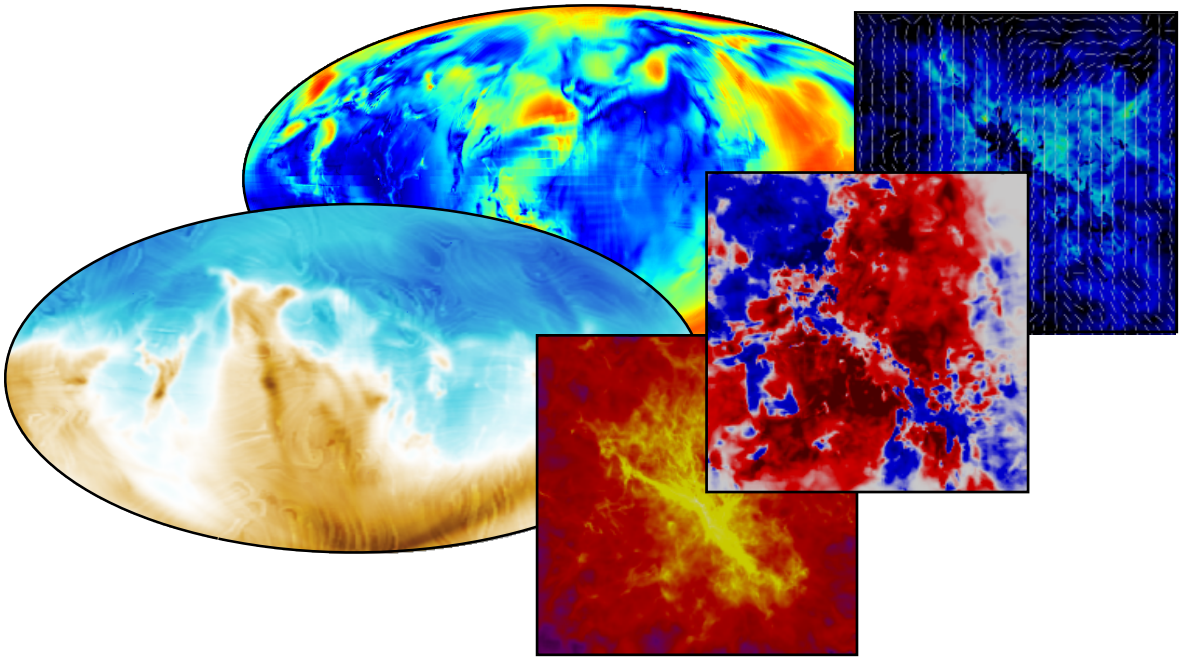
POLARIS

(Polarized Radiation Simulator)

User manual for POLARIS v4.12.04

last revised: August 15, 2024



Stefan Reissl[1,2] and Robert Brauer[2]

[1]Heidelberg University, Institute for Theoretical Physics,
Albert-Überle-Str. 2, 69120 Heidelberg, Germany
reissl@uni-heidelberg.de

[2]University of Kiel, Institute of Theoretical Physics and Astrophysics,
Leibnizstraße 15, 24118 Kiel, Germany
polaris@astrophysik.uni-kiel.de

Front page images

Oval panels: All-sky maps of intensity and degree of linear polarization of a post-processed SILCC simulation provided by Philipp Girichidis.
Squared panels: Degree of linear and circular polarization as well as line of sight magnetic field strength of a post-processed MHD collapse simulation provided by Daniel Seifried.

Figure: Institutes that were or are still involved in the development of POLARIS. From left to right: Institute of Theoretical Physics and Astrophysics at the CAU Kiel (ITAP), the Center for Astronomy at the University Heidelberg (ZAH), and the Institute of research into the fundamental laws of the Universe at CEA Saclay (IRFU).

Introduction

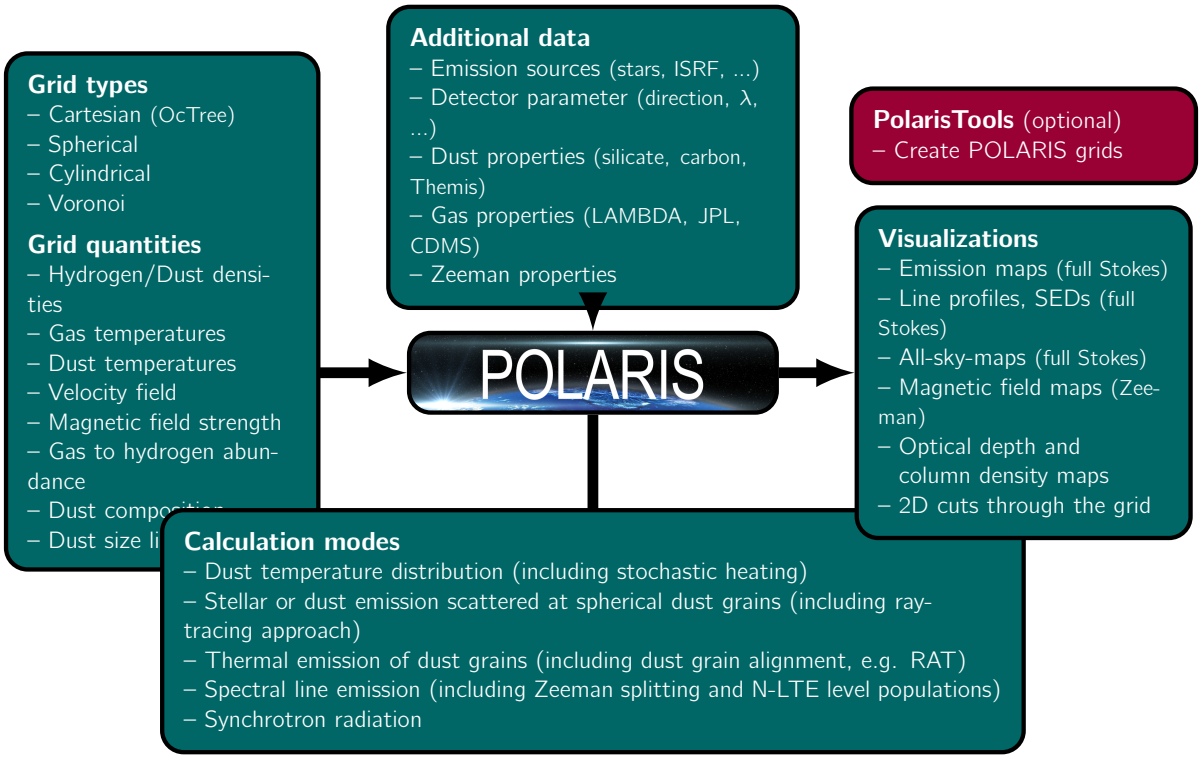
Table 1: *History of POLARIS and related publications.*

Apr. 1999	*Development*: MC3D in version 1 (basis for POLARIS, Wolf et al. 1999)
Feb. 2003	*Development*: MC3D in version 2 (basis for POLARIS, Wolf 2003)
2010	*Development*: MC3D in version 4 (basis for POLARIS)
2014	*Development*: Start of POLARIS development
June 2014	*Publication*: Reissl et al. (2014)
July 2015	*Development*: Mol3D (basis for line RT in POLARIS, Ober et al. 2015)
Apr. 2016	*Publication*: Brauer et al. (2016)
Sept. 2016	*Publication*: Reissl et al. (2016)
2017	*Development*: Final merge of MC3D and Mol3D into POLARIS
Sept. 2017	First POLARIS workshop in Heidelberg
May 2017	*Publication*: Brauer et al. (2017b)
July 2017	*Publication*: Reissl et al. (2017)
Nov. 2017	*Publication*: Brauer et al. (2017a)
May 2018	*Publication*: Reissl et al. (2018a)
July 2018	*ASCL Code Record*: First public release of POLARIS (Reissl et al. 2018b)
Jan. 2019	*Publication*: Seifried et al. (2019)
...	...
July 2020	POLARIS is hosted on GitHub and actively maintained and developed
...	...

Notes: *For a complete and up-to-date list of publications, see NASA/ADS.*
Legend: - *Cite one of these when using POLARIS in general*
- *Additionally, cite this when using line transfer / Zeeman splitting*

POLARIS is a three-dimensional (3D) Monte-Carlo (MC) continuum and line radiative transfer (RT) code. The aim of POLARIS is to provide a tool to investigate the observability of characteristic physical quantities of analytical astrophysical models as well as complex magneto-hydrodynamic (MHD) simulations. Hence, POLARIS is capable of simulating the direct and scattered thermal emission of dust grains, the direct and scattered emission of stars, the emission of spectral lines of various gas species, and synchrotron radiation of cosmic ray electrons and thermal electrons. Resulting from these simulations, POLARIS provides synthetic intensity and polarization maps, spectral energy distributions (SED) and line spectra. One of the key features of POLARIS is the consideration of the magnetic field. Via dust grain alignment and Zeeman splitting, POLARIS is capable of providing predictions for investigations of magnetic fields based on the dust and the gas phase. To achieve this, the code makes use of a full set of physical quantities as input to simulate

Figure 1: Illustration how POLARIS works and which kind of simulations can be performed.



synthetic observational data (density, temperature, velocity, magnetic field, dust grain properties, spectroscopy databases, and different sources of radiation). This combination of features makes POLARIS a unique radiative transfer code with various applications not only related to magnetic fields. So far, POLARIS consists of the work of three PhD students (Stefan Reissl, Robert Brauer, Florian Ober) which were supervised by Sebastian Wolf at Kiel university. After their PhD, Stefan Reissl and Robert Brauer are still developing, improving, and using POLARIS in their positions as postdoctoral researchers. In Table 1, an overview of the history of POLARIS is shown. In addition, the table includes papers whose results were fully or partially obtained by POLARIS. Most of these studies needed the unique capabilities of POLARIS to obtain their results.

Special thanks

Without the generous help of many people that give comments or find bugs, POLARIS would not be in its great shape. Therefore, Stefan and Robert want to say thank you to the following people:

- Robi Banerjee
- Robert Brunngräber
- Vincent Guillet
- Ralf Klessen
- Bastian Körtgen
- Florian Ober
- Eric Pellegrini
- Daniel Seifried
- Valeska Validivia
- Steffi Walch
- Sebastian Wolf

Thank You!

Copyright

Contents

List of Figures

List of Tables

Listings

1 Installation

1.1 Download

POLARIS can be downloaded from the homepage or cloned from the GitHub repository via:

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

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:

- GNU Compiler Collection (`gcc`) with `OpenMP` support, `icc`, or `clang++`
- `cmake`, or `ninja`
- `Python` version >= 3.6 (packages: *numpy*, *setuptools*)

(MacOS user only) It is recommended to use Homebrew (a package manager for macOS) to install the required packages (see below).
(Linux server/cluster user only) Some Linux servers have not a recent version of `gcc` with `OpenMP` support installed. However, most server/cluster systems offer the use of environment `modules` to update to a recent version of `gcc` or `OpenMP`.[1]

1.2 Installation

Linux

To install POLARIS on your computer, open a terminal/console and change to the POLARIS directory:

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

Subsequently, 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
```

[1] See `https://modules.sourceforge.net/` for more information about the usage.

For more information, type:

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

POLARIS can now be executed from any newly opened terminal/console by using the command `polaris` followed by a command file (see Sect. 2.1). To use it in already open terminals/consoles, execute the following command to update the environmental paths:

```
source ~/.bashrc
```

For creating custom grids, PolarisTools provides the command `polaris-gen` that is also installed when using the compile script. Use `polaris-gen -h` for more information or see the quickstart guide.

Alternatively, the user can build POLARIS by doing the following steps:

```
sudo apt update -yy && sudo apt install -yy ninja-build libccfits-dev libcfitsio-dev
mkdir -p build
cd build
CC=gcc CXX=g++ cmake ../src -DCMAKE_BUILD_TYPE=Release -GNinja
ninja && ninja test
python3 ../tools/setup.py install --user &>/dev/null
```

The POLARIS executable can now be found in the build directory. POLARIS can be executed using the command `./polaris` followed by a command file (see Sect. 2.1). For creating custom grids, `polaris-gen` can be found in `tools/scripts`.

MacOS

To install POLARIS on your computer, open a terminal/console and change to the POLARIS directory:

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

Then do the following steps to build POLARIS:

```
brew install llvm libomp ninja ccfits cfitsio
mkdir -p build
cd build
CC="$(brew --prefix llvm)/bin/clang" CXX="$(brew --prefix llvm)/bin/clang++" cmake ../
    ↪ src -DCMAKE_BUILD_TYPE=Release -GNinja
ninja && ninja test
python3 ../tools/setup.py install --user &>/dev/null
```

The POLARIS executable can now be found in the build directory. POLARIS can be executed using the command `./polaris` followed by a command file (see Sect. 2.1). For creating custom grids, `polaris-gen` can be found in `tools/scripts`.

Windows

An installer to use POLARIS with Windows is not available yet.

2 Input files

2.1 The command file

A simulation with POLARIS can be executed by providing the path to a command file as a single argument:

```
polaris PATH/TO/THE/command_file
```

Predefined commands in a pseudo XML style are implemented in the code and allow the user to create a script with sequences of simulations. The structure of the command file is intended to be simple and suggestive. A complete list of commands is provided in Tables 2.4 to 2.10 at the end of this section.

The POLARIS parser does not distinguish between tab and whitespace nor does it care about the number of tabs and whitespaces between each command. Lines marked with a # or a ! are ignored and can be used as comments. The command file consists of an arbitrary number of tasks. Each task block has the following form:

```
<task>  0/1
# commands ...
</task>
```

The number 1 is optional. Task blocks with a 0 will be skipped completely. Commands that are common to all tasks can be defined in an extra block in the beginning as the first block of the command file with:

```
<common>  0/1
# commands ...
</common>
```

When identical commands appear in both the common-block and in a task-block the command written in the task-block has priority. The order of commands inside each task block is mostly irrelevant. Exceptions are gas species (see Sect. 3.8.2), background sources (see Sect. 3.2), and detectors (see Sect. 3.5) which are numbered in the order of theirs appearance. Each task-block needs a command that defines the pipeline ID (simulation type). A detailed description of each simulation type is provided in Sect. 3.

A command file example to perform the calculation of the dust temperature can be found in Listing 2.1. In this example, the command `CMD_TEMP` runs a simulation for heating the dust by considering different photon emitting sources (see Sect. 3.2). In this case, a single star and the interstellar radiation field (ISRF) are considered as sources. The ISRF source requires an external file for the spectral energy distribution (SED). In contrast to dust polarization and emission, the LRT (see Sect. 3.8) and the synchrotron runs do not require a dust model (see Sect. 3.9).

A dust model consists of a arbitrary number of grain material, mass fraction, size distribution, and size ranges. In the above example the dust model is defined in the common block. In this particular case a dust grain model mixture of 62.5 % silicate and 37.5 % graphite is used with a size distribution of $N_{\mathrm{d}}(a) \propto a^{-3.5}$. The values 0.005×10^{-6} and 0.25×10^{-6} are the minimal and maximal dust grain radii in meters. The available grain size range can be found in the dust parameters file (see Sect. 2.4). With `<path_grid>`, the path to the input grid is defined. Each simulation of a POLARIS pipeline requires a grid with a set of physical input parameters in a

Table 2.1: List of commands that changed compared to previous versions.

previously	current version
CMD_RAYTRACING	CMD_DUST_EMISSION
CMD_MCPOL	CMD_DUST_SCATTERING
CMD_LINETRANSFER	CMD_LINE_EMISSION
<plot_inp_midplanes>	no longer exists
<plot_out_midplanes>	no longer exists
all the detector commands	see Sect. 3.5
<gas_species vel_channels = >	max_vel and vel_channels moved to detector
wavelengths & grain sizes	moved from indices to physical values in SI

predefined geometry. The units of the grid quantities need to be in SI units or in cgs units, if <path_grid_cgs> is used. The creation of such a grid is described in Sect. 2.3 in greater detail. With <path_out>, the path to the resulting output files is defined.

POLARIS can plot the input and output 3D distributions of the physical parameters of a particular grid as gnuplot files. The next optional lines define the number of data points and vectors as well as the step for the grid lines to be plotted. Additionally, the values of the xy-, xz-, and yz- midplane files can be written as fits files in a regular raster.

Hence, the command <write_out_midplanes> defines the resolution of the mid-plane files with predefined number of bins (e.g. 256). In POLARIS the problem of radiative transfer is solved in a parallelized way using the OpenMP library. The number of processors can be defined with the command <nr_threads>. POLARIS works internally strictly in SI unity for all input parameters. Physical quantities predefined in the grid need possibly to be converted (see Sect. 2.3.6 for details). With <conv_dens> 0.01, the grid density is converted from cgs in SI. Finally, the dust-to-gas mass ratio is defined by <mass_fraction> If the <mass_fraction> is set to zero, the fractions of the dust components are used instead and do not have to be summed up to one (see Sect. 3.6).

This sample script is just a general overview. A detailed description of the manifold features of POLARIS and the corresponding commands to control them is provided in the following sections.

HINTS:

- No extra directory needs to be created in advance. POLARIS creates automatically all the necessary directories defined by the output path.
- All paths have to be written in quotation marks.

2.2 Changed feature with respect to older versions

The version 4.00 is the first public release of the POLARIS code. In order to keep consistency we renamed some commands. However, people used previous versions for their projects and publications (versions less than 4.00). Hence, we give a short list of changed commands in Table 2.1.

2.3 Grid files

In POLARIS RT simulations can be performed with four different grid geometries so far. The available grid types are spherical, cylindrical, octree, and Voronoi. All the grids have in common

Listing 2.1: Example command file.

```
<common>
#parameters of the optical properties of the dust
<dust_component> "PATH/TO/POLARIS/input/dust/silicate_oblate.dat" 0.625 -3.5 0.005e-6
    ↪ 0.25e-6
<dust_component> "PATH/TO/POLARIS/input/dust/graphite_oblate.dat" 0.375 -3.5 0.005e-6
    ↪ 0.25e-6
</common>

<task> 1
#pipeline ID for dust heating
<cmd> CMD_TEMP

#A star as radiation source
<source_star nr_photons = "1e6"> 1.15e2 4.16e2 1.8e1 7 9e3

#ISRF as radiation source
<source_isrf nr_photons = "500000"> "PATH/TO/POLARIS/input/interstellar_radiation_
    ↪ field.dat"

#path of the input grid file
<path_grid> "PATH/TO/YOUR/grid.dat"

#parameters files for all output data
<path_out> "PATH/WHERE/THE/RESULTS/SHOULD/BE/SAVED/"

#Nr. of gnuplot points and maximal grid level to be plotted
<nr_plot_points> 4000 # optional
<nr_plot_vectors> 4000 # optional

<max_plot_lines> 3 # optional

#Nr. of bins of the input and output mid-plane plots
<write_out_midplanes> 256 # optional

#Nr. of threads used for parallel computing
<nr_threads> 8

#conversion factor from cgs in SI units
<conv_dens> 0.01 # optional

#dust to gas ratio
<mass_fraction> 0.01 # optional
</task>
```

Table 2.2: IDs for the implemented POLARIS grid types.

type	octree	spherical	cylindrical	voronoi
ID_{grid}	20	30	40	50

that they start with a standardized header followed by a grid specific data section. All quantities are stored in a **binary** format.

The grid type itself is characterized by ID_{grid} (unsigned short, 2 bytes) which is the very first number in each grid file is shown in Table 2.2. After the grid ID_{grid} follows a number N_{phys} (unsigned short, 2 bytes) that defines the amount of physical quantities for each grid cell and a list of IDs (unsigned short, 2 bytes) to identify each quantity (see Table 2.3). With exception to n_g/ρ_g, n_d/ρ_d, and T_d, all identifiers can only appear once in the header. POLARIS RT simulations require at least a gas density n_g/ρ_g as physical quantity in each grid cell. All other quantities are optional and depend on the kind of chosen simulation pipeline (see Sect. 3) Instead of number densities n, mass densities ρ of the gas and dust can be defined as well. However, a grid cannot be used, if densities of both units are defined.

2.3.1 Multiple dust compositions

There are two different methods to define the distribution of multiple dust compositions inside the grid. Either the dust mixture index ID_{dust} is set for each grid cell or multiple dust density distributions (or gas densities via gas-to-dust mass ratio) are defined. With the first method, the radiative transfer is performed by using in each cell the dust composition related to the set ID_{dust} (see 3.6 for defining multiple dust compositions). This is the suggested method for grids based on MHD/HD simulations, since they are usually only supporting one density distribution and adding a value to each cell is easily realized. A more sophisticated approach is the usage of multiple density distributions (n_d/ρ_d or n_g/ρ_g). For this method, the user needs to define as much dust compositions as there are density distributions. With this, the radiative transfer is performed by using in each cell a mixture of the dust compositions that depends on the ratio between all density distributions. This method is well suited for grids based on analytical models since multiple density distributions can easily be created analytically.

HINTS:

- If you want a specific cell to be skipped in an RT simulation set its density to zero.
- POLARIS shifts the center of the grid to be in the center of the external coordinate system. The coordinates of the radiating sources (see Sect. 3.2) have possibly to be adapted accordingly!

2.3.2 Spherical grid

After defining the header, the spherical grid can be defined by an additional sequence of 8 numbers R_{min}, R_{max}, N_r, N_φ, N_ϑ, f_r, f_φ, f_ϑ. Here, the first two, R_{min} and R_{max}, respectively, define the inner and outer radius of the sphere followed by the number of cells in r-direction N_r, φ-direction N_φ, and ϑ-direction N_ϑ, respectively. The boundary of the grid is a sphere with a radius of R_{max}. The shape parameter f_r, f_φ, and f_ϑ determine the distribution of steps along the r-direction, φ-direction, and the ϑ-direction with:

Radial direction

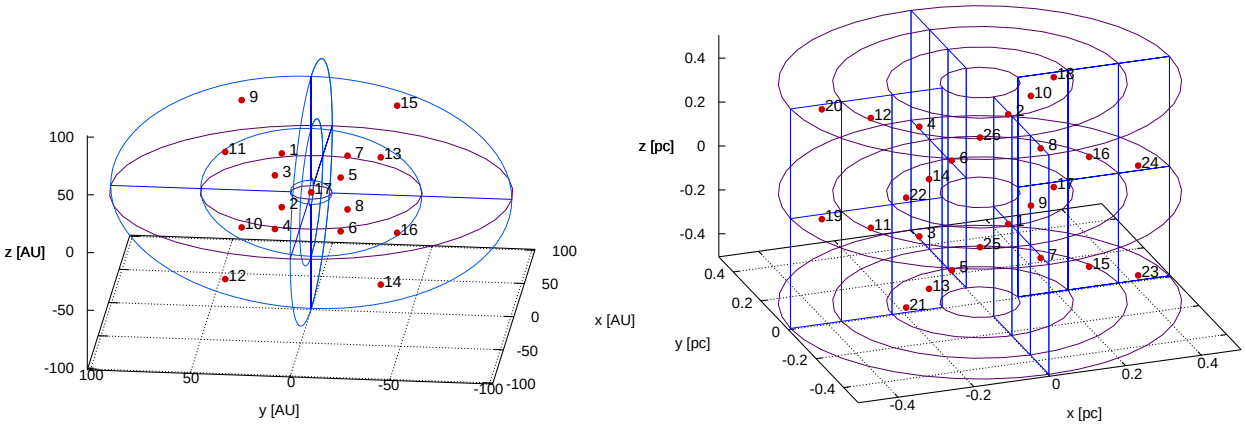


Figure 2.1: Exemplary representations of the POLARIS supported spherical grid (left) and cylindrical grid (right). The numbers represent the order of cells characteristic for each type of grid.

- $f_r = 0$
  The header has to be followed by $N_r - 1$ values (double, 8 bytes) that define the location of each i-th radial cell border with $r_2 < \cdots < r_i < \cdots < r_{N_r}$ whereas $r_1 = R_{\text{min}}$ and $r_{N_r+1} = R_{\text{max}}$, respectively.
- $f_r < 0$
  The cell borders along the r-direction are equally distributed to create N_r cells.
- $f_r = 1$
  The i-th cell border in the r-direction is calculated according to:

$$r_i = R_{\text{min}} + (R_{\text{max}} - R_{\text{min}}) \sin\left(\frac{i\pi}{N_r}\right). \tag{2.1}$$

- $f_r > 1$
  The i-th cell border in the r-direction is calculated according to:

$$r_i = R_{\text{min}} + \frac{(f_r^i - 1)(R_{\text{max}} - R_{\text{min}})}{f_r^{N_r} - 1}. \tag{2.2}$$

Phi direction

- $f_\varphi = 0$
  The header has to be followed by $N_\varphi - 1$ values (double, 8 bytes) that define the location of each i-th phi cell border with $\varphi_2 < \cdots < \varphi_i < \cdots < \varphi_{N_\varphi}$ whereas $\varphi_1 = 0$ and $\varphi_{N_\varphi+1} = 2\pi$, respectively.
- $f_\varphi \neq 0$
  The cell borders along the φ-direction are equally distributed to create N_φ cells.

Theta direction

- $f_\vartheta = 0$
  The header has to be followed by $N_\vartheta - 1$ values (double, 8 bytes) that define the location of each i-th theta cell border with $\vartheta_2 < \cdots < \vartheta_i < \cdots < \vartheta_{N_\vartheta}$ whereas $\vartheta_1 = 0$ and $\vartheta_{N_\vartheta+1} = \pi$, respectively.
- $f_\vartheta < 0$
  The cell borders along the ϑ-direction are equally distributed to create N_ϑ cells.
- $f_\vartheta = 1$
  The i-th cell border in the ϑ-direction is calculated according to:

$$\vartheta_i = \begin{cases} \frac{\pi}{2} \sin\left(\frac{i\pi}{N_\vartheta - 1}\right), & \text{if } i < \frac{N_\vartheta}{2} \\ \pi\left(1 - \frac{1}{2}\sin\left(\frac{i\pi}{N_\vartheta - 1}\right)\right), & \text{if } i \geq \frac{N_\vartheta}{2} \end{cases} \tag{2.3}$$

- $f_\vartheta > 1$
  The i-th cell border in the ϑ-direction is calculated according to:

$$\vartheta_i = \begin{cases} \frac{\pi}{2} - \frac{(f_\vartheta^{N_\vartheta/2-i} - 1)\frac{\pi}{2}}{f_\vartheta^{N_\vartheta/2} - 1}, & \text{if } i < \frac{N_\vartheta}{2} \\ \frac{\pi}{2} + \frac{(f_\vartheta^{i-N_\vartheta/2} - 1)\frac{\pi}{2}}{f_\vartheta^{N_\vartheta/2} - 1}, & \text{if } i \geq \frac{N_\vartheta}{2} \end{cases} \tag{2.4}$$

If multiple shape parameters f are zero, the lists of cell borders have to be in the same order as the shape parameters itself. As for the order of cells the grid runs over ϑ, φ, and r. Hence, the position of each cell within the grid is defined by its order of appearance in the grid file. The cell at the center is the last cell in the list leading to a total amount of $N_\mathrm{c} = N_r \times N_\varphi \times N_\vartheta + 1$ cells. This gives for the spherical grid file the following form:
$\mathrm{ID}_\mathrm{grid}$ (unsigned short, 2 bytes, 30 = spherical)
N_phys (unsigned short, 2 bytes, maximal 21 possible physical quantities in each cell)
$\mathrm{ID}_{n_\mathrm{g}}$ $\mathrm{ID}_{T_\mathrm{g}}$ ... (unsigned short, 2 bytes, optionally and in arbitrary order)
R_min R_max (double, 8 bytes)
N_r N_φ N_ϑ (unsigned short, 2 bytes)
f_r f_φ f_ϑ (double, 8 bytes)
n_g T_g ... (double, 8 bytes, in the same order as in the header, 1st cell)
...
n_g T_g ... (double, 8 bytes, last outer cell, $(N_\mathrm{c} - 1)$-th cell)
n_g T_g ... (double, 8 bytes, center cell, N_c-th cell)

The spherical grid geometry as shown in Fig. 2.1 on the left-hand side would result from the following sequence of values:

```
30 4 0 7 8 9
1.496e+11 1.496e+12
2 4 2
1.005 -1 -1
1.72e6 -1.647e+04 1.647e+04 0.0
1.72e6 -1.647e+04 1.647e+04 0.0
1.72e6 -1.647e+04 -1.647e+04 0.0
1.72e6 -1.647e+04 -1.647e+04 0.0
1.72e6 1.647e+04 -1.647e+04 0.0
1.72e6 1.647e+04 -1.647e+04 0.0
1.72e6 1.647e+04 1.647e+04 0.0
```

```
1.72e6 1.647e+04 1.647e+04 0.0
1.72e6 -3.897e+04 3.897e+04 0.0
1.72e6 -3.897e+04 3.897e+04 0.0
1.72e6 -3.897e+04 -3.897e+04 0.0
1.72e6 -3.897e+04 -3.897e+04 0.0
1.72e6 3.897e+04 -3.897e+04 0.0
1.72e6 3.897e+04 -3.897e+04 0.0
1.72e6 3.897e+04 3.897e+04 0.0
1.72e6 3.897e+04 3.897e+04 0.0
1.72e6 0.0 0.0 0.0
```

This example is a spherical grid containing four physical quantities, a constant gas number density of $n_\mathrm{g} = 1.72 \times 10^6\,\mathrm{m}^{-3}$ and a toroidal gas velocity component ($v_{\mathrm{g},x}$, $v_{\mathrm{g},y}$, $v_{\mathrm{g},z}$) rotation around the z-axis. The geometry is defined by an inner radius of $R_\mathrm{min} = 10\,\mathrm{au}$ and an outer radius of $R_\mathrm{max} = 100\,\mathrm{au}$ and a number of cells of $N_r = 2$, $N_\varphi = 4$, $N_\vartheta = 2$ cells in r-, φ-, and ϑ-direction with the shape parameters $f_r = 1.005$, $f_\varphi = -1$, and $f_\vartheta = -1$.

2.3.3 Cylindrical grid

The structure of the cylindrical grid is quite similar to the spherical one. The header is followed by R_min, R_max, Z_max, N_r, N_φ, N_z, f_r, f_φ, f_z. For the cylindrical grid R_min and R_max define the inner and outer radius followed by the number of cells in r-direction N_r, φ-direction N_φ, and z-direction N_z, respectively. The boundary of the grid is a cylinder with a radius of R_max and the z-direction extends from $-Z_\mathrm{max}$ to Z_max. The factors f_r, f_φ, and f_z serve the same function as introduced in Sect. 2.3.2 with the following modifications:

Radial direction

- The options of the radial direction of the spherical grid

Phi direction

- The options of the φ-direction of the spherical grid
- $f_\varphi = -1$
  The header has to be followed by N_r values (double, 8 bytes) that define the number of phi cells $N_{\varphi,i}$ in the radial ring i. The value N_φ will be ignored in this case.

Z direction

- The options of the ϑ-direction of the spherical grid (replace ϑ with z)
- $f_z = -1$
  The header has to be followed by N_r values (double, 8 bytes) that define the vertical width of each cylindrical cell $h_{z,i}$ in the radial ring i. After distributing N_z cells centered around the midplane, the leftover space to Z_max will be ignored (set to be empty).

The grid cells run over z, φ, and r. The innermost cylinder cells start at $-Z_\mathrm{max}$ and run to Z_max resulting in a total amount of $N_\mathrm{c} = N_z(N_r \times N_\varphi + 1)$ cells. Hence, the position of each cell within the grid is defined by its order of appearance in the grid file. The quantities in the cylindrical grid file appear in the following order:
$\mathrm{ID}_\mathrm{grid}$ (unsigned short, 2 bytes, 40 = cylindrical)
N_phys (unsigned short, 2 bytes, maximal 21 possible physical quantities in each cell)
$\mathrm{ID}_{n_\mathrm{g}}$ $\mathrm{ID}_{T_\mathrm{g}}$ ... (unsigned short, 2 bytes, optionally and in arbitrary order)
R_min R_max Z_max(double, 8 bytes)
N_r N_φ N_z(unsigned short, 2 bytes)

f_r f_φ f_z (double, 8 bytes)
n_g T_g ... (double, 8 bytes, in the same order as in the header, 1st cell)
...
n_g T_g ... (double, 8 bytes, last outer cell)
n_g T_g ... (double, 8 bytes, first center cell at $-Z_{max}$)
n_g T_g ... (double, 8 bytes, last center cell at Z_{max}, N_c-th cell)

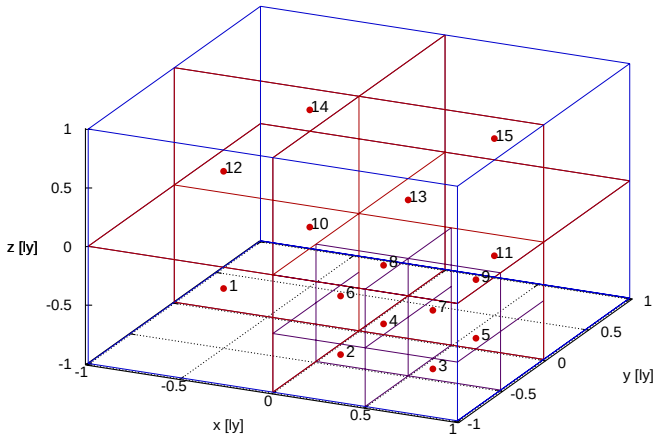
The following sequence of values grid is an as shown in Fig. 2.1 on the right-hand side:

```
40 4 0 4 5 6
3.086E+018 1.543E+019
3 4 2
1.001 -2 -2
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
7.36e2 0.0e+0 0.0e+0 1.0e-7
```

This is an example of a cylindrical grid that contains four physical quantities, a constant gas number density of $n_g = 7.36 \times 10^2\,\mathrm{m}^{-3}$ and a magnetic field with a constant magnitude of $10^{-7}\,\mathrm{T}$ in the z-direction. The geometry is defined by an inner radius of $R_{min} = 0.1\,\mathrm{pc}$ and an outer radius of $R_{max} = 0.5\,\mathrm{pc}$. The number of cells are $N_r = 3$, $N_\varphi = 4$, $N_z = 2$ in r-, φ-, and z-direction with the shape parameters $f_r = 1.005$, $f_\varphi = -2$ and $f_z = -2$.

2.3.4 Octree grid

For the octree grid one number (double, 8 bytes) follows after the header defining the dimension (d_{max}) of the entire cube. Finally, the grid refinement has to be represented by a sequence of numbers where the first number (unsigned short, 2 bytes) defines whether the cell is a leaf (1) or a branch (0). The second number (unsigned short, 2 bytes) is the grid level. In case of a leaf it follows a data section with numbers (4 bytes) which must match the exact order of identifies of the physical quantities defined in the header. The boundary of the grid is a cube with a side length



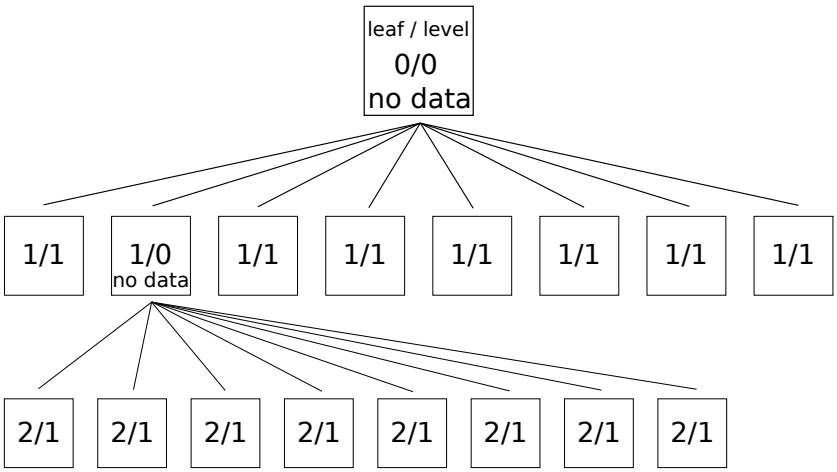


Figure 2.2: Exemplary octree grid (top) and its representation as graph (bottom) where level 0 represent the entire cube. The numbers represent the order of cells defined by their position within the octree graph.

of d_{max}. Hence, the position of each cell within the grid is defined by its order of appearance in the grid file. In general the octree data format has the following form:
ID_{grid} (unsigned short, 2 bytes, 20 = octree)
N_{phys} (unsigned short, 2 bytes, maximal 21 possible physical quantities in each cell)
ID_{n_g} ID_{T_g} ... (unsigned short, 2 bytes, optionally and in arbitrary order)
d_{max} (double, 8 bytes, dimension of the cube at level 0)
0/1 (unsigned short, 2 bytes, 0 = branch, 1 = leaf)/(unsigned short, 2 bytes, level 0 = entire cube)
n_g T_g ... (float, 4 bytes, in the same order as in the header)
...
0/1 (unsigned short, 2 bytes) level (unsigned short, 2 bytes)
n_g T_g ... (float, 4 bytes, last cell)

The grid refinement as shown in Fig. 2.2 would result from following sequence of values:

20 2 0 3

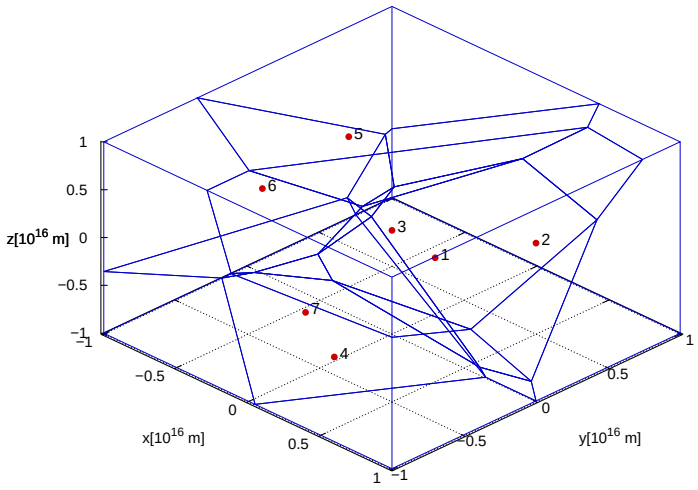


Figure 2.3: Exemplary voronoi grid. The numbers represent the order of cells defined by their appearance.

```
1.892e16
0 0
1 1 1.43e5 25
1 0
2 1 5.56e7 31
2 1 5.56e7 31
2 1 5.56e7 31
2 1 5.56e7 31
2 1 5.56e7 31
2 1 5.56e7 31
2 1 5.56e7 31
2 1 5.56e7 31
1 1 1.43e5 25
1 1 1.43e5 25
1 1 1.43e5 25
1 1 1.43e5 25
1 1 1.43e5 25
1 1 1.43e5 25
1 1 1.43e5 25
```

This is an example of an octree grid with a maximal grid refinement of two and a side length of 2 ly. It contains mostly a constant gas number density and dust temperature of $n_g = 1.43 \times 10^5\,\mathrm{m}^{-3}$ and $T_g = 25\,\mathrm{K}$, respectively, with a level one refinement. A smaller region has a level two refinement with $n_g = 5.56 \times 10^7\,\mathrm{m}^{-3}$ and $T_g = 31\,\mathrm{K}$.

2.3.5 Voronoi grid

A Voronoi grid is unstructured and can easily be imagined as many connected soap bubbles. However, the order in which the cells are written to the grid files is not arbitrary. Each Voronoi cell knows the IDs of its neighboring cells. These IDs are identical with the order of appearance of the

cells in the grid file. After the header follows an additional sequence of two numbers (double, 8 bytes), the number of cells N_c in the grid and the side length d_{max} of the cube. The boundary of the grid is a cube with a side length of d_{max}. Hence, the position of each cell within the grid is defined by the three numbers of its x-, y-, and z-coordinates. After the coordinates follows the volume of the cell and the list of physical parameters as defined in the header. The next number gives the number of neighbors N_{ne} followed by the IDs of the neighboring cells. If the ID is negative then the ID stands for a wall of the surrounding cube ($1 = x$, $2 = -x$, $3 = y$, $4 = -y$, $5 = z$, and $6 = -z$ wall).

ID_{grid} (unsigned short, 2 bytes, 50 = voronoi)
N_{phys} (unsigned short, 2 bytes, maximal 21 possible physical quantities in each cell)
ID_{n_g} ID_{T_g} ... (unsigned short, 2 bytes, optionally and in arbitrary order)
N_c d_{max} (double, 8 bytes)
x y z (float, 4 bytes) volume (double, 8 bytes) n_g T_g ... (float, 4 bytes) N_{ne} ID_1 ... $ID_{N_{ne}}$ (integer, 4 bytes, first cell)
x y z (float, 4 bytes) volume (double, 8 bytes) n_g T_g ... (float, 4 bytes) N_{ne} ID_1 ... $ID_{N_{ne}}$ (integer, 4 bytes)
...
x y z (float, 4 bytes) volume (double, 8 bytes) n_g T_g ... (float, 4 bytes) N_{ne} ID_1 ... $ID_{N_{ne}}$ (integer, 4 bytes, last cell)

The grid refinement as shown in Fig. 2.3 would result in the following sequence of values:

```
50 3 0 2 3
7 2.0e16
0.4e16 -0.8e16 -0.4e16 0.72e48    1.1e5 10 20 8 -5 -2 -3 2 0 1 5 6
-0.1e16 -0.5e16 -0.5e16 1.26e48     1.1e5 10 20 8 1 5 0 -5 -1 3 -3 2
0.4e16 0.6e16 -0.2e16 1.86e48    1.1e5 10 20 11 3 0 6 -5 -1 -4 -2 2-6 4 5
-0.6e16 -0.3e16 0.3e16 1.15e48    1.1e5 10 20 9 0 4 2 1 6 -1 -6 -3 3
0.3e16 0.0e16 0.0e16 0.70e48    1.1e5 10 20 7 3 -2 2 6 1 5 4
0.3e16 -0.3e16 0.5e16 1.17e48    1.1e5 10 20 9 3 5 6 4 -6 1 -2 0 -3
-0.4e16 0.1e16 0.7e16 1.13e48    1.1e5 10 20 7 -6 -4 -1 1 5 2 0
```

This is an example of a Voronoi grid with seven cells. The content is a constant gas number density of $n_g = 1.1 \times 10^5\,m^{-3}$, a dust temperature of $T_d = 10\,K$, and a gas temperature of $T_g = 20\,K$.

2.3.6 Unit conversion

POLARIS works internally strictly in SI units. In order to convert the physical parameters of the input grids, POLARIS provides a number of commands for the command file. The command `<conv_dens>` converts the densities of the input grid into number densities [m^{-3}]. Lengths and velocities can be converted in [m] and [m/s] by `<conv_len>` and `<conv_vel>`, respectively. The magnetic field can be converted into Tesla with the command `<conv_mag>`. Finally, the command `<mass_fraction>` defines the ratio of dust mass to gas mass ratio in order to calculate the dust density distribution, with the help of the average molecular wight defined by the command `<mu>`, if the dust density is not provided. The following example shows the conversion factors to convert `cgs` to `SI`:

```
# cgs into SI
<conv_dens> 1e6 # cm^3 in m^3
<mu> 2.0
<conv_len> 0.01 # cm in m
<conv_mag> 1e-4 # G in T
<conv_vel> 0.01 # cm/s in m/s
<mass_fraction> 0.01 # dust mass is 1% of the total gas mass
```

HINTS:

- If a conversion of units was applied, the parameters of any newly written grid will be stored in `SI`. You have to consider this in the follow-up calculations accordingly.
- Instead of defining the conversion factors for cgs to SI, the command `<path_grid_cgs>` can be used to read a grid that uses strictly cgs units.

2.4 Dust properties

2.4.1 Dust cross sections

Some of POLARIS RT simulations are dependent on the optical properties of dust grains. Such properties can individually be pre-calculated and tabulated, for instance with MIEX (Wolf & Voshchinnikov 2004) or DDSCAT (Draine & Flatau 2013). The choice of grain size, dust materials, and size distribution is crucial in the simulation of synthetic observations. Hence, POLARIS can handle an arbitrary number of tabulated dust grain properties and creates a dust mixture dependent on user defined parameters. The input tables are plain text and follow the same commenting rules as the input command file (see Sect. 2.1). In contrast to the command files, here, the order of all values matters.

Each dust parameters file starts with an arbitrary string describing the dust grain material. In the following line the number of dust grain sizes N_a, the number of wavelength N_λ, the number of inclination angles N_i, the ratio of minor to major dust grain axes $s < 1$, the density ϱ_d [kg/m^3] of the dust grain material itself, the sublimation temperature T_sub [K] of the dust grain material, a geometrical factor δ_RAT relevant for RAT alignment ($\delta_\mathrm{RAT} = 1$ for a sphere, Draine & Weingartner 1996, 1997), and finally a number that defines the alignment behavior (0 = not aligned, 1 = aligned according to chosen alignment mechanism) are listed. The next two lines are the exact dust grain sizes and wavelength, respectively, corresponding to the numbers of N_a and N_λ. Next, it follows a table running over wavelength λ and dust grain radii a. The dust grain radius a is defined to be the grain radius of a sphere of equivalent volume. This table contains the efficiencies of extinction ($Q_{\mathrm{ext},\perp}$, $Q_{\mathrm{ext},||}$), scattering ($Q_{\mathrm{sca},\perp}$, $Q_{\mathrm{sca},||}$), and absorption ($Q_{\mathrm{abs},\perp}$, $Q_{\mathrm{abs},||}$) for light polarized perpendicular and parallel with respect to the grains minor principle axis. Furthermore, the table contains the circular polarization efficiency ΔQ_circ, and – one for each inclination angle – the efficiencies $Q_{\Gamma,1}$ ... Q_{Γ,N_i} for RAT alignment (see Sect. 3.7.3). Based on the chosen phase function (see Sect. 3.7.1), additional parameters – one for each inclination angle – have to be provided in the table:

- For the Henyey-Greenstein function, g_1 ... g_{N_i},
- for the Draine Henyey-Greenstein function, g_1 ... g_{N_i} and α_1 ... α_{N_i},
- for the three-parameter Henyey-Greenstein function, $g_{1,1}$ ... g_{1,N_i}, $g_{2,1}$ ... g_{2,N_i} and w_1 ... w_{N_i}

are necessary. Later, the different cross-sections are calculated by POLARIS with $C = \pi a^2 Q$. In the general form the quantities in a dust parameters file have to be arranged as follows:

description string
#radii wavel. inclinations aspect_ratio density sub_temp δ align
N_a N_λ N_i s ϱ_d[kg/m^3] T_sub[K] δ_RAT 0/1
#grain radii [m]
a_1 ... a_{N_a}
#wavelength [m]

λ_1 ... λ_{N_λ}
$\#Q_{\text{ext},\perp}$ $Q_{\text{ext},||}$ $Q_{\text{abs},\perp}$ $Q_{\text{abs},||}$ $Q_{\text{sca},\perp}$ $Q_{\text{sca},||}$ ΔQ_{circ} $Q_{\Gamma,1}$ ... Q_{Γ,N_i} g_1 ... g_{N_i}
$Q_{\text{ext},\perp,1,1}$ $Q_{\text{ext},||,1,1}$ $Q_{\text{abs},\perp,1,1}$ ...
$Q_{\text{ext},\perp,1,2}$ $Q_{\text{ext},||,1,2}$ $Q_{\text{abs},\perp,1,2}$ ...
...
$Q_{\text{ext},\perp,1,N_a}$ $Q_{\text{ext},||,1,N_a}$ $Q_{\text{abs},\perp,1,N_a}$ ...
$Q_{\text{ext},\perp,2,1}$ $Q_{\text{ext},||,2,1}$ $Q_{\text{abs},\perp,2,1}$ ...
$Q_{\text{ext},\perp,2,2}$ $Q_{\text{ext},||,2,2}$ $Q_{\text{abs},\perp,2,2}$ ...
...
$Q_{\text{ext},\perp,N_\lambda,N_a}$ $Q_{\text{ext},||,N_\lambda,N_a}$ $Q_{\text{abs},\perp,N_\lambda,N_a}$ ...

For example, the POLARIS astro-silicate parameters file of oblate dust grains is created as follows:

```
#string ID
astronomical silicate (oblate shaped)
#nr_radii nr_wave. inc_angles aspect_ratio density sub_temp delta align
112 100 31 0.5 3800 1200 1.95675 1
#a_eff
5.00E-9 7.83E-9 1.08E-8 1.39E-8 1.72E-8 2.06E-8 ...
#wavelength
9.00E-8 2.63E-7 4.48E-7 6.46E-7 8.57E-7 1.00E-6 ...
#Qext1 Qext2 Qabs1 Qabs2 Qsca1 Qsca2 dQcirc Qtrq0 ...
8.84E-1 4.65E-1 8.66E-1 4.56E-1 1.80E-2 9.22E-3 2.71E-2 9.08E-9 ...
1.52E-2 1.76E-2 1.97E-2 2.17E-2 2.34E-2 2.48E-2 2.51E-2 2.67E-2 ...
...
1.95E-4 1.94E-4 1.94E-4 1.93E-4 1.93E-4 1.92E-4 1.92E-4 1.92E-4 ...
```

2.4.2 Scattering matrices

In the most general case the Müller matrix for scattering has also 4×4 entries that determine the polarization state of photons for each scattering event. Since such a matrix depends at least on the grain size a, dust material, wavelength λ, and the scattering angles ϑ and φ, the parameter space can be enormous (see Sect. 3.7.5 for details about the physics of scattering). This holds even more for the upcoming POLARIS mode for scattering on non-spherical dust grains. Hence, POLARIS does not read the files for the scattering matrices except for the simulation mode `CMD_DUST_SCATTERING` and Mie scattering (`PH_MIE`, see Sect. 3.7.1 for more information about phase functions). Considering the parameter space, the scattering matrices are stored in an extra directory. The directory has the same name as the dust parameters file defined in Sect. 2.4. This directory contains several files with scattering matrices for each wavelength. The files are labeled as `'wIDXXX.sca'` where `XXX` represents the wavelength index, e.g. `001`. Each matrix file in turn contains then the data over grain size, incident angles i, scattering angle φ and ϑ and a list of the actual entries of the matrix (in future we may split the files also for different grain sizes). For instance,

$M_{1,1,1,1,11}$ $M_{1,1,1,1,12}$ $M_{1,1,1,1,33}$ $M_{1,1,1,1,34}$ $M_{1,1,1,2,11}$ $M_{1,1,1,2,12}$ $M_{1,1,1,2,33}$ $M_{1,1,1,2,34}$ ...
$M_{1,1,1,N_\vartheta,11}$ $M_{1,1,1,N_\vartheta,12}$ $M_{1,1,1,N_\vartheta,33}$ $M_{1,1,1,N_\vartheta,34}$ $M_{1,1,2,1,11}$ $M_{1,1,2,1,12}$ $M_{1,1,2,1,33}$ $M_{1,1,2,1,34}$ ...
$M_{1,1,N_\varphi,N_\vartheta,11}$ $M_{1,1,N_\varphi,N_\vartheta,12}$ $M_{1,1,N_\varphi,N_\vartheta,33}$ $M_{1,1,N_\varphi,N_\vartheta,34}$ $M_{1,2,1,1,11}$ $M_{1,2,1,1,12}$ $M_{1,2,1,1,33}$ $M_{1,2,1,1,34}$
... $M_{N_a,N_i,N_\varphi,N_\vartheta,11}$ $M_{N_a,N_i,N_\varphi,N_\vartheta,12}$ $M_{N_a,N_i,N_\varphi,N_\vartheta,33}$ $M_{N_a,N_i,N_\varphi,N_\vartheta,34}$

Here, all data must be stored in a **binary format** as a **4 byte float** for each parameter. Additionally, the directory needs also to contain the file `'scat.inf'`. This file contains only three lines specify the number of bins and matrix entries as **plain text**. In the general form the file `'scat.inf'` looks like this:

N_a N_λ N_i N_φ N_ϑ
N_M
N_{11} N_{12} N_{13} N_{14} N_{21} N_{22} N_{23} N_{24} N_{31} N_{32} N_{33} N_{34} N_{41} N_{42} N_{43} N_{44}

Here, N_a is the number of dust grain sizes, N_λ is the number of wavelengths, N_i is the number of incident angles, N_φ is the number of scattering angles in φ-direction, and N_ϑ is the number of scattering angles in ϑ-direction. Dependent on the dust grain model not all entries of the scattering matrix are necessary and some may be equal. Hence, the total number of entries ($N_M \leq 16$) defined by the next line. The last line defines the scattering matrix itself. Here, the number is the position of entries in the binary file while its position defines the position within the scattering matrix. As an example:

```
#nr. of dust species #wav. #inc. angles #phi angle #theta angle
1    231     1     1     181

#data length
4
#position of matrix elements
#M11 M12 M13 M14 M21 M22 M23 M24 M31 M32 M33 M34 M41 M42 M43 M44
1   2   0   0   2   1   0   0   0   0   3   4   0   0  -4   3
```

This example shows a data set of scattering matrices for a single dust grains size $N_a = 1$, for $N_\lambda = 231$ wavelength, one scattering angle φ-direction $N_\varphi = 1$, and $N_\vartheta = 181$ in ϑ-direction. The total number of entries is $N_M = 4$. In this example, the scattering matrix itself is built up exactly as shown in Eq. (3.88).

2.4.3 Dust refractive index

As an alternative, the dust grains can be assumed to be homogeneous spheres in order to apply the Mie scattering theory (Mie 1908; van de Hulst 1957; Bohren & Huffman 1983). Hereby, the optical properties of the particles are calculated with the approach by Wolf & Voshchinnikov (2004). For this purpose, a file containing the refractive index of the composition has to be provided. The input tables are again plain text and similar to the dust cross-section files. However, they need to have the ending '`.nk`'. Each dust refractive index file starts with an arbitrary string describing the dust grain material. In the following line the number of wavelengths N_λ, the number of inclination angles N_i, the aspect ratio $s < 1$ of the dust grains, the density ϱ_d [kg/m^3] of the dust grain material itself, the sublimation temperature T_sub [K] of the dust grain material, a geometrical factor δ_RAT relevant for RAT alignment ($\delta_\mathrm{RAT} = 1$ for a sphere, Draine & Weingartner 1996, 1997), and finally a number that defines the alignment behavior (0 = not aligned, 1 = aligned according to chosen alignment mechanism) are listed. At the moment, only spherical dust grains can be calculated based on the refractive index. However, information about the non-spherical shape and alignment are already contained in the dust refractive index file for future use. The following lines have to contain the wavelength λ as well as the real and imaginary component of the refractive index n and k separated with at least a space character.

description string
nr. of wavelengths inclinations aspect_ratio density sub_temp δ align
N_λ N_i s ϱ_d[kg/m^3] T_sub[K] δ_RAT 0/1
λ_1 n_1 k_1
λ_2 n_2 k_2
...
λ_N n_N k_N

For example, the POLARIS asto-silicate parameters file of spherical dust grains is created as follows:

```
#string ID
astronomical silicate

#nr. of wavelength #inc. angles #aspect ratio #density [kg/m^3] #sub.temp #delta #
    ↪ align
100 1 1 3500 1200 0 0

#wavelength real refractive index imaginary refractive index
4.9999999999999774e-08 8.271977641350000132e-01 2.685931626490000168e-01
5.564875580140000232e-08 7.621913301310000444e-01 3.246343812520000038e-01
6.193568044479999963e-08 6.387779917170000044e-01 5.901285004609999607e-01
6.893287112919999462e-08 8.504294128460000435e-01 8.380542302629999662e-01
...
2.000000000000000042e-03 3.432783175560000011e+00 2.465826813000000090e-02
```

2.4.4 Heat capacities / Enthalpies

The heat capacities (or enthalpies) are required to calculate the stochastic heating (i.e. quantum or single photon heating). The required `'calorimetry.dat'` file has to be in a sub-directory named after the dust parameters file at the location where the dust parameters files are. This file must contain the heat capacity (or enthalpy) for different temperatures (optional: grain sizes). An index is used to specify if heat capacities (0) or enthalpies (1) are used (type of calorimetry). The file has the following structure in case of heat capacities:

```
# nr. of temperatures
```
N_{temp}
```
# temperature: T [K]
```
T_0 T_1 T_1 ... T_{N-1}
```
# type of calorimetry
0
# heat capacity C [J/K/m^3]
# C(T_0, a_0), C(T_0, a_1), C(T_0, a_2), ...
# C(T_1, a_0), C(T_1, a_1), C(T_1, a_2), ...
```
$C(T_0, a_0)$, $C(T_0, a_1)$, ... $C(T_0, a_{N-1})$
$C(T_1, a_0)$, $C(T_1, a_1)$, ... $C(T_1, a_{N-1})$
...
$C(T_{N-1}, a_0)$, $C(T_{N-1}, a_1)$, ... $C(T_{N-1}, a_{N-1})$

For example, the POLARIS heat capacity file for the `'aPyM5.dat'` dust is as follows:

```
#nr. of temperatures
30
# temperature: T [K]
0.1 0.14522119293692692 0.2109016609370204 0.30626685580037694 ...
# heat capacity C [J/K/m^3]
# C(T_0, a_0), C(T_0, a_1), C(T_0, a_2), ...
# C(T_1, a_0), C(T_1, a_1), C(T_1, a_2), ...
0.8959215850801721 0.8959215850801721 0.8959215850801721 ...
2.412681532591634 2.412681532591634 2.412681532591634 ...
5.135704178483553 5.135704178483553 5.135704178483553 ...
10.972358781119063 10.972358781119063 10.972358781119063 ...
...
6801426.356038569 6801426.356038569 6801426.356038569 ...
```

2.5 The gas species parameters

2.5.1 LAMDA molecular database

POLARIS line radiative transfer (LRT) simulations need the pre-calculated quantum numbers, energy levels, Einstein coefficients, and collision rates characteristic for each species of gas species. These molecular parameters files are publicly available at the website of the Leiden Atomic and Molecular Database (LAMDA, van der Tak et al. 2020). The provided LAMDA file format is fully supported by POLARIS without any intermediate conversion step. Lines beginning with a ! are comments and will be ignored by the POLARIS command parser.

Since the LAMDA database is an external source, we just provide a short description of the file format in this manual as far as it is relevant for the POLARIS code. For additional information, we refer to the corresponding website. The first parameter in a LAMDA file is a string with the name of the gas species. The next number is the molecular weight followed by the numbers of pre-calculated energy levels. In the following table the energy levels, the energy on each level itself, and the quantum number of each level are listed. The number of pre-calulated transitions is in the line followed by a second table. This table provides the an unique ID for each transition the number of energy levels between which the transition occurs, the Einstein A coefficient, as well as the characteristic frequency of the transition. The unique transition ID in this table is relevant for the POLARIS command files (see Sect. 2.1) and the Zeeman parameters file (see Sect. 2.5.2) in order to select the desired transitions. The next four lines define the number of considered collision partners to calculate the collisional excitation, a string identifying the collision partner and the reference of this data, the number of collision transitions, and finally the number of collision temperatures. The tables are at the end of the file and provide the physical parameters for all of permutations of collision partners, transitions and collision temperatures.

2.5.2 Zeeman parameters files

To consider the Zeeman splitting in LRT simulations, additional parameters complementing the LAMDA molecular parameters file are required. For further details about the underlying physical relevance of these parameters see Sect. 3.8.2. These additional parameters are listed in an extra file and shipped with the POLARIS package for different gas species. Please contact the developers, if this file is not existent for your desired gas species.

At the top is the name of the gas species corresponding to that of the LAMDA parameters file and in the second line is the radius of the gas species. The next number defines the amount of lines with Zeeman lines. In the following sections come the upper and lower level Landè g factors, the number of Zeeman upper and lower sub-levels, as well as the line strength between all Zeeman sub-levels.

In its general form the Zeeman file is ordered as follows (see Sect. 3.8 for details):

```
!Molecule name
!Molecule radius for collision calculations
!Number of transitions N_tr with Zeeman effect
!Corresponding transition index in LAMDA database of the first transition
!Lande factor of upper level of the first transition
!Lande factor of lower level of the first transition
!Number of Zeeman sub-levels in the upper level of the first transition
!Number of Zeeman sub-levels in the lower level of the first transition
!Line strength of π and σ± transitions for all permutations quantum numbers M' and M''
...
!Lande factor of upper level of the N_tr-th transition
!Lande factor of lower level of the N_tr-th transition
```

```
!Number of Zeeman sub-levels in the upper level of the N_tr-th transition
!Number of Zeeman sub-levels in the lower level of the N_tr-th transition
!Line strength of π and σ_± transitions for all permutations quantum numbers M' and M''
```

How the Zeeman parameters file of the gas species OH would look like, can be seen in Listing 2.2.

Listing 2.2: The Zeeman parameters file of the gas species OH.

```
!Molecule name
OH
!Molecule radius for collision calculations
0.958e-10
!Number of transitions with Zeeman effect
2
!Transition index in LAMDA database
2
!Lande factor of upper level
1.16828926744
!Lande factor of lower level
1.16828926744
!Number of Zeeman sub-levels in the upper level
3
!Number of Zeeman sub-levels in the lower level
3
!Line strength of π transition (M'=-1 -> M''=-1)
0.5
!Line strength of π transition (M'=0 -> M''=0)
0.0
!Line strength of π transition (M'=1 -> M''=1)
0.5
!Line strength of σ+ transition (M'=-1 -> M''=0)
0.25
!Line strength of σ+ transition (M'=0 -> M''=1)
0.25
!Line strength of σ− transition (M'=1 -> M''=0)
0.25
!Line strength of σ− transition (M'=0 -> M''=-1)
0.25
!Transition index in LAMDA database
3
!Lande factor of upper level
0.700973560462
!Lande factor of lower level
0.700973560462
!Number of Zeeman sub-levels in the upper level
5
!Number of Zeeman sub-levels in the lower level
5
```

Listing 2.2: The Zeeman parameters file of the gas species OH (continued).

```
!Line strength of π transition (M'=-2 -> M''=-2)
0.4
!Line strength of π transition (M'=-1 -> M''=-1)
0.1
!Line strength of π transition (M'=0 -> M''=0)
0.0
!Line strength of π transition (M'=1 -> M''=1)
0.1
!Line strength of π transition (M'=2 -> M''=2)
0.4
!Line strength of σ+ transition (M'=-2 -> M''=-1)
0.1
!Line strength of σ+ transition (M'=-1 -> M''=0)
0.15
!Line strength of σ+ transition (M'=0 -> M''=1)
0.15
!Line strength of σ+ transition (M'=1 -> M''=2)
0.1
!Line strength of σ- transition (M'=2 -> M''=1)
0.1
!Line strength of σ- transition (M'=1 -> M''=0)
0.15
!Line strength of σ- transition (M'=0 -> M''=-1)
0.15
!Line strength of σ- transition (M'=-1 -> M''=-2)
0.1
```

Table 2.3: Identifier of the physical quantities in a POLARIS grid. (not yet fully supported or anticipated for a future version of POLARIS; only used internally and not designed to be set by user)

physical quantity	description	ID
n_g [m^{-3}]	gas number density	0
n_d [m^{-3}]	dust number density	1
T_d [K]	dust temperature	2
T_g [K]	gas temperature	3
B_x [T]	magnetic field in x-direction	4
B_y [T]	magnetic field in y-direction	5
B_z [T]	magnetic field in z-direction	6
$v_{g,x}$ [m/s]	gas velocity in x-direction	7
$v_{g,y}$ [m/s]	gas velocity in y-direction	8
$v_{g,z}$ [m/s]	gas velocity in z-direction	9
p_x [$kg\,m^{-2}\,s^{-1}$]	radiative pressure in x-direction	10
p_y [$kg\,m^{-2}\,s^{-1}$]	radiative pressure in y-direction	11
p_z [$kg\,m^{-2}\,s^{-1}$]	radiative pressure in z-direction	12
a_{alg} [m]	dust grain alignment radius	13
a_{min} [m]	minimal dust grain radius	14
a_{max} [m]	maximal dust grain radius	15
q	exponent of the grain size distribution	16
σ_{mol}	ratio of a molecular species mass to the total gas mass	17
v_{turb} [m/s]	turbulent gas velocity	18
PDA	PDA photon count	19
ID_O	ID for the opiate database	20
ID_{dust}	ID for an individual dust component	21
n_{th} [m^{-3}]	number density of thermal electrons	22
T_e [K]	temperature thermal electrons	23
n_{CR} [m^{-3}]	number density of cosmic ray electrons	24
γ_{min}	minimal Lorentz-factor	25
γ_{max}	maximal Lorentz-factor	26
p	power-law index of cosmic ray electrons	27
ρ_g [$kg\,m^{-3}$]	gas mass density	28
ρ_d [$kg\,m^{-3}$]	dust mass density	29
E_x [$W\,m^{-1}\,m^{-2}$]	radiation field density in x-direction	30
E_y [$W\,m^{-1}\,m^{-2}$]	radiation field density in y-direction	31
E_z [$W\,m^{-1}\,m^{-2}$]	radiation field density in z-direction	32
E [$W\,m^{-1}\,m^{-2}$]	total radiation field density	33
$\langle\cos(\vartheta)\rangle$	average angle between radiation and magnetic field	34
$\langle\gamma\rangle$	anisotropy factor of the radiation field	35

Table 2.4: Available general commands and their default values for the POLARIS command files. The red names in brackets show for which simulation type (pipeline ID) the command can be used.

Command	Description
Simulation types	
<cmd> CMD_TEMP or CMD_TEMP_RAT or CMD_RAT or CMD_DUST_SCATTERING or CMD_DUST_EMISSION or CMD_LINE_EMISSION or CMD_SYNCHROTRON	Set the simulation type of the current task
	Default: No simulation will be performed without this command
General	
<path_grid> "/PATH/TO/GRID"	Path to the used grid
(ALL)	Default: No grid considered
<path_out> "/PATH/TO/RESULTS"	Path to the POLARIS results
(ALL)	Default: No grid considered
<start> START_INDEX	Set the first considered detector (index as occurrence in command file)
(DUST_EMISSION, LINE_EMISSION, SYNCHROTRON)	Default: Start with first detector
<stop> STOP_INDEX	Set the last considered detector (index as occurrence in command file)
(DUST_EMISSION, LINE_EMISSION, SYNCHROTRON)	Default: Stop after last detector
<conv_dens> FACTOR_DENS	Set the conversion factor for the density
(ALL)	Default: 1.0
<conv_len> FACTOR_LEN	Set the conversion factor for the length
(ALL)	Default: 1.0
<conv_mag> FACTOR_MAG	Set the conversion factor for the magnetic field strength
(ALL)	Default: 1.0
<conv_vel> FACTOR_VEL	Set the conversion factor for the velocity
(ALL)	Default: 1.0
<nr_threads> NR_THREADS	Set the number of processor cores on which POLARIS can run
(ALL)	Default: 1

Table 2.5: Available detector commands and their default values for the POLARIS command files. The red names in brackets show for which simulation type (pipeline ID) the command can be used.

Command	Description
Detectors	
<detector_dust ...>	Detector for dust emission. See Sect. 3.5
(DUST_EMISSION)	Default: No dust detector
<detector_dust_healpix ...>	Detector for dust emission (healpix background grid). See Sect. 3.5
(DUST_EMISSION)	Default: No dust detector
<detector_dust_polar ...>	Detector for dust emission (polar background grid). See Sect. 3.5
(DUST_EMISSION)	Default: No dust detector
<detector_dust_slice ...>	Detector for dust emission (slice background grid). See Sect. 3.5
(DUST_EMISSION)	Default: No dust detector
<detector_dust_mc ...>	Detector for emission scattered at the dust grains. See Sect. 3.5
(DUST_SCATTERING)	Default: No dust scattering detector
<detector_line ...>	Detector for line emission. See Sect. 3.5
(LINE_EMISSION)	Default: No line detector
<detector_line_healpix ...>	Detector for line emission (healpix background grid). See Sect. 3.5
(LINE_EMISSION)	Default: No line detector
<detector_line_polar ...>	Detector for line emission (polar background grid). See Sect. 3.5
(LINE_EMISSION)	Default: No line detector
<detector_line_slice ...>	Detector for line emission (slice background grid). See Sect. 3.5
(LINE_EMISSION)	Default: No line detector
<gas_species>	Gas species for line emission. See Sect. 3.8
(LINE_EMISSION)	Default: No gas species
<detector_sync ...>	Detector for synchrotron emission. See Sect. 3.5
(SYNCHROTRON)	Default: No synchrotron detector
<detector_sync_healpix ...>	Detector for synchrotron emission (healpix background grid). See Sect. 3.5
(SYNCHROTRON)	Default: No synchrotron detector
<max_subpixel_lvl> MAX_LEVEL	Set the maximum level of subpixeling
(DUST_EMISSION, LINE_EMISSION)	Default: 1

Table 2.6: Available radiation source commands and their default values for the POLARIS command files. The red names in brackets show for which simulation type (pipeline ID) the command can be used.

Command	Description
Radiation sources	
<source_star nr_photons = >	Stellar radiation source (see Sect. 3.2)
(TEMP, RAT, DUST_SCATTERING, DUST_EMISSION)	Default: No stellar radiation source
<source_starfield nr_photons = >	Starfield radiation source (see Sect. 3.2)
(TEMP, RAT, DUST_SCATTERING)	Default: No starfield radiation source
<source_isrf nr_photons = >	Interstellar radiation field as radiation source (see Sect. 3.2)
(TEMP, RAT)	Default: No ISRF radiation source
<source_background nr_photons = >	Background radiation source (see Sect. 3.2)
(DUST_EMISSION, LINE_EMISSION, SYNCHROTRON)	Default: A basic background source will be initialized if required
<source_dust nr_photons = >	Use the dust grains as radiation source (activates self-scattering calculations, see Sect. 3.2)
(DUST_SCATTERING)	Default: The dust grains are not considered calculating the radiation scattered at dust grains
<axis1> AXIS1_X AXIS1_Y AXIS1_Z	Rotation axis for first rotation angle
(ALL)	Default: (1, 0, 0)
<axis2> AXIS2_X AXIS2_Y AXIS2_Z	Rotation axis for second rotation angle
(ALL)	Default: (0, 1, 0)

Table 2.7: Available dust commands and their default values for the POLARIS command files. The red names in brackets show for which simulation type (pipeline ID) the command can be used.

Command	Description
Dust	
<dust_component> or <dust_component id = >	Set the considered dust grain compositions (see Sect. 3.6)
(ALL)	Default: No dust grains used
<align> ALIG_PA or ALIG_RAT or ALIG_IDG or ALIG_GOLD or ALIG_INTERNAL or ALIG_NONPA	Considered alignment mechanism of non-spherical dust grains
(DUST_EMISSION)	Default: Random alignment
<sub_dust> 1 (yes) or 0 (no)	Enables/Disables sublimation of dust grains (sublimation temperature is set in the dust catalog)
(TEMP, TEMP_RAT)	Default: No sublimation
<f_highJ> F_HIGHJ	Set the ratio of grains at high-J attractor point for the radiative torque alignment (See Sect. 3.7.3, RAT)
(DUST_EMISSION)	Default: 0.25
<f_c> F_C	Set the correlation factor for the (see Sect. 3.7.3)
(DUST_EMISSION)	Default: 0.6
<Q_ref> Q_ref	Reference value for the radiative torque efficiency (see Eq. 3.66)
(RAT, TEMP_RAT)	Default: 0.4
<alpha_Q> alpha_Q	Exponent for the radiative torque efficiency power law, a positive alpha_Q gives a negative exponent (see Eq. 3.66)
(RAT, TEMP_RAT)	Default: 3.0
<R_rayleigh> R_rayleigh	Rayleigh reduction factor non-perfect alignment or RAT alignment if imperfect internal alignment is not used (see Sect. 3.7.3, RAT)
(TEMP, RAT, TEMP_RAT, DUST_EMISSION)	Default: 1.0
<adj_tgas> TEMP_FACTOR	Overwrite the gas temperature with the dust temperature times TEMP_FACTOR
(TEMP, TEMP_RAT)	Default: Keep gas temperature as it is in the grid
<mass_fraction> MASS_FRACTION	Set gas-to-dust mass ratio
(ALL)	Default: 0.01

Table 2.8: Available dust commands and their default values for the POLARIS command files (continued). The red names in brackets show for which simulation type (pipeline ID) the command can be used.

Command	Description
Dust	
<dust_offset> 1 (yes) or 0 (no)	Enables/Disables the use of the dust temperature of the input grid as an offset (see Sect. 3.7.2)
(TEMP, TEMP_RAT)	Default: Do not use the previous dust temperature
<radiation_field> 1 (yes) or 0 (no)	Enables/Disables the saving of the radiation field (required by <stochastic_heating>, see Sect. 3.7.2)
<rt_scattering> 0	Disables the usage of the radiation field to include scattering in the ray-tracing
(DUST_EMISSION)	Default: Use the radiation field to add scattered light to thermal emission
<full_dust_temp> 1 (yes) or 0 (no)	Enables/Disables the calculation of the dust temperature for each dust grain size (see Sect. 3.7.2)
(TEMP, TEMP_RAT)	Default: Calculate the dust temperature only for the effective dust grain size
<stochastic_heating> SIZE_LIMIT	Consider the emission of stochastically heated dust grains with a size < SIZE_LIMIT (see Sect. 3.7.2)
(DUST_EMISSION)	Default: Calculate the dust emission only from equilibrium temperature(s)
<larm_f> LARM_F	Set the prefactor for alignment limitation of the magnetic field (see, Eq. 3.63)
(TEMP, RAT, TEMP_RAT, DUST_EMISSION)	Default: 4.1×10^{-19}
<phase_function> PH_HG, PH_DHG, PH_TTHG, PH_MIE, PH_ISO, or <phase_function id = >	Set the phase function used for all dust grains (Henyey-Greenstein, Mie or isotropic, see Sect. 3.7.1), or for each considered dust grain composition (id =)
(TEMP, RAT, DUST_SCATTERING)	Default: PH_ISO
<enfsca> 1 (yes) or 0 (no)	Enables/Disables the use of enforced first scattering
(TEMP, RAT, DUST_SCATTERING)	Default: Use enforced first scattering
<peel_off> 1 (yes) or 0 (no)	Enables/Disables the use of the peel-off technique (see Sect. 3.3.3)
(DUST_SCATTERING)	Default: Use the peel-off technique
<acceptance_angle> AC_ANGLE	Set the acceptance angle, if peel-off is not used
(DUST_SCATTERING)	Default: 1°

Table 2.9: Available gas commands and their default values for the POLARIS command files. The red names in brackets show for which simulation type (pipeline ID) the command can be used.

Command	Description
	Gas
`<mu> MU`	Average atomic mass unit per gas particle
(ALL)	Default: 2.0
`<vel_maps> 1` (yes) or `0` (no)	Enables/Disables the creation of velocity channel maps
(LINE_EMISSION)	Default: No velocity channel map will be created
`<vel_is_speed_of_sound> 1` (yes) or `0` (no)	Enables/Disables the interpretation of the velocity of the grid as multiple of the speed of sound
	Default: The velocity field of the grid is considered to have SI units
`<kepler_star_mass> MASS_OF_CENTRAL_STAR`	Set the mass of the central star and enable the use of Keplerian rotation as the velocity field
(LINE_EMISSION)	Default: Use the velocity field of the grid
`<turbulent_velocity> V_TURBULENT`	Set the turbulent velocity of the gas phase to add extra Doppler broadening
(LINE_EMISSION)	Default: $0\,\mathrm{m\,s^{-1}}$

Table 2.10: Available visualization commands and their default values for the POLARIS command files. The red names in brackets show for which simulation type (pipeline ID) the command can be used.

Command	Description
Visualization	
`<write_inp_midplanes> INP_MIDPLANE_PIXEL`	Enables the creation of midplane `'.fits'` files (input grid, see Sect. 4.2.1)
(ALL)	Default: No midplane files
`<write_out_midplanes> OUT_MIDPLANE_PIXEL`	Enables the creation of midplane `'.fits'` files (output grid, see Sect. 4.2.1)
(ALL)	Default: No midplane files
`<write_3d_midplanes> PLANE` or `PLANE NR_SLICES` or `PLANE NR_SLICES Z_MIN, Z_MAX`	Enables the creation of 3D midplane files (see Sect. 4.2.1). `PLANE` = 1, 2, or 3 for xy, xz, or yz plane.
(ALL)	Default: No 3D midplane files
`<write_radiation_field>` ID	Enables the inclusion of the radiation field into the midplane `'.fits'` files (1: u_{rad}, 2: J, 3: $\vec{J}$; see Sect. 4.2.1)
(ALL)	Default: No radiation field included
`<write_g_zero>` 1 (yes) or 0 (no)	Enables/Disables the inclusion of the Mathis field G_0 value into the midplane `'.fits'` files (see Sect. 4.2.1; Mathis et al. 1983; Camps et al. 2015)
(ALL)	Default: No G_0 value included
`<midplane_zoom> ZOOM_FACTOR`	Set the zoom onto the midplane cut images
(ALL)	Default: No midplane zoom
`<write_dust_files>` 1 (yes) or 0 (no)	Enables/Disables the creation of Gnuplot files to visualize the dust properties
(ALL)	Default: No Gnuplot files
`<nr_plot_points> GNU_POINTS`	Enables/Disables the creation of Gnuplot files to visualize the grid (how many scalar points, see Sect. 4.3)
(ALL)	Default: No Gnuplot files
`<nr_plot_vectors> GNU_VECTORS`	Enables/Disables the creation of Gnuplot files to visualize the grid (how many vectors, see Sect. 4.3)
(ALL)	Default: No Gnuplot files
`<max_plot_lines> GNU_LINES`	Enables/Disables the creation of Gnuplot files to visualize the grid (how many lines points, see Sect. 4.3)
(ALL)	Default: No Gnuplot files
`<amira_inp_points> INP_AMIRA_POINTS`	Enables/Disables the creation of AMIRA files (input grid, see Sect. 4.4)
(ALL)	Default: No AMIRA files
`<amira_out_points> OUT_AMIRA_POINTS`	Enables/Disables the creation of AMIRA files (output grid, see Sect. 4.4)
(ALL)	Default: No AMIRA files

3 POLARIS pipeline

The POLARIS code works in independent distinct simulation modes. The choice of modes depends on the scientific interest of the user. All modes can be combined into a single command file. In this section we describe the different POLARIS modes and outline the underlying physical principles of dust and molecular line, and synchrotron RT, respectively, as well as the implemented numerical methods.

3.1 Photon propagation

3.1.1 The Stokes vector

The polarization state of radiation along its path can be quantified by the four-component Stokes vector $\vec{S} = (I, Q, U, V)^{\mathsf{T}}$ where the parameter I is the total intensity, Q and U describe the state of linear polarization, and V is for circular polarization. A bundle of polarized light can be projected on two different coordinate systems (see Fig. 3.1). Additionally, with the decomposition of polarized radiation in two clockwise and counterclockwise rotating waves gives the components the Stokes vector defined by

$$\vec{S} = \begin{pmatrix} I \\ Q \\ U \\ V \end{pmatrix} = \begin{pmatrix} |E_{\mathrm{x}}|^2 + |E_{\mathrm{y}}|^2 \\ |E_{\mathrm{x}}|^2 - |E_{\mathrm{y}}|^2 \\ |E_{45}|^2 - |E_{-45}|^2 \\ |E_{\mathrm{cw}}|^2 - |E_{\mathrm{ccw}}|^2 \end{pmatrix}. \tag{3.1}$$

The degree of polarizazion is defined as the ratio of polarized flux to total flux. Thus, according to the Stokes formalism, the degree of total polarization P, degree of linear polarization P_{l}, and degree of circular polarization P_{c} are determined by

$$P = \frac{\sqrt{U^2 + Q^2 + V^2}}{I} \in [0, 1], \tag{3.2}$$

$$P_{\mathrm{l}} = \frac{\sqrt{U^2 + Q^2}}{I} \in [0, 1], \tag{3.3}$$

$$P_{\mathrm{c}} = \frac{V}{I} \in [-1, 1], \tag{3.4}$$

and the position angle χ as observed on the plane of the sky is

$$\tan(2\chi) = \frac{U}{Q}. \tag{3.5}$$

$P_{\mathrm{c}} > 0$ stands for circular polarization with the electric field vector rotating counter clockwise and $P_{\mathrm{c}} < 0$ stands a rotation clockwise towards the observer Additionally, each photon package in POLARIS keeps also track about the optical depth and column density along its path.

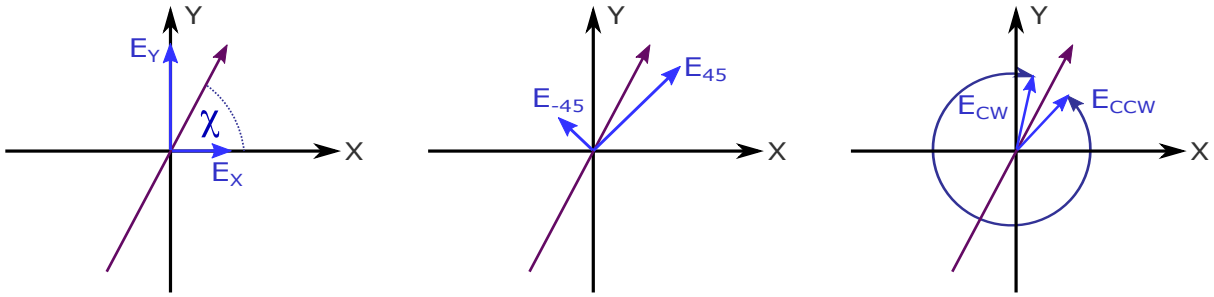


Figure 3.1: Representation of the same electric field vector in three different coordinate systems. Here, the quantity χ is the orientation angle as observed on the plane of the sky.

3.1.2 General radiative transfer equation

The POLARIS code solves the RT equation in all four Stokes parameters simultaneously. This problem can be expressed as

$$\frac{\mathrm{d}}{\mathrm{d}\ell}\vec{S} = -\hat{R}(\alpha)\hat{K}\hat{R}^{-1}(\alpha)\vec{S} + \vec{J}. \tag{3.6}$$

Here, $\hat{R}(\alpha)$ is a rotation matrix, and $\vec{J}$ is the contribution due to emission, and $\mathrm{d}\ell$ is the infinitesimal path length element. The quantity $\hat{K}$ is the Müller matrix describing the extinction, absorption, and scattering, respectively. Both $\hat{K}$ as well as $\vec{J}$ are defined by the characteristic physics of radiation interacting with gas species, free electrons, or dust, respectively. In the most general form the RT problem can be written as (e.g., Martin 1974; Whitney & Wolff 2002)

$$\frac{\mathrm{d}}{\mathrm{d}\ell}\begin{pmatrix} I \\ Q \\ U \\ V \end{pmatrix} = -\begin{pmatrix} \alpha_\mathrm{I} & \alpha_\mathrm{Q} & \alpha_\mathrm{U} & \alpha_\mathrm{V} \\ \alpha_\mathrm{Q} & \alpha_\mathrm{I} & \kappa_\mathrm{V} & \kappa_\mathrm{U} \\ \alpha_\mathrm{U} & -\kappa_\mathrm{V} & \alpha_\mathrm{I} & \kappa_\mathrm{Q} \\ \alpha_\mathrm{V} & -\kappa_\mathrm{U} & -\kappa_\mathrm{Q} & \alpha_\mathrm{I} \end{pmatrix}\begin{pmatrix} I \\ Q \\ U \\ V \end{pmatrix} + \begin{pmatrix} j_\mathrm{I} \\ j_\mathrm{Q} \\ j_\mathrm{U} \\ j_\mathrm{V} \end{pmatrix}. \tag{3.7}$$

Refer to Sect. 3.7.5 for the definition of the extinction and emission coefficients. Dependent on the physical problem some of the coefficients can be eliminated by rotating the polarized radiation from the lab frame into the target frame meaning the frame of the magnetic field or the symmetry axis of the dust grain, respectively. From the definition of the Stokes vector follows for the rotation:

$$\hat{R}(\alpha) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(2\alpha) & -\sin(2\alpha) & 0 \\ 0 & \sin(2\alpha) & \cos(2\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{3.8}$$

Implemented in the POLARIS is the Runge-Kutta Fehlberg (*RFK45*) solver in order to provide a high accuracy solution to the RT problem. This method uses an inbuilt step size ℓ correction to keep the error within below a threshold ϵ_err by comparing the forth order Runge Kutta approximation

$$y_{k+1} = y_k + \frac{25}{216}k_1 + \frac{1408}{2565}k_3 + \frac{2197}{4101}k_4 - \frac{1}{5}k_5 \tag{3.9}$$

with the fifth order solution

$$\hat{y}_{k+1} = y_k + \frac{25}{216}k_1 + \frac{1408}{2565}k_3 + \frac{2197}{4101}k_4 - \frac{1}{5}k_5 \tag{3.10}$$

Here, the factors are defined by

$$k_1 = \ell f(t_k, y_k), \tag{3.11}$$

$$k_2 = \ell f(t_k + \frac{1}{4}\ell, y_k + \frac{1}{4}k_1), \tag{3.12}$$

$$k_3 = \ell f(t_k + \frac{3}{8}\ell, y_k + \frac{3}{32}k_1 + \frac{9}{32}k_2), \tag{3.13}$$

$$k_4 = \ell f(t_k + \frac{12}{13}\ell, y_k + \frac{1932}{2197}k_1 - \frac{7200}{2197}k_2 + \frac{7296}{2197}k_3), \tag{3.14}$$

$$k_5 = \ell f(t_k + \ell, y_k + \frac{439}{216}k_1 + 8k_2 - \frac{3680}{513}k_3 - \frac{854}{4104}k_4), \tag{3.15}$$

$$k_6 = \ell f(t_k + \frac{1}{2}\ell, y_k + \frac{8}{27}k_1 + 2k_2 - \frac{3544}{2565}k_3 + \frac{1859}{4104}k_4 - \frac{11}{40}k_5). \tag{3.16}$$

Here, $f(t_k, y_k)$ represents the each of the Stokes component of Eq. (3.7). Finally, the total error is calculated with:

$$\epsilon = \frac{|\hat{y}_{k+1} - y_{k+1}|}{\epsilon_{\text{err}}\,|\hat{y}_{k+1} + \epsilon_{\text{abs}}|}. \tag{3.17}$$

We calculate the error for each of the Stokes parameters separately and select then the largest one in order to avoid overshoots in the *RFK45* solver. If $\epsilon > 0$ a smaller step size ℓ_{new} for the next integration needs to be determined according to

$$\ell_{\text{new}} = \min\left(0.9\ell_{\text{old}}\epsilon^{-0.2};\ 0.25\ell_{\text{old}}\right). \tag{3.18}$$

Otherwise, in the case of $\epsilon \leq 0$ the integration step is within the demanded error limit and the equation system of RT is considered to be solved. Dependent on the chosen POLARIS RT simulation mode this process stops when a photon package leaks from the grid or hits a detector, respectively. In POLARIS the errors are set to be $\epsilon_{\text{err}} = 10^{-6}$ and $\epsilon_{\text{abs}} = 10^{-30}$ by default. As a final fail save cells are completely skipped when no solution is found within a maximal number of 1.5×10^6 solver steps to ensure for POLARIS to terminate in any case. The errors and maximal step size are defined in the file src/Typedefs.h. If you want to perform simulations in more extreme environments (e.g. high densities) you may change the values of `rel_err`, `abs_err`, and `MAX_SOLVER_STEPS` to optimize the balance between accuracy and simulation run-time for you own purposes.

3.1.3 Monte-Carlo (MC) photon transfer

This MC photon transfer scheme is applied in the grain alignment radius mode (`CMD_RAT`, see Sect. 3.7.4), the dust heating mode (`CMD_TEMP`, see Sect. 3.7.2), and in dust polarization runs with scattering (`CMD_DUST_SCATTERING`, see Sect. 3.7.5). The MC photon transfer in POLARIS works with photon packages instead of single photon in order to make the propagation of light more efficient. Each photon package starts at a source (see Sect. 3.2) with a direction, energy per unit time, polarization state (Stokes vector) and one distinct wavelength. The probability of radiation-dust interaction is determined by the path length through the ith cell ℓ, the cross section of extinction C_{ext}, and the dust number density n_{d} of the dust in each cell. A random number $z \in [0, 1)$ is chosen to sample an optical depth from the statistical function

$$\tau_{\text{st}} = -\ln(1 - z). \tag{3.19}$$

In general the path length ℓ_i is defined to be between the two cell walls where the photon package enters and leaves, respectively, the i-th cell. The total optical depth of the photon package accumulates along all the paths ℓ_i through each i-th cells is:

$$\tau_\mathrm{ext} = \sum_{i=1}^{N} C_{\mathrm{ext},i} n_{\mathrm{d},i} \ell_\mathrm{i}, \tag{3.20}$$

where $C_{\mathrm{ext},i}$ and $n_{\mathrm{d},i}$ are the local dust extinction cross section and the dust number density, respectively.

A photon package interacts with the dust when the condition

$$\tau_\mathrm{ext} < \tau_\mathrm{st} \tag{3.21}$$

is fulfilled in a particular cell. In this case the position of the photon package and, subsequently, the path length within that cell is adjusted to ensure $\tau_\mathrm{ext} = \tau_\mathrm{st}$. The nature of the interaction itself can be either scattering, or absorption and re-emission. The wavelength-specific dust grain albedo determines the probability of scattering,

$$\varpi(\lambda) = \frac{C_{\mathrm{sca},\lambda}}{C_{\mathrm{abs},\lambda} + C_{\mathrm{sca},\lambda}} \in [0, 1]. \tag{3.22}$$

Here, $C_{\mathrm{abs},\lambda}$ and $C_{\mathrm{sca},\lambda}$ are the dust cross sections of absorption and scattering, respectively, and λ is the wavelength of the photon package. Note, that the cross sections are also dependent on the grain radius a. In the case of $z > \varpi$ the photon package scatters on the dust grain. The scattering event changes the direction of the photon package by an angle of ψ, dependent on the characteristic phase function $\Phi(\psi)$ (see also Sect. 3.7.1). In the case of $z < \varpi$ the photon package gets absorbed and instantaneously re-emitted by the dust grain in order to sustain LTE. Here, the direction of the thermal radiation is isotropic but the wavelength changes after re-emission. Assuming that the dust grain radiates like a black body, the new wavelength has to be sampled from a modified black body spectrum. The sampling function depends on the simulation mode. In order ensure the correct remission for dust grains heated by radiation the sampling function is

$$p(\lambda, T_\mathrm{d}) = \frac{\int_\lambda^{\lambda+\mathrm{d}\lambda} C_{\mathrm{abs},\lambda} \frac{\mathrm{d}B_\lambda(T_\mathrm{d})}{\mathrm{d}T} \mathrm{d}\lambda}{\int_{\lambda_\mathrm{min}}^{\lambda_\mathrm{max}} C_{\mathrm{abs},\lambda} \frac{\mathrm{d}B_\lambda(T_\mathrm{d})}{\mathrm{d}T} \mathrm{d}\lambda} \tag{3.23}$$

with $B_\lambda(T_\mathrm{d})$ being the Planck function (see Bjorkman & Wood 2001). In the mode for determining the alignment radius, the dust temperature remains constant and the new wavelength is sampled from

$$p(\lambda, T_\mathrm{d}) = \frac{\int_\lambda^{\lambda+\mathrm{d}\lambda} C_{\mathrm{abs},\lambda} B_\lambda(T_\mathrm{d}) \mathrm{d}\lambda}{\int_0^\infty C_{\mathrm{abs},\lambda} B_\lambda(T_\mathrm{d}) \mathrm{d}\lambda}, \tag{3.24}$$

The mode for polarized light due to dust scattering is monochromatic meaning without any absorption and re-emission at all.

3.1.4 MC noise estimation

The modes of dust grain heating (see Sect. 3.7.2), polarization by scattering (see Sect. 3.7.5), and the calculation of the radiation field (see Sect. 3.7.4) are based on the MC method, where photon packages propagate on probabilistic path was through the grid. The POLARIS modes making use of the MC method are based on quantities sampled by random numbers from physically motivated probability distribution functions. Consequently, the results are prone to noise that scales with the number N_ph of considered photon packages. The noise in MC simulations can be quantified by $\propto 1/\sqrt{N_\mathrm{ph}}$. Hence, the reduction of noise scales not linearly with the number of applied photon packages. See Sect. 4.2.4 for an estimation of the error of Monte-Carlo simulations.

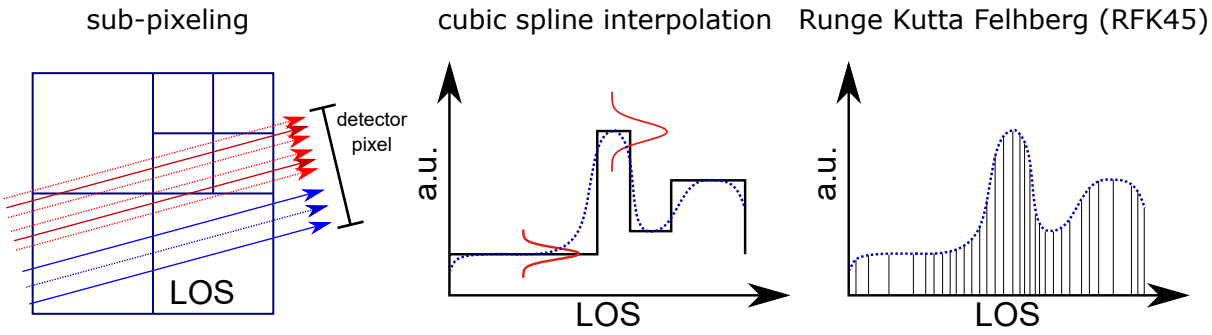


Figure 3.2: Solving the RT problem by ray-tracing follows the steps of sub-pixeling (see Sect. 3.3.1), spline interpolation, and subsequent integration applying the *RFK45* method.

3.1.5 Ray-tracing

For the dust emission and LRT it is more sufficient to trace single rays through the grid towards the observer instead of performing a full MC RT simulation and is applied in the POLARIS modes of dust polarization by thermal re-emission (`CMD_DUST_EMISSION`) and LRT (`CMD_LINE_EMISSION`). A way to quantify the range of wavelength where ray-tracing is applicable is by the albedo of the dust grain (see Eq. 3.22). As a rule of thumb scattering becomes irrelevant for $\varpi < 0.1 - 0.01$ and ray-tracing can be applied. Using ray-tracing comes with the advantage of a reduced simulation run time and an excellent SNR. Both, dust polarization and LRT require to solve the RT equations in the stokes vector formalism (see also Sect. 3.1.1). In order to reduce artifacts caused by grid geometry, we interpolate values with a cubic spline interpolation. This approach is especially advisable to avoid special problems concerning LRT (see Sect. 3.8).

A POLARIS simulation run considering ray-tracing and different alignment mechanisms and detector types may look like Listing 3.1. The emission from radiation scattered at dust grains can be considered in the ray-tracing technique if the radiation field was saved in `CMD_TEMP` simulations (see Sect. 3.7.2). If this was not the case, the radiation field can be calculated right before the `CMD_DUST_EMISSION` simulation, by defining a combination of stellar and/or dust sources in the cmd file. However, this approach to consider scattered light is only physically sufficient if the radiation field is dominated by radiation coming from a single direction and the regions dominating the scattered light have to be optically thin (usually working for circumstellar disks). Use the command `<rt_scattering> 0`, if the scattered light should not be considered, but the radiation field is stored in the grid.

3.2 Photon emitting sources

In a RT simulation a photon package starts its life cycle with a characteristic direction, energy per unit time, wavelength, and an initial degree of linear and circular polarization mimicking a certain type of photon emitting sources. In order to cover the broad variety of radiation emitting sources in astrophysics POLARIS provides several classes of sources.

Star: This source takes take no possible irregularities associated with the surface of the star such as sun spots or limb darkening into consideration. Hence, a star is considered to be a simple point source. The starting direction of each photon package is random. The distribution of wavelength is either sampled from

$$z = \frac{\int_{\lambda_i}^{\lambda_i+\Delta\lambda} L_\lambda \mathrm{d}\lambda}{\int_0^\infty L_\lambda \mathrm{d}\lambda} \tag{3.25}$$

Listing 3.1: Example command file to calculate the polarized dust emission.

```
<task> 1
#polarization by aligned dust
<cmd> CMD_DUST_EMISSION

#a star in the center as radiation source
#used for radiation field calculation if not done with CMD_TEMP
<source_star nr_photons = "1e6"> 0 0 0 1 6000

#the thermal emission of the dust grains only used
#for radiation field calculation if not done with CMD_TEMP
<source_dust nr_photons = "1e4">

#silicate as only dust components
<dust_component> "PATH/TO/POLARIS/input/dust/silicate_oblate.dat" 0.7 -3.5 24.2e-9
    ↪ 1.13e-6
<dust_component> "PATH/TO/POLARIS/input/dust/graphite_oblate.dat" 0.3 -3.5 24.2e-9
    ↪ 1.13e-6

#path of the input grid file
<path_grid> "PATH/WHERE/THE/RESULTS/SHOULD/BE/SAVED/rat/grid_rat.dat"

#oath for all output data
<path_out> "PATH/WHERE/THE/RESULTS/SHOULD/BE/SAVED/dust_polarization/"

#considered alignment mechanisms
<align> ALIG_IDG
<align> ALIG_RAT
<align> ALIG_INTERNAL

#additional alignment parameters
<f_c> 0.6 #correlation
<f_highJ> 0.5 #ration of grains at highJ attractor point

#definition of the 1st rotation axis (e.g. x-axis)
<axis1> 1 0 0

#definition of plane detectors (e.g. rotation arround x-axis)
<detector_dust nr_pixel = "320"> 93 96 1 1 0 0 3.1e18 1
<detector_dust nr_pixel = "320"> 93 96 1 1 45 0 3.1e18 1
<detector_dust nr_pixel = "320"> 93 96 1 1 90 0 3.1e18 1

#definition of spherical detectors (e.g. different pos.)
<detector_dust_healpix nr_sides = "256"> 96 96 1 1 1.65e+16 0 0 -80 80 -45 45
<detector_dust_healpix nr_sides = "256"> 96 96 1 1 1.65e+16 0 0 -80 80 -45 45

#uses 32 cores
<nr_threads> 32

#dust to gas mass ratio
<mass_fraction> 0.01 # optional

#stochastic heating for grains with a radius of up to 50nm
<stochastic_heating> 5e-8
</task>
```

where $z \in [0, 1)$ is a random number or the emission is calculated for `WL_STEPS` wavelengths between `WL_MIN` and `WL_MAX` as defined src/Typedefs.h (Default: $\lambda \in [0.1 \times 10^{-6}\,\mathrm{m},\ 2000 \times 10^{-6}\,\mathrm{m}]$). The second option is used when calculating RAT alignment radii or if the radiation field has to be calculated. In the first case, each newly created photon package carries an energy per unit time of

$$\dot{E} = \frac{L}{N_{\mathrm{ph}}} \tag{3.26}$$

away, where N_{ph} is the total number of photon packages. In the second case, each newly created photon package carries an energy per unit time of

$$\dot{E} = \pi 4 \pi R_{\odot}^2 \frac{B_{\lambda}(T)}{N_{\mathrm{ph}}} \tag{3.27}$$

away, where N_{ph} is the number of photon packages (emitted per wavelength) and the parameter q and u are modifying the emission by setting $Q = qI$ and $U = uI$. A star can be defined by the following commands:

<source_star nr_photons = "N_{ph}"> x [m] y [m] z [m] R_* [$\mathrm{R}_{\odot}$] T_* [K] q u

Alternatively, the SED of the star can be defined by an external file with :

<source_star nr_photons = "N_{ph}"> x [m] y [m] z [m] "path"

Examples:

```
<source_star nr_photons = "5e6"> 1.15e2 4.16e2 1.8e1 12 17000
<source_star nr_photons = "8e9"> 2.32e9 19.9e7 1.3e8 "/PATH/TO/THE/STELLAR/sed.dat"
```

Starfield: This diffuse source is intended to mimic a spacial extended source such as a star field located in a region small in comparison to the entire model space This source is similar to the star source. However, the photon packages start not from a single point. The starting point of the emitted photon packages follow a 3D Gaussian distribution meaning that the probability to start at a position toward the center of the sphere is higher. A star can be defined by the commands

<source_starfield nr_photons = "N_{ph}"> x [m] y [m] z [m] R_* [$\mathrm{R}_{\odot}$] T_* [K] μ_{f} [m]

or

<source_starfield nr_photons = "N_{ph}"> x [m] y [m] z [m] μ_{f} [m] "path"

where μ_{f} is the variance of the 3D Gaussian distribution.

Examples:

```
<source_starfield nr_photons = "5e6"> 1.15e2 4.16e2 1.8e1 12 23e4 1e18
<source_starfield nr_photons = "8e9"> 2.32e9 19.9e7 1.3e8 1e18 "/PATH/TO/THE/STELLAR/
    ↪ spec.dat"
```

ISRF: The ISRF contributes also to the overall radiation field inside the model space and is therefore of relevance for the calculation of the RAT alignment efficiency and the dust temperature. As for the point source the ISRF source sends photon packages in random direction. However, the starting point is from the surface of a sphere with a radius from the center to the outermost edges of the grid (or times a factor f_{radius}). The ISRF emission spectrum can be defined by an external SED file or by providing the G_0 value for the Mathis ISRF (for the Mathis ISRF, see Mathis et al. 1983; Camps et al. 2015). The emission of the surface has to obey Lambert's cosine law. Thus, the direction of the photon package is calculated from $z = \cos^2(\theta)$, where z is a random number. However, for increasing radius, most photon packages do not interact with the model space. For this reason, the direction of the emission is biased and the energy of the photon package is weighted

accordingly. The new direction is based on $\cos^n(\theta)$, so $z = \cos^{n+1}(\theta)$, and the weight is $\cos^{1-n}(\theta)$. By default, $n = 50 f_{\text{radius}}$.

This type of source can only be defined once per simulation. The ISRF source can be defined by the commands:

<source_isrf nr_photons = "N_{ph}"> "path"

or

<source_isrf nr_photons = "N_{ph}"> "path" f_{radius}

or

<source_isrf nr_photons = "N_{ph}"> G_0

or

<source_isrf nr_photons = "N_{ph}"> G_0 f_{radius}

Example:

```
<source_isrf nr_photons = "1e7">
    ↪ "/PATH/TO/POLARIS/input/interstellar_radiation_field.dat"
<source_isrf nr_photons = "1e7">
    ↪ "/PATH/TO/POLARIS/input/interstellar_radiation_field.dat" 3
<source_isrf nr_photons = "1e7"> 1
<source_isrf nr_photons = "1e7"> 1 3
```

Background: The background source is designed to simulate a collection of objects outside of the grid. The background source is defined by a pixel matrix where each pixel has its characteristic SED, intensity, and state of linear and circular polarization. This type of source defines also the starting values of the stokes vector for ray-tracing and LRT simulations. All POLARIS simulations can contain an arbitrary number of distinct background sources. The background source is defined by an set of parameters or an external file containing a pixel matrix:

<source_background nr_photons = "N_{ph}"> f [$a.u.$] T_{eff} [K] q u v α_1 [°] α_2 [°]
<source_background nr_photons = "N_{ph}"> "path" α_1 [°] α_2 [°]

In the case of an constant background source the intensity follows $I_\lambda = f \times B_\lambda(T_{\text{eff}})$ with a a scaling factor f and an effective temperature T_{eff}, where as the quantities $q = Q/I$, $u = U/I$,and $v = V/I$ are the Stokes parameters normalized by intensity. The angles α_1 [°] and α_2 [°] are the rotation angles around the user defined axis (see Sect. 3.4).

Examples:

```
<source_background nr_photons = "1e5"> 1e18 30000 1 0 0 90 45
<source_background nr_photons = "1e5"> "/PATH/TO/THE/background_source.dat" 90 45
```

Dust: Since the dust grains have a certain temperature, they contribute also to the radiation field in the model space (available for `CMD_DUST_SCATTERING` simulations to consider self-scattering and `CMD_DUST_EMISSION` simulations to calculate the radiation field). The dust containing cells are considered as separate photon package emitting sources and the individual cell i is chosen for the emission at a certain wavelength λ according to the following equation:

$$z = \frac{L_{i,\lambda}}{\sum_{i=0}^{N_{\text{cells}}} L_{i,\lambda}} \tag{3.28}$$

where $z \in [0, 1)$ is a random number and L_i is defined as follows:

$$L_{i,\lambda} = 4\pi N_{i,\text{d}} C_{\text{abs},\lambda} B_\lambda(T_{i,\text{d}}) \tag{3.29}$$

The photon package starts from a random position evenly distributed in the entire cell volume. Since the photons packages are sampled over all cells via the total dust emission, the energy per unit time and wavelength of an individual photon package is determined by the dust properties with a modified black body spectrum as follows:

$$\dot{E}_\lambda = \frac{4\pi}{N_{\text{ph}}} \sum_{i=0}^{N_{\text{cells}}} N_{i,\text{d}} C_{\text{abs},\lambda} B_\lambda(T_{i,\text{d}}) \tag{3.30}$$

where $N_{i,\text{d}}$ is the number of dust grains in the cell i. The dust as a source of radiation can simply switched on and off by writing the tag

<source_dust nr_photons = "N_{ph}">

In the command file, N_{ph} defines the number of photon packages to be emitted from each cell.

Example:

```
<source_dust nr_photons = "1e6">
```

Laser: The laser radiation source is a very narrow band emission source with a normal distribution as the emission profile. It is designed for testing models or applications in other fields of physics (like plasma physics). It is available for `CMD_DUST_EMISSION` and `CMD_DUST_SCATTERING` simulations. The emitted flux at a wavelength λ can be calculated with the following equation:

$$F_\lambda = \frac{P}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(\lambda-\lambda_0)^2}{2\sigma^2}\right), \tag{3.31}$$

where P is the total power of the laser (in Watts), λ_0 is the central wavelength, and σ is the standard deviation that is calculated from the FWHM with the following equation:

$$\sigma = \frac{\text{FWHM}}{\sqrt{8\ln(2)}}. \tag{3.32}$$

The laser is located at the position x, y, z and points in the direction d_x, d_y, d_z. The laser as a source of radiation can simply switched on and off by writing the tag

<source_laser nr_photons = "N_{ph}"> x [m] y [m] z [m] d_x [m] d_y [m] d_z [m] P [W] λ_0 [m] FWHM [m] q u

In the command file, N_{ph} defines the number of photon packages to be emitted. The parameter q and u are modifying the emission by setting $Q = qI$ and $U = uI$.

Example:

```
<source_laser nr_photons = "1e6"> 0 0 0 0 0 0 1e-3 1e-6 1e-9 0 0
```

HINTS:

- The external file for the SED is a simple table in the form
  `#λ [m]` L_λ `[W/m]`
  ... ...
  ... ...
  ... ...
- The external file for the matrix of the background source is a simple table in the form
  `#λ [m]` L_λ `[W/m] q u v`
  ... ...
  ... ...
  ... ...

- A background source is required for any RT simulation based on ray-tracing. If no background source was defined in the command file a default source will automatically be created with all parameters set to zero.
- The star source can also be used to add the direct attenuated emission of a star to a ray-tracing simulation (e.g. if the scattered light is considered in ray-tracing simulations via the radiation field).
- The range of wavelengths has to be equal or larger than the wavelength range defined in the src/Typedefs.h (Default: $\lambda \in [0.1 \times 10^{-6}\,\mathrm{m},\ 2000 \times 10^{-6}\,\mathrm{m}]$). This range can be adjusted by changing `WL_MIN`, `WL_MAX`, and `WL_STEPS` in the src/Typedefs.h file (use `./compile.sh -u` afterwards). Intermediate values are calculated by means of cubic spline interpolation.

3.3 Optimization techniques

A 3D MC RT simulation is challenging concerning run time and the capabilities of available computational equipment. For this reason POLARIS is equipped with several optimization algorithms to perform RT calculations on a reasonable time and a high SNR.

3.3.1 Sub-pixeling

Using only one ray per detector pixel one may lose information of sub-structures in the model smaller than the size of of the bin (see Fig. 3.2). In POLARIS each detector pixel starts with four rays as a first step. Here, POLARIS keeps track of the exact cells the ray passes. If two neighboring rays pass exactly the same cells the rays are combined to a single ray. In the opposite case, when neighboring rays pass different cells each ray is again split into four sub-rays. Since no RT equations are solved in this step, the splitting of rays comes with almost no additional time. Once, the optimal number of rays per pixel is known POLARIS starts with the ray-tracing along each ray. However, allowing POLARIS to split rays down to the smallest sizes can result in an enormous slow down for the actual ray-tracing since the RT problem needs to be solve for all the rays separately. Hence, it is recommended to limit the maximal level of splitting. The command is: `<max_subpixel_lvl> N_split`

Hence, each detector pixel gets hit by a maximal number of $4^{N_{\mathrm{split}}}$ rays. When the sub-pixel level is not set the level is 3 by default.

Example:

```
<max_subpixel_lvl> 3
```

HINTS:

- The polar detector's sub-pixeling is *not* adaptive but relies on creating a sufficiently resolved polar background grid based on the spherical or cylindrical grid cells resolution. The resolution is thereby chosen as

$$N_{r,\mathrm{sub}} = 2^{N_{\mathrm{split}}} \frac{r_{i+1} - r_i}{\text{pixel width}} \tag{3.33}$$

with N_{sub} denoting the number of radial rings between r_{i+1} and r_i of the polar background grid. The number of φ-cells per ring is chosen subsequently depending on the subpixeled radii:

$$N_{\varphi,\mathrm{sub}}\left(r_{i,\mathrm{sub}}\right) = 2\pi \frac{r_{i+1,\mathrm{sub}}}{r_{i+1,\mathrm{sub}} - r_{i,\mathrm{sub}}} \tag{3.34}$$

For more details on the polar detector, see Sect. 3.5.

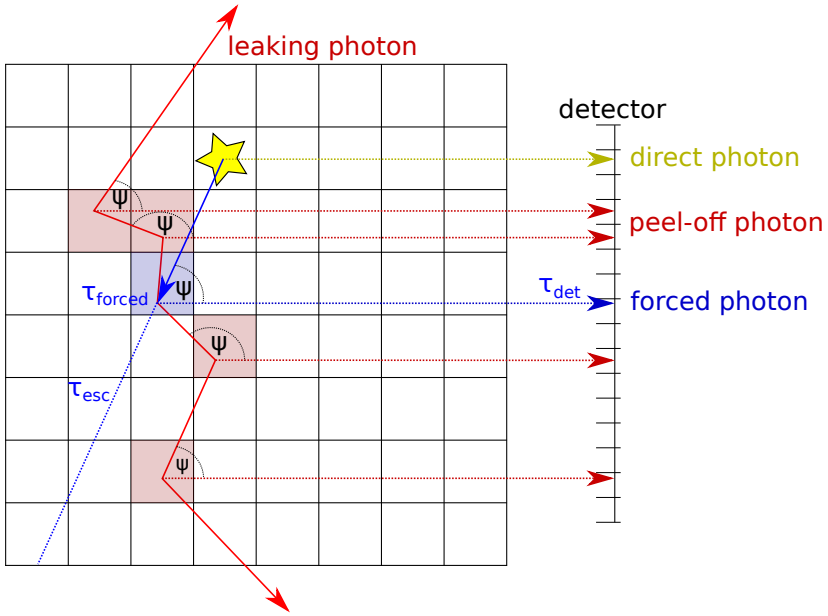


Figure 3.3: Sketch of the implemented forced first scattering scheme and the peel-off technique.

3.3.2 Enforced first scattering

In optically thin regions an interaction between dust and radiation is highly unlikely and a significantly high amount of photon packages can escape the grid without any interaction at all. In POLARIS the photon package can be forced to interact at least once with the dust according to the method presented in Cashwell & Everett (1957) in order to increase the SNR. For each newly created photon package, an optical depth $\tau_{\rm esc}$ is calculated from the starting point to the boundaries of the grid along the current direction. A new optical depth $\tau_{\rm forced}$ to the first interaction point is then randomly selected from

$$\tau_{\rm forced} = -\ln[1 - z(1 - f_{\rm e})] \tag{3.35}$$

with a random number z and the factor $f_{\rm e} = \exp(-\tau_{\rm esc})$. In order to compensate for the forced interaction the photon package p_0 has to be split in two packages p_1 and p_2 with their energy adjusted by $E_1 = (1 - f_{\rm e})E_0$ and $E_2 = f_{\rm e}E_0$ at the forced interaction point. From this point on each photon package follows independently the usual propagation scheme (see Sect. 3.1.3 and also Fig. 3.3) until it reaches to the boundaries of the grid or a detector. The command that enables the forced first scattering technique is `<enfsca> 1`. It can only be used for simulations with `CMD_DUST_SCATTERING`.

Example:

```
<enfsca> 1 # forced first scattering will be applied
```

3.3.3 Peel-off technique

Most of the photon packages do not leak from the grid with directions towards a detector and a huge amount of information is lost. This results also in a bad signal-to noise ratio. In order to overcome this weakness POLARIS makes use of the peel-off technique (see Yusef-Zadeh et al. 1984). Here, at each scattering point a fraction of light is also scattered in direction of the detector. The energy of the peel-off photon package $E_{\rm po}$ has to be adjusted by

$$E_{\rm po} = E_{\rm pp}\Phi(\psi)\exp(-\tau_{\rm det}) \tag{3.36}$$

where the probability of scattering in direction of the detector is determined by the phase function $\Phi(\psi)$ (see Sect. 3.7.1 and Fig. 3.3) and the angle ψ is between original direction and detector direction. The optical depth τ_{det} is integrated along the line of sight between the position of the scattering event and the detector position. Although, the optical depth towards the observer has to be calculated for each scattering event, the peel-off technique results in synthetic images with an excellent signal-to-noise ratio even with a reduced number of photon packages. The command that enables the peel-off technique is `<peel_off> 1`. It can only be used for simulations with `CMD_DUST_SCATTERING`.

Example:

```
<peel_off> 1 # peel-off technique will be applied
```

3.3.4 Wavelength range selection

Stars and star fields emit photons in the full wavelength range of its black body SED which depends on its characteristic temperature. However, wavelengths at both ends of the SED can be neglected since they do not contribute significantly to the total emitted energy of the source. This reduces the number of wavelength to be considered and subsequently the computational efforts. POLARIS adjusts the SED of any source automatically so that maximal 10^{-6} % of the total luminosity gets lost. This technique is standard in POLARIS and can not be switched off. However, if the radiation field has to be saved in the grid by using `<radiation_field> 1`, each wavelength will be used and no wavelength will be neglected.

3.4 Grid rotation

The grid is by default orientated in an external coordinate system with the xy-plane towards the observer and the z-axis along the LOS. In order to re-orientate the grid for dust scattering, dust ray-tracing, or line RT two arbitrary rotation axis can be defined.

```
<axis1> x [a.u.] y [a.u.] z [a.u.]
<axis2> x [a.u.] y [a.u.] z [a.u.]
```

By default the first rotation axis the x-axis (100) and the second rotation axis is along the y-axis (001).

Examples:

```
<axis1> 1 0 0 # defining the x axis as first rotation axis
<axis2> 1 1 1 # defining a diagonal direction as second rotation axis
```

Later, these rotation axis define the orientation of the grid with respect to the detector (see Sect. 3.5).

HINTS:

- The vectors defining the two axis will automatically be normalized.
- The rotations are performed subsequently. This means that the second axis is defined in the frame *after* the first rotation. In practice it is therefore expedient to perform the position angle rotation first and the inclination rotation second. Alternatively, the rotation axis for inclination can be defined along the projected major axis of the disk.

3.5 Detector parameters

The POLARIS code provides two main classes of detectors. The plane detector and the spherical (healpix) detector. The plane detector should be used when the observer is far away from the grid and the rays can be considered to be parallel from each other. This kind of detector is available for MC dust scattering (`CMD_DUST_SCATTERING`), ray-tracing for thermal emission of dust grains (`CMD_DUST_EMISSION`), LRT (`CMD_LINE_EMISSION`), and synchrotron RT (`CMD_SYNCHROTRON`). A plane detector is characterized by its distance, viewing angles, wavelength, and the number of pixel. The exact command for a detector depends on the kind of POLARIS simulation mode.

Dust scattering: For observations considering dust scattering (`CMD_DUST_SCATTERING`) a plane detector is defined by:

`<detector_dust_mc nr_pixel = "`N_{pixel}`">` λ_{min} λ_{max} N_λ α_1 `[deg]` α_2 `[deg]` D `[m]`

where N_{pixel} is the number of pixel, λ_{min} is the shortest wavelength, λ_{max} is the longest wavelength, N_λ is the amount of wavelengths used for simulation (if 1, only λ_{min} is used), α_1 and α_2 are the rotation angles around the first and second rotation axis (see Sect. 3.4), and D is the distance to the observer.

An optional feature is the zoom of the detector. This can be done by adding the desired side-lengths to the command line (available for each plane detector).

`<detector_dust_mc nr_pixel = "`N_{pixel}`">` ... d_x `[m]` d_y `[m]`

Another optional feature is the definition of individual numbers of pixel for each direction (available for each plane detector).

`<detector_dust_mc nr_pixel = "`$N_{\text{pixel,x}}$`*`$N_{\text{pixel,y}}$`">` ...

Examples:

```
<detector_dust_mc nr_pixel = "256"> 1e-6 2e-6 2 15.1 0 4.31998E+18
<detector_dust_mc nr_pixel = "256*128"> 1e-6 1e-5 2 35.1 0 4.31998E+18 149597870700.0
    ↪ 149597870700.0
```

Dust emission: For observations considering ray-tracing of the dust thermal emission (`CMD_DUST_EMISSION`) a plane detector is defined by:

`<detector_dust nr_pixel = "`N_{pixel}`">` λ_{min} λ_{max} N_λ $\text{ID}_{\text{source}}$ α_1 `[deg]` α_2 `[deg]` D `[m]`

The only difference to the scattering detector is the definition of $\text{ID}_{\text{source}}$. This ID is related to the index of the defined background sources (has to be 1, if no background source is defined). In contrast to the scattering detector, the ray-tracing detectors can make use of a background grid that is not the same as the used 2D Cartesian pixel map. Other options are `'polar'` or `'slice'`. They are chosen by writing `<detector_dust_polar` or `<detector_dust_slice`.

The `'polar'` grid can only be used if the underlying 3D grid is spherical or cylindrical and it should be used if the 3D grid contains a lot of cell close to the center ($f_r > 1.0$). For spherical or cylindrical grids, the usage of the polar ray-tracing detector is recommended. The amount of rays is arranged to fit to the spherical or cylindrical distribution of cells. Subsequently, the polar detector is mapped onto a cartesian detector with the definded number of pixels. Hereby, two methods are used to calculate the resulting Stokes vector in each pixel:

Interpolation: In this method, an iteratation over all cartesian pixel is performed. For each cartesian pixel, the Stokes vector of the nearest polar pixel and its neighboring polar pixels are interpolated to get the resulting Stokes vector.

This method is good if the cartesian detector has more pixels than the polar detector, if not, polar pixels (i.e., flux) will be ignored.

Nearest: In this method, an iteratation over all polar pixels is performed. For each polar pixel, the Stokes vector is added to the nearest corresponding cartesian pixel.

This method is good if cartesian detector has less pixels than the polar detector, if not, cartesian pixels potentially have zero flux. However, the total flux is maintained.

Thus, if the resulting cartesian detector has more pixels than there are polar pixels, POLARIS uses the interpolation method.

The `'slice'` background grid has only a analyzing purpose. Here, the 2D Cartesian background grid is put inside of the grid as a horizontal slice. Each ray-tracing pixel starts at different positions in the grid. For instance, this can be used to create contour height plots of the optical depth.

In addition to the zoom feature, the ray-tracing detector can also be shifted in both directions. This can be done by adding the desired shift of the detector to the command line.

<detector_dust nr_pixel = "N_{pixel}"> ... d_x [m] d_y [m] Δx [m] Δy [m]

Examples:

```
<detector_dust nr_pixel = "256"> 1e-6 2e-6 2 1 15.1 0 4.31998E+18
<detector_dust nr_pixel = "256*128"> 1e-6 2e-6 2 1 35.1 0 4.31998E+18 149597870700.0
    ↪ 149597870700.0 14959787070.0 -14959787070.0
<detector_dust_polar nr_pixel = "256"> 1e-6 2e-6 2 1 35.1 0 4.31998E+18
<detector_dust_slice nr_pixel = "256"> 1e-6 2e-6 2 1 35.1 0 4.31998E+18
```

Line emission: For observations considering ray-tracing of the spectral line emission (`CMD_LINE_EMISSION`) a plane detector is defined by:

<detector_line nr_pixel = "N_{pixel}" vel_channels = "N_{vel}"> $ID_{gas\ species}$ $ID_{transition}$ ID_{source} $v_{min,max}$ [m/s] α_1 [deg] α_2 [deg] D [m]

where N_{vel} is the number of the applied velocity bins, $ID_{gas\ species}$ is the ID of a gas species (atom/molecules) previously defined in the command file, $ID_{transition}$ is the ID of the line transition corresponding to the ID in the Leiden molecular database file (see Sect. 2.5), and $v_{min,max}$ defines the upper and lower limits of the velocity range.

Examples:

```
<detector_line nr_pixel = "256" vel_channels = "35"> 1 2 1 1e5 15.1 0 4.31998E+18
<detector_line nr_pixel = "256*128" vel_channels = "101"> 1 5 1 1e5 35.1 0 4.31998E+18
    ↪  149597870700.0 149597870700.0 14959787070.0 -14959787070.0
<detector_line_polar nr_pixel = "256" vel_channels = "35"> 1 5 1 1e5 35.1 0 4.31998E
    ↪ +18
<detector_line_slice nr_pixel = "256" vel_channels = "101"> 1 5 1 1e5 35.1 0 4.31998E
    ↪ +18
```

HINTS:

- Use an odd number of velocity bins N_{vel} to ensure to observe at the exact transition frequency ν_{ij}

Synchrotron: Defining a detector for a POLARIS synchrotron run is similar to the definition of a dust detector for ray-tracing. For observations considering synchrotron emission and Faraday rotation (`CMD_SYNCHROTRON`) a plane detector is defined by:

<detector_sync nr_pixel = "N_{pixel}"> λ_{min} λ_{max} N_λ ID_{source} α_1 [deg] α_2 [deg] D [m]

The zoom feature of the synchrotron detector as well as the shifting in both directions can be defined by:

<detector_sync nr_pixel = "N_{pixel}"> ... d_x [m] d_y [m] Δx [m] Δy [m]

Examples:

```
<detector_sync nr_pixel = "256"> 0.71 0.81 2 1 15.1 0 4.31998E+18
<detector_sync nr_pixel = "256*128"> 0.71 0.81 2 35.1 0 4.31998E+18 149597870700.0
    ↪ 149597870700.0 14959787070.0 -14959787070.0
```

HEALPIX detector: A spherical detector should be used when the observer has a position within the grid and the rays are coming from the outside (a surrounding sphere). This kind of detector is not yet available for the dust scattering mode and is purely based on ray-tracing. For spherical detectors the ray-tracing starts from of a sphere with a diameter that is larger than the side length of the grid and ends at the exact position of the observer. The starting points are equally distributed on the surface of the sphere according to the HEALPIX pixelation . This makes the POLARIS spherical detector directly comparable to Planck data and comes with the advantage that each point on the sphere is surrounded the same surface area.

In order to perform ray-tracing with aligned dust grains, the spherical detector can be defined by:

<detector_dust_healpix nr_sides = "N_{sides}"> λ_{min} λ_{max} N_λ ID_{source} r_x [m] r_y [m] r_z [m]
↪ l_{min} [deg] l_{max} [deg] b_{min} [deg] b_{max} [deg]

Here, r_x, r_y, and r_z is the position of the observer within the grid and N_{sides} defines the number of sides according to the HEALPIX algorithm. The HEALPIX algorithm requires for the number of sides N_{sides} to be to the base of 2 e.g. $1, 2, 4, 8$ .... The total number of detector pixel is then defined by $N_{pixel} = 12\ N_{sides}^2$. By default the spherical detector covers the full sky and points in direction of the center of the grid. The quantities l_{min}, l_{max}, b_{min}, and b_{max} are optional and define the minimum and maximum of the longitude and latitude, respectively, of the area on the sphere to be observed. However, in contrast to the galactic coordinate system the angles are defined to be $l \in [-180, 180]$ and $b \in [-90, 90]$.

Example:

```
<detector_dust_healpix nr_sides = "32"> 1e-6 1e-5 2 1 1.65e+20 0 0 -80 80 -45 45
```

A LRT detector is defined by:

<detector_line_healpix nr_sides = "N_{sides}" vel_channels = "N_{vel}"> ID_{gas} species $ID_{transition}$
ID_{source} $v_{min,max}$ [m/s] r_x [m] r_y [m] r_z [m] l_{min} [deg] l_{max} [deg] b_{min} [deg] b_{max} [deg
↪] v_x [m/s] v_y [m/s] v_z [m/s]

Here, the quantities v_x v_y, and v_z are also optional and define the velocity of the observer relative to the grid.

Example:

```
<detector_line_healpix nr_sides = "32" vel_channels = "35"> 1 2 1 1e5 1.96e+20 0 0
    ↪ -180 180 -3.5 3.5 1.5e3 -1e3 0
```

Following our previous scheme, the HEALPIX synchrotron detector is defined by:

<detector_sync_healpix nr_sides = "N_{sides}"> λ_{min} λ_{max} N_λ ID_{source} r_x [m] r_y [m] r_z [m]
↪ l_{min} [deg] l_{max} [deg] b_{min} [deg] b_{max} [deg]

Example:

```
<detector_sync_healpix nr_sides = "512"> 0.03 0.7 11 1.65e+20 0 0 -80 80 -45 45
```

3.6 Dust components

The dust grain model can be defined by:

`<dust_component> "path"` Ξ_i `q` a_{min} a_{max}

where `"path"` is the path to a single dust parameters file, Ξ_i is the mass fraction of the material, q is the exponent of the grain size power-law distribution $N_\text{d}(a) \propto a^q$, and a_{min} and a_{max} are the dust grain radii wich have to be in the range as defined in the dust parameters file (see Sect. 2.4). POLARIS can handle an arbitrary number of materials. In addition, more options to define the dust model can be applied by using the following definition:

`<dust_component> "path" "size_keyword"` Ξ_i $\varrho_{\text{material}}$ $[\text{kg}/\text{m}^3]$ a_{min} a_{max} p_0 p_1 ...

Here, $\varrho_{\text{material}}$ is the material density of the dust composition (a value of 0 uses the material density of the dust parameters file). As the `"size_keyword"`, several size distributions can be used:

- `'plaw'` (power-law distribution)

$$N_\text{d}(a) = a^{p_0} \tag{3.37}$$

- `'plaw-ed'` (power-law distribution + exponential decay)

$$N_\text{d}(a) = a^{p_0} \cdot \exp\left(-\left(\frac{a-p_1}{p_2}\right)^{p_3}\right) \tag{3.38}$$

- `'plaw-cv'` (power-law distribution + curvature)

$$N_\text{d}(a) = a^{p_0} \cdot \left(1+|p_2|\left(\frac{a}{p_1}\right)^{p_3}\right)^{\text{sign}(p_2)} \tag{3.39}$$

- `'plaw-cv-ed'` (power-law distribution + curvature + exponential decay)

$$N_\text{d}(a) = a^{p_0} \cdot \exp\left(-\left(\frac{a-p_1}{p_2}\right)^{p_3}\right) \cdot \left(1+|p_5|\left(\frac{a}{p_4}\right)^{p_6}\right)^{\text{sign}(p_5)} \tag{3.40}$$

- `'logn'` (log-normal distribution)

$$N_\text{d}(a) = \exp\left(-0.5\left(\frac{\ln(a)-\ln(p_0)}{p_1}\right)^2\right) \tag{3.41}$$

Another optional feature is the usage of multiple dust mixtures in the grid (see Sect. 2.3.1). To enable this feature, the dust components need an additional index, which can be defined as follows (not dependent on following parameters):

`<dust_component id = "ID`$_{\text{dust}}$`"> ...`

Multiple dust components with the same ID_{dust} will be mixed together to a dust mixture as long as their total mass fraction equals to one. This is useful if multiple dust compositions with different size distributions should be used, since each mixture (i.e. each ID_{dust}) can have its own distribution. In particular, different dust distributions within the same ID_{dust} can cause unexpected behaviour of POLARIS. Furthermore, each mixture can have different scattering phase functions:

`<phase_function id = "ID`$_{\text{dust}}$`"> ...`

If no ID_{dust} is given, a value of zero is assumed.

Examples:

```
<dust_component> "PATH/TO/POLARIS/input/dust/silicate_oblate.dat" 0.625 -3.5 5e-9 0.25
    ↪ e-6
<dust_component id = "1"> "PATH/TO/POLARIS/input/dust/silicate_oblate.dat" 0.625 -3.5
    ↪ 5e-9 0.25e-6
<dust_component> "PATH/TO/POLARIS/input/dust/silicate_oblate.dat" "plaw" 1.0 3500 5e-9
    ↪  1.13e-6 -3.5
```

Dust properties such as cross sections or the scattering matrix are weighted over the grain size distribution $N_\mathrm{d}(a)$ between a minimal a_min and maximal a_max grain size and the different materials are mixed. Here, we define $N_{\mathrm{mat},i}$ as the amount of dust grain materials of the i-th mixture and Ξ_j are the corresponding material fractions with $\sum_{j=1}^{N_{\mathrm{mat},i}} \Xi_j = 1$. For example, the average cross-section in a grid cell is

$$C = \frac{1}{\sum_{i=1}^{N_\mathrm{mix}} n_{\mathrm{d},i}} \sum_{i=1}^{N_\mathrm{mix}} n_{\mathrm{d},i} \times \frac{\int_{a_\mathrm{min}}^{a_\mathrm{max}} \pi a^2 \sum_{j=1}^{N_{\mathrm{mat},i}} \Xi_j Q_j(a) N_{\mathrm{d},i}(a) \mathrm{d}a}{\int_{a_\mathrm{min}}^{a_\mathrm{max}} N_{\mathrm{d},i}(a) \mathrm{d}a}, \tag{3.42}$$

where n_d is the dust number density and N_mix the amount of total dust mixtures.

3.7 Dust continuum radiative transfer

3.7.1 Phase functions

Light propagating through a dusty medium will change its direction by scattering on a single dust grains. The angular distribution of scattering on a grain at a given wavelength is given by the phase function Φ. Many phase functions have been proposed to account for dust with different shapes, sizes, and properties of grain materials. POLARIS allows to choose the kind of phase function used in the different MC based modes.

Isotropic scattering In the case of isotropic scattering each direction has the same probability after the scattering event. The same sampling process is also applied by the thermal remission. The new photon package direction $\vec{r}'$ is sampled from a uniform distribution of points on the surface of a unit sphere and depends on the three parameters $u = 1 - 2z_1$, $t = 2\pi z_2$, and $r = \sqrt{1-u^2}$ and can be calculated by

$$\vec{r}' = (r\cos(t), r\sin(t), u)^T . \tag{3.43}$$

Here, the quantities z_1 and z_2 are random numbers. This phase function is available for all MC based POLARIS modes. In order to activate isotropic scattering the line

```
<phase_function> PH_ISO
```

needs to be in the command file. The isotropic phase function is the standard phase function.

Henyey-Greenstein scattering In contrast to isotropic scattering, anisotropic scattering is characterized by a preferential scattering directions. The Henyey-Greenstein (HG, Henyey & Greenstein 1941) phase function is the most widely used parametrization of the phase function and provides a good single-parameter approximation with

$$\Phi_\mathrm{HG}(\psi, g) = \frac{1}{4\pi} \frac{1-g^2}{[1+g^2-2g\cos(\psi)]^{3/2}}. \tag{3.44}$$

This phase function depends on scattering angle ψ and the asymmetry parameter $g \in [-1, 1]$ where $g = 0$ means isotropic scattering, $g = 1$ backward scattering, and $g = 1$ forward scattering, respectively. The anisotropy parameter g is pre-calculated dependent on material (i.e., refractive index), grain size, and wavelength, or provided by the dust parameters file (see Sect. 2.4). The HG phase function can be selected with:

```
<phase_function> PH_HG
```

Alternatively, a modified Henyey-Greenstein additionally introduces the parameter α (Draine 2003):

$$\Phi_{\mathrm{DHG}}(\psi, g, \alpha) = \frac{1 + \alpha \cos^2(\psi)}{1 + \alpha(1 + 2g^2)/3} \Phi_{\mathrm{HG}}(\psi, g), \tag{3.45}$$

where Φ_{HG} is defined by Eq. (3.44). For $\alpha = 0$, Eq. (3.45) reduces to the Henyey-Greenstein phase function (Eq. 3.44). For $g = 0$ and $\alpha = 1$, it reduces to the Rayleigh scattering phase function. The parameters g and α have to be provided by the dust parameter file (see Sect. 2.4). The Draine HG phase function can be selected with:

```
<phase_function> PH_DHG
```

Alternatively, a three-parameter Henyey-Greenstein is a linear combination of two HG phase functions (Irvine 1965, 1968):

$$\Phi_{\mathrm{TTHG}}(\psi, g_1, g_2, w) = w\Phi_{\mathrm{HG}}(\psi, g_1) + (1 - w)\Phi_{\mathrm{HG}}(\psi, g_2), \tag{3.46}$$

where Φ_{HG} is defined by Eq. (3.44). The parameters g_1, g_2 and w have to be provided by the dust parameter file (see Sect. 2.4). The Draine HG phase function can be selected with:

```
<phase_function> PH_TTHG
```

Mie-scattering In the case of homogeneous and spherical dust grains a solution to the scattering problem can be found with Mie scattering theory. Here, the phase function of scattering can be expanded in terms of an infinite series of associated Legendre polynomials. The coefficients of the series can be calculated with the refractive index of the dust grain material. This allows to calculate the new scattering direction by utilizing the pre-calculated values of the scattering matrix. The S_{11} element of the scattering matrix can be used by POLARIS to calculate the phase function (see Sect. 3.7.5 for details). The Mie scattering phase function is only available for the MC dust polarization mode and can be activated by

```
<phase_function> PH_MIE
```

Finally, polarization due to scattering is only considered in the case of Mie-scattering (see Sect. 3.7.5).

3.7.2 Dust heating

The command that identifies the dust heating mode is `CMD_TEMP`. This mode is a combination of the methods of continuous absorption (Lucy 1999) and immediate re-emission (Bjorkman & Wood 2001). As a minimal requirement the input grid needs the 3D gas density distribution. Additionally, a dust model and at least one photon emitting source is required. However, if the grid contains no further physical parameters, no follow-up RT calculations (e.g. dust polarization) are possible. The dust shape and the possible alignment of non-spherical dust grains has no influence on the resulting dust dust temperature distribution. For this POLARIS mode the dust grains are considered to be spherical in the dust heating mode. This reduces the required computational efforts dramatically.

In the beginning of the dust heating mode the emissivity

$$j_\mathrm{l}(T_\mathrm{d}) = n_\mathrm{d} \int C_{\mathrm{abs},\lambda} B_\lambda(T_\mathrm{d}) \mathrm{d}\lambda \tag{3.47}$$

is pre-calculated for a range of dust temperatures $T_\mathrm{d} \in [1\,\mathrm{K},\ 4000\,\mathrm{K}]$ and tabulated. Each photon package starts with an energy per unit time $\epsilon/\Delta t$ and a wavelength λ. In each cell with volume V_cell, we store the energy per unit time

$$\dot{E} = n_\mathrm{d} \frac{\epsilon}{V_\mathrm{cell} \Delta t} \sum_\mathrm{i} C_{\mathrm{abs},\lambda_i} \times \ell_i + \Delta\dot{E} \tag{3.48}$$

that all passing photon packages deposit along their paths ℓ during the MC dust heating simulation. The quantity $\Delta\dot{E}$ describes the offset in the dust heating simulation in the case of an already exiting dust temperature. If the photon package interacts by means of absorption and re-emission (see also Sect. 3.1.3) a new wavelength is sampled from

$$z = \frac{\int_{\lambda_\mathrm{min}}^{\lambda_i} C_{\mathrm{abs},\lambda} \frac{\mathrm{d}B_\lambda(T_{\mathrm{d,i}})}{\mathrm{d}T} \mathrm{d}\lambda}{\int_{\lambda_\mathrm{min}}^{\lambda_\mathrm{max}} C_{\mathrm{abs},\lambda} \frac{\mathrm{d}B_\lambda(T_{\mathrm{d,i}})}{\mathrm{d}T} \mathrm{d}\lambda} \tag{3.49}$$

to LTE. Since the temperature changes by each absorption event the new wavelength cannot be sampled from the Planck function but by its temperature derivative in order to ensure a correct spectrum of the emitted photons (compare Eq. 3.24). Finally, we compare $\dot{E}$ with the tabulated and cubic spline interpolated values of $j(T_\mathrm{d})$ in the end of the MC simulation in order to determine the dust temperature T_d. The newly calculated grid (`'grid_temp.dat'`) will the be stored in the defined output directory. If the grid contains no dust temperature at all the grid will be extended by one additional physical parameter.

We provide a minimal example command file for the dust heating mode in Listing 3.2. As a radiation source a single solar like star at the center of the grid is defined sending 10^6 photon packages into the grid. The dust model consists only of silicate grains. Besides the path to the dust parameters file, no further parameters are defined. This means we consider the full size range with an size distribution exponent of $q = -3.5$ for the dust model. Again, we define the path of the input grid and the path for the resulting grid and additional output files (e.g. plots). In this simulation we use the maximal possible number of threads of the machine (`<nr_threads> -1`). For the dust-to-gas mass ratio we use the usual value of 1 %. POLARIS can not directly calculate a gas temperature. However, a fixed ratio of gas temperature T_d to dust temperature T_d can be assumed with the command `<adj_tgas>`. Any possible input gas temperature is lost if this command is defined.

If stochastic heating should be considered for the thermal emission of the dust grains, either the radiation field **or** the effective temperature due to stochastic heating has to be saved in the grid with the command `<radiation_field> 1` **or** `<stochastic_heating> SIZE_LIMIT` (see also Sect. 3.7.5). Saving the radiation field offers more flexibility and precision, since the probability distribution of temperatures will be used in the dust emission and the maximum size can be varied between each dust emission simulation. However, saving the radiation field significantly increases the grid memory size (by around 400 doubles per cell) and the calculation of the dust emission takes more time if the stochastic heating is considered by using the probability distribution of temperatures that is calculated out of the radiation field.

Additionally, a dust temperature can also already be contained in the input grid e.g. as a result of a MHD simulation. This temperature may emerge from different physical processes (e.g. gas-dust collisions or compression) other than radiation-dust interaction. Hence, it may be of interest to combine both dust temperatures by defining the command `<dust_offset>` (0=off and 1=on). Here, POLARIS performs the reverse calculation by comparing the input dust temperature with the

Listing 3.2: Example command file to calculate the dust temperature distribution.

```
<task> 1
#pipeline ID for dust heating
<cmd> CMD_TEMP

#A star in the center as radiation source
<source_star nr_photons = "1e6"> 0 0 0 1 6000

#Silicate as only dust components
<dust_component> "PATH/TO/POLARIS/input/dust/silicate_oblate.dat"

#path of the input grid file
<path_grid> "PATH/TO/YOUR/grid.dat"

#path for all output data
<path_out> "PATH/WHERE/THE/RESULTS/SHOULD/BE/SAVED/temp/"

#Maximal nr. of available threads for parallel computing
<nr_threads> -1

#dust to gas mass ratio
<mass_fraction> 0.01 # optional

#dust to gas temperature ratio
<adj_tgas> 10 # optional

#dust to gas temperature ratio
<dust_offset> 1 # optional

#calculate the dust temperature for each dust grain size individually
<full_dust_temp> 0 # optional

#save the 3D radiation field in the grid to use it for e.g. stochastic heating
<radiation_field> 0 # optional

#Save the effective temperature from stochastic heating for
#dust grains with a size up to 10 nm
<stochastic_heating> 1e-8
</task>
```

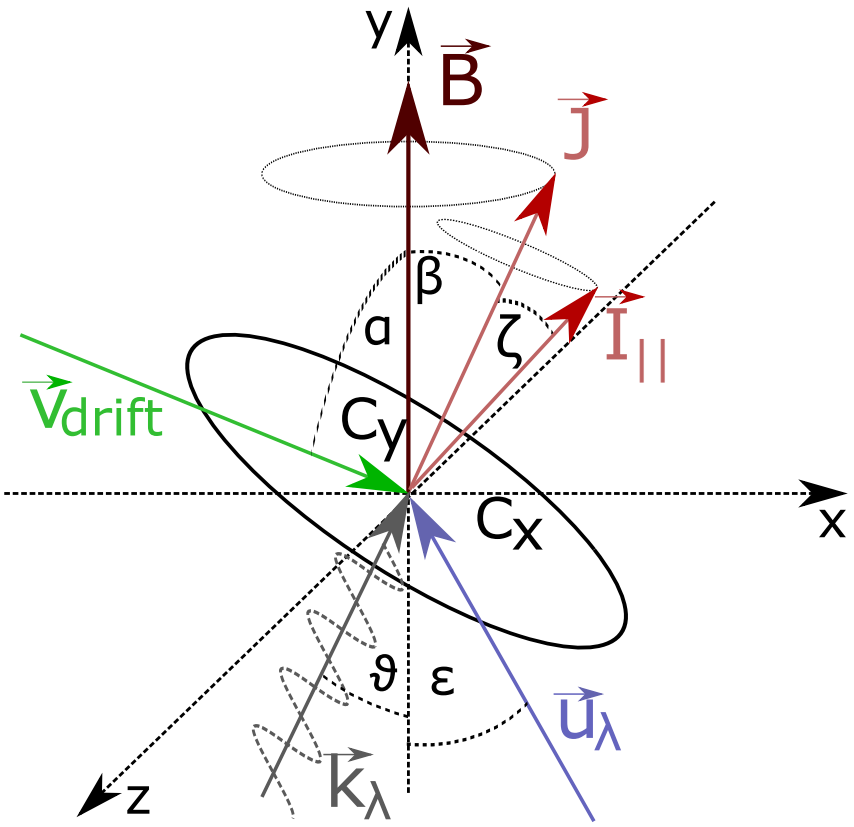


Figure 3.4: Geometrical configuration of a dust grain partially aligned with its angular momentum $\vec{J}$ to the magnetic field direction $\vec{B}$. The precession of $\vec{J}$ around $\vec{B}$ defines the cone angle β and the precession of maximum moment of inertia $\vec{I}$ around $\vec{J}$ defines the angle of internal alignment ζ. Here, the angle ϑ is defined to be between direction of incident light $\vec{k}$ and $\vec{B}$. The angles α and ϵ are between the direction of the supersonic velocity stream $\vec{v}_g$ and $\vec{B}$ and between the anisotropy $\vec{u}_\lambda$ in the radiation field and $\vec{B}$, respectively. The vector $\vec{v}'_{rel}$ is the projection of $\vec{v}_g$ on the direction of the magnetic field direction $\vec{B}$.

pre-calculated values of $j(T_d)$ in order to determine the offset of $\Delta\dot{E}$ in each cell (see Eq. 3.48). The resulting final dust temperature is higher as the sum of both separate temperatures.

Instead of only calculating the dust temperature distribution for an average dust grain size, a dust temperature can be calculated for each dust grain size individually by adding the following command to the `CMD_TEMP` simulation:

```
<full_dust_temp> 1
```

The resulting temperature containing grid will be adjusted accordingly (see Sect. 4.1) and any follow-up simulation of the thermal emission with this grid will use the temperatures of each grain size. As a consequence, the considered dust composition cannot not be changed after the temperature calculation (which in general should not be done).

3.7.3 Grain alignment theories

Besides Zeeman splitting the extensive treatment of imperfectly aligned dust grains in the RT simulations is one of the unique features of the POLARIS code. A unifying theory of dust grain alignment is still missing. For this reason POLARIS allows to choose from the major classes of

dust grain alignment theories developed so far. In this section we intent to provide a short summary about this theories and their physical implications.

Considering the violent environment in most astrophysical processes the perfect alignment of non-spherical dust grains with respect to the magnetic field direction is permanently disturbed (see Fig. 3.4). This leads to the precession angles of β and ζ. The angle β is defined to be between the angular momentum of the dust grain $\vec{J}$ and the magnetic field $\vec{B}$, and ζ is between the maximum moment of inertia $\vec{I}$ and $\vec{J}$. The larger the precession angles are, the more the non-spherical dust grains appear to be spherical. Consequently, the resulting polarization becomes reduced. A measurement to quantify this is the so called Rayleigh reduction factor

$$R = \left\langle G\left(\cos^2(\beta)\right) G\left(\cos^2(\zeta)\right)\right\rangle \in [0,1] \tag{3.50}$$

where $R = 1$ stands for the maximal possible polarization of a dust grain and $R = 0$ when polarization is completely suppressed (Greenberg 1968; Lee & Draine 1985; Roberge & Lazarian 1999; Lazarian 2007). The Rayleigh reduction factor is defined by the precession angles averaged over distribution functions where

$$G(z) = \frac{3}{2}\left(z - \frac{1}{3}\right) = \frac{3z}{2} - \frac{1}{2}. \tag{3.51}$$

In order to reduce the computational time we assume the precession angles to be mostly independent with

$$\langle G(X)\, G(Y)\rangle \approx \langle G(X)\rangle \times \langle G(Y)\rangle\,(1 + f_c)\,, \tag{3.52}$$

where f_c is a correlation factor and $f_c = 0.6$ stands for no correlation at all. For most cases $f_c \approx 0.6$ and of minor relevance. However, if required it can be changed by the user with the command `<f_c> 0.6`. The distribution functions in turn depend heavily on the considered dust grain alignment theory.

Imperfect internal (`II`) alignment:

In order to consider the effects of internal alignment in your RT calculations include `<align> ALIG_INTERNAL` in your command file.

Even without an external disturbance a non-spherical dust grain can not be expected to align perfectly with the magnetic field. Due to internal thermal fluctuations the dust grain performs a precession of $\vec{I}_{||}$ around $\vec{J}$ with an opening angle of ζ between both quantities (see Lazarian & Roberge 1997, for details). This effect is related to the internal temperature of the dust. Hence, the distribution function of `II` alignment follows a Boltzmann distribution

$$f(\zeta) \approx \exp\left(-\alpha\left[1 + (h-1)\sin^2(\zeta)\right]\right), \tag{3.53}$$

where $\alpha = J_{\text{eff}}^2/(2I_{||}k_{\text{B}}T_{\text{d}})$, $h = I_{||}/I_{\perp} = 2/(s^2{+}1)$ is the ratio of the minimal and maximal momenta of inertia, $s < 1$ is the aspect ratio of the grain, and J_{eff} is the rms value of the angular momentum. The calculation of J_{eff} in turn depends on the additionally considered alignment mechanisms (see the following Sects.). This substitutions allow to calculate the Rayleigh reduction factor with $x = \alpha(h-1)$:

$$\left\langle G\left(\cos^2(\zeta)\right)\right\rangle = \frac{e^x}{\sqrt{\pi x}\ \text{erfi}\left(\sqrt{x}\right)} - \frac{1}{2x}, \tag{3.54}$$

where erfi is the imaginary error function.

Imperfect Davis – Greenstein (IDG) alignment

The IDG alignment can be switched on by adding `<align> ALIG_IDG` to the command file. This theory provides a steady state solution for dust grain alignment by balancing the disturbances on the grain by random gas collisions with alignment effects of paramagnetic relaxation (Davis-Greenstein effect) in the dust grain material (Davis & Greenstein 1951; Spitzer & McGlynn 1979). Solving the equations of state the Rayleigh reduction factor can be calculated with

$$\left\langle \cos^2(\beta) \right\rangle = \left(1 - \sqrt{\frac{\xi^2(a)}{1-\xi^2(a)}} \right) \arcsin \frac{\sqrt{1-\xi^2(a)}}{1-\xi^2(a)}, \tag{3.55}$$

where

$$\xi^2(a) = \frac{a + \delta_0 T_\mathrm{d}/T_\mathrm{g}}{a + \delta_0} \tag{3.56}$$

and a is the dust grain radius. Here, the critical parameter is

$$\delta_0 = 2.07 \times 10^{20} \frac{B^2}{n_\mathrm{g} T_\mathrm{d} \sqrt{T_\mathrm{g}}}, \tag{3.57}$$

which provides an upper threshold of dust grain alignment Dust grains with sizes $a > \delta_0$ do not contribute significantly to polarization. The constant 2.07×10^{20} is adequate for most paramagnetic materials. However, it can be changed by the user with the command `<delta0>`. As for the angular momentum of internal alignment, we use the approximation:

$$J_\mathrm{eff}^2 \approx 2 I_{||} k_\mathrm{B} T_\mathrm{g} \sqrt{(1 + s^{-2}/2)\,(1 + T_\mathrm{d}/T_\mathrm{g})}. \tag{3.58}$$

GOLD (magneto mechanical) alignment

The original GOLD alignment described only the aligment of dust grains in the presence of a velocity field (Gold 1952). Later, this theory was extended to model also the influence of the magnetic field (magneto mechanical alignment, see Lazarian 1995; Lazarian et al. 1996; Lazarian & Efroimsky 1996). For considering GOLD alignment, write `<align> ALIG_GOLD` in your command file. In GOLD alignment, two forces act on the dust grain simultaneous caused by an directed gas stream and the alignment by the magnetic field. This leads to a dependency of dust grain alignment on the angle α between the predominate direction of the velocity stream and the magnetic field direction. As a solution for the Rayleigh reduction factor (Lazarian 1996, 1997), the GOLD alignment theory provides

$$\left\langle \cos^2(\beta) \right\rangle = \begin{cases} \frac{\sqrt{-\gamma_\mathrm{g}}\ \arcsin\left(\sqrt{-\gamma_\mathrm{s}/(1+\gamma_\mathrm{g})}\right)}{\gamma_\mathrm{s} \arctan\left(\sqrt{\gamma_\mathrm{s}\gamma_\mathrm{g}/(1+\gamma_\mathrm{s}+\gamma_\mathrm{g})}\right)} - \frac{1}{\gamma_\mathrm{s}} & \text{if } \gamma_\mathrm{s} < 0 \text{ and } \gamma_\mathrm{g} < 0 \\ \frac{\sqrt{-\gamma_\mathrm{g}}\ \operatorname{arsinh}\left(\sqrt{\gamma_\mathrm{s}/(1+\gamma_\mathrm{g})}\right)}{\gamma_\mathrm{s} \operatorname{artanh}\left(\sqrt{-\gamma_\mathrm{s}\gamma_\mathrm{g}/(1+\gamma_\mathrm{s}+\gamma_\mathrm{g})}\right)} - \frac{1}{\gamma_\mathrm{s}} & \text{if } \gamma_\mathrm{s} > 0 \text{ and } \gamma_\mathrm{g} < 0 \\ \frac{\sqrt{\gamma_\mathrm{g}}\ \arcsin\left(\sqrt{-\gamma_\mathrm{s}/(1+\gamma_\mathrm{g})}\right)}{\gamma_\mathrm{s} \operatorname{artanh}\left(\sqrt{-\gamma_\mathrm{s}\gamma_\mathrm{g}/(1+\gamma_\mathrm{s}+\gamma_\mathrm{g})}\right)} - \frac{1}{\gamma_\mathrm{s}} & \text{if } \gamma_\mathrm{s} < 0 \text{ and } \gamma_\mathrm{g} > 0 \\ \frac{\sqrt{\gamma_\mathrm{g}}\ \operatorname{arsinh}\left(\sqrt{\gamma_\mathrm{s}/(1+\gamma_\mathrm{g})}\right)}{\gamma_\mathrm{s} \arctan\left(\sqrt{\gamma_\mathrm{s}\gamma_\mathrm{g}/(1+\gamma_\mathrm{s}+\gamma_\mathrm{g})}\right)} - \frac{1}{\gamma_\mathrm{s}} & \text{if } \gamma_\mathrm{s} > 0 \text{ and } \gamma_\mathrm{g} > 0 \end{cases} \tag{3.59}$$

with the anisotropy factor of the gas stream

$$\gamma_\mathrm{s} = -\frac{1}{2} \frac{\langle \vec{v}_\mathrm{rel} \rangle - 3 \langle \vec{v}'_\mathrm{rel} \rangle}{\langle \vec{v}_\mathrm{rel} \rangle - \langle \vec{v}'_\mathrm{rel} \rangle}. \tag{3.60}$$

Here, $\vec{v}'_\mathrm{rel}$ is the projection of the relative drift velocity $\vec{v}_\mathrm{rel}$ between the gas and dust component on the magnetic field direction, $\gamma_\mathrm{g} = (s^2 - 1)/2$ is the grain non-sphericity (Lazarian 1994), and s

is the aspect ratio of the grain. In order to calculate the `II` alignment, the angular momentum of GOLD alignment can be approximated (Lazarian 1997) by

$$J_{\text{eff}}^2 \approx k_{\text{B}} I_{||} (2/h + 1) \left(\frac{T_{\text{g}} + T_{\text{d}}}{2} + \frac{\mu m_{\text{H}} v_{\text{rel}}^2}{6 k_{\text{B}}} \right), \tag{3.61}$$

with the molecular mass μm_{H} of the gas. The GOLD alignment requires an supersonic velocity stream with a Mach number $M > 1$. Dust grains with smaller number are assumed to be not aligned. Therefore, POLARIS calculates

$$M = \frac{v_{\text{rel}}}{\sqrt{k_{\text{B}} T_{\text{g}} / \mu m_{\text{H}}}}, \tag{3.62}$$

for each cell of the grid. The average mass of a gas particle in these calculations can be controlled with the command `<mu>`.

As for the coupling efficiency between magnetic field and dust grain alignment, we demand (Lazarian & Hoang 2007; Hoang & Lazarian 2009; Hughes et al. 2009):

$$|\vec{B}| > 4.1 \times 10^{-19} \frac{a n_{\text{g}} T_{\text{d}} \sqrt{T_{\text{g}}}}{s^2}, \tag{3.63}$$

$$\Rightarrow a_{\text{lim}} \approx 2.439 \times 10^{18} \frac{|\vec{B}| s^2}{n_{\text{g}} T_{\text{d}} \sqrt{T_{\text{g}}}}, \tag{3.64}$$

where $s < 1$ is the aspect ratio of the grain and B the magnetic field strength in tesla (T). Thus, an upper limit of grain size a_{lim} is defined for alignment (Eq. 3.64). The same limitation of the magnetic field applies also to RAT alignment. In the case of other dust grain materials you can use the command `<larm_f>` to alter the prefactor of Eq. (3.63). It needs to be emphasized that this model of mechanical alignment turned out to be unreliable and might be updated on a short term to keep up with current research on that field.

Radiative torque (RAT) alignment

The RAT alignment theory can be switches on with the command `<align> ALIG_RAT`. This alignment balances the influences of random gas bombardment, paramagnetic dissipation in the dust grain material, and the radiative pressure due to an isotropic radiation field. The critical quantities here are the local mean energy density $\overline{u}_\lambda$ is the wavelength specific anisotropy factor γ_λ varies between $\gamma_\lambda = 1$ for an unidirectional radiation field and $\gamma_\lambda = 0$ for completely isotropic radiation. Both, mean energy density $\overline{u}_\lambda$ and anisotropy factor γ_λ can be calculated by POLARIS in an extra RT mode (see Sect. 3.7.4). Given that $\overline{u}_\lambda$ and γ_λ are known, POLARIS is able to determine the characteristic dust grain size a_{alg} at which dust grains start to align.

In particular, POLARIS calculates the torques Γ_{rad} generated by an anisotropic radiation field which can cause a non-spherical dust grain to gain angular momentum (Hoang & Lazarian 2014):

$$\Gamma_{\text{rad}} = \pi a^2 \int \left(\frac{\lambda}{2\pi} \right) \gamma_\lambda \cos(\epsilon) Q_\Gamma(a, \lambda) \overline{u}_\lambda \mathrm{d}\lambda, \tag{3.65}$$

where a is the equivalent radius of the particle. The radiative torque efficiency can be approximated by a power law

$$Q_\Gamma(a, \lambda) = \begin{cases} Q_\Gamma^{\text{ref}} & \text{if} \lambda \leq 1.8a, \\ Q_\Gamma^{\text{ref}} \times \left(\frac{\lambda}{1.8a} \right)^{-\alpha_{\text{Q}}} & \text{otherwise}, \end{cases} \tag{3.66}$$

where $Q_\Gamma^{\rm ref}$ and $\alpha_{\rm Q}$ depend on the grain shape and grain material (Lazarian & Hoang 2007; Hoang & Lazarian 2014; Reissl et al. 2020). The POLARIS code assumes a value of $Q_\Gamma^{\rm ref} = 0.4$ and $\alpha_{\rm Q} = 3.0$, however these values can be changed with the command `<Q_ref>` or `<alpha_Q>`. Following Hoang & Lazarian (2008), POLARIS assumes that dust grains are aligned if

$$\frac{t_{\rm drag}\Gamma_{\rm rad}}{J_{\rm th}} \geq 3, \tag{3.67}$$

where $t_{\rm drag}$ is the characteristic timescale of the gas drag acting on the dust grains (collisions with gas particles as well as emission of IR photons), and $J_{\rm th}$ is the grain angular momentum in the abscence of any aligning torques since dust grain rotation is at thermal equilibrium with the gas (see Reissl et al. 2020, for details).

An analytically distribution function for the precession angle β and, subsequently, the Rayleigh reduction factor is unknown so far. However, the Rayleigh reduction factor can be estimated. Solving the equations of state reveals two attractor points in the parameters space of RAT theory with different angular momenta. Dust grains at the attractor point with a high angular momentum (high-J) can be assumed to aligned perfectly where as the second attractor point is internally not perfectly aligned or rather $\langle G(\cos^2(\zeta))\rangle < 1$. Defining $f_{\rm high-J}$ to be the fraction of dust grains that settle at the high-J attractor point the Rayleigh reduction factor can be approximated with

$$R = \begin{cases} f_{\rm high-J} + (1 - f_{\rm high-J}) \left\langle G\left(\cos^2(\zeta)\right)\right\rangle & \text{if } a_{\rm alg} \leq a < a_{\rm lim} \\ 0 & \text{otherwise} \end{cases} \tag{3.68}$$

The upper limit of grain size $a_{\rm lim}$ is defined in Eq. (3.64). The angular momentum for `II` alignment is assumed to be $J_{\rm eff}^2 \approx 2I_{||}k_{\rm B}T_{\rm g}$. The POLARIS code assumes a value of $f_{\rm high-J} = 0.25$ however this values can be changed with the command `<f_highJ>`.

If `II` alignment is not used, POLARIS sets

$$R = \begin{cases} 1 & \text{if } a_{\rm alg} \leq a < a_{\rm lim} \\ 0 & \text{otherwise.} \end{cases} \tag{3.69}$$

The default value of $R = 1$ can be changed with the command `<R_rayleigh>`.

Combined of Rayleigh reduction factors

Since all these alignment theories have their place and are most likely simultaneously at work POLARIS allows to calculate a combined Rayleigh reduction factor with

$$R_\Sigma = \frac{R_{\rm IDG} + R_{\rm RAT} + R_{\rm GOLD} + R_{\rm IDG}R_{\rm RAT} + R_{\rm IDG}R_{\rm GOLD} + R_{\rm RAT}R_{\rm GOLD} + 3R_{\rm IDG}R_{\rm RAT}R_{\rm GOLD}}{1 + 2R_{\rm IDG}R_{\rm RAT} + 2R_{\rm IDG}R_{\rm GOLD} + 2R_{\rm RAT}R_{\rm GOLD} + 2R_{\rm IDG}R_{\rm RAT}R_{\rm GOLD}}. \tag{3.70}$$

However, this is a first approximation and has to be taken with care. In order to combine the alignment theories you can list any number of combination of alignment commands in your command file e.g.

```
<align> ALIG_INTERNAL
<align> ALIG_RAT
<align> ALIG_GOLD
<align> ALIG_IDG
```

3.7.4 The grain alignment radius

The command that identifies the RAT alignment radius mode is `CMD_RAT`. In order to solve Eq. (**??**) the local mean energy density $\overline{u}_\lambda$ and the wavelength specific anisotropy factor γ_λ need to be determined. Here, POLARIS uses a MC mode where we store the direction and magnitude of the energy density $\vec{u}_\lambda$ in each cell. Following Lucy (1999), for the mean energy density per path length and wavelength, we derive

$$\vec{u}_{\lambda,i} = \frac{\epsilon \ell_i}{c \Delta t V_{\text{cell}}} \frac{\vec{k}}{|\vec{k}|}, \tag{3.71}$$

where $\vec{k}$ is the wave vector of each photon package. This allows to calculate the mean energy density

$$\overline{u}_\lambda = \sum_{i=1}^{N_{\text{ph}}} |\vec{u}_{\lambda,i}| \tag{3.72}$$

and the anisotropy parameter

$$\gamma_\lambda = \frac{1}{\overline{u}_\lambda} \left| \sum_{i=1}^{N_{\text{ph}}} \vec{u}_{\lambda,i} \right| \in [0;1] \tag{3.73}$$

The RAT alignment radius mode is the most memory demanding one since four values (8 bytes, double, $(u_x, u_y, u_z)_\lambda$, and $\overline{u}_\lambda$) per wavelength and cell need to be stored and updated when a new photon package passes the cell. Finally, POLARIS solves Eq. (**??**) in order to calculate the minimal dust grain alignment radius a_{alg} in each cell. A new grid will be created that contains all the physical input quantities extended by the alignment radius a_{alg} and stored as `'grid_rat.dat'` in the output path. We provide a minimal example command file for the RAT alignment mode in Listing 3.3.

POLARIS also allows the calculation of both, the dust temperature and RAT alignment radius at once by using `CMD_TEMP_RAT` instead of `CMD_TEMP` or `CMD_RAT`. The output grid will be named `'grid_temp.dat'` and includes both quantities.

3.7.5 Dust Intensity and polarization maps

Polarization by non-spherical dust grains

Focusing on extinction the entries of the cross sections matrix can be calculated with (Lazarian 2007):

$$C_{\text{ext,x}} = \langle C_{\text{ext}} \rangle + \frac{R}{3} \left(C_{\text{ext},\perp} - C_{\text{ext},||} \right) \left(1 - 3 \sin^2(\vartheta) \right), \tag{3.74}$$

$$C_{\text{ext,y}} = \langle C_{\text{ext}} \rangle + \frac{R}{3} \left(C_{\text{ext},\perp} - C_{\text{ext},||} \right). \tag{3.75}$$

where $C_{\text{ext,x}}$ is the extinction cross section along the grains major axis and $C_{\text{ext,y}}$ is the cross section along the minor axis while the angle ϑ is between incident light and magnetic field direction (see Fig. 3.4), and

$$\langle C_{\text{ext}} \rangle = \frac{2C_{\text{ext},\perp} + C_{\text{ext},||}}{3}, \tag{3.76}$$

is the average cross section of a spherical dust grain of equivalent volume. Note that cross sections and the Rayleigh reduction factor are also dependent on wavelength and grain radius.

For oblate dust grains the cross section of extinction is

$$C_{\text{ext}} = \frac{C_{\text{ext,x}} + C_{\text{ext,y}}}{2}, \tag{3.77}$$

Listing 3.3: Example command file to calculate the dust temperature distribution.

```
<task> 1
#pipeline ID for RAT alignment radius simulation
<cmd> CMD_RAT

#A star in the center as radiation source
<source_star nr_photons = "1e6"> 0 0 0 1 6000

# Silicate as only dust components
<dust_component> "PATH/TO/POLARIS/input/dust/silicate_oblate.dat"

#path of the input grid file
<path_grid> "PATH/WHERE/THE/RESULTS/SHOULD/BE/SAVED/temp/grid_temp.dat"

#path for all output data
<path_out> "PATH/WHERE/THE/RESULTS/SHOULD/BE/SAVED/rat/"

#Maximal nr. of threads for parallel computing
<nr_threads> -1

#dust to gas mass ratio
<mass_fraction> 0.01 # optional
</task>
```

the cross section of linear polarization is defined as

$$C_{\text{pol}} = \frac{R}{2}\left(C_{\text{ext},\perp} - C_{\text{ext},||}\right)\sin^2(\vartheta). \tag{3.78}$$

The Rayleigh reduction factor depends on the selected dust grain alignment theories as introduced in Sect. 3.7.3 and is calculated by POLARIS for each radiation-dust interaction separately. Similar, it is

$$C_{\text{abs}} = \frac{C_{\text{abs,x}} + C_{\text{abs,y}}}{2}, \tag{3.79}$$

$$C_{\text{pabs}} = \frac{R}{2}\left(C_{\text{abs},\perp} - C_{\text{abs},||}\right)\sin^2(\vartheta). \tag{3.80}$$

Grain alignment characterized by a user defined Rayleigh reduction factor can be used with the command `ALIG_NONPA` and passing a value to the command `<R_rayleigh>`. In addition, if $R = -1$ is passed, the Rayleigh reduction factor is set to the absolute value of the magnetic field, while the direction of alignment is the magnetic field direction. In this case, the *magnetic field* is used as a parameter for the aligment direction and efficiency and thus should be normalized from 0 to 1.

In contrast, the cross-sections of perfectly aligned dust grains can be calculated efficiently using (Reissl et al. 2014)

$$C_{\text{ext}} = \frac{C_{\text{ext},\perp} + C_{\text{ext},||}}{2} + \frac{C_{\text{ext},\perp} - C_{\text{ext},||}}{2}\cos^2(\vartheta), \tag{3.81}$$

$$C_{\text{abs}} = \frac{C_{\text{abs},\perp} + C_{\text{abs},||}}{2} + \frac{C_{\text{abs},\perp} - C_{\text{abs},||}}{2}\cos^2(\vartheta), \tag{3.82}$$

and setting $R = 1$ for C_{pol} and C_{pabs} in Eqs. (3.78) and (3.80). Perfect grain alignment can be used with the command `ALIG_PA`. The direction of alignment is the magnetic field direction.

The reference frame of the dust grain is defined by the alignment direction. For a Stokes vector rotated into the reference frame of the dust, the dust RT the extinction coefficients are $\alpha_U = \alpha_V = \kappa_U = \kappa_V = 0$. The same hold for the emissivity coefficients $j_U = j_V = 0$. Hence, Eq. (3.7) is completely defined by the remaining extinction coefficients

$$\alpha_I = \sum_{i=1}^{N_{mix}} n_{d,i} \times \frac{\int_{a_{min}}^{a_{max}} \sum_{j=1}^{N_{mat,i}} \Xi_j C_{ext,j}(a) N_{d,i}(a) da}{\int_{a_{min}}^{a_{max}} N_{d,i}(a) da}, \tag{3.83}$$

$$\alpha_Q = \sum_{i=1}^{N_{mix}} n_{d,i} \times \frac{\int_{a_{min}}^{a_{max}} \sum_{j=1}^{N_{mat,i}} \Xi_j C_{pol,j}(a) N_{d,i}(a) da}{\int_{a_{min}}^{a_{max}} N_{d,i}(a) da}, \tag{3.84}$$

$$\kappa_Q = -\sum_{i=1}^{N_{mix}} n_{d,i} \times \frac{\int_{a_{min}}^{a_{max}} \sum_{j=1}^{N_{mat,i}} \Xi_j C_{circ,i}(a) N_{d,i}(a) da}{\int_{a_{min}}^{a_{max}} N_{d,i}(a) da}, \tag{3.85}$$

as well as the emission coefficients

$$j_I = \sum_{i=1}^{N_{mat}} n_{d,i} \times \frac{\int_{a_{min}}^{a_{max}} \sum_{j=1}^{N_{mat,i}} \Xi_j C_{abs,j}(a) B_\lambda(T_{d,i}) N_{d,i}(a) da}{\int_{a_{min}}^{a_{max}} N_{d,i}(a) da}, \tag{3.86}$$

$$j_Q = \sum_{i=1}^{N_{mat}} n_{d,i} \times \frac{\int_{a_{min}}^{a_{max}} \sum_{j=1}^{N_{mat,i}} \Xi_j C_{pabs,j}(a) B_\lambda(T_{d,i}) N_{d,i}(a) da}{\int_{a_{min}}^{a_{max}} N_{d,i}(a) da}, \tag{3.87}$$

where n_d is the dust number density, C_{circ} is the circular polarization cross section, and $B_\lambda(T_d)$ is the Planck function for a given dust temperature T_d.

HINTS:

- Both commands (`ALIG_NONPA`, `ALIG_PA`) ignore the grain alignment radius a_{alg} and upper limit of grain size a_{lim} and can not be combined with other grain alignment theories.

Emission of stochastically heated dust grains

The emission of stochastically heated dust grains can be calculated in `CMD_DUST_EMISSION` simulations. This is done for the smallest dust grains whose heat capacity is so small that the assumption of thermal equilibrium is not valid anymore. In POLARIS the stochastic heating algorithm is implemented according to the work of Camps et al. (2015). The required heat capacities will be taken from the file `'heat_capacity.dat'` in the sub-directory dust_catalog_name/ where the chosen dust catalog file is saved (see Sect. 2.4.4). In the command file, this calculation can be enabled with the following command:

`<stochastic_heating>` $a_{s,max}$

The probability distribution of temperatures due to stochastic heating will be calculated from the radiation field for all dust grain sizes up to $a_{s,max}$ to calculate the thermal emission of the dust grains in `CMD_DUST_EMISSION` simulations. Therefore, the `CMD_TEMP` simulation had to be executed with the command `<radiation_field> 1`. An alternative way is to calculate an effective temperature out of the pobability distribution of temperatures in `CMD_TEMP` simulations by using `<stochastic_heating>` $a_{s,max}$ **and** using `<radiation_field> 0` there (see also Sect. 3.7.2). If this happened and the grid contains not the radiation field or `<stochastic_heating>` is not defined in the `CMD_DUST_EMISSION` simulation, these effective temperatures are used. To avoid unwanted behavior, please use only one of the approaches (`<stochastic_heating>` at `CMD_TEMP` **or** `CMD_DUST_EMISSION`).

Polarization by scattering on spherical dust grains

The probability for a scattering event is quantified by the optical depth (see Sect. 3.1.3). Rather than converting the radiation to internal energy, it emits the same amount of energy in a different direction and the polarization state of light becomes altered where as the wavelength remains the same. When describing polarized light at the physical level of radiative transfer theory, the change in the Stokes vector can be calculated with the help of the Müller matrix or scattering matrix $\hat{M}(\psi)$. The elements of the Müller matrix are dependent and for spherical dust grains we get

$$\hat{M}(\psi) = \begin{pmatrix} M_{11} & M_{12} & 0 & 0 \\ M_{12} & M_{11} & 0 & 0 \\ 0 & 0 & M_{33} & M_{34} \\ 0 & 0 & -M_{34} & M_{33} \end{pmatrix}. \tag{3.88}$$

Note that the scattering matrix elements are also a function of material, grain size, wavelength and scattering angle. The parameters of the Stokes vector before $\vec{S}$ and after $\vec{S}'$ each scattering event is determined by the Müller matrix with $\vec{S}' \propto \hat{M}(\psi)\vec{S}$. The direction of propagation $\vec{r}'$ after the scattering defines the so called scattering plane with the original direction of radiation $\vec{r}$. The coordinate system of the Stokes vector $(\vec{r}, \vec{h}, \vec{v})$ is usually defined with respect to an external coordinate system (see Fig. 3.5). Here, $\vec{v}$ is parallel to the z-axis of the external coordinate system and perpendicular to the direction of propagation $\vec{r}$ with $\vec{h} = \vec{v} \times \vec{r}$. In order to apply the Müller matrix the Stokes vector has to be transformed into the same coordinate system as the scattering plane $(\vec{p}, \vec{s}, \vec{v})$ where $\vec{p}$ is in the scattering plane and $\vec{s} = \vec{p} \times \vec{r}$. The rotation angle α is defined to be between the vectors $\vec{s}$ and $\vec{h}$ and the scattering angle ψ between $\vec{r}'$ and $\vec{r}$ is defined by

$$\cos(\psi) = \vec{r}' \cdot \vec{r}. \tag{3.89}$$

After the scattering event, the stokes vector is in the new coordinate system $(\vec{r}', \vec{h}', \vec{v}')$. In order to align the new coordinate system again with the external coordinate system, the Stokes vector has to be rotated by an angle of α'. Finally, the change in the Stokes vector because of scattering can be calculated with

$$\vec{S}' = \hat{R}(-\alpha)\hat{M}(\psi)\hat{R}(\alpha')\vec{S}'. \tag{3.90}$$

After each scattering event a new direction of the photon package has to be determined. For scattering on spherical dust grains the phase function is $\Phi(\psi) \propto M_{11}(\psi)$. An example file for dust scattering may be written as Listing 3.4.

3.8 Line radiative transfer (LRT)

In contrast to dust RT the in LRT a transition between two characteristic energy level $E_\mathrm{i} \leftrightarrow E_\mathrm{j}$ of a certain gas species occurs at a distinct frequency ν_ij. Here, the sub-indices i and j refer to the upper and lower energy level, respectively. The probability for the different kinds of possible transition are characterized by the Einstein coefficients A_ij for spontaneous emission, B_ij for stimulated emission, and B_ji for absorption. Additionally, we consider the collisional (de-)excitation with the collision rate $C_\mathrm{ij} = n_\mathrm{c}\gamma_\mathrm{ij}$ where n_c is the number density of the collision partners and γ_ij is the collision rate. The number density n_c can be taken from the input grid while the transitions coefficients are provided by a LAMDA molecular parameters file (see Sect. 2.5).

Focusing on a single line transition between the upper level j to the lower level i the RT equation can be written as

$$\frac{\mathrm{d}I_\mathrm{ij}}{\mathrm{d}\ell} = -\kappa_\mathrm{ij} I_\mathrm{ij} + j_\mathrm{ij} \tag{3.91}$$

where emission coefficient $j_\mathrm{i,j}$ and absorption coefficient $\alpha_\mathrm{i,j}$. Here, polarization does not play a role and the RT problem is fully described by the propagation of the intensity I. In POLARIS the

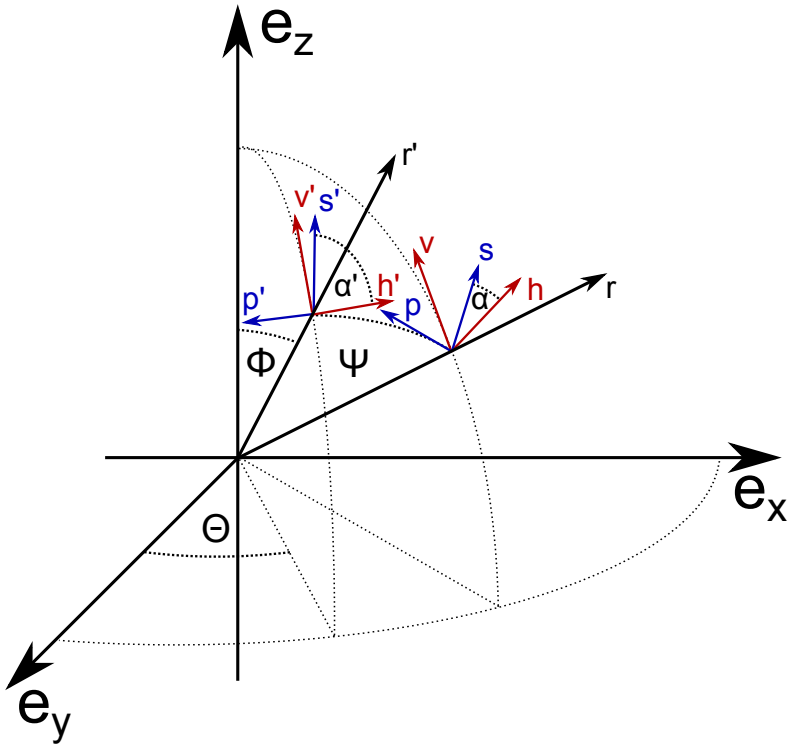


Figure 3.5: Geometrical configuration of scattering on spherical dust grains. The scattered light is redirected along the scattering plane by an angle of ψ. In order to apply the Müller matrix $\hat{M}(\psi)$, the target coordinate system $(\vec{p}, \vec{s}, \vec{r})$ has to be transformed in the lab coordinate system $(\vec{v}, \vec{h}, \vec{r})$ along the angle α.

thermal movement of gas species is the considered to be the dominant cause of line broadening. The cubic spline interpolation of the velocity (see Sect. 3.3.1 and Sect. 3.1.5) and the RKF45 solver ensures that the broadened transition lines smoothly overlap. Hence, the characteristic transition frequency ν_{ij} is no longer a single peak but follow a Gaussian distribution function

$$\Phi_{i,j}(\nu) = \frac{c}{\sqrt{\pi} v_{tot} \nu} \exp\left(- \left[\frac{c(\nu - \nu_{i,j})}{v_{tot} \nu} \right]^2 \right) \tag{3.92}$$

Here, c is the speed of light and the total thermal velocity is

$$v_{tot} = \sqrt{\left(\frac{2 k_B T_g}{m_{mol}} \right)^2 + v_{turb}^2} \tag{3.93}$$

where v_{turb} is the micro-turbulent component and $2 k_B T_g / m_{mol}$ is the kinetic velocity of the gas species. Hence, the coefficients of absorption and emission can be written as

$$j_{i,j} = \frac{h\nu}{4\pi} N n_i A_{i,j} \Phi_{i,j}(\nu) \tag{3.94}$$

and

$$\alpha_{i,j} = \frac{h\nu}{4\pi} N \left(n_j B_{j,i} - n_i B_{i,j} \right) \Phi_{i,j}(\nu) \tag{3.95}$$

respectively, where n_i is the number of electrons on the i-th level. Balancing the full RT equation with the contributions of collisional excitation and de-excitation and the mean intensity

$$J_{i,j} = \frac{1}{4\pi} \int \int I_\nu \Phi_{i,j}(\nu) \mathrm{d}\Omega \mathrm{d}\nu \tag{3.96}$$

Listing 3.4: Example command file to calculate the stellar emission scattered at dust grains.

```
<task> 1
# polarization by dust scattering
<cmd> CMD_DUST_SCATTERING

#A star in the center as radiation source
<source_star nr_photons = "1e6"> 0 0 0 1 6000

# Silicate as only dust components
<dust_component> "PATH/TO/POLARIS/input/dust/silicate_oblate.dat" 0.7 -3.5 24.2e-9
    ↪ 1.13e-6
<dust_component> "PATH/TO/POLARIS/input/dust/graphite_oblate.dat" 0.3 -3.5 24.2e-9
    ↪ 1.13e-6

#path of the input grid file
<path_grid> "PATH/WHERE/THE/RESULTS/SHOULD/BE/SAVED/temp/grid_temp.dat"

#path for all output data
<path_out> "PATH/WHERE/THE/RESULTS/SHOULD/BE/SAVED/mc_dust/"

# phase function has is Mie scattering
<phase_function> PH_MIE

# enforced first scattering is switched on
<enfsca> 1

# definition of the first detector
<detector_dust_mc nr_pixel = "290"> 1e-6 2e-6 2 0 0 4.31998E+18
# definition of a second detector
<detector_dust_mc nr_pixel = "290"> 1e-6 2e-6 2 0 0 4.31998E+18

# star as radiation source
<source_star nr_photons = "1000000"> 0.0001 0.0001 0.0001 2 4000

uses 32 cores
<nr_threads> 32

#dust to gas mass ratio
<mass_fraction> 0.01 # optional
</task>
```

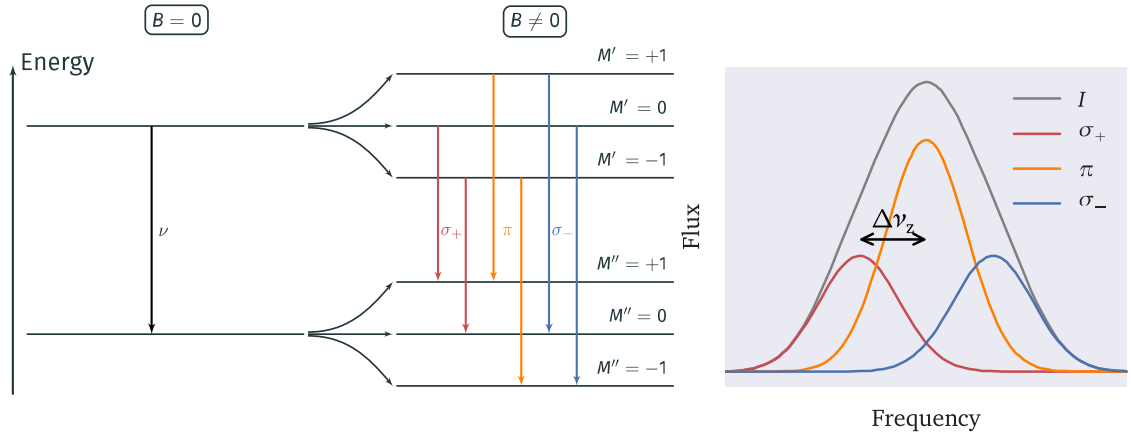


Figure 3.6: Splitting of molecular lines transitions in the absence and presence of a magnetic field (*left*) and line profile of the π and $\sigma_{\pm}$ transitions with thermal, natural, as well as pressure broadening (*right*).

gives

$$\sum_{i>j} \left[n_i A_{ij} + \left(n_i B_{ij} - n_j B_{ji}\right) J_{i,j}\right] - \sum_{i<j} \left[n_j A_{ji} + \left(n_j B_{ji} - n_i B_{ij}\right)\right] + \sum_i \left(n_i C_{ij} + n_j C_{ji}\right) \tag{3.97}$$

This system of equations is solved with the help of Gaussian elimination. The command to define all the parameters for a gas species to be observed is

```
<gas_species> "path" POP a
```

Here, the path gives the LAMDA gas species parameters file, *a* is the abundance with respect to the total gas density, and `POP` stands for the index for the different implemented methods for calculating the level populations (1: LTE, 2: FEP, 3: LVG; see Sect. 3.8.1).

Example:

```
<gas_species> "/PATH/TO/POLARIS/input/gas/co.dat" 2 1e-7
```

The defined gas species can then selected by a line detector with an ID coinciding with the order of appearance in the command file (see Sect. 3.5 for defining the line detector).

3.8.1 Level population approximations

In order to perform LRT simulations one needs to calculate the level populations. Several numerical methods are known to calculate level populations considering different local conditions.

Local thermodynamic equilibrium (LTE)

For this approximation it is assumed that the different level populations are thermalized meaning that excitation temperature T_{exc} and kinetic gas temperature T_g are equal:

$$T_g = T_{exc} = \frac{h\nu_{i,j}}{k_B} \left[\ln\left(\frac{g_i n_j}{g_j n_i}\right)\right]^{-1}. \tag{3.98}$$

Here, n_i id the number of electrons in the i-th level of the gas species, g_i is the corresponding statistical weight and $\nu_{i,j}$ is the frequency characteristic for that transition. Hence, the level populations

follow the Boltzmann distribution with

$$\frac{n_\mathrm{i}}{n_\mathrm{j}} = \frac{g_\mathrm{i}}{g_\mathrm{j}} \exp\left(\frac{h\nu_\mathrm{i,j}}{k_\mathrm{B} T_\mathrm{g}}\right) \tag{3.99}$$

This approximation can be adequate when the optical depth of a transition is so high that the collisional and radiative excitations and de-excitation, respectively, are approximately in equilibrium. In order to apply LTE in LRT simulations the flag `POP_LTE` need to be added in the command line defining the gas species.

Full escape probability (FEP)

FEP assumes that any newly created photon can escape the grid without any further interactions. The level populations can then be calculated by simply applying an external radiation field $J_\mathrm{i,j} = J_\mathrm{ext}$ to Eq. (3.97). Hence, the FEP method applies for low molecular abundances. Considering FEP in a POLARIS run the flag `POP_LTE` needs to be added.

Large velocity gradient (LVG)

This approximation assumes that a photon can easily escape from the grid because of a large gradient in the velocities of neighboring regions. For the LVG the mean intensity is due to the local transition and an external radiation field

$$J_\mathrm{i,j} = (1-\beta) S_\mathrm{i,j} + \beta J_\mathrm{ext} \tag{3.100}$$

Here, β is the probability for the photons to escape from the grid and is given by

$$\beta = \frac{1 - \exp(\tau_\mathrm{i,j})}{\tau_\mathrm{i,j}} \tag{3.101}$$

where as

$$S_\mathrm{i,j} = \frac{j_\mathrm{i,j}}{\alpha_\mathrm{i,j}} = \frac{n_\mathrm{i} A_\mathrm{i,j}}{n_\mathrm{j} B_\mathrm{ji} - n_\mathrm{i} B_\mathrm{ij}} \tag{3.102}$$

is the source function of the transition. For this approximation the required optical depth $\tau_\mathrm{i,j}$ can be calculated as

$$\tau_\mathrm{ij} = \frac{c^3 n_\mathrm{g} A_\mathrm{ij}}{8\pi\nu_\mathrm{ij}^3 \left|dv/dl\right|} \left(\frac{g_\mathrm{i}}{g_\mathrm{j}} n_\mathrm{j} - n_\mathrm{i}\right) \tag{3.103}$$

given a large enough velocity gradient $|dv/d\ell|$. This approximation can be applied when the variations in velocity are larger than the local thermal and micro-turbulent velocities. Using LVG requires the tag `POP_LVG`. At the beginning of each LRT run POLARIS pre-calculates the level population for each cell of the grid. The methods of LTE and FEP require little computational efforts. In contrast, LVG is more time consuming for a large number of cells since its need to solve Eq. (3.97) iteratively. However, the calculation of level populations is parallelized in POLARIS.

HINTS:

- If the abundance a is negative, this number means the abundance is taken from the input grid.

An exemplary LRT command file in POLARIS may look like as shown in Listing 3.5.

Listing 3.5: Example command file to calculate the spectral line emission of a gas species.

```
<task> 1
# command that defines the line radiative transfer
<cmd> CMD_LINE_EMISSION

# no subpixeling
<max_subpixel_lvl> 0

# a star in the center as radiation source
<source_star nr_photons = "1e6"> 0 0 0 1 6000

# graphite as a single dust component
<dust_component> "PATH/TO/POLARIS/input/dust/graphite_oblate.dat"

#path of the input grid file
<path_grid> "PATH/WHERE/THE/RESULTS/SHOULD/BE/SAVED/temp/grid_temp.dat"

#path for all output data
<path_out> "PATH/WHERE/THE/RESULTS/SHOULD/BE/SAVED/mc_dust/"

# definition of the rotation axes
<axis1> 1 0 0
<axis2> 0 1 1

# 1st gas species is SO
<gas_species> "/home/data/so.dat" 1 1e-7
# 2nd gas species is CO
<gas_species> "/home/data/co.dat" 1 1e-5

#detector for 1st gas species with transition nr. 4 in LAMDA file)
<detector_line nr_pixel = "380" vel_channels = "35"> 1 4 1 1e5 0 0 4.73E+016
#detector for 2nd gas species with transition nr. 3 in LAMDA file)
<detector_line nr_pixel = "380" vel_channels = "35"> 2 3 1 1e5 0 0 4.73E+016

uses 32 cores
<nr_threads> 32

#dust to gas mass ratio
<mass_fraction> 0.01 # optional
</task>
```

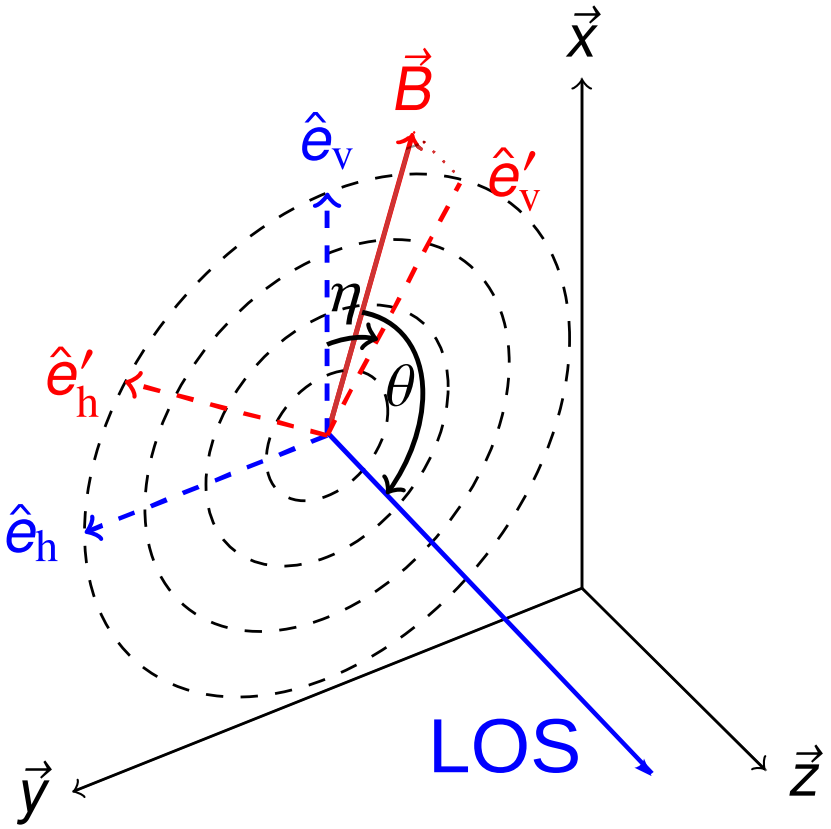


Figure 3.7: Definition of the coordinate systems and rotation angle for the LRT with Zeeman effect. The external coordinate system given by the unit vectors $\hat{e}_v$ and $\hat{e}_v$ and the coordinate system of the Stokes vector is defined by $\hat{e}'_v$ and $\hat{e}'_v$. The angle η is between the two vectors $\hat{e}_v$ and $\hat{e}'_v$ while the angle θ is between the magnetic field $\vec{B}$ and the LOS.

3.8.2 LRT with Zeeman effect

In POLARIS, the implementation of LRT including Zeeman splitting is based on the work of Larsson et al. (2014). The energy levels of a gas species can split into separate distinct sub-level when a gas species is exposed by an external magnetic field (see Sect. 3.6). This splitting of spectral lines is referred as to Zeeman effect. Here, the degeneracy of the sub-level is given by $2J+1$ and the magnetic quantum number can get the values of

$$m_J = [-J, -J+1, ..., J-1, J]\,. \tag{3.104}$$

The energy of any transition is given by:

$$\Delta E = -\left(\vec{\mu}\vec{B}\right) = g_J \mu_0 \Delta m_J B. \tag{3.105}$$

Here, B is the magnetic field strength, μ_0 is the Bohr magneton. The Landè factor g_J is defined as a fraction of quantum numbers and a gas species specific constant g_s by

$$g_J = g_s \frac{J(J+1) + S(S+1) + N(N+1)}{2J(J+1)}. \tag{3.106}$$

The difference in energy between different sub-level allows to calculate the characteristic frequency of each transition

$$\nu_0 = \frac{B\mu_0}{h}\left(g_{J'} m_{J'} - g_{J''} m_{J''}\right) \tag{3.107}$$

where $'$ and $''$ indicate the upper and lower sub-level, respectively.

However, not all permutations of transitions are possible. The rules for allowed transitions are $\Delta m_{\mathrm{J}} = \pm 1$ (called $\sigma_{\pm}$ transition), $\Delta m_{\mathrm{J}} = 0$ (π transition) if $\Delta J = 0, \pm 1$. As with dust polarization the LRT problem needs to be solved in the Stokes vector formalism, since the Zeeman effect leads to light polarization. The equation of LRT including the Zeeman effect is similar to Eq. (3.7) an can be written as:

$$\frac{\mathrm{d}\vec{I}_\nu}{\mathrm{d}\ell} = -\left(\hat{K}_{\nu,\mathrm{A}} + \hat{K}_{\nu,\mathrm{B}}\right)\left(\vec{I}_\nu - \vec{S}_\nu\right). \tag{3.108}$$

The matrices $\hat{K}_{\nu,\mathrm{A}}$ and $\hat{K}_{\nu,\mathrm{B}}$ are for extinction and magneto-optical effects. For extinction the matrix can be calculated as

$$\hat{K}_{\nu,\mathrm{A}} = \frac{n_{\mathrm{i}}}{2} s_{\mathrm{N}',\mathrm{N}'',\mathrm{J}',\mathrm{J}''} \sum_{\mathrm{m}',\mathrm{m}''} \left(s_{\mathrm{m}',\mathrm{m}''} F_{\mathrm{A}}(\nu', a)\hat{A}_{\mathrm{m}',\mathrm{m}''}\right) \tag{3.109}$$

and the magneto mechanical matrix is defined to be:

$$\hat{K}_{\nu,\mathrm{B}} = n_{\mathrm{i}} s_{\mathrm{N}',\mathrm{N}'',\mathrm{J}',\mathrm{J}''} \sum_{\mathrm{m}',\mathrm{m}''} \left(s_{\mathrm{m}',\mathrm{m}''} F_{\mathrm{B}}(\nu', a)\hat{B}_{\mathrm{m}',\mathrm{m}''}\right). \tag{3.110}$$

Here, $s_{\mathrm{N}',\mathrm{N}'',\mathrm{J}',\mathrm{J}''}$ is the line strength for the $N' \rightarrow N''$ and $J' \rightarrow J''$ transition and $s_{\mathrm{m}',\mathrm{m}''}$ is the relative line strength between the Zeemman $m_{J'} \rightarrow m_{J''}$ sub-level.

The angles θ and η are defined to be between magnetic field and line of sight and the coordinate system of the Stokes vector and the coordinate system of the model space (see Fig. 3.7 for details) in order to take care of an arbitrary orientations. This allow to calculate the polarization rotation matrices

$$\hat{A}_{\sigma\pm} = \begin{pmatrix} 1+\cos^2(\theta) & \cos(2\eta)\sin^2(\theta) & \sin(2\eta)\sin^2(\theta) & \mp\cos(\theta) \\ \cos(2\eta)\sin^2(\theta) & 1+\cos^2(\theta) & 0 & 0 \\ \sin(2\eta)\sin^2(\theta) & 0 & 1+\cos^2(\theta) & 0 \\ \mp\cos(\theta) & 0 & 0 & 1+\cos^2(\theta) \end{pmatrix} \tag{3.111}$$

and

$$\hat{A}_{\pi} = \begin{pmatrix} \sin^2(\theta) & -\cos(2\eta)\sin^2(\theta) & -\sin(2\eta)\sin^2(\theta) & 0 \\ -\cos(2\eta)\sin^2(\theta) & \sin^2(\theta) & 0 & 0 \\ \sin(2\eta)\sin^2(\theta) & 0 & \sin^2(\theta) & 0 \\ \mp\cos(\theta) & 0 & 0 & 1+\cos^2(\theta) \end{pmatrix} \tag{3.112}$$

for the $\sigma_{\pm}$ and pi transitions, respectively. For the magneto-optical effects the rotation matrices are defined as

$$\hat{B}_{\sigma\pm} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & \mp 2\cos(\theta) & \sin(2\eta)\sin^2(\theta) \\ 0 & \mp 2\cos(\theta) & 0 & -\cos(2\eta)\sin^2(\theta) \\ 0 & \sin(2\eta)\sin^2(\theta) & -\cos(2\eta)\sin^2(\theta) & 0 \end{pmatrix} \tag{3.113}$$

and

$$\hat{B}_{\pi} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\sin(2\eta)\sin^2(\theta) \\ 0 & 0 & 0 & -cos(2\eta)\sin^2(\theta) \\ 0 & \sin(2\eta)\sin^2(\theta) & -\cos(2\eta)\sin^2(\theta) & 0 \end{pmatrix}. \tag{3.114}$$

Subsequently, for the polarization exist two extreme cases. In the first case the magnetic field is parallel along the line of sight. Here, the Zeeman effect is governed by the $\sigma_{\pm}$ transitions and radiation becomes only circularly polarized. In the second case B is perpendicular along the line of

ΔJ	π	$\sigma_{\pm}$
+1	$\frac{3(J+1)^2-3m_J^2}{2(J+1)(2J+1)(2J+3)}$	$\frac{3(J+1\pm m_J)(J+2\pm m_J)}{4(J+1)(2J+1)(2J+3)}$
0	$\frac{3m_J^2}{J(J+1)(2J+1)}$	$\frac{3(J\mp m_J)(J+1\pm m_J)}{2J(J+1)(2J+1)}$
-1	$\frac{3J^2-3m_J^2}{2J(2J-1)(2J+1)}$	$\frac{3(J\mp m_J)(J-1\mp m_J)}{4J(2J-1)(2J+1)}$

Table 3.1: Relative line strength $s_{m',m''}$ for different ΔJ, π, and $\sigma_{\pm}$ transitions.

sight coming with the $\sigma_{\pm}$ transitions polarized vertically to the magnetic field and the π transition polarized horizontally to the magnetic field. The relative line strength $s_{m',m''}$ is defined for each transition as a characteristic fraction of quantum numbers (see Table 3.1).

As with simple LRT without Zeeman effects the characteristic Zeeman transitions are thermally broadened caused by different relative velocities of the gas species with respect to the observer. Additionally, Zeeman transitions are naturally broader as a consequence of quantum mechanics. Energy states with a shorter decay rate is associated with a larger uncertainty in energy and vise versa because of Heisenberg's uncertainty principle. Consequently, the transition frequency is broader dependent on the decay rate of each level. Occasional collisions with other gas species can cause collisional de-excitations. This depends on the pressure of the gas and each line shape broadens even more. In Eq. (3.109) and Eq. (3.110), the functions $F_A(\nu', a)$ and $F_B(\nu', a)$ are the line shape function that models the thermal, natural, as well as pressure broadening of line transition. The parameters are defined to be as

$$\nu' = \frac{\nu_0 + \Delta\nu_0 - \nu}{\nu_D}, \tag{3.115}$$

$$a = \frac{\gamma}{4\pi\Delta\nu_D}, \tag{3.116}$$

where $\Delta\nu_D$ is the Doppler broadening width and γ is the pressure broadening width. Both $F_A(\nu', a)$ and $F_B(\nu', a)$ can be combined resulting in the Fadeeva function defined as

$$w(z) = F_A(\nu', a) + iF_B(\nu', a) = \frac{1}{2\pi}\int_{-\infty}^{\infty}\frac{e^{-y^2}}{z-y}dy \tag{3.117}$$

with $z = \nu' + ia$. In the POLARIS code the solution to this problem is solved with the help of the C++ Fadeeva package[1]. The Fadeeva function may in principle also be applied for simple LRT. However, the LAMDA files do not provide the necessary paramters to do so. This may be solved in a future version of POLARIS. For Zeeman splitting there is a correlation between the Stokes V parameter and the first velocity derivative $dI/d\nu$

$$V = \frac{dI}{d\nu}\Delta\nu\cos(\theta) \tag{3.118}$$

where

$$\Delta\nu = \frac{\mu_0}{h}\left(g_{J'}m_{J'} - g_{J''}m_{J''}\right). \tag{3.119}$$

Bohr magneton μ_0 and the Planck constant h are known and the quantities $g_{J'}$ and $m_{J'}$ can be pre-calculated for each gas species. Hence, the line of sight magnetic field strength $B\cos(\theta)$ can be calculated with POLARIS by simulating the Stokes I and V parameters and the subsequent fitting of $dI/d\nu$ onto V. In POLARIS a LRT run including the Zeeman effect is almost identical to

[1] See http://ab-initio.mit.edu/wiki/index.php/Faddeeva_Package, Copyright © 2012 Massachusetts Institute of Technology

a simple RT as presented in the previous section. However, simulating the Zeeman effect requires the quantities of the Landè factor (see Eq. 3.106) and the relative line strength (see Table 3.1). The values for these quantities need to be delivered in an extra file (see Sect. 2.5.2) corresponding to the Leiden molecular file. Instead of a single path for the LAMDA file on need to add a second path to the corresponding Zeeman file:

```
<gas_species> "path" POP a "path Zeeman"
```

Example:

```
<gas_species> "/PATH/TO/POLARIS/input/gas/oh.dat" 1 1e-5 "/PATH/TO/POLARIS/input/gas/
    ↪ oh_zeeman.dat"
```

An exemplary LRT command file in POLARIS with Zeeman splitting may look like as shown in Listing 3.6.

3.9 Synchrotron RT

When a moving electron is forced on a curved path or an orbit it emits radiation. In particular free electrons in the ISM gyrate around the magnetic field lines. This leads to polarized radiation as well as rotation of the polarization plane of background radiation (Faraday effect). For a complete synchrotron RT, one needs to consider two different species of electrons: cosmic ray (CR) electrons and thermalized relativistic electrons. Synchrotron intensity as well as linear and circular polarization emerges mostly from CR electrons where as thermal electrons dominate Faraday rotation (FR) and Faraday conversion (FC). The RT problem is identical to that of Eq. (3.7). The synchrotron required coefficients can be calculated by integrals over modified Bessel functions. However, due to performance reasons, the implementation in POLARIS follows the approach of applying fitting functions approximating the exact integral solutions. These fitting functions provide high accuracy solutions for typical ISM-like conditions. The implementation in POLARIS is based on the functions presented in Pandya et al. (2016) and Dexter (2016).

3.9.1 CR electrons

The energy of CR electrons follow a power-law distribution

$$N_{\mathrm{CR}}(\gamma) = \begin{cases} n_{\mathrm{CR}} \gamma^{p} (p-1) \left(\gamma_{\mathrm{min}}^{p-1} - \gamma_{\mathrm{max}}^{p-1} \right) & \text{if } \gamma_{\mathrm{min}} < \gamma < \gamma_{\mathrm{max}} \\ 0 & \text{otherwise} \end{cases} \tag{3.120}$$

where $\gamma = (1-\beta^2)^{1/2}$ is the Lorentz factor, $\beta = v/c$, n_{CR} is the CR electron density p is a power-law index. The distribution has sharp lower and upper cut-offs at γ_{min} and γ_{max}, respectively. By rotating the Stokes vector in the direction of the magnetic field, some of the coefficients can be eliminated. For extinction this leads to $\alpha_{\mathrm{U}} = \kappa_{\mathrm{U}} = 0$ and for the emissivity coefficient $j_{\mathrm{U}} = 0$. The

Listing 3.6: Example command file to calculate the spectral line emission with Zeeman splitting of a gas species.

```
<task> 1
# command that defines the line radiative transfer
<cmd> CMD_LINE_EMISSION

# no subpixeling
<max_subpixel_lvl> 0

# a star in the center as radiation source
<source_star nr_photons = "1e6"> 0 0 0 1 6000

# graphite as a single dust component
<dust_component> "PATH/TO/POLARIS/input/dust/graphite_oblate.dat"

#path of the input grid file
<path_grid> "PATH/WHERE/THE/RESULTS/SHOULD/BE/SAVED/temp/grid_temp.dat"

#path for all output data
<path_out> "PATH/WHERE/THE/RESULTS/SHOULD/BE/SAVED/mc_dust/"

# definition of the rotation axes
<axis1> 1 0 0
<axis2> 0 1 1

# 1st gas_species is SO
# this gas_species is WITHOUT Zeeman effect
<gas_species> "PATH/TO/POLARIS/input/gas/so.dat" 1 1e-7

# 2nd gas_species is OH
# the abundance is taken from the 3rd position (hence -3) of the grid ratios
# this gas_species is WITH Zeeman effect
<gas_species> "PATH/TO/POLARIS/input/gas/oh.dat" 1 -3 "PATH/TO/POLARIS/input/gas/so_
    ↪ zeeman.dat"

# detector for 1st gas species with transition nr. 4 in LAMDA file)
<detector_line nr_pixel = "380" vel_channels = "35"> 1 4 1 1e5 0 0 4.73E+016
# detector for 2nd gas_species with transition nr. 2 in LAMDA file AND Zeeman file)
<detector_line nr_pixel = "380" vel_channels = "35"> 2 2 1 1e5 0 0 4.73E+016
# etector for 2nd gas_species with transition nr. 3 in LAMDA file AND Zeeman file)
<detector_line nr_pixel = "380" vel_channels = "35"> 2 3 1 1e5 0 0 4.73E+016

#uses 32 cores
<nr_threads> 32

#dust to gas mass ratio
<mass_fraction> 0.01 # optional
</task>
```

remaining coefficients of emissivity can be approximately be calculated by

$$j_{\mathrm{I}}(\lambda) = \gamma_{\mathrm{min}}^{1-p} \frac{1}{\lambda_{\mathrm{c}}} \frac{n_{\mathrm{CR}} e^2 3^{\frac{p}{2}} (p-1) \sin(\vartheta)}{2(p+1)\left(\gamma_{\mathrm{min}}^{1-p} - \gamma_{\mathrm{max}}^{1-p}\right)} \times \Gamma\left(\frac{3p-1}{12}\right) \Gamma\left(\frac{3p+19}{12}\right) \left(\frac{\lambda_{\mathrm{c}}}{\lambda \sin(\vartheta)}\right)^{-\frac{p-1}{2}}, \tag{3.121}$$

$$j_{\mathrm{Q}}(\lambda) = j_{\mathrm{I}}(\lambda) \left(-\frac{p+1}{p+7/3}\right), \tag{3.122}$$

$$j_{\mathrm{V}}(\lambda) = j_{\mathrm{I}}(\lambda) \left(-\frac{171}{250} \frac{\lambda_{\mathrm{c}} p^{\frac{1}{2}}}{3\lambda \tan(\vartheta)}\right) \tag{3.123}$$

where $\Gamma(x)$ is the gamma function, the angle θ is between the direction of light propagation and the magnetic field, and the quantity

$$\lambda_{\mathrm{c}} = \frac{2\pi m_{\mathrm{e}} c^2}{eB} \tag{3.124}$$

is defined to be the characteristic cyclotron wavelength. It follow that the maximal possible degree of linear polarization is directly connected to the power-law index p since:

$$\max(P_{\mathrm{I}}) = \frac{|j_{\mathrm{Q}}|}{j_{\mathrm{I}}} = \frac{p+1}{p+7/3}. \tag{3.125}$$

The approximate solutions of CR electron emission coefficients are

$$\alpha_{\mathrm{I}}(\lambda) = \gamma_{\mathrm{min}}^{1-p} \frac{n_{\mathrm{CR}} e^2}{\nu m_{\mathrm{e}} c} \frac{3^{\frac{p+1}{2}} (p-1)}{4\left(\gamma_{\mathrm{min}}^{1-p} - \gamma_{\mathrm{max}}^{1-p}\right)} \times \Gamma\left(\frac{3p+12}{12}\right) \Gamma\left(\frac{3p+22}{12}\right) \left(\frac{\lambda_{\mathrm{c}}}{\lambda \sin(\vartheta)}\right)^{-\frac{p+2}{2}}, \tag{3.126}$$

$$\alpha_{\mathrm{Q}}(\lambda) = \frac{996\ \alpha_{\mathrm{I}}(\lambda)}{1000} \left(-\frac{3(p-1)^{\frac{43}{500}}}{4}\right), \tag{3.127}$$

$$\alpha_{\mathrm{V}}(\lambda) = \alpha_{\mathrm{I}}(\lambda)\, k_{\mathrm{V}}(\vartheta) \left[-\frac{7}{4}\left(\frac{71p}{100} + \frac{22}{625}\right)^{\frac{197}{500}}\right] \times \left[\left(\sin^{-\frac{48}{25}}(\vartheta) - 1\right)^{\frac{64}{125}} \left(\frac{\lambda_{\mathrm{c}}}{\lambda}\right)^{-\frac{1}{2}}\right]. \tag{3.128}$$

In contrast to polarized RT with dust, for synchrotron there is also a conversion between the Stokes components I and V as well as Q and U. However, for CR electrons the Faraday coefficients have just an minor effect on linear polarization compared to those of thermal electrons and we consider them approximately to be $\kappa_{\mathrm{Q}} = \kappa_{\mathrm{V}} = 0$.

3.9.2 Thermal electrons

Thermal electrons follow a Maxwell Jüttner distribution (a relativistic Maxwellian energy distribution):

$$N_{\mathrm{th}}(\gamma) = \frac{n_{\mathrm{th}} \gamma^2 \beta \exp(-\gamma/\Theta)}{\Theta K_2(1/\Theta)} \tag{3.129}$$

where n_{th} is the thermal electron density and the dimensionless temperature is defined as

$$\Theta = \frac{k_{\mathrm{B}} T_{\mathrm{e}}}{m_{\mathrm{e}} c^2} \tag{3.130}$$

Here, k_B is the Boltzmann constant, T_e is the electron temperature and m_e is the electron mass. For typical ISM and molecular cloud conditions $\Theta \ll 1$. Hence, thermal electrons do not contribute to emission and absorption and we assume $j_{I,Q,V} = \alpha_{I,Q,V} = 0$. In the case of thermal electrons the coefficients of FR and FC can be written as

$$\kappa_Q(\lambda, \vartheta) = -\frac{1}{4\pi^2} \frac{n_{th} e^4 B^2}{m_e^3 c^6} \lambda^3, \tag{3.131}$$

$$\kappa_V(\lambda, \vartheta) = -\frac{1}{2\pi} \frac{n_{th} e^2 B}{m_e^2 c^4} \lambda^2 \cos(\vartheta). \tag{3.132}$$

3.9.3 Synchrotron run with both electrons species

A synchrotron run in POLARIS requires at least a magnetic field component. Furthermore, a thermal electron number density n_{th} or a CR electron distribution meaning the quantities of number density of CR electrons n_{CR}, the minimal Lorentz-factor γ_{min}, the maximal Lorentz-factor γ_{max}, and the power-law index p need to be provided by the grid (see Table 2.3 for details). When both CR electron as well thermal electrons are defined in the grid POLARIS solves simultaneously the RT equation without the FR and FC coefficients κ_Q and κ_V, respectively, as well as the full RT equation. This allows to investigate the influence of Faraday effects in detail. Consequently, the resulting POLARIS synchrotron detector file contains two times the information compared to a POLARIS dust detector file. In detail the detector fits file contains a set of quantities consisting of the Stokes I, Q, U, V parameters calculated with $\kappa_Q = \kappa_V = 0$ followed by a placeholder and the thermal electron column density N_{th}. The following set contains the I, Q, U, V for the full set of coefficients including Faraday effects as well as the quantity $\lambda^2 \times \kappa_V$ and the CR electron column density $N_{CR} = 0$. An exemplary synchrotron command file is shown in Listing 3.7.

Listing 3.7: Example command file to calculate the synchrotron polarization and emission.

```
<task> 1
<cmd> CMD_SYNCHROTRON #command that defines the synchrotron ray-tracing

#plane synchrotron detector
<detector_sync nr_pixel = "800"> 7.35E-01 7.35E-01 1 1 45.0 45.0 1.543e+19

#maximal sub-pixel level
<max_subpixel_lvl> 0

#input grid incuding gas and electron densities
<path_grid> "/home/User/synchrotron/grid.dat"

#path for the results
<path_out> "/home/User/synchrotron/results/pol_syn/"

#number of threads
<nr_threads> 64

#plotting of the input midplanes
<write_inp_midplanes> 128

#definition of the rotation axis
<axis1> 1 1 0 #optional
<axis2> 0 1 1 #optional
</task>
```

4 Output data

POLARIS creates automatically all the directories it needs to store the resulting simulation data and plots if they do not already exist. Hence, there's no need for the user to create any extra directory in advance. The output path is completely defined by the command:

```
<path_out> "output_path"
```

were the path needs to follow the rules of the applied operating system. Example:

```
# Linux or MacOS path
<path_out> "/home/user/polaris_projects/results/important_project01/"

# Windows path
<path_out> "C:\user\polaris_projects\results\important_project01\"
```

If a directory is missing along the hierarchy of the path it will be created by POLARIS from there. It is also recommended to use absolute paths instead of paths relative to the directory containing the POLARIS executable. Additionally, POLARIS creates two the two directory for data/ and plots/ in the output_path/. After each POLARIS simulation, the results are stored in the `data` directory. The only exception is the resulting grid file. Here, POLARIS saves each newly created grid directly into the output_path/.

4.1 Output grids

The POLARIS simulations to calculate the dust temperature distribution (`CMD_TEMP`) and the calculation of the maximal dust grain alignment radius a_{alg} (`CMD_RAT`) create a new grid including this information. These grids have exactly the same file format as the input grid. If there is no data position reserved in the input grid, then the data length of the grid will be extended. In this case the total size of the resulting grid will be larger than the input grid. However, if the input grid already contains the values of dust temperature or alignment radius, these parameters will be overwritten. For the dust temperature distribution, the amount of additional data positions varies depending on the chosen command. If `<full_dust_temp>` is enabled, a data position of the dust temperature for each dust grain size will be created and filled in the grid. If `<stochastic_heating> max_size` is set, the dust grains with a size less than the `max_size` get, in addition to `<full_dust_temp>`, a data position for each temperature defined in the `'heat_capacity.dat'` file.

The new grids have the file names `'grid_temp.dat'` in the case of dust heating and `'grid_rat.dat'` in the case of the maximal dust grain alignment radius a_{alg}, respectively. These grids are the only files that are directly written into the output path instead of the data/ directory.

HINT: Each newly created grid is in SI and no further unit conversion is required for any followup simulation.

4.2 Detector files

Most results from POLARIS simulations are saved as compressed '.fits.gz' files.[1] To realize this, POLARIS takes advantage of the CCfits and cfitsio packages. The POLARIS package is shipped with both libraries and the installer takes care of their installation. In the following, the structure of the '.fits' files for each simulation will be described. To use uncompressed '.fits' files, the user has to change the file extension at FITS_COMPRESS_EXT in src/Typedefs.h. These files can be opened with, for example, SAOImageDS9 or a python script using astropy[2].

4.2.1 Midplane cuts

With every run of POLARIS cuts through the midplanes of the grid can be created. Up to two different midplane '.fits' files will be created with each task that included the <write_inp_midplanes> and/or <write_out_midplanes> commands. The 'input_midplane.fits' file can be created by each run of POLARIS and includes cuts of the quantities contained in the input grid. Each simulation with POLARIS that is creating an output grid (see above) can also create an 'output_midplane.fits' file that contains the cuts for each quantity that changed or was created due to the simulation (for additional quantities see Table 2.10). A description of the structure of these midplane '.fits' files can be found in Listing 4.1, which shows the header and additional comments. The header of the midplane files contains three to four different representations (units) of the spatial axes. With a viewing program like ds9, switching between these representations can be done by changing the WCS.

If the command <write_3d_midplanes> is used, the midplanes are created slightly different. Instead of having slices through each plane ('xy', 'xz', 'yz'), only one plane is used and multiple slices at different positions of the third axis are created to obtain a 3-dimensional impression of the grid quantities.

4.2.2 Emission maps

Simulations of the thermal dust grain emission CMD_DUST_EMISSION, the radiation scattered at dust grains CMD_DUST_SCATTERING and synchrotron emission CMD_SYNCHROTRON will create an emission map with the filename 'polaris_detector_nrXXXX.fits'. The 'X' stand for the detector index with four digits (e.g. 0001 for first detector defined in the 'cmd_file'). In addition to the spatial axes, one axis is provided for the different wavelength IDs and one for the quantities like intensity and optical depth A description of the structure of these emission map '.fits' files can be found in Listing 4.2, which shows the header and additional comments.

4.2.3 Emission SEDs

Simulations of the thermal dust grain emission CMD_DUST_EMISSION and the radiation scattered at dust grains CMD_DUST_SCATTERING will create a spectral energy distribution from the emission maps with the filename 'polaris_detector_nrXXXX_sed.fits'. The 'X' stand for the detector index with four digits (e.g. 0001 for first detector defined in the 'cmd_file'). These files are essentially a sum over the spatial axes of the emission maps to provide fast access to the SED of large simulations A description of the structure of these emission SED '.fits' files can be found in Listing 4.3, which shows the header and additional comments.

[1] General information about '.fits' files can be found at https://fits.gsfc.nasa.gov/fits_documentation.html

[2] See https://docs.astropy.org/en/stable/generated/examples/io/plot_fits-image.html for a guide on how to read and plot an image from a FITS file with astropy.

Listing 4.1: Header of an 'input_midplane.fits' file including comments to explain the file structure.

```
SIMPLE = T / file does conform to FITS standard
BITPIX = -64 / number of bits per data pixel
NAXIS = 4 / number of data axes
NAXIS1 = 256 / length of data axis 1
NAXIS2 = 256 / length of data axis 2
NAXIS3 = 3 / different planes (xy, xz, yz)
NAXIS4 = 13 / different quantities (see MIDPLANEX)
EXTEND = T / FITS dataset may contain extensions
COMMENT FITS (Flexible Image Transport System) format is defined in 'Astronomy
COMMENT and Astrophysics', volume 376, page 359; bibcode: 2001A&A...376..359H
CTYPE1 = 'PARAM ' / type of unit 1
CRVAL1 = -44704051205273.4 / value of axis 1
CRPIX1 = 1 / pixel where CRVAL1 is defined
CDELT1 = 350620009453.125 / delta of axis 1
CUNIT1 = 'm ' / unit of axis 1
CTYPE1B = 'PARAM ' / type of unit 1
CRVAL1B = -298.828125 / value of axis 1
CRPIX1B = 1 / pixel where CRVAL1 is defined
CDELT1B = 2.34375 / delta of axis 1
CUNIT1B = 'AU ' / unit of axis 1
CTYPE1C = 'PARAM ' / type of unit 1
CRVAL1C = -0.00144875963301446 / value of axis 1
CRPIX1C = 1 / pixel where CRVAL1 is defined
CDELT1C = 1.13628206510938E-05 / delta of axis 1
CUNIT1C = 'pc ' / unit of axis 1
CTYPE2 = 'PARAM ' / type of unit 2
CRVAL2 = -44704051205273.4 / value of axis 2
CRPIX2 = 1 / pixel where CRVAL2 is defined
CDELT2 = 350620009453.125 / delta of axis 2
CUNIT2 = 'm ' / unit of axis 2
CTYPE2B = 'PARAM ' / type of unit 2
CRVAL2B = -298.828125 / value of axis 2
CRPIX2B = 1 / pixel where CRVAL2 is defined
CDELT2B = 2.34375 / delta of axis 2
CUNIT2B = 'AU ' / unit of axis 2
CTYPE2C = 'PARAM ' / type of unit 2
CRVAL2C = -0.00144875963301446 / value of axis 2
CRPIX2C = 1 / pixel where CRVAL2 is defined
CDELT2C = 1.13628206510938E-05 / delta of axis 2
CUNIT2C = 'pc ' / unit of axis 2
HIERARCH MIDPLANE1 = 'gas_density' / quantity of 1. image
HIERARCH MIDPLANE2 = 'gas_temperature' / quantity of 2. image
HIERARCH MIDPLANE3 = 'delta ' / quantity of 3. image
HIERARCH MIDPLANE4 = 'mag_total' / quantity of 4. image
HIERARCH MIDPLANE5 = 'mag_x ' / quantity of 5. image
HIERARCH MIDPLANE6 = 'mag_y ' / quantity of 6. image
HIERARCH MIDPLANE7 = 'mag_z ' / quantity of 7. image
HIERARCH MIDPLANE8 = 'vel_total' / quantity of 8. image
HIERARCH MIDPLANE9 = 'vel_x ' / quantity of 9. image
HIERARCH MIDPLANE10 = 'vel_y ' / quantity of 10. image
HIERARCH MIDPLANE11 = 'vel_z ' / quantity of 11. image
HIERARCH MIDPLANE12 = 'mach ' / quantity of 12. image
HIERARCH MIDPLANE13 = 'larm ' / quantity of 13. image
```

Listing 4.2: Header of an 'polaris_detector_nrXXXX.fits' file including comments to explain the file structure.

```
SIMPLE = T / file does conform to FITS standard
BITPIX = -64 / number of bits per data pixel
NAXIS = 4 / number of data axes
NAXIS1 = 256 / length of data axis 1
NAXIS2 = 256 / length of data axis 2
NAXIS3 = 1 / the wavelength axis
NAXIS4 = 6 / different quantities (see CUNIT4)
EXTEND = T / FITS dataset may contain extensions
COMMENT FITS (Flexible Image Transport System) format is defined in 'Astronomy
COMMENT and Astrophysics', volume 376, page 359; bibcode: 2001A&A...376..359H
CTYPE1 = 'PARAM ' / type of unit 1
CRVAL1 = -44704051205273.4 / value of axis 1
CRPIX1 = 1 / pixel where CRVAL1 is defined
CDELT1 = 350620009453.125 / delta of axis 1
CUNIT1 = 'm ' / unit of axis 1
CTYPE1A = ... See midplane header ...
CTYPE2 = 'PARAM ' / type of unit 2
CRVAL2 = -44704051205273.4 / value of axis 2
CRPIX2 = 1 / pixel where CRVAL2 is defined
CDELT2 = 350620009453.125 / delta of axis 2
CUNIT2 = 'm ' / unit of axis 2
CTYPE2A = ... See midplane header ...
CTYPE3 = 'PARAM ' / type of unit 3
CRVAL3 = 1 / value of axis 3
CRPIX3 = 1 / pixel where CRVAL3 is defined
CDELT3 = 1 / delta of axis 3
CUNIT3 = 'Wavelength index' / unit of axis 3
CTYPE4 = 'PARAM ' / type of unit 4
CRVAL4 = 1 / value of axis 4
CRPIX4 = 1 / pixel where CRVAL4 is defined
CDELT4 = 1 / delta of axis 4

CUNIT4 = 'I, Q, U, V [Jy/px], optical depth, column density [m^-2]' / unit of axis 4
ETYPE = 'thermal emission' / type of emission
  ... or ...
CUNIT4 = 'I, Q, U, V, I_direct, I_scat [Jy/px]' / unit of axis 4
ETYPE = 'scattered emission / direct stellar emission' / type of emission
  ... or ...
CUNIT4 = 'I, Q, U, V, [Jy/px], ...' / unit of axis 4
ETYPE = 'synchotron emission' / type of emission

ID = 1 / detector id
HIERARCH WAVELENGTH1 = 0.000849466675 / value of 1. wavelength
DISTANCE= 4.31994861405407E+18 / distance to object
RAXIS1X = 1. / rotation axes 1 (x component)
RAXIS1Y = 0. / rotation axes 1 (y component)
RAXIS1Z = 0. / rotation axes 1 (z component)
RANGLE1 = 0. / rotation angle 1 [\si{\degree}]
RAXIS2X = 0. / rotation axes 2 (x component)
RAXIS2Y = 1. / rotation axes 2 (y component)
RAXIS2Z = 0. / rotation axes 2 (z component)
RANGLE2 = 0. / rotation angle 2 [\si{\degree}]
DETGRID = 'Plane / Cartesian background grid' / description of the detector grid
```

Listing 4.3: Header of an 'polaris_detector_nrXXXX_sed.fits' file including comments to explain the file structure.

```
SIMPLE = T / file does conform to FITS standard
BITPIX = -64 / number of bits per data pixel
NAXIS = 3 / number of data axes
NAXIS1 = 1 / the wavelength axis
NAXIS2 = 1 / dummy to have one line, if opened as image
NAXIS3 = 6 / different quantities (see CUNIT4)
EXTEND = T / FITS dataset may contain extensions
COMMENT FITS (Flexible Image Transport System) format is defined in 'Astronomy
COMMENT and Astrophysics', volume 376, page 359; bibcode: 2001A&A...376..359H
CTYPE1 = 'PARAM ' / type of unit 1
CRVAL1 = 1 / value of axis 1
CRPIX1 = 1 / pixel where CRVAL1 is defined
CDELT1 = 1 / delta of axis 1
CUNIT1 = 'Wavelength index' / unit of axis 1
CTYPE2 = 'PARAM ' / type of unit 2
CRVAL2 = 1 / value of axis 2
CRPIX2 = 1 / pixel where CRVAL2 is defined
CDELT2 = 1 / delta of axis 2
CUNIT2 = 'None ' / unit of axis 2

CUNIT3 = 'I, Q, U, V [Jy/px], optical depth' / unit of axis 3
ETYPE = 'thermal emission' / type of emission
  ... or ...
CUNIT3 = 'I, Q, U, V, I_scat [Jy/px]' / unit of axis 3
ETYPE = 'scattered emission / direct stellar emission' / type of emission

ID = 1 / detector id
HIERARCH WAVELENGTH1 = 0.000849466675 / value of 1. wavelength
DISTANCE= 4.31994861405407E+18 / distance to object
RAXIS1X = 1. / rotation axes 1 (x component)
RAXIS1Y = 0. / rotation axes 1 (y component)
RAXIS1Z = 0. / rotation axes 1 (z component)
RANGLE1 = 0. / rotation angle 1 [\si{\degree}]
RAXIS2X = 0. / rotation axes 2 (x component)
RAXIS2Y = 1. / rotation axes 2 (y component)
RAXIS2Z = 0. / rotation axes 2 (z component)
RANGLE2 = 0. / rotation angle 2 [\si{\degree}]
DETGRID = 'Plane / Cartesian background grid' / description of the detector grid
```

4.2.4 Statistical maps

Simulations of the radiation scattered at dust grains `CMD_DUST_SCATTERING` will additionally create a statistical map with the filename `'polaris_detector_nrXXXX_stats.fits'`. The `'X'` stand for the detector index with four digits (e.g. 0001 for first detector defined in the `'cmd_file'`). In addition to the spatial axes, one axis is provided for the different wavelength IDs and one for the quantities. The first quantity is the total number of photon packages N_{ph} per pixel. Subsequently, there are the relative error R and the variance of the variance VOV of each Stokes I, Q, U, V, and polarized intensity $PI = \sqrt{Q^2 + U^2 + V^2}$. Following Camps & Baes (2018), the relative error R and the variance of the variance VOV are defined as

$$R = \frac{\sqrt{M_2}}{\overline{x}\sqrt{N}}, \tag{4.1}$$

$$VOV = \frac{1}{N}\frac{M_4 - M_2^2}{M_2^2}, \tag{4.2}$$

where $\overline{x}$ is the sample mean and M_k is the kth central moment of the distribution:

$$\overline{x} = \frac{1}{N}\sum x_i, \tag{4.3}$$

$$M_k = \frac{1}{N}\sum (x_i - \overline{x})^k. \tag{4.4}$$

Hereby, x_i is the contribution of the ith photon package to X, where the quantity X represents the Stokes parameters. Based on Camps & Baes (2018), the results of the Monte-Carlo simulation are *reliable* if $R < 0.1$, *questionable* if $0.1 < R < 0.2$, and *unreliable* if $R > 0.2$. As suggested by Gordon et al. (2001), the uncertainty in X can be written as

$$\sigma_X = X R_x \tag{4.5}$$

if the relative error R_x is smaller than 0.1.

4.2.5 Velocity channel maps

Simulations of the line radiative transfer `CMD_LINE_EMISSION` will create velocity channel maps for each velocity channel with the filenames `'vel_channel_maps_species_XXXX_line_YYYY_vel_ZZZZ.fits'`. The `'X'` stand for the gas species index with four digits, the `'Y'` stand for the transition index with four digits and the `'Z'` stand for the current velocity channel (e.g. 0001_line_0001_vel_0001 for first gas species, the first transition defined in the `'cmd_file'` and the first velocity channel, which is located at `-MAXVEL`). In addition, the column density and the magnetic field analysis (for Zeeman simulations) will be written in an extra file (`'vel_channel_maps_species_XXXX_line_YYYY_extra.fits'`). A description of the structure of these velocity channel maps `'.fits'` files can be found in Listing 4.4, which shows the header and additional comments.

4.2.6 Integrated velocity channel map

Simulations of the line radiative transfer `CMD_LINE_EMISSION` will create an integrated velocity channel map with the filename `'int_channel_map_species_XXXX_line_YYYY.fits'`. The `'X'` stand for the gas species index with four digits and the `'Y'` stand for the transition index with four digits (e.g. 0001_line_0001 for first gas species and first transition defined in the `'cmd_file'`). A description of the structure of these integrated velocity channel map `'.fits'` files can be found in Listing 4.5, which shows the header and additional comments.

Listing 4.4: Header of a 'vel_channel_maps_species_XXXX_line_YYYY_vel_ZZZZ.fits' file including comments to explain the file structure.

```
SIMPLE = T / file does conform to FITS standard
BITPIX = -64 / number of bits per data pixel
NAXIS = 3 / number of data axes
NAXIS1 = 256 / length of data axis 1
NAXIS2 = 256 / length of data axis 2
NAXIS3 = 6 / length of data axis 3
EXTEND = T / FITS dataset may contain extensions
COMMENT FITS (Flexible Image Transport System) format is defined in 'Astronomy
COMMENT and Astrophysics', volume 376, page 359; bibcode: 2001A&A...376..359H
CTYPE1 = 'PARAM ' / type of unit 1
CRVAL1 = -44704051205273.4 / value of axis 1
CRPIX1 = 1 / pixel where CRVAL1 is defined
CDELT1 = 350620009453.125 / delta of axis 1
CUNIT1 = 'm ' / unit of axis 1
CTYPE1A = ... See midplane header ...
CTYPE2 = 'PARAM ' / type of unit 2
CRVAL2 = -44704051205273.4 / value of axis 2
CRPIX2 = 1 / pixel where CRVAL2 is defined
CDELT2 = 350620009453.125 / delta of axis 2
CUNIT2 = 'm ' / unit of axis 2
CTYPE1A = ... See midplane header ...
CTYPE3 = 'PARAM ' / type of unit 3
CRVAL3 = 1 / value of axis 3
CRPIX3 = 1 / pixel where CRVAL3 is defined
CDELT3 = 1 / delta of axis 3
CUNIT3 = 'I, Q, U, V [Jy/px], optical depth, column density [m^-2]' / unit of a
HIERARCH GAS_SPECIES = 'C18O ' / name of the observed gas_species
TRANS = 1 / transition index number (see leiden database)
HIERARCH LEVEL_UPPER = 2 / upper energy level index number (see leiden dat
HIERARCH LEVEL_LOWER = 1 / lower energy level index number (see leiden dat
FREQ = 109782173400. / frequency of the simulated transition
VCH = 1 / current velocity channel
CHANNELS= 35 / number of velocity channels
MAXVEL = 3000. / maximum velocity of the velocity channels (-max
ZEEMAN = F / is zeeman splitting in the simulations consider
DISTANCE= 4.31994861405407E+18 / distance to object
RAXIS1X = 1. / rotation axes 1 (x component)
RAXIS1Y = 0. / rotation axes 1 (y component)
RAXIS1Z = 0. / rotation axes 1 (z component)
RANGLE1 = 90. / rotation angle 1 [\si{\degree}]
RAXIS2X = 0. / rotation axes 2 (x component)
RAXIS2Y = 1. / rotation axes 2 (y component)
RAXIS2Z = 0. / rotation axes 2 (z component)
RANGLE2 = 0. / rotation angle 2 [\si{\degree}]
```

Listing 4.5: Header of an `'int_channel_map_species_XXXX_line_YYYY.fits'` file including comments to explain the file structure.

```
SIMPLE = T / file does conform to FITS standard
BITPIX = -64 / number of bits per data pixel
NAXIS = 3 / number of data axes
NAXIS1 = 128 / length of data axis 1
NAXIS2 = 128 / length of data axis 2
NAXIS3 = 6 / different quantities (see CUNIT3)
EXTEND = T / FITS dataset may contain extensions
COMMENT FITS (Flexible Image Transport System) format is defined in 'Astronomy
COMMENT and Astrophysics', volume 376, page 359; bibcode: 2001A&A...376..359H
CTYPE1 = 'PARAM ' / type of unit 1
CRVAL1 = -44528741200546.9 / value of axis 1
CRPIX1 = 1 / pixel where CRVAL1 is defined
CDELT1 = 701240018906.25 / delta of axis 1
CUNIT1 = 'm ' / unit of axis 1
CTYPE1A = ... See midplane header ...
CTYPE2 = 'PARAM ' / type of unit 2
CRVAL2 = -44528741200546.9 / value of axis 2
CRPIX2 = 1 / pixel where CRVAL2 is defined
CDELT2 = 701240018906.25 / delta of axis 2
CUNIT2 = 'm ' / unit of axis 2
CTYPE2A = ... See midplane header ...
CTYPE3 = 'PARAM ' / type of unit 3
CRVAL3 = 1 / value of axis 3
CRPIX3 = 1 / pixel where CRVAL3 is defined
CDELT3 = 1 / delta of axis 3
CUNIT3 = 'I, Q, U, V [Jy/px], optical depth, column density [m^-2]' / unit of axis 3
HIERARCH GAS_SPECIES = 'C18O ' / name of the observed gas_species
TRANS = 1 / transition index number (see leiden database)
HIERARCH LEVEL_UPPER = 2 / upper energy level index number (see leiden database)
HIERARCH LEVEL_LOWER = 1 / lower energy level index number (see leiden database)
FREQ = 109782173400. / frequency of the simulated transition
CHANNELS= 201 / number of velocity channels
MAXVEL = 3000. / maximum velocity of the velocity channels (-max
ZEEMAN = F / is zeeman splitting in the simulations consider
DISTANCE= 4.31994861405407E+18 / distance to object
RAXIS1X = 1. / rotation axes 1 (x component)
RAXIS1Y = 0. / rotation axes 1 (y component)
RAXIS1Z = 0. / rotation axes 1 (z component)
RANGLE1 = 90. / rotation angle 1 [\si{\degree}]
RAXIS2X = 0. / rotation axes 2 (x component)
RAXIS2Y = 1. / rotation axes 2 (y component)
RAXIS2Z = 0. / rotation axes 2 (z component)
RANGLE2 = 0. / rotation angle 2 [\si{\degree}]
```

4.2.7 Line spectrum

Simulations of the line radiative transfer `CMD_LINE_EMISSION` will create a line spectrum with the filename `'line_spectrum_species_XXXX_line_YYYY.fits'`. The `'X'` stand for the gas species index with four digits and the `'Y'` stand for the transition index with four digits (e.g. 0001_line_0001 for first gas species and first transition defined in the `'cmd_file'`). A description of the structure of these line spectrum `'.fits'` files can be found in Listing 4.6, which shows the header and additional comments.

4.2.8 Healpix map (all-sky-map)

If the healpix background grid is chosen for ray-tracing simulations (`CMD_DUST_EMISSION`, `CMD_SYNCHROTRON`, `CMD_LINE_EMISSION`), the resulting `'.fits'` files will have a different structure. Due to the standardized structure of these files, they can easily be post-processed by programs like `healpy`. A description of the structure of these healpix `'.fits'` files can be found in Listing 4.7, which shows the header and additional comments.

4.3 Gnuplot

Several results of POLARIS can be plotted as a Gnuplot script. The Gnuplot software allows creating plots as well as the 3D representation of scientific data. POLARIS can create Gnuplot scripts for the 3D data of the physical parameters of the input grids and output grids. The data representation is either point-like (e.g. density, temperatures) or vector-like (e.g. magnetic field, velocities). Writing such files is optional and can be switched on by adding the lines

```
<nr_plot_points> N_p
<nr_plot_vectors> N_v
<max_plot_lines> N_l
```

to the command file. A Gnuplot script is in plain text. Hence, it is not recommended plotting the maximal amount of grid points, vectors, and lines. Here, they can be limited by the numbers N_p, N_v, and N_l, respectively. Example:

```
<nr_plot_points> 4000
<nr_plot_vectors> 4000
<max_plot_lines> 300
```

The POLARIS code plots automatically also the cross-sections of the defined dust model (see Sect. 3.7.5). This is for the single dust materials as well as the dust mixtures. Dependent on the selected POLARIS simulation mode a source file will be created. These files contain 3D representation of the position and parameters of the defined stars and starfields, respectively (see Sect. 3.2). All the Gnuplot files require no further editing by the user and can directly be used for interpretation, representation, and analysis of the resulting simulation data.

4.4 AMIRA

AMIRA is a software for the visualization of large scientific data but is not for free. POLARIS supports the AMIRA format for 3D data. An AMIRA file can be created for the physical parameters of the input grid (e.g. gas temperature or density) as well as the output grid (e.g. dust temperature, RAT alignment radius). Adding the following commands to the command file results in an AMIRA file for each of the physical parameters sampled over N_{bins} bins in each direction:

```
<amira_inp_points> N_bins
<amira_out_points> N_bins
```

Listing 4.6: Header of an 'line_spectrum_species_XXXX_line_YYYY.fits' file including comments to explain the file structure.

```
SIMPLE = T / file does conform to FITS standard
BITPIX = -64 / number of bits per data pixel
NAXIS = 3 / number of data axes
NAXIS1 = 201 / different velocity channels
NAXIS2 = 1 / dummy to have one line, if opened as image
NAXIS3 = 4 / different quantities (see CUNIT3)
EXTEND = T / FITS dataset may contain extensions
COMMENT FITS (Flexible Image Transport System) format is defined in 'Astronomy
COMMENT and Astrophysics', volume 376, page 359; bibcode: 2001A&A...376..359H
CTYPE1 = 'VELO ' / type of unit 1
CRVAL1 = -3000. / value of axis 1
CRPIX1 = 1 / pixel where CRVAL1 is defined
CDELT1 = 30. / delta of axis 1
CUNIT1 = 'm/s ' / unit of axis 1
CTYPE2 = 'PARAM ' / type of unit 2
CRVAL2 = 1 / value of axis 2
CRPIX2 = 1 / pixel where CRVAL2 is defined
CDELT2 = 1 / delta of axis 2
CUNIT2 = 'I, Q, U, V [Jy/px], optical depth, column density [m^-2]' / unit of a
HIERARCH GAS_SPECIES = 'C18O ' / name of the observed gas_species
TRANS = 1 / transition index number (see leiden database)
HIERARCH LEVEL_UPPER = 2 / upper energy level index number (see leiden database)
HIERARCH LEVEL_LOWER = 1 / lower energy level index number (see leiden database)
FREQ = 109782173400. / frequency of the simulated transition
CHANNELS= 201 / number of velocity channels
MAXVEL = 3000. / velocity of the velocity channels (-maxvel to maxvel)
ZEEMAN = F / is zeeman splitting in the simulations consider
DISTANCE= 4.31994861405407E+18 / distance to object
RAXIS1X = 1. / rotation axes 1 (x component)
RAXIS1Y = 0. / rotation axes 1 (y component)
RAXIS1Z = 0. / rotation axes 1 (z component)
RANGLE1 = 90. / rotation angle 1 [\si{\degree}]
RAXIS2X = 0. / rotation axes 2 (x component)
RAXIS2Y = 1. / rotation axes 2 (y component)
RAXIS2Z = 0. / rotation axes 2 (z component)
RANGLE2 = 0. / rotation angle 2 [\si{\degree}]
```

Listing 4.7: Header of the first extension of a healpix '.fits' file including comments to explain the file structure.

```
XTENSION= 'BINTABLE' / binary table extension
BITPIX = 8 / 8-bit bytes
NAXIS = 2 / 2-dimensional binary table
NAXIS1 = 48 / width of table in bytes
NAXIS2 = 12288 / number of rows in table
PCOUNT = 0 / size of special data area
GCOUNT = 1 / one data group (required keyword)
TFIELDS = 6 / different quantities/wavelengths/frequencies
TTYPE1 = 'I_STOKES (WAVELENGTH = 8.500000e-04 [m])' / label for field 1
TFORM1 = 'D ' / data format of field: 8-byte DOUBLE
TUNIT1 = 'Jy/px ' / physical unit of field
TTYPE2 = 'Q_STOKES (WAVELENGTH = 8.500000e-04 [m])' / label for field 2
TFORM2 = 'D ' / data format of field: 8-byte DOUBLE
TUNIT2 = 'Jy/px ' / physical unit of field
TTYPE3 = 'U_STOKES (WAVELENGTH = 8.500000e-04 [m])' / label for field 3
TFORM3 = 'D ' / data format of field: 8-byte DOUBLE
TUNIT3 = 'Jy/px ' / physical unit of field
TTYPE4 = 'V_STOKES (WAVELENGTH = 8.500000e-04 [m])' / label for field 4
TFORM4 = 'D ' / data format of field: 8-byte DOUBLE
TUNIT4 = 'Jy/px ' / physical unit of field
TTYPE5 = 'OPTICAL_DEPTH (WAVELENGTH = 8.500000e-04 [m])' / label for field 5
TFORM5 = 'D ' / data format of field: 8-byte DOUBLE
TTYPE6 = 'COLUMN_DENSITY' / label for field 6
TFORM6 = 'D ' / data format of field: 8-byte DOUBLE
TUNIT6 = 'm^-2 ' / physical unit of field
EXTNAME = 'HEALPIX_EXTENSION' / name of this binary table extension
PIXTYPE = 'HEALPIX ' / Pixel algorigthm
ORDERING= 'RING ' / Ordering scheme
INDXSCHM= 'IMPLICIT' / Indexing scheme
NSIDE = 32 / Resolution Parameter
FIRSTPIX= 0 / First pixel (0 based)
LASTPIX = 12287 / Last pixel (0 based)
CROTA2 = 0 / Rotation Angle (Degrees)

ETYPE = 'thermal emission' / type of emission
  ... or ...
ETYPE = 'synchotron emission' / type of emission
  ... or ...
HIERARCH GAS_SPECIES = 'C18O ' / name of the observed gas_species
TRANS = 1 / transition index number (see leiden database)
HIERARCH LEVEL_UPPER = 2 / upper energy level index number (see leiden dat
  ...

ID = 1 / detector id
HIERARCH OBS_POSITION_X = 0. / x-axis position of observer
HIERARCH OBS_POSITION_Y = 0. / y-axis position of observer
HIERARCH OBS_POSITION_Z = 0. / z-axis position of observer
HIERARCH OBS_VELOCITY_X = 0. / velocity of observer in x direction
HIERARCH OBS_VELOCITY_Y = 0. / velocity of observer in y direction
HIERARCH OBS_VELOCITY_Z = 0. / velocity of observer in z direction
HIERARCH LONGITUDE_MIN = 0. / minimum considered galactic longitude
HIERARCH LONGITUDE_MAX = 6.28318530717959 / maximum considered galactic longitud
HIERARCH LATITUDE_MIN = 0. / minimum considered galactic latitude
HIERARCH LATITUDE_MAX = 3.14159265358979 / maximum considered galactic latitude
```

Example:

```
<amira_inp_points> 100
<amira_out_points> 500
```

The AMIRA file is in plane text with a short header and a data section running over the z, y, and x-direction. Writing an AMIRA file is a problem that cannot be parallized and can take up to one hour. Hence, the number of bins N_{bins} has to be chosen carefully.

Bibliography

Bjorkman, J. E. & Wood, K. 2001, ApJ, 554, 615

Bohren, C. F. & Huffman, D. R. 1983, Absorption and scattering of light by small particles (Wiley)

Brauer, R., Wolf, S., & Flock, M. 2017a, A&A, 607, A104

Brauer, R., Wolf, S., & Reissl, S. 2016, A&A, 588, A129

Brauer, R., Wolf, S., Reissl, S., & Ober, F. 2017b, A&A, 601, A90

Camps, P. & Baes, M. 2018, ApJ, 861, 80

Camps, P., Misselt, K., Bianchi, S., et al. 2015, A&A, 580, A87

Cashwell, E. D. & Everett, C. J. 1957, A practical manual on the Monte Carlo method for random walk problems (Los Alamos Scientific Laboratory)

Davis, Leverett, J. & Greenstein, J. L. 1951, ApJ, 114, 206

Dexter, J. 2016, MNRAS, 462, 115

Draine, B. T. 2003, ApJ, 598, 1017

Draine, B. T. & Flatau, P. J. 2013, User Guide for the Discrete Dipole Approximation Code DDSCAT 7.3, arXiv:1305.6497

Draine, B. T. & Weingartner, J. C. 1996, ApJ, 470, 551

Draine, B. T. & Weingartner, J. C. 1997, ApJ, 480, 633

Gold, T. 1952, MNRAS, 112, 215

Gordon, K. D., Misselt, K. A., Witt, A. N., & Clayton, G. C. 2001, ApJ, 551, 269

Greenberg, J. M. 1968, in Nebulae and Interstellar Matter, ed. B. M. Middlehurst & L. H. Aller (University of Chicago Press), 221

Henyey, L. G. & Greenstein, J. L. 1941, ApJ, 93, 70

Hoang, T. & Lazarian, A. 2008, MNRAS, 388, 117

Hoang, T. & Lazarian, A. 2009, ApJ, 697, 1316

Hoang, T. & Lazarian, A. 2014, MNRAS, 438, 680

Hughes, A. M., Wilner, D. J., Cho, J., et al. 2009, ApJ, 704, 1204

Irvine, W. M. 1965, ApJ, 142, 1563

Irvine, W. M. 1968, ApJ, 152, 823

Larsson, R., Buehler, S. A., Eriksson, P., & Mendrok, J. 2014, J. Quant. Spectr. Rad. Transf., 133, 445

Lazarian, A. 1994, MNRAS, 268, 713

Lazarian, A. 1995, ApJ, 451, 660

Lazarian, A. 1996, in Astronomical Society of the Pacific Conference Series, Vol. 97, Polarimetry of the Interstellar Medium, ed. W. G. Roberge & D. C. B. Whittet, 425

Lazarian, A. 1997, ApJ, 483, 296

Lazarian, A. 2007, J. Quant. Spectr. Rad. Transf., 106, 225

Lazarian, A. & Efroimsky, M. 1996, ApJ, 466, 274

Lazarian, A., Efroimsky, M., & Ozik, J. 1996, ApJ, 472, 240

Lazarian, A. & Hoang, T. 2007, MNRAS, 378, 910

Lazarian, A. & Roberge, W. G. 1997, ApJ, 484, 230

Lee, H. M. & Draine, B. T. 1985, ApJ, 290, 211

Lucy, L. B. 1999, A&A, 344, 282

Martin, P. G. 1974, ApJ, 187, 461

Mathis, J. S., Mezger, P. G., & Panagia, N. 1983, A&A, 128, 212

Mie, G. 1908, Ann. Phys., 330, 377

Ober, F., Wolf, S., Uribe, A. L., & Klahr, H. H. 2015, A&A, 579, A105

Pandya, A., Zhang, Z., Chandra, M., & Gammie, C. F. 2016, ApJ, 822, 34

Reissl, S., Guillet, V., Brauer, R., et al. 2020, A&A, 640, A118

Reissl, S., Seifried, D., Wolf, S., Banerjee, R., & Klessen, R. S. 2017, A&A, 603, A71

Reissl, S., Stutz, A. M., Brauer, R., et al. 2018a, MNRAS, 481, 2507

Reissl, S., Wolf, S., & Brauer, R. 2016, A&A, 593, A87

Reissl, S., Wolf, S., & Brauer, R. 2018b, POLARIS: POLArized Radlation Simulator, Astrophysics Source Code Library, record ascl:1807.001

Reissl, S., Wolf, S., & Seifried, D. 2014, A&A, 566, A65

Roberge, W. G. & Lazarian, A. 1999, MNRAS, 305, 615

Seifried, D., Walch, S., Reissl, S., & Ibáñez-Mejía, J. C. 2019, MNRAS, 482, 2697

Spitzer, L., J. & McGlynn, T. A. 1979, ApJ, 231, 417

van de Hulst, H. C. 1957, Light Scattering by Small Particles (John Wiley & Sons)

van der Tak, F. F. S., Lique, F., Faure, A., Black, J. H., & van Dishoeck, E. F. 2020, Atoms, 8, 15

Whitney, B. A. & Wolff, M. J. 2002, ApJ, 574, 205

Wolf, S. 2003, Comput. Phys. Commun., 150, 99

Wolf, S., Henning, T., & Stecklum, B. 1999, A&A, 349, 839

Wolf, S. & Voshchinnikov, N. V. 2004, Comput. Phys. Commun., 162, 113

Yusef-Zadeh, F., Morris, M., & White, R. L. 1984, ApJ, 278, 186

Index