OpTool User Guide

Carsten Dominik & Michiel Min

July 2020, version 0.9.9

Introduction

This tool allows to produce complex dust particle opacities right from the command line. It is derived[1] from Michiel Min's OpacityTool for the DIANA program. `optool` adds a much more flexible command line interface, a curated list of built-in materials, easy access to external material properties, more output options, and significant speed gains.

Capabilities

- stand-alone tool, fully command line driven, no input files need to be edited
- flexible command line interface to specify input in a compact and simple way
- full scattering matrix output in several formats, including for RADMC-3D
- combining materials through mixing into a complex grain with porosity
- DHS method to model shape effects and low-porosity aggregates
- a useful collection of compiled-in materials for standard applications in astronomy
- an easy way to use external refractive index data for more specialized applications
- multi-core support through OpenMP to make even complex computations very fast

Terms of use

`optool` is distributed under the MIT license and can be used, changed and redistributed freely. The implemented physics is the same as in OpacityTool. As a reference for this tool, please point to the GitHub repository. If you use it to publish papers, please cite the papers establishing the methods used, as well as to the original laboratory papers that published the refractive index measurements.

- `optool`: `https://github.com/cdominik/optool.git`
- DHS model for irregular grains: Min et al. 2005, A&A, 432, 909
- DIANA standard Opacities: Woitke, Min et al. 2016, A&A 586, 103
- Third party software: Toon et al. 1981, Applied Optics 20, 3657
- References to refractive index data used in your particular application.

[1] For another derivative of OpacityTool with a different set of applications, check out SIGMA, the *Simple Icy Grain Model for Aggregates*, by Lefévre et al. (2020).

Examples

A simple grain made only of the default pyroxene, for the default grain size distribution ($a^{-3.5}$ powerlaw from 0.05 to 3000μm), on the default wavelength grid (0.05μm to 1cm).

```
optool pyr
```

Include the scattering matrix in the produced output

```
optool pyr -s
```

Reproduce the DIANA standard dust model, using a specific pyroxene (70% Mg) and carbon, in a mass ratio 0.87/0.13, and with a porosity of 25%.

```
optool pyr-mg70 0.87  c 0.13  -p 0.25
```

List the built-in materials

```
optool -c
```

Add an ice mantle (built-in data from Warren+08) that is 20% of the core mass

```
optool pyr-mg70 0.87  c 0.13  -m ice-w 0.2  -p 0.25
```

Like the previous example, but use ice refractive index data from a separate file.

```
optool pyr-mg70 0.87  c 0.13  -p 0.25  -m data/ice_hudgins.dat 0.2
```

Pure ice grains in a narrow size distribution from 1 to 3 microns, with 15 sample sizes following an $f(a) \propto a^{-2.5}$ powerlaw size distribution. Also, restrict the wavelength range to 10-100μm, and turn off DHS to get perfect spheres.

```
optool ice  -a 1 3 2.5 15  -l 10 100 -fmax 0
```

For silicon carbide, compute the opacity of a single grains size (2.5μm) at λ=8.9μm.

```
optool -a 2.5 -l 8.9 sic
```

Represent the default dust model (DIANA, you also get this when you do not give any materials at all) in 42 grain sizes, and produce input files for RADMC-3D, one for each grain size, with full scattering matrix, chopping 3 degrees from the scattering peak.

```
optool -na 42 -d -s -radmc -chop 3
```

Use the Bruggeman rule to blend refractive index data of three orientations of a crystaline material, using the wavelength grid given in one of those files. Output to `blended.lnk`.

```
optool -b gra_x.lnk 0.33 gra_y.lnk 0.33 gra_z.lnk 0.33 -l gra_x.lnk
```

Compiling `optool`

On most systems, you can download and compile `optool` with these simple steps, using the freely available GNU FORTRAN compiler `gfortran`.

```
git clone https://github.com/cdominik/optool.git
cd optool
make
```

The executable is called `optool`, and you should put it on your execution path. To use the Intel fortran compiler, to use multiple cores[2] for speed, or to be able to write FITS files[3], use one or more of the following parameters during compilation:

```
make ifort=true       # use the Intel fortran compiler iFort
make multi=true       # enable multicore support
make  fits=true       # add support for writing FITS files
```

You can also find binaries for Mac and Linux at my homepage.

command line arguments

`-h` Show a compact help message about command line options.

Grain composition and geometry

`-c` List available built-in materials (the keys for the **-c** and **-m** options).

`[-c] KEY-or-FILE1 [MFRAC]`
: Specify a material to include in the grain. `KEYorFILE` can be the key for a builtin material, or the path to the correct `lnk` file. `MFRAC` is the *mass* fraction (default 1.0) of the material. You can give up to 10 materials to build up the grain. Mass fractions do not have to add up to one, they will be renormalized. All materials will be mixed together using the *Bruggeman* rule, and vacuum can be added through the porosity. A **-c** switch before each `KEY-or-FILE` is optional.

`-m KEY-or-FILE [MFRAC]`
: Like **-c**, but use this material as grain mantle that will be added using the *Maxwell-Garnett* rule. Only one **-m** switch is allowed.

`-p POROSITY [P_MANTLE]`
: Porosity, the *volume* fraction of vacuum, a number smaller than 1. The default is 0. A single value will apply to both core and mantle, but a second value will be specific for the mantle (and may be 0).

`-fmax VHMAX`
: Maximum *volume* fraction of the inner hole for the DHS approach. The default is 0.8. Zero means to use solid spheres, i.e. perfectly regular grains.

[2] If you do turn multicore support on, please also run "`make selftest`" to test if everything works properly. I have had problems with the OpenMP setup of an older version of `gfortran`.

[3] This requires the `cfitsio` library to be installed on your system.

Grain size distribution

`-a AMIN [AMAX [APOW [NA]]]`
: Specify (minimum) grain radius, and optionally maximum grain radius, the size distribution powerlaw and the number of size bins. You may also use options to set individual values with **-amin**, **-amax**, **-apow**, **-na**. The defaults are 0.05 μm, 3000 μm, 3.5, and *10 per size decade with a fixed minimum of 5*, respectively. If only a single size is specified with **-a**, then $a_{max}=a_{min}$ and $n_a=1$ are implied.

Wavelength grid

`-l LMIN [LMAX [NLAM]]`
: Specifiy the (minimum) wavelength, and optionally the maximum wavelength and the number of wavelengths points for the construction of the wavelength grid. The default values are 0.05 μm, 10000 μm, and 300, respectively. You may also use the options **-lmin**, **-lmax**, and **-nlam** (or **-nl**) to set individual values. If only one wavelength is specified with **-l**, then $\lambda_{max}=\lambda_{min}$ and $n_\lambda=1$ are implied.

`-l FILE`
: Read the wavelength grid from `FILE`. The file may start with comment lines, and the first non-comment line needs to contain the number of wavelength values in the data block below it. In the data block, the first column is expected to hold the wavelength values, in μm. For example, an `lnk` file could be used here.

Controlling the output

The standard output is the file `dustkappa.dat`, with the opacities and the asymmetry parameter g. The following options control and extend the output.

`-o [DIR]`
: Put the output files in directory `DIR` instead of the current working directory. `./output` will be used if `DIR` is not specified.

`-s [NANG]`
: Include the full scattering matrix in the output. `NANG` can optionally specify the number of equally-spaced angular grid points to cover the range of angles between 0 and 180 degrees. The default for `NANG` is 180 and should normally not be changed.

`-chop [NDEG]`
: Cut out the first `NDEG` (2 if unspecified) degrees of the forward scattering peak and compensate by a reduction in the scattering cross section.

`-d [NSUB]`
: Divide the computation up into `NA` parts to produce a file for each grain size. Each size will actually be an average over a small range of `NSUB` grains around the real size, to smear out resonances. The default for `NSUB` is 5.

`-fits`
: Write `dustkappa.fits` with the absorption cross sections and scattering matrix elements, instead of ASCII output. With the `-d` switch, `NA` files will be written.

`-radmc [LABEL]`
: RADMC-3D uses a different angular grid and normalization for the scattering matrix, so the output has to be adapted for it. The extension of the files will be changed to `.inp`, and if you specify `LABEL`, it will be used in the file name(s).

`-t [TMIN [TMAX [NT]]]`
: Compute mean opacities per g of dust mass, $\kappa_{\rm Planck}$ and $\kappa_{\rm Rosseland}$, in the given temperature interval, in `nt` logarithmic steps, with output to `dustkapmean.dat`. The parameters default to 10K, 10000K, and 200, respectively.

`-b` Only write the refractive index data resulting the mixing process to `blended.lnk`.

Material properties

`optool` needs refractive index data to work. For your convenience, a useful list of materials is compiled into `optool`, but you can also find and use other data. No matter where the data is from, you should *always* cite the original laboratory papers.

Built-in materials

To access one of the built-in materials, specify the corresponding key string like `pyr-mg70` instead of the path to an `lnk` file. In each material class I have selected a useful default, accessible with an even simpler generic key.

-c Key generic	**-c Key** full key	**Material**	**State**	ρ g/cm^3	$\lambda_{\rm min}$ μm	$\lambda_{\rm max}$ μm	**Reference**
	pyr-mg100	$MgSiO_3$	amorph	2.71	0.2	500	Dorschner+95
	pyr-mg95	$Mg_{0.95}Fe_{0.05}SiO_3$	amorph	2.74	0.2	500	Dorschner+95
	pyr-mg80	$Mg_{0.8}Fe_{0.2}SiO_3$	amorph	2.9	0.2	500	Dorschner+95
pyr	pyr-mg70	$Mg_{0.7}Fe_{0.3}SiO_3$	amorph	3.01	0.2	500	Dorschner+95
	pyr-mg60	$Mg_{0.6}Fe_{0.4}SiO_3$	amorph	3.1	0.2	500	Dorschner+95
	pyr-mg50	$Mg_{0.5}Fe_{0.5}SiO_3$	amorph	3.2	0.2	500	Dorschner+95
	pyr-mg40	$Mg_{0.4}Fe_{0.6}SiO_3$	amorph	3.3	0.2	500	Dorschner+95
ens	pyr-c-mg96	$Mg_{0.96}Fe_{0.04}$SiO3	cryst	2.8	**2.0**	**99**	Jäger+98
ol	ol-mg50	$MgFeSiO_4$	amorph	3.71	0.2	500	Dorschner+95
	ol-mg40	$Mg_{0.8}\ Fe_{1.2}\ SiO_4$	amorph	3.71	0.2	500	Dorschner+95
for	ol-c-mg100	$Mg_2\ SiO_4$	cryst	3.33	**3.0**	250	Steyer+74
c	c-z	C	amorph?	1.8	0.05	1 (4)	Zubko+96
	c-p	C	amorph	1.8	0.11	800	Preibisch+93
gra	c-gra	C graphite	cryst	2.16?	0.001	1000	uncertain
	c-org	CHON organics	amorph	1.4	0.1	1 (5)	uncertain
	c-nano	C nano-diamond	cryst	2.3	0.02	**110**	Mutschke+04
ice	ice-w	Water ice	cryst	0.92	0.04	2 (6)	Warren+08
iron	fe-c	Fe	metal	7.87	0.1	1 (5)	Henning+96
	fes	FeS	metal	4.83	0.1	1 (5)	Henning+96
	sic	SiC	cryst	3.22	0.001	1000	Laor93
cor	cor-c	Al_2O_3	cryst	4.0	0.5	**40**	Koike+95

External refractory index files (`lnk` files)

`optool` can use external refractive index data in files with the following format:

- The file may start with several comment lines (lines starting with `!`, `#`, or `*`).
- The next line contains two numbers, the number of wavelengths N_λ and the specific weight ρ of the material in g/cm^3
- Then follow three columns of data: $\lambda[\mu\text{m}]$, and the real and imaginary parts of the refractive index, n and k.

You can find refractive index data in the Jena database, and associated with original papers in the literature. Don't forget to add the line with N_λ and ρ! If for some reason it is not convenient to add that line to the file, `optool` will count the lines for you and you can specify the density after the mass fraction, like this: `optool -c path/to/file.lnk 0.7 3.42`

Output files

dustkappa.dat

This is an ASCII file containing the basic opacity results. It starts with a comment section describing the dust model, followed by the format number (3, currently), followed by the number of wavelengths in the grid, both on lines by themselves. Then follows a block with these columns:

1. wavelength λ [micron]
2. mass absorption cross section κ_{abs} [cm^2/g]
3. mass scattering cross section κ_{sca} [cm^2/g]
4. asymmetry parameter g

dustkapscatmat.dat

ASCII file with cross sections and full scattering matrix. The comment section at the start of the file explains the structure. See the appendix for information about the normalization of the scattering matrix. And see the `-radmc` switch which will modify the output to make sure it can be used as an input file for RADMC-3D.

dustkappa.fits

The FITS-file (ending in '.fits') is written instead of the ASCII output when using the `-fits` switch. It has two HDU blocks. The first block contains the cross sections per unit mass. This is a $N_\lambda \times 4$ matrix with these columns:

1. wavelengths in [μm]
2. mass extinction cross section κ_{ext} in [cm^2/g]
3. mass absorption cross section κ_{abs} in [cm^2/g]
4. mass scattering cross section κ_{sca} in [cm^2/g]

The second HDU block contains the scattering matrix elements. It is a $N_\lambda \times 6 \times N_{\text{ang}}$ matrix, containing the 6 elements of the scattering matrix for N_{ang} equidistant scattering angles from forward scattering (element 0) to backward scattering (element N_{ang}-1), for each wavelength value. The stored matrix elements are F_{11}, F_{12}, F_{22}, F_{33}, F_{34}, and F_{44}.

dustkapmean.dat
: This file will only be written with the `-t` switch. It contains 3 columns: (1) T [K], (2) $\kappa_{\rm Planck}$, (3) $\kappa_{\rm Ross}$, both in cm^2 per gram of *dust*. Note that dust evaporation is not considered, and that a wide wavelengths coverage is needed for good results.

Inspecting the computed optical properties

To try out `optool` you could use one of these commands

```
make test              # for a simple size-integrated opacity computation
    ... or ...
make testdiv           # for opacities as a function of grain size
```

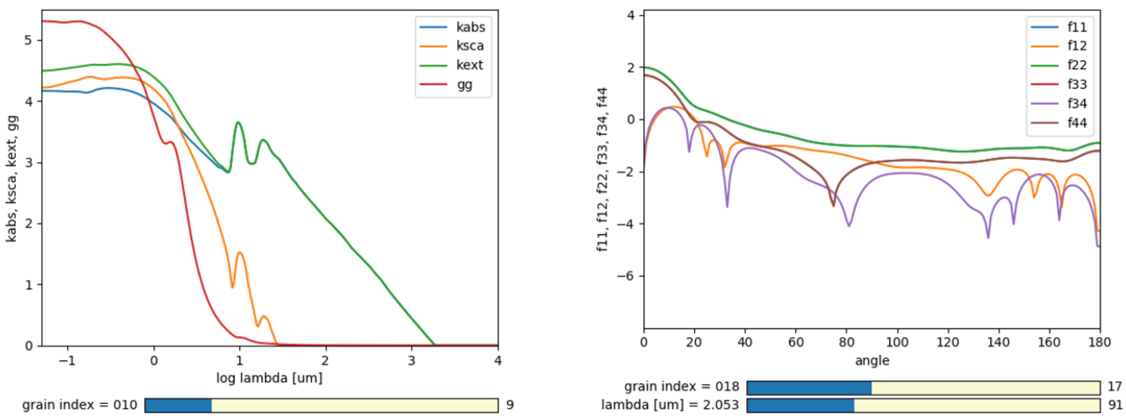

Figure 1: Screenshot of the plots created by running `ipython -i optool_plot.py`. Note that we plot the logarithm of the absolute value of the scattering matrix, in order to deal with range and sign issues.

The commands will run `optool` with the standard DIANA material properties, and then use the python script `optool_plot.py` to plot the computed opacities. You will get:

- a plot showing the opacities $\kappa_{\rm abs}$, $\kappa_{\rm sca}$, and $\kappa_{\rm ext}$ as a function of wavelength, along with the asymmetry parameter g. Note that the red g curve does not have its own scale, imagine the y axis going from 0 to 1 for g.
- a plot showing the scattering matrix elements as a function of scattering angle, and with sliders to go through grain sizes and wavelengths. The y axis of the plot is actually $\log_{10}(|F_{ij}|)$, so the downward peaks are actually places where the matrix element goes through zero.

If you want to run the plotter yourself, `ipython -i path/to/optool_plot.py` is the correct command to do so, from the directory where the output files are located.

Acknowledgments

- Michiel Min for the DIANA OpacityTool and all the incredible work that went into it. `optool` is a direct derivative of that tool and reuses almost all of its code.
- Charléne Lefévre for SIGMA, which triggered me to add a grain mantle using the Maxwell-Garnett rule.
- Kees Dullemond for his python plotting routine `viewarr` (available on github), and code for computing Planck and Rosseland means opacities.
- Jeroen Bouwman for some pointers to refractive index data.

Appendix

Units

Due to conventions in our field, the input and output of `optool` uses the following units

- **microns** for grain sizes and wavelengths
- $\mathbf{g/cm^3}$ for mass densities of materials
- $\mathbf{cm^2\ g^{-1}}$ for opacities $\kappa_{\rm abs}$, $\kappa_{\rm sca}$, and $\kappa_{\rm ext}$
- $\mathbf{sr^{-1}}$ or $\mathbf{cm^2\ g^{-1}\ sr^{-1}}$ for the scattering matrix elements, see below.
- **Kelvin** for temperatures

Scattering Matrix: The fine print

Phase function normalization

A number of different normalizations for the scattering matrix are being used in the literature and in computational tools. The differences are significant, and it is important to be aware of the choice. For `optool` we are using a convention in which the average over all directions of the 1-1 element of the scattering matrix equals unity, i.e.

$$\oint_{(4\pi)} F_{11}(\lambda,\Theta) d\Omega = 4\pi \tag{1}$$

See Hovenier (2004) for a discussion of this normalization. `optool` can also produce output for RADMC-3D which uses a different normalization, namely

$$\oint_{(4\pi)} Z_{11}(\lambda,\Theta) d\Omega = \kappa_{\rm sca}(\lambda) \tag{2}$$

Forward-scattering peak

Particles that are much larger than the wavelength of the considered radiation can show extreme forward scattering, to an extend that a very significant fraction of the *scattered* radiation is sent into just a few degrees around the forward direction (scattering angle zero). This can be difficult to handle for radiative transfer codes which have limited angular resolution or limited sampling. Some codes have an internal way to deal with this. MCMax3D, for example, has the `nspike` keyword to deal with this issue. Others, like RADMC-3D require this to be taken care of by the process that creates the opacity files. This is the purpose of the `-chop` switch in `optool`. It specifies a number of degrees (for example 3 might be a good value) around the forward scattering direction. Inside that cone, the scattering matrix gets limited to the value at the edge of the cone. To compensate and ensure energy conservation, the scattering cross section will be reduced accordingly, and, depending on the chosen normalization, the scattering matrix itself may also have to be scaled. As a result, the radiation that would be *scattered* into this narrow range of angles will be treated as if it did have *no interaction at all* with the grain.

Angular grid

`optool` uses an angular grid in one degree steps from 0 to 180 degrees. The full degrees are the cell *interfaces* of that grid. `optool` computes the scattering matrix at the cell *midpoints*, i.e. at 0.5°, 1.5° etc to 179.5°, for a total of 180 values. The scattering matrix is normalized in this way, so that a numerical integral gives the correct result.
However, if you are using `optool` to produce scattering cross sections of RADMC-3D, the conventions are different. RADMC-3D requires the values of the scattering matrix on the cell *boundaries*, so at 0°, 1° etc to 180°, for a total of 181 values. For the input files for RADMC-3D, we interpolate and extend the computed values to the cell boundaries. With strongly forward-scattering grains, this does require a small renormalization to make sure the the full solid angle integration gives again accurate results. `optool` is handling this renormalization fully automatically.

How to ingest refractive index data for another material

Additional refractive index data tables can be compiled into the code. Here is how:

1. Give your `lnk` file a name exactly like `pyr-mg70-Dorschner1995.lnk`, where `pyr-mg70` is the key to access the material and `Dorschner1995` (the text after the final `-`) is the reference.
2. Put this file into the `lnk_data` directory.
3. Optionally edit `lnk_data/lnk-help.txt`, so that `optool -c ?` will list the new material. Note that, in order to define generic keys, optool looks for pairs like `genkey -> fullkey` in this file.
4. Run `make ingest` to update `ref_ind.f90`, now including your new material.
5. Recompile and install the code.

Bibliography

- Dorschner, J. et al. 1995, A&A 300, 503
- Henning, Th. and Stognienko, R. 1996, A&A 311,291
- Hovenier, J, 2004, Report available on ADS.

- Jäger, C. et al. 1998, A&A 339, 904
- Koike, C. et al. 1995, Icarus 114, 203
- Laor, A. and Draine, B., ApJ 402, 441
- Lefèvre, C.; Min,M. et al. 2020, A&A (submitted)
- Min, M. et al. 2005, A&A, 432, 909
- Min, M. et al. 2016, A&A, 585, 13
- Mutschke, H. et al. 2004, A&A 423, 983
- Toon, O. & Ackerman,T. 1981, Applied Optics 20, 3657
- Woitke, P.; Min, M. et al. 2016, A&A 586, 103
- Preibisch, Th. et al. 1993, A&A 279, 577
- Steyer, T. 1974, PhD Thesis, The University of Arizona
- Warren, S. and Brandt,R. 2008, JGRD,113, D14220
- Zubko, V. et al. 1996, MNRAS 282,1321