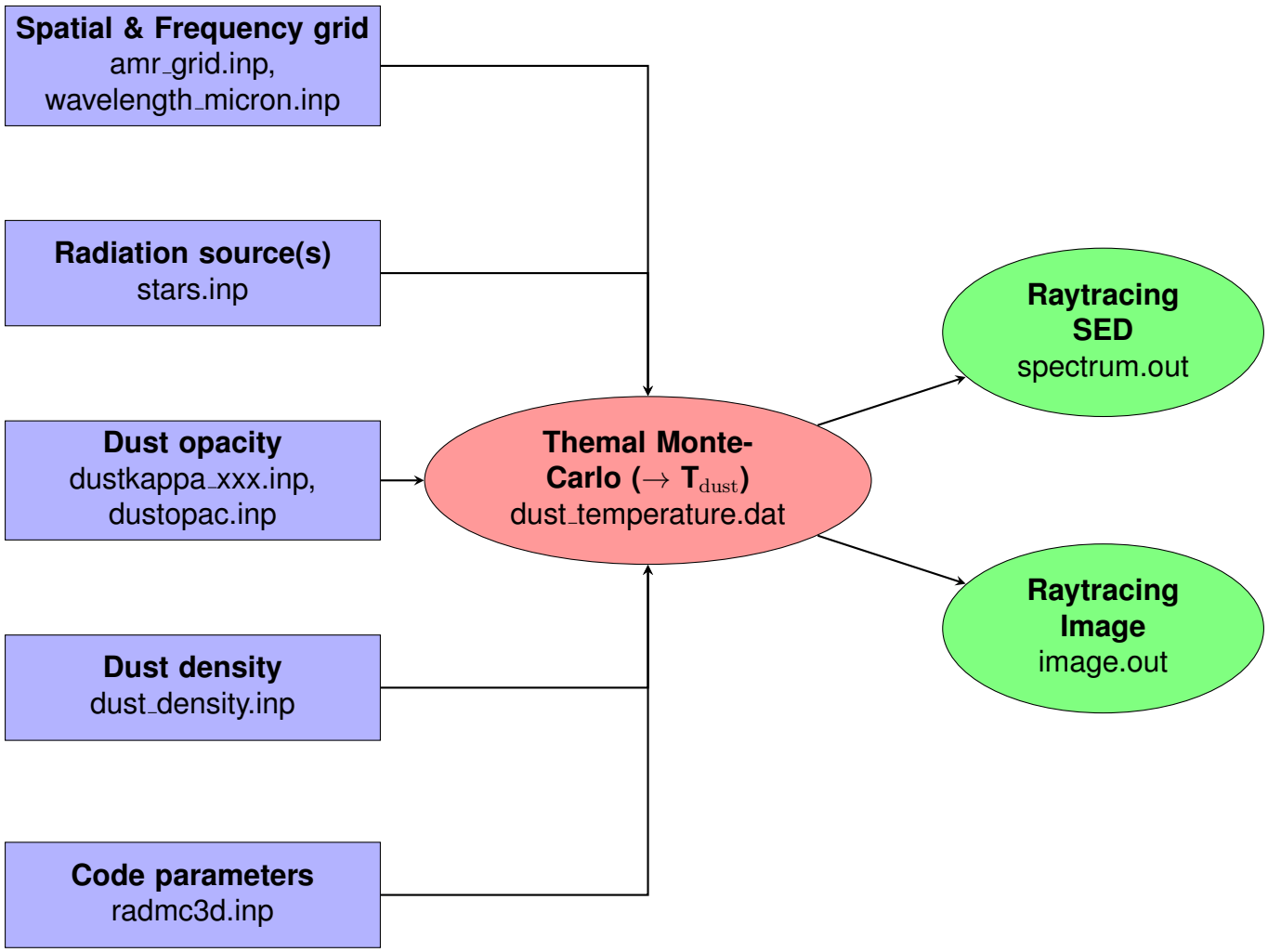


Figure 1: Structure of a dust continuum model. Inputs are marked with blue, intermediate calculation, data products are in red while green marks the output of the simulation.

radmc3dPy commands

First let us create a directory for our model. Then go to this directory and start python.

1 Import `radmc3dPy` .

```
>>> import radmc3dPy
```

2 Check which models are available:

```
>>> radmc3dPy.setup.get_model_names()
['ppdisk']
```

3 Create a parameter file with the default values

```
>>> radmc3dPy.analyze.write_default_parfile('ppdisk')
```

Exit python and open the created 'problem_params.inp' file with a text editor and change the parameters if needed.

4 Set up the model and create all necessary input files.

```
>>>radmc3dPy.setup.problem_setup_dust('ppdisk')
```

5 Then run RADMC-3D from the shell with the Monte-Carlo simulation to calculate the dust temperature.

```
$>radmc3d mctherm
```

Alternatively we can also make a system call from within Python, e.g.:

```
>>>os.system('radmc3d mctherm')
```

6 After the thermal Monte-Carlo run has finished we can make an image from within pyton.

```
$>radmc3dPy.image.makeimage(npix=400, sizeau=200, wav=880., incl=45, posang=43.)
```

7 After RADMC-3D finished we can read the image and plot it.

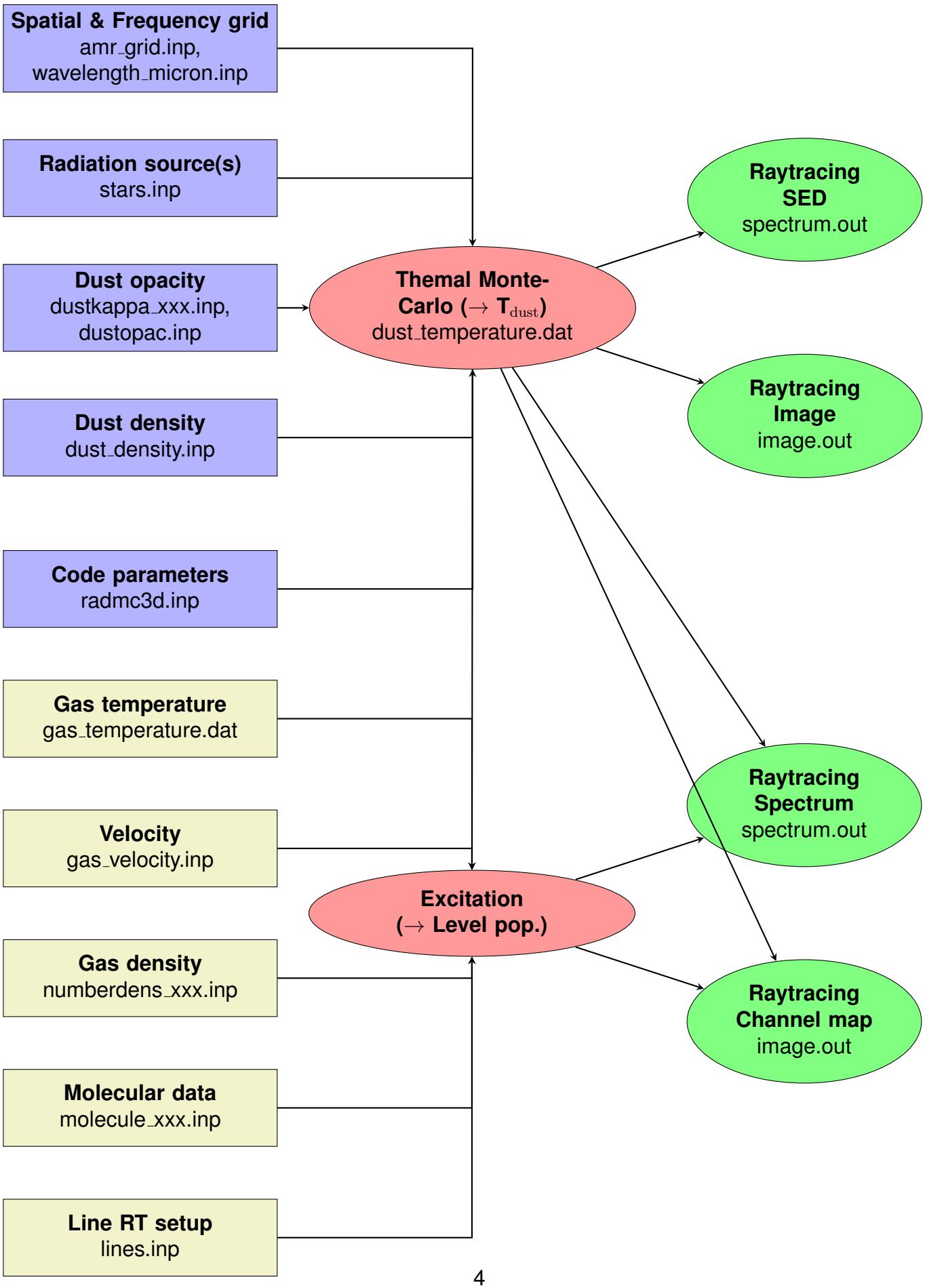
```
>>>imag=radmc3dPy.image.readimage()
>>>radmc3dPy.image.plotimage(imag)
```

8 We can also convolve the image with an arbitrary elliptical gaussian Gaussian beam

```
>>>conv_imag = imag.imconv(fwhm=[0.05, 0.1], pa=40., dpc=140.)
```

The fwhm of the Gaussian beam should be in arcsec, the positing angle of the major axis of the beam ellipse should be in degrees, and the distance to the source in pc (dpc keyword) should be given.

Gas model


Spatial & Frequency grid
amr_grid.inp,
wavelength_micron.inp
Radiation source(s)
stars.inp
Dust opacity
dustkappa_xxx.inp,
dustopac.inp
Dust density
dust_density.inp
Code parameters
radmc3d.inp
Gas temperature
gas_temperature.dat
Velocity
gas_velocity.inp
Gas density
numberdens_xxx.inp
Molecular data
molecule_xxx.inp
Line RT setup
lines.inp
Themal Monte-Carlo (→ T$_{\rm dust}$)
dust_temperature.dat
Excitation
(→ Level pop.)
Raytracing SED
spectrum.out
Raytracing Image
image.out
Raytracing Spectrum
spectrum.out
Raytracing Channel map
image.out

radmc3dPy commands

1 Import `radmc3dPy` .

```
>>> import radmc3dPy
```

2 Check which models are available:

```
>>> radmc3dPy.setup.get_model_names()
['ppdisk']
```

3 Create a parameter file with the default values

```
>>> radmc3dPy.analyze.write_default_parfile('ppdisk')
```

4 Exit python and open the created 'problem_params.inp' file with a text editor and change the parameters if needed. Create all necessary imput files.

```
>>>radmc3dPy.setup.problem_setup_gas('ppdisk')
```

5 This time we can skip the thermal Monte-Carlo simulation and go directly to raytracing, calculating images and spectra.

```
$>radmc3d image npix 400 sizeau 200 incl 45.  phi 0.  posang 43.  iline 3 vkms 1.0
```

6 This command calculates a single channel map at

```
>>>imag=radmc3dPy.image.readimage()
>>>radmc3dPy.image.plotimage()
```