POLARIS Quickstart Guide

Download

Download zip package from the homepage or clone the github repository via:

```
git clone https://github.com/polaris-MCRT/POLARIS.git
```

HINT: It is recommended to clone the git repository into the home directory. If downloaded from the homepage, extract the zip file into the home directory via:

```
unzip -q POLARIS-master-basic.zip -d ~/
```

Requirements

The following packages are required for the installation:

- gcc (preferred), icc, or clang++
- cmake (preferred), or ninja
- python3 (packages: *numpy*, *setuptools*)

Installation (Linux)

Open a terminal/console and move into the POLARIS directory:

```
cd /YOUR/POLARIS/PATH/
```

Run the installation script:

```
./compile.sh -f
```

For the first installation, the option `-f` is required to install the CCfits and cfitsio libraries. Alternatively, these libraries can be installed with a package manager (root permissions are required):

```
sudo apt update
sudo apt install libccfits-dev libcfitsio-dev
```

If these packages are installed on the system, simply install POLARIS via

```
./compile.sh
```

For more information, type:

```
./compile.sh -h
```

POLARIS can now be executed from any newly opened terminal/console. However, to use it in already open terminals/consoles, execute the following command to update the environmental paths:

```
source ~/.bashrc
```

HINT: Please refer to the manual (Sect. 1.2) for installation on **macOS**. An installer to use POLARIS with Windows is not available yet.

Start a simulation

POLARIS simulations are performed by parsing a command file with the simulation parameters. Exemplary `.cmd` command files for temperature, thermal emission, and scattered stellar emission simulations can be found in

- `projects/disk/example/temp/`,
- `projects/disk/example/dust/`, and
- `projects/disk/example/dust_mc/`, respectively.

Parameters of the model such as density distribution or magnetic field direction are stored in a separate grid file (for detailed information, see the manual, Sect. 2.3). The simulations use an exemplary (binary) grid file `grid.dat` of a circumstellar disk which can be found in `projects/disk/`.

To start the temperature simulation (`temp`), move into the POLARIS directory and execute `polaris` followed by the command file:

```
cd /YOUR/POLARIS/PATH/
polaris projects/disk/example/temp/POLARIS.cmd
```

The results are stored at `projects/disk/example/temp/data/` as `.fits.gz` files. These files can be opened with, for example, SAOImageDS9, or a python script using astropy.

Simulations are performed similarly for thermal emission (`dust`) and stellar scattered radiation (`dust_mc`). Please refer to the command list in the `projects` folder or the manual (Table 2.4 - 2.10) for available options of the command file.

HINT: For thermal emission simulations, a temperature simulation has to be performed first.

HINT: The previous results will be overwritten, if the same command file is used. Please change <path_out> in the command file to use a new directory for the new results.

HINT: If users write their own command file, before starting the simulation, please check <dust_component>, <path_grid>, and <path_out> in the command file for the correct (absolute) paths.

Create a grid

POLARIS includes PolarisTools, a Python package to create custom grid files for POLARIS. The (binary) grid file can be created with the command `polaris-gen`.

Predefined models

There are already two models available:

Circumstellar disk with a power-law density distribution

$$\rho(r, z) = \rho_0 \left(\frac{r}{r_0}\right)^{-\alpha} \times \exp\left[-\frac{1}{2}\left(\frac{z}{h(r)}\right)^2\right]$$

$$h(r) = h_0 \left(\frac{r}{r_0}\right)^{\beta}$$

Default values: $r_0 = 100$ au, $h_0 = 10$ au, $\beta = 1.1$, $\alpha = 3(\beta - 0.5)$, inner disk radius $r_{\text{in}} = 0.1$ au, outer disk radius $r_{\text{out}} = 100$ au, and total gas mass $M_{\text{gas}} = 10^{-3}$ $M_\odot$ with a dust to gas mass ratio of 0.01.

Sphere with a constant density distribution

$$\rho(r) = \rho_0$$

Default values: inner radius $r_{\mathrm{in}} = 0.1$ au, outer radius $r_{\mathrm{out}} = 100$ au, and total gas mass $M_{\mathrm{gas}} = 10^{-4}$ $\mathrm{M}_\odot$ with a dust to gas mass ratio of 0.01. In addition, the sphere model has a magnetic field with a toroidal geometry and a strength of 10^{-10} T.

By default, the density distribution is normalized to the given total mass, so the value of ρ_0 is calculated accordingly.

To create a grid file, use

```
polaris-gen model_name grid_filename.dat
```

where `model_name` is either `disk`, or `sphere`. The (binary) grid file will be stored at `projects/model_name/`. To modify specific parameters of the model, for instance a total gas mass of 10^{-5} $\mathrm{M}_\odot$ and an inner radius of 1 au, type:

```
polaris-gen model_name grid_filename.dat --gas_mass 1e-5M_sun --inner_radius 1AU
```

For more information, type:

```
polaris-gen -h
```

Extra parameter

To modify further model specific parameter values, the user can parse a list of parameter values using the option `--extra` followed by the keywords and the corresponding value (int, float, or str). By default, the user can parse

- 5 values for the `disk` model: reference radius `ref_radius` (r_0 in meter), reference scale height `ref_scale_height` (h_0 in meter), `alpha` (α), `beta` (β), and `tapered_gamma` (γ),
- 2 values for the `sphere` model: the geometry of the magnetic field `mag_field_geometry` (toroidal, vertical, or radial) and the magnetic field strength `mag_field_strength` (in tesla).

For example, the disk density profile can be modified to $r_0 = 50$ au and $\beta = 1.25$ with

```
polaris-gen disk grid_filename.dat --extra ref_radius 50*149597870700 beta 1.25
```

or the magnetic field of the sphere to a radial geometry with

```
polaris-gen sphere grid_filename.dat --extra mag_field_geometry radial
```

By adding `tapered_gamma` (γ) to the extra parameter of the disk model, the density distribution has an additional exponential taper at large radii (e.g. Andrews et al. 2009)

$$\rho(r,z) = \rho_0 \left(\frac{r}{r_0}\right)^{-\alpha} \exp\left[-\left(\frac{r}{r_0}\right)^{2-\gamma}\right] \times \exp\left[-\frac{1}{2}\left(\frac{z}{h(r)}\right)^2\right]$$

Additional parameter values to modify the model can be defined in the function `update_parameter` in the file `tools/polaris_tools_modules/model.py`.

Hint: For any changes in the files, the user has to recompile with:

```
./compile.sh -u
```

Custom model

For a more complex model modification, it is recommended that users define their own models in `tools/polaris_tools_custom/model.py`. Therein, each model is defined as a class with a corresponding entry in the dictionary at the top of `model.py`. Similar, to create a grid file for a custom model, use

```
polaris-gen model_name grid_filename.dat
```

where `model_name` is the name of the model in the dictionary of `model.py`.

Hint: For any changes in the files, the user has to recompile with:

```
./compile.sh -u
```

Convert a grid file

Users can also write and edit their own grid file. For this purpose, the command `polaris-gen` has an ascii to binary converter (and vice versa) for converting grid files. To convert an existing ascii grid file to a binary grid file, use

```
polaris-gen model_name grid_filename.txt --convert ascii2binary
```

To convert an existing binary grid file to an ascii grid file, use

```
polaris-gen model_name grid_filename.dat --convert binary2ascii
```

The input grid file has to be located in `projects/model_name/` and the new output grid file will be stored at `projects/model_name/`. For the general structure and available options in the grid file, please read the manual (Sect. 2.3).