Users' Guide

V4.3

J.Ph. Bernard, I. Choubani, M. Compiègne, N. Flagey, V. Guillet,
A. Hughes, D. Paradis, L. Verstraete & N. Ysard
dustemwrap@irap.omp.eu

April 17, 2023

Contents

1 Preamble **2**

2 Structure of this document **3**

3 DustEMWrap installation **4**
3.1 **DustEMWrap** dependencies 6
3.2 **DustEMWrap** and **IDL** 6

4 An overview of the DustEMWrap framework **7**
4.1 Running DustEM using **DustEMWrap** 7
4.2 Running DustEM iteratively using **DustEMWrap** 7

5 Available dust models **9**

6 Dust model parameters **10**

7 Format requirements for input data **12**
7.1 SED and extinction curve data . 12
7.2 ISRF data 13

8 Known instrument filters **14**

9 Color corrections **15**

10 Plug-ins **19**

11 Getting started **22**

12 Modification history **23**

References **24**

Appendices **25**

1 Preambule

We present **DustEMWrap** V4.3, an **IDL** extension to the DustEM **fortran** code. **DustEMWrap** allows the user to iteratively call the DustEM **fortran** executable while modifying the input parameters from within the **IDL** environment. The capabilities of **DustEMWrap** are continuously evolving. The core features of the current **DustEMWrap** V4.3 release include:

1. Color corrections within photometric bands applied to the DustEM spectra, including for JWST instruments
2. Iterative fitting of astronomical SEDs in the near-infrared to radio regime
3. Generation of pre-computed DustEM tables to make fast comparisons with astronomical data
4. Optional inbuilt and user-defined plug-ins to model various emission processes and modify default parameters of the DustEM **fortran** code
5. Iterative fitting of linearly polarised emission (Stokes IQU) due to thermal dust and synchrotron radiation.
6. A helper tool that allows users to extract an SED (StokesIQU or Stokes I only) from input WCS FITS files.

DustEMWrap provides an easy implementation of any operation requiring multiple calls to DustEM. It incorporates transmission curves and color correction rules for a large set of astronomical instruments. The current fitting strategy is a generic χ^2 minimisation process, as implemented in Craig Markwardt's MPFIT package.

DustEMWrap supports the use of plug-ins (i.e. user-defined components that contribute to the emission/extinction or modify default DustEM inputs) to fit an observed SED. This gives **DustEMWrap** the ability to construct model SEDs from custom model predictions, as well as the physical interstellar dust models available in DustEM. **DustEMWrap** is thus a flexible tool for the fitting of continuum astronomical data in the infrared to radio regime.

2 Structure of this document

In Section 3, we describe the installation of **DustEMWrap**, including the necessary modifications during the installation of the DustEM `fortran` code to ensure that **DustEMWrap** and the DustEM `fortran` function together correctly. Code dependencies of **DustEMWrap** are outlined in 3.1.

Section 4 provides an overview of how DustEM and **DustEMWrap** work. The section addresses an interested non-expert, and should be accessible to new users. More detailed information is available in the **DustEMWrap** Developers' Guide (available by contacting `dustemwrap@irap.omp.eu`).

Section 5 lists the set of physical interstellar dust models that can be invoked using the current release of **DustEMWrap**.

Section 6 summarises the main parameters of the interstellar dust models, and shows how to specify the free parameters that will included in the fit during a **DustEMWrap** run.

Section 7 describes the required format for input files that describe an observational SED, extinction curve or ISRF for **DustEMWrap**.

Section 8 describes the library of instrument filters that are currently included in **DustEMWrap**.

Section 9 describes how **DustEMWrap** implements the color corrections associated with the instrument filters.

Section 10 explains the plug-in formalism and describes the default plug-ins that are provided with the current release.

In Section 11, we present the example routines included in the current release that demonstrate some common use cases of the **DustEMWrap** code.

Key events in the development and modification history of **DustEMWrap** are outlined in Section 12.

In the appendices, we provide:

- a minimal example of an `idl_startup` configuration file that is required to run **DustEMWrap**;
- a link to API documentation for all the **IDL** routines provided in the current **DustEMWrap** release;
- a description of the **IDL** system variables used by **DustEMWrap**;
- a description of the format of the binary FITS file that may be used to store the results from a **DustEMWrap** run.

3 DustEMWrap installation

This section describes how to obtain and install the current version of the DustEM fortran and **DustEMWrap** codes.

The DustEM fortran code and its documentation are available at :
http://www.ias.u-psud.fr/DUSTEM

The current release of **DustEMWrap** is distributed via the website:
http://dustemwrap.irap.omp.eu

The main steps of the installation (in order) are:

1. Download the DustEM fortran release. The **DustEMWrap** V4.3 release has been tested for V4.3 of DustEM.

2. Edit the DM_constants.f90 file in the src folder of the DustEM fortran release by setting the data_path variable to point to the empty directory on your disk where you want **DustEMWrap** to write out temporary data while it is executing.
   Warning: This should be an empty directory to which you have exclusive read/write access, since the contents will be erased by DustEMWrap each time the code is executed..
   For instance:
   `CHARACTER (len=100) :: data_path='/path_to_user_home/tmp/dustem/'`

3. Recompile the DustEM fortran following the instructions given in the DustEM documentation.[1]

4. Obtain **DustEMWrap** from the **DustEMWrap** website and place it in a dedicated directory. For the rest of this Guide, we assume that **DustEMWrap** is located in
   /path_to_user_home/Soft/DUSTEM_WRAP/
   and that the DustEM fortran package is in
   /path_to_user_home/Soft/DUSTEM/

5. Include the following lines in your idl_startup file, ensuring that the variables match your installation. The values below are given for the example install described in this document:

   ```
   defsysv,'!dustem_wrap_soft_dir','/path_to_user_home/Soft/DUSTEM_WRAP/'
   defsysv,'!dustem_soft_dir','/path_to_user_home/Soft/DUSTEM/'
   defsysv,'!dustem_which','RELEASE'
   defsysv,'!dustem_dat','/path_to_user_home/tmp/dustem/'
   defsysv,'!dustem_res','/path_to_user_home/tmp/dustem/'
   defsysv,'!dustem_f90_exec','/path_to_user_home/Soft/DUSTEM/'+'/src/dustem'
   ```

   Note that the / character is needed at the end of paths.

   The variable !dustem_f90_exec must point to the DustEM fortran executable as produced in step 3.

6. Include the following line in your idl_startup file, which will add the **DustEMWrap** code to your **IDL** path
   `!path=!path+':'+expand_path('+'+!dustem_wrap_soft_dir+'/src/idl/')`

[1] Compiling the DustEM fortran requires the user to specify a directory for the input and output of the Meudon PDR code (dir_PDR variable in the DM_constants.f90 file). This interoperability is not available via **DustEMWrap**. **DustEMWrap** users can set the dir_PDR variable according to the DustEM fortran instructions, i.e. to the absolute path to the DustEM out/ sub-directory, or to the data_path directory defined above.

7. **DustEMWrap** needs several external libraries (see Sect. 3.1 for a complete description). A static version of these libraries is distributed with the **DustEMWrap** code. If you prefer to use them (rather than your pre-existing system versions), or if you don't have these external libraries already installed on your system, then you should also include the following line in your `idl_startup` file, which adds the `idl_extern` directory to your path :
`!path=!path+':'+expand_path('+'+!dustem_wrap_soft_dir+'/src/idl_extern')`
Your path should be constructed so that the `idl_extern` routines are privileged over versions elsewhere in your system.

8. Open an **IDL** session and test your installation using the `dustem_init` command, e.g.
`IDL> dustem_init,/wraptest,/plot`
If this command executes successfully, a graphical window will display the emission from the default dust model used by the `DustEM fortran` code (see Figure 1), and the results from running the `DustEM fortran` are printed to the terminal.

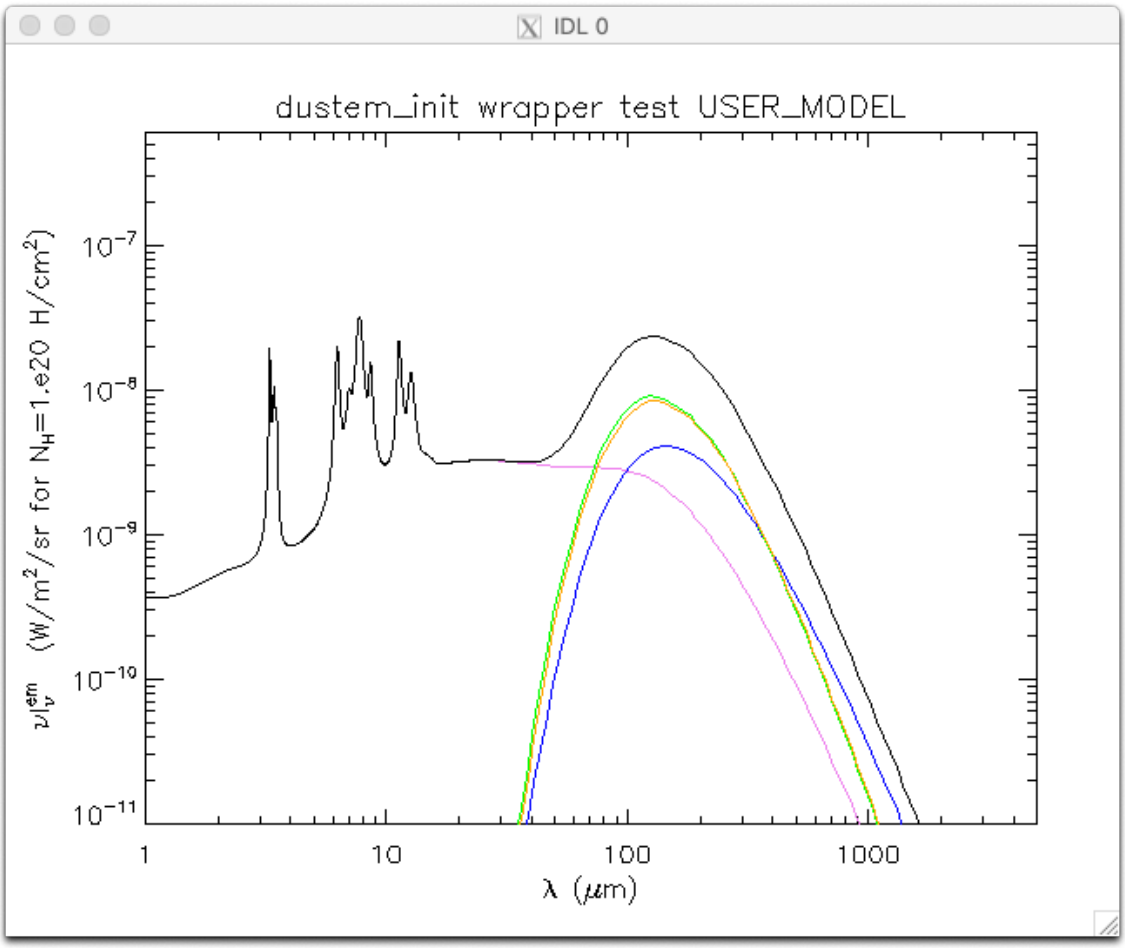

Figure 1: *Screen grab illustrating the graphical output of* `IDL> dustem_init,/wraptest,/plot`. *The emission due to individual grain populations in the model is indicated in color, and the total (i.e. all grain populations combined) dust emission is overplotted as a black curve.*

There are a few options to the `dustem_init`, e.g. you can choose the physical dust model via the `model=` keyword. See
`IDL> dustem_init,/help`
for more information.

3.1 DustEMWrap dependencies

DustEMWrap uses routines from the following IDL Libraries:

- Coyote Library : https://github.com/idl-coyote/
- HEALPix IDL routines: https://sourceforge.net/projects/healpix/
- IDL Astronomy Library: https://github.com/wlandsman/IDLAstro/
- ISO Calibration (ICE) Library: contact dustemwrap@irap.omp.eu
- JPB IDL Library: contact dustemwrap@irap.omp.eu
- Markwardt IDL Library: http://www.physics.wisc.edu/~craigm/idl/idl.html
- TextoIdl: http://physics.mnstate.edu/craig/textoidl/

These libraries can be downloaded and installed according to the default instructions for each library. Alternatively, the routines from these libraries that are used by **DustEMWrap** are included in the **DustEMWrap** distribution (as described in optional installation step 7 above). If you encounter unexpected errors when using the latest versions of the above **IDL** libraries, please let us know (dustemwrap@irap.omp.eu).

3.2 DustEMWrap and IDL

DustEMWrap was originally written in the IDL language (ca. 2010, IDL version 6) on a Mac. It has subsequently been developed and regularly tested on a Mac Book Pro and Linux machines (CentOS/Ubuntu) running IDL versions up to and including 8.4. The current development team has less experience running **DustEMWrap** under Windows, and welcomes any feedback.

Since 2021, **DustEMWrap** has been tested on Macs and Linux machines running the following free alternatives to IDL:

- **GDL**, GNU Data Language, a free implementation similar to IDL.
  See https://gnudatalanguage.github.io/
- **Fawlty (FL)**, an IDL8 (Interactive Data Language) compatible compiler.
  See http://www.flxpert.hu/fl/

The current version of **DustEMWrap** exhibits a similar computation time under **GDL**, **FL** and **IDL**, measured using a limited number of test cases. If you encounter issues running **DustEMWrap** under these **IDL** alternatives, please get in touch since we aim to achieve full compatibility by the next release.

4 An overview of the DustEMWrap framework

4.1 Running DustEM using DustEMWrap

DustEMWrap has five main functionalities that allow the user to run DustEM from within an **IDL** session:

- Reading the fortran input files. These files contain parameters that are used by the DustEM fortran code, such as dust composition, interstellar radiation field (ISRF), dust optical cross sections, etc. They are described in detail in the DustEM documentation.
- Storing the above inputs into **IDL** variables. The main variable is an **IDL** system variable (called !dustem_params) which is available from any **IDL** routine. This allows the **DustEMWrap** user to modify the input parameters from within their **IDL** session.
- Writing the updated input parameters into a set of files that will be read by the DustEM fortran code. These modified files are written to the directory specified in !dustem_dat.
- Launching DustEM with the appropriate, user-modified input files. This is done via the dustem_run.pro function.
- Reading the results of a DustEM run, and storing them into **IDL** structures that may then be used for plotting and further analysis.

The output structures returned by the execution of dustem_run.pro are :

1. The emission output structure, st.dustem:
WAV : emission wavelength (microns)
EM_GRAIN_1 : emission due to grain type ♯ 1 ($4\pi\nu I_\nu$ in erg/s/cm^2/H)
EM_GRAIN_2 : emission due to grain type ♯ 2 ($4\pi\nu I_\nu$ in erg/s/cm^2/H)
etc ...
EM_TOT : integrated emission from all grain types ($4\pi\nu I_\nu$ in erg/s/cm^2/H)

2. The extinction output structure, st.ext:
WAV : extinction wavelength (in microns)
ABS_GRAIN : Absorption of each grain type per dust mass (in cm^2/g)
SCA_GRAIN : Scattering of each grain type per dust mass (in cm^2/g)
EXT_TOT : the total extinction due to all grain types

A minimal example of how to run DustEM from within **DustEMWrap** (and perform the above operations) is presented in Sect. 11.

4.2 Running DustEM iteratively using DustEMWrap

In addition to allowing a user to run the DustEM fortran from within **IDL**, **DustEMWrap** also provides tools to iteratively fit SEDs with DustEM. In this case, the fit is performed according to the steps below:

- The observational data is read and stored in a dedicated **IDL** system variable (called !dustem_data). This is performed using dustem_set_data.pro. SEDs and extinction curves can be constructed from within an **IDL** routine or read from a text file (with the extension .xcat). Example input files are included in the current **DustEMWrap** release in the Data/EXAMPLE_OBSDATA/ subdirectory. A description of the required input data format is given in Sect. 7.
- The parameters to fit are selected by the user, along with their allowed range and initial values. The user can define a maximum of four vectors: (1) the parameter description vector containing the free parameters of the dust model and any plug-ins, followed by

one or more keyword pertaining to each parameter; (2) the initial values vector that contains the initial values of the free parameters; (3) a fixed parameter description vector that indicates the parameters to be held fixed; and (4) a fixed parameter initial values vector that contains the values of the fixed parameters. For each of the free parameters, control is provided via vectors that indicate if the parameter is upper/lower-bounded (`ulimed/llimed`) and specify the values of the upper and lower bounds `ulim/llim`.[2]

- The minimization is performed using the `dustem_mpfit_data.pro`, which uses the `dustem_mpfitfun.pr` and `dustem_mpfit.pro` functions.[3]. During minimization, the output spectrum of `DustEM` is color-corrected according to the flux convention used by each instrument and transmission information for each filter. Derivatives of the model with respect to the variable input parameters are computed numerically.

- The fit results – i.e. best-fit parameter values and associated errors, χ^2 and reduced χ^2 for the fit – are recovered, and optionally saved for later use.

A minimal example of how to run `DustEM` iteratively from within **`DustEMWrap`** (and perform the above operations) is presented in Sect. 11. A screen grab showing the output of a typical **`DustEMWrap`** run to fit total intensity data is shown in Figure 2. Additional panels and windows are displayed when **`DustEMWrap`** is used to fit polarization and/or extinction data.

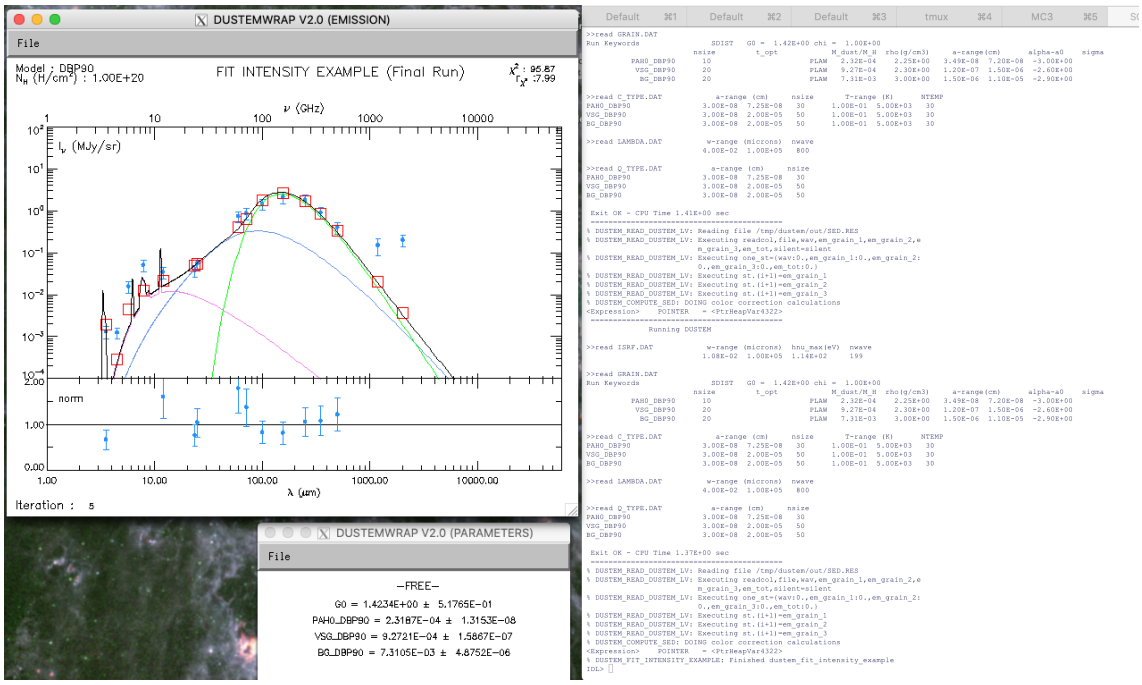

Figure 2: *Screen grab illustrating the output from running* **`DustEMWrap`** *to fit a total intensity (Stokes I) SED. The red points show the SED data, composed of spectral data (dots) and photometric data (squares). The yellow (spectral) and blue (photometric) dots show the best fitting SED. The dashed lines show the spectra for the various dust components used in the fit. The best fit values of free parameters, their uncertainties and the values of fixed parameters are reported in two small auxiliary GUI windows. The dust model, number of iterations, χ^2 and reduced χ^2 are indicated in the main plotting window.*

[2]Tied free parameters are not implemented in the current release of **`DustEMWrap`**.

[3]These are direct copies of the `mpfitfun.pro` and `mpfit.pro` functions from the Markwardt IDL Library. They have been renamed in the **`DustEMWrap`** distribution to avoid conflicts and assist debugging

5 Available dust models

The physical interstellar dust models that can be invoked via **DustEMWrap** are:

- `MC10` : Compiégne et al., 2011, A&A, 525, 103
- `DBP90` : Desert, Boulanger & Puget, 1990, A&A, 237, 215
- `DL01` : Draine & Li, 2001, ApJ, 554, 778
- `DL07` : Draine & Li, 2007, ApJ, 657, 810
- `G17_MODELA` : Guillet et al., 2018, A&A, 610, 16
- `G17_MODELB` : Guillet et al., 2018, A&A, 610, 16
- `G17_MODELC` : Guillet et al., 2018, A&A, 610, 16
- `G17_MODELD` : Guillet et al., 2018, A&A, 610, 16
- `J13` : Jones et al., 2013, A&A, 558 , 62
- `WD01_RV5P5B` : Weingartner & Draine, 2001, ApJ, 548, 296 with $R_V = 5.5$

Note that Guillet et al. (2018) models specifically address the polarisation of the thermal dust emission. **DustEMWrap** therefore requires Stokes Q and Stokes U information to perform SED-fitting with these models.

The DustEMWrap team is currently working to incorporate the 'astrodust' model presented in Hensley & Draine (2022). To keep informed of the next release that includes this model, please subscribe to our mailing list.

6 Dust model parameters

DustEMWrap allows the user to fit an observational SED by optimising a user-defined combination of dust model and plug-in parameters. During the fit, parameters of the dust model and plugins may be fixed, upper- and/or lower-bounded, or left free to vary.

During a **DustEMWrap** run, the parameters of the dust model are stored in the system variable `!dustem_params`. The dust grain properties are accessed via the structure `(*!dustem_params).grains`, where `(*!dustem_params).grains(0)` is the first grain type in the model, `(*!dustem_params).grains(1)` is the second grain type in the model, etc. The dust grain properties that can be fit correspond to the quantities defined in the `GRAIN_MDL.DAT` files that are found in the `data` subdirectory of the `DustEM fortran` distribution.

Specifically, for each grain type in a dust model, there are:

- `MDUST_O_MH` : the abundance of the grain type relative to H
- `RHO` : the grain material specific mass density in g/cm^{-3}
- `AMIN` : the minimum grain size in cm
- `AMAX` : the maximum grain size in cm
- `ALPHA_O_A0` : the centroid (A0) of a log-normal, or the index of a power law (ALPHA) that describes the grain size distribution.
- `AT` : shape parameter 1 of an exponential decay to a power-law grain size distribution
- `AC` : shape parameter 2 of an exponential decay to a power-law grain size distribution
- `GAMMA` : shape parameter 3 of an exponential decay to a power-law grain size distribution
- `AU` : shape parameter 1 of curvature term of a power-law grain size distribution
- `ZETA` : shape parameter 2 of curvature term of a power-law grain size distribution
- `ETA` : shape parameter 3 of curvature term of a power-law grain size distribution

The other tags in the structure identify the grain type and its internal description in the code, but cannot be adjusted during the fit. These are

- `GRAIN_TYPE` : the name of the grain type
- `NSIZE` : the number of intervals used to numerically define the grain size distribution
- `TYPE_KEYWORDS` : keywords that specify the form of the grain size distribution, and whether polarisation information is specified in the model. This cannot be included in the fit, but the user can set non-default values via the `grain_keywords` keyword of `dustem_init.pro`.

The `DustEM fortran` Users' Guide provides a detailed explanation of all the dust model parameters. The most common parameters that users wish to fit are the abundance of the different dust grain types specified by a dust model, and the intensity of the local ISRF that heats the dust.[4] Taking the `DBP90` model with three grain types as an example (see also Section 11), a user might wish to determine the best-fitting relative abundance of PAHs, small grains and large grains from their observational data, also leaving the ISRF intensity as a free parameter. In this case, the parameters that should be included in the fit are:

- `(*!dustem_params).G0` : the ISRF intensity (the same for all dust grain types).
- `(*!dustem_params).grains(0).MDUST_O_MH` : the abundance of the first grain type (here `PAH0_DBP90`)

[4]By default, the spectral shape of the ISRF used by `DustEM` is the standard Mathis et al (1983) radiation field.

- `(*!dustem_params).grains(1).MDUST_O_MH` : the abundance of the second grain type (here `VSG_DBP90`)
- `(*!dustem_params).grains(2).MDUST_O_MH` : the abundance of the third grain type (here `BG_DBP90`)

At the start of any **`DustEMWrap`** run, the dust grain properties are initialised with their default values, as read from the `GRAIN_MDL.DAT` files. Their initial values may be modified by the user before launching the fit (see also Section 11). Parameters that the user excludes from the fit will be held fixed at their default values.

As well as parameters of the dust models, users may wish to include parameters of any plug-ins in their fit. The use of plug-ins is described in Section 10 (see also Section 11).

7 Format requirements for input data

7.1 SED and extinction curve data

The observational SED and extinction curve data that is used by **DustEMWrap** is handled by the `dustem_set_data.pro` routine. The easiest way to construct the **IDL** structure containing this information is to read it from a text file (extension `.xcat`) using the `read_xcat.pro` routine. Input SED files for **DustEMWrap** can also be constructed during an **IDL** session by generating an empty structure (using `dustem_initialize_sed.pro`), filling it with the appropriate data and saving it into a `.xcat` file using the `write_xcat.pro` routine. The equivalent routine to create an empty extinction curve is `dustem_initialize_ext.pro`. The examples presented in Section 11 illustrate these methods. Undefined or missing values in any column requiring numerical data should be specified in the SED and extinction curvey `.xcat` files as `-32768`.

The information in the SED `.xcat` files includes the following fields:

- INSTRU: Instrument name.
- FILTER: Filter name. The list of filters known to **DustEMWrap** are described in Sect. 8. The `SPECTRUM` value indicates spectral data, for which no color correction is required.
- WAVE: This is the reference wavelength for filter measurements and the wavelength of the data for spectral data. For broadband data, the value in this field is for user convenience, and will be replaced by **DustEMWrap** during SED-fitting by the fiducial value for the filter. The fiducial wavelengths used by **DustEMWrap** for each filter are defined in the `instrument_description.xcat` file (see Section 8).
- STOKESI: This is the brightness (specific intensity) or flux value. **DustEMWrap** computes many of its outputs (e.g. dust abundances) assuming a gas column density of $N_{\rm H} = 10^{20}\,{\rm cm}^{-2}$. The user can change this reference gas column density via the `!dustem_HCD` system variable, i.e by setting `*!dustem_HCD`$= 10^{21}$.
- STOKESQ: Stokes Q for a gas column density of $N_{\rm H} = 10^{20}\,{\rm cm}^{-2}$
- STOKESU: Stokes U for a gas column density of $N_{\rm H} = 10^{20}\,{\rm cm}^{-2}$
- LARGEP: Polarized intensity for a gas column density of $N_{\rm H} = 10^{20}\,{\rm cm}^{-2}$
- SMALLP: Polarization fraction for a gas column density of $N_{\rm H} = 10^{20}\,{\rm cm}^{-2}$
- PSI: Polarization angle in degrees
- SIGMAII: Variance of Stokes I
- SIGMAQQ: Variance of Stokes Q
- SIGMAUU: Variance of Stokes U
- SIGMAIQ: Covariance of Stokes I and Stokes Q
- SIGMAIU: Covariance of Stokes I and Stokes U
- SIGMAQU: Covariance of Sokes Q and Stokes U
- SIGMA_LARGEP: Variance of LARGEP
- SIGMA_SMALLP: Variance of SMALLP
- SIGMA_PSI: Variance of PSI

The information in the extinction curve `.xcat` files includes the following fields:

- INSTRU: For extinction curves, this should always be set to `EXTINCTION`.
- FILTER: For extinction curves, this should always be set to `SPECTRUM`.
- WAVE: This is the wavelength of the data.
- EXT_I: This is the total extinction cross-section, assuming a gas column density of $N_{\rm H} = 10^{20}\,{\rm cm}^{-2}$.
- EXT_Q: Stokes Q extinction cross-section for a gas column density of $N_{\rm H} = 10^{20}\,{\rm cm}^{-2}$
- EXT_U: Stokes U extinction cross-section for a gas column density of $N_{\rm H} = 10^{20}\,{\rm cm}^{-2}$
- EXT_P: Polarized extinction cross-section for a gas column density of $N_{\rm H} = 10^{20}\,{\rm cm}^{-2}$
- EXT_SMALLP: Extinction polarization fraction for a gas column density of $N_{\rm H} = 10^{20}\,{\rm cm}^{-2}$
- PSI: Extinction polarization angle in degrees
- SIGEXTII: Variance of Stokes I
- SIGEXTQQ: Variance of Stokes Q
- SIGEXTUU: Variance of Stokes U
- SIGEXTIQ: Covariance of Stokes I and Stokes Q
- SIGEXTIU: Covariance of Stokes I and Stokes U
- SIGEXTQU: Covariance of Sokes Q and Stokes U
- SIGEXTP: Variance of LARGEP
- SIGEXTSMALLP: Variance of SMALLP
- SIGEXT_PSI: Variance of PSI

Users interested in polarisation should note that the SED and extinction curve fitting procedures implemented in **`DustEMWrap`** work exclusively with Stokes IQU parameters. The columns containing polarised intensity/extinction (P), polarisation fraction (SMALLP), polarisation angle (PSI) and their uncertainties are included only for user convenience, and are not used by **`DustEMWrap`** during calculations. The function `dustem_fill_sed_dependent_columns.pro` may be used to calculate polarised intensity, polarisation fraction and polarisation angle from a given set of Stokes IQU values, and to complete the corresponding columns of an SED input file. The equivalent routine for an extinction curve is `dustem_fill_ext_dependent_columns.pro`.

7.2 ISRF data

By default, the radiation field that heats the dust in the `DustEM fortran` code corresponds to the description of standard ISRF by Mathis et al (1983). **`DustEMWrap`** users can modify or replace this by an alternative ISRF description using a text file and the plugin routine `dustem_plugin_modify_isrf.pro`. In this case, the input text file should contain two columns, the wavelength (in microns) and the flux $4\pi I\nu$ in erg/s/cm2/Hz. The wavelength range over which the ISRF is sampled should cover 0.01 to 10^5 microns in 200 steps. The current version of **`DustEMWrap`** includes the routine `dustem_create_rfield.pro`, which may be useful for generating alternative ISRFs. The examples presented in Section 11 illustrate methods to modify the input ISRF.

8 Known instrument filters

The instrument filters known to **DustEMWrap** are listed in Tab. 1. Filters are attributed a unique name that starts with the instrument name, followed by a number starting from 1 and increasing with the reference wavelength of the filter. **DustEMWrap** reads the filter information from the file called `instrument_description.xcat`, which is distributed at the top-level of the **DustEMWrap** code.

For convenience, **DustEMWrap** includes several routines that allow the user to obtain information about the filters by specifying the unique filter name, e.g.
`IDL> dustem_filter2wav('MIRI1')`
`5.6361961`
will return the reference wavelength of the `MIRI1` filter,
`IDL> dustem_filter2fluxconv('SPIRE2')`
`nuInu=cste`
will return the flux convention of the `SPIRE2` filter. See the **DustEMWrap** API documentation on the **DustEMWrap** website for more examples (`http://dustemwrap.irap.omp.eu`).

9 Color corrections

Photometric instruments measure astronomical spectra within a finite bandwidth in frequency. The spectral response within the bandwidth is set by the optical elements of the instrument, such as the filters. The measurements in each filter (or bandwidth) are defined with respect to a reference wavelength λ_0. The in-band measurements correspond to the intensity at λ_0 of a spectrum with a specific spectral shape (called the flux convention) that yields the same power as was measured by the instrument. Different missions and instruments have adopted different flux conventions. A popular flux convention is the one originally adopted by the IRAS experiment and corresponds to "nu*Inu=cste", where the spectral shape of the fiducial spectrum is $I_\nu \propto \nu^{-1}$. More recently, JWST has opted for a $I_\lambda = \mathrm{cste}$ flux convention.

Color correction factors (K) are defined as the factor by which a given spectrum at wavelength λ_0 ($I_\nu(\nu_0)$) must be multiplied to obtain the measurement in a given photometric band using the adopted flux convention ($\tilde{I}^0_\nu$):

$$\tilde{I}^0_\nu = K \times I_\nu(\nu_0), \tag{1}$$

The color correction depends on the spectral shape of the instrument transmission and the intrinsic spectral shape of the source.

For the "nu*Inu=cste" flux convention ($I_\nu = \tilde{I}^0_\nu \times (\nu/\nu_0)^{-1}$), equating the in-band power of the reference spectrum and the actual spectrum leads to :

$$\int_0^\infty \tilde{I}^0_\nu (\frac{\nu}{\nu_0})^{(-1)} T(\nu)\,\mathrm{d}\nu = \int_0^\infty T(\nu) I_\nu\,\mathrm{d}\nu, \tag{2}$$

where $T(\nu)$ is the filter transmission. The color correction factor for this flux convention can therefore be computed as:

$$K = \frac{1}{\nu_0 I_\nu(\nu_0)} \times \frac{\int_0^\infty T(\nu) I_\nu\,\mathrm{d}\nu}{\int_0^\infty T(\nu)\nu^{-1}\,\mathrm{d}\nu}, \tag{3}$$

or equivalently for a wavelength integration:

$$K = \frac{\lambda_0}{I_\nu(\lambda_0)} \times \frac{\int_0^\infty T(\lambda) I_\nu \lambda^{-2}\,\mathrm{d}\lambda}{\int_0^\infty T(\lambda)\lambda^{-1}\,\mathrm{d}\lambda}. \tag{4}$$

For the $I_\lambda = \mathrm{cste}$ flux convention, the same reasoning leads to

$$\int_0^\infty \tilde{I}^0_\nu (\frac{\nu}{\nu_0})^{(-2)} T(\nu)\,\mathrm{d}\nu = \int_0^\infty T(\nu) I_\nu\,\mathrm{d}\nu, \tag{5}$$

$$K = \frac{1}{\nu_0^2 I_\nu(\nu_0)} \times \frac{\int_0^\infty T(\nu) I_\nu\,\mathrm{d}\nu}{\int_0^\infty T(\nu)\nu^{-2}\,\mathrm{d}\nu}, \tag{6}$$

which, for a wavelength integration, corresponds to:

$$K = \frac{\lambda_0^2}{I_\nu(\lambda_0)} \times \frac{\int_0^\infty T(\lambda) I_\nu \lambda^{-2}\,\mathrm{d}\lambda}{\int_0^\infty T(\lambda)\,\mathrm{d}\lambda}. \tag{7}$$

In **DustEMWrap**, the color corrections are computed using the routine `dustem_cc.pro`, which computes the SED value ($\tilde{I}^0_\nu$) for a set of instrument filters, given an input spectrum specified by its wavelength and specific intensity values. The routine optionally returns the color correction factors. The flux convention used for the color correction can be specified by the user but defaults to the convention listed in the distributed file `instrument_description.xcat`.

Computing color correction can be time-consuming. In order to save CPU time, starting with V1.2 of **DustEMWrap**, color corrections are not evaluated when computing model derivatives for each parameters. Instead the color correction value from the previous model evaluation is used (as stored in the `DustEM` system variable called `!dustem_previous_cc`).

Table 1: List of **DustEMWrap** filters in Version V4.3

Instrument	Filter	Filter Name[#]	Approx. Wavelength[†] [μm]
Photometric instruments			
AKARI	AKARI1	N2	2.4
AKARI	AKARI2	N3	3.2
AKARI	AKARI3	N4	4.1
AKARI	AKARI4	S7	7.
AKARI	AKARI5	S9W	9.
AKARI	AKARI6	S11	11.
AKARI	AKARI7	L15	15.
AKARI	AKARI8	L18W	18.
AKARI	AKARI9	L24	24.
AKARI	AKARI10	N24	65.
AKARI	AKARI11	WIDE-S	90.
AKARI	AKARI12	WIDE-L	140.
AKARI	AKARI13	N160	160.
ARCHEOPS	ARCHEOPS1	KS3.545GHz	550.
ARCHEOPS	ARCHEOPS2	KS3.353GHz	849.
ARCHEOPS	ARCHEOPS3	KS3.217GHz	1381.
ARCHEOPS	ARCHEOPS4	KS3.143GHz	2096.
BOLOCAM	BOLOCAM1	BOLOCAM_1p1mm	1106.
DIRBE	DIRBE1	DIRBE_1p25um	1.25
DIRBE	DIRBE2	DIRBE_2p2um	2.2
DIRBE	DIRBE3	DIRBE_3p5um	3.5
DIRBE	DIRBE4	DIRBE_4p9um	4.9
DIRBE	DIRBE5	DIRBE_12um	12.
DIRBE	DIRBE6	DIRBE_25um	25.
DIRBE	DIRBE7	DIRBE_60um	60.
DIRBE	DIRBE8	DIRBE_100um	100.
DIRBE	DIRBE9	DIRBE_140um	140.
DIRBE	DIRBE10	DIRBE_240um	240.
GISMO	GISMO1	GISMO_2mm	2000.
HAWCPLUS	HAWCPLUS1	A	53.
HAWCPLUS	HAWCPLUS2	B	63.
HAWCPLUS	HAWCPLUS3	C	89.
HAWCPLUS	HAWCPLUS4	D	155.
HAWCPLUS	HAWCPLUS5	E	214.
HFI	HFI1	HFI857GHz	349.8
HFI	HFI2	HFI550GHz	550.1
HFI	HFI3	HFI353GHz	849.3
HFI	HFI4	HFI217GHz	1381.5
HFI	HFI5	HFI143GHz	2096.5
HFI	HFI6	HFI100GHz	2997.9
IRAC	IRAC1	IRAC3p6	3.5
IRAC	IRAC2	IRAC4p4	4.5
IRAC	IRAC3	IRAC5p8	5.7
IRAC	IRAC4	IRAC8	7.8
IRAS	IRAS1	IRAS1	12.
IRAS	IRAS2	IRAS2	25.
IRAS	IRAS3	IRAS3	60.
IRAS	IRAS4	IRAS4	100.
IRS	IRS1	PUI_BLUE	15.8
IRS	IRS2	PUI_RED	22.3
ISOCAM	ISOCAM1	SW4	2.8
ISOCAM	ISOCAM2	SW7	3.0
ISOCAM	ISOCAM3	SW2	3.3
ISOCAM	ISOCAM4	SW1	3.6
ISOCAM	ISOCAM5	SW6	3.7
ISOCAM	ISOCAM6	SW9	3.9
ISOCAM	ISOCAM7	SW8	4.1
ISOCAM	ISOCAM8	SW5	4.0
ISOCAM	ISOCAM9	SW11	4.3
ISOCAM	ISOCAM10	SW3	4.5
ISOCAM	ISOCAM11	LW1	4.5
ISOCAM	ISOCAM12	SW10	4.6
ISOCAM	ISOCAM13	LW4	6.0
ISOCAM	ISOCAM14	LW5	6.8
ISOCAM	ISOCAM15	SW2	6.7
ISOCAM	ISOCAM16	SW6	7.7
ISOCAM	ISOCAM17	SW7	9.6
ISOCAM	ISOCAM18	SW8	11.3
ISOCAM	ISOCAM19	SW10	12.0
ISOCAM	ISOCAM20	SW3	14.3
ISOCAM	ISOCAM21	SW9	14.9
ISOPHOT	ISOPHOTP1	P3.29	3.3
ISOPHOT	ISOPHOTP2	P3.6	3.6
ISOPHOT	ISOPHOTP3	P4.85	4.8
ISOPHOT	ISOPHOTP4	P7.3	7.3
ISOPHOT	ISOPHOTP5	P7.7	7.7
ISOPHOT	ISOPHOTP6	P10	10.
ISOPHOT	ISOPHOTP7	P11.3	11.3
ISOPHOT	ISOPHOTP8	P11.5	12.
ISOPHOT	ISOPHOTP9	P12.8	12.8
ISOPHOT	ISOPHOTP10	P16	15.
ISOPHOT	ISOPHOTP11	P20	20.
ISOPHOT	ISOPHOTP12	P25	25.
ISOPHOT	ISOPHOTP13	P60	60.
ISOPHOT	ISOPHOTP14	P100	100.
ISOPHOT	ISOPHOTC1	C50	65.
ISOPHOT	ISOPHOTC2	C60	60.
ISOPHOT	ISOPHOTC3	C70	80.
ISOPHOT	ISOPHOTC4	C90	90.
ISOPHOT	ISOPHOTC5	C100	100.
ISOPHOT	ISOPHOTC6	C105	105.
ISOPHOT	ISOPHOTC7	C120	120.
ISOPHOT	ISOPHOTC8	C135	150.
ISOPHOT	ISOPHOTC9	C160	170.00
ISOPHOT	ISOPHOTC10	C180	180.00
ISOPHOT	ISOPHOTC11	C200	200.00
LABOCA	LABOCA1	LABOCA345GHz	870.
LFI	LFI1	LFI70GHz	4286.
LFI	LFI2	LFI44GHz	6818.
LFI	LFI3	LFI30GHz	10000.

Table 2: List of **DustEMWrap** filters in Version V4.3 (continued)

Instrument	Filter	Filter Name[#]	Approx Wavelength[†] [μm]
Photometric instruments			
MIPS	MIPS1	MIPS24	24.
MIPS	MIPS2	MIPS70	71.
MIPS	MIPS3	MIP160	156.
MIRI	MIRI1	F0560W	5.64
MIRI	MIRI2	F0770W	7.65
MIRI	MIRI3	F1000W	9.95
MIRI	MIRI4	F1065C	10.60
MIRI	MIRI5	F1140C	11.30
MIRI	MIRI6	F1130W	11.31
MIRI	MIRI7	F1280W	12.82
MIRI	MIRI8	F1500W	15.06
MIRI	MIRI9	F1550C	15.51
MIRI	MIRI10	F1800W	17.97
MIRI	MIRI11	F2100W	20.80
MIRI	MIRI12	F2300C	22.66
MIRI	MIRI13	F2550W	25.32
MSX	MSX1	B1	4.3
MSX	MSX2	B2	4.4
MSX	MSX3	A	8.3
MSX	MSX4	C	12.1
MSX	MSX5	D	14.7
MSX	MSX6	E	21.3
NIKA2	NIKA21	NIKA2_1mm	1153.
NIKA2	NIKA22	NIKA2_2mm	1999.
NIRCAM[‡]	NIRCAM1	F070W	0.70
NIRCAM	NIRCAM2	F090W	0.90
NIRCAM	NIRCAM3	F115W	1.15
NIRCAM	NIRCAM4	F140M	1.41
NIRCAM	NIRCAM5	F150W	1.50
NIRCAM	NIRCAM6	F162M	1.63
NIRCAM	NIRCAM7	F164N	1.64
NIRCAM	NIRCAM8	F150W2	1.66
NIRCAM	NIRCAM9	F182M	1.84
NIRCAM	NIRCAM10	F187N	1.87
NIRCAM	NIRCAM11	F200W	1.99
NIRCAM	NIRCAM12	F210M	2.10
NIRCAM	NIRCAM13	F212N	2.12
NIRCAM	NIRCAM14	F250M	2.50
NIRCAM	NIRCAM15	F277W	2.77
NIRCAM	NIRCAM16	F300M	2.99
NIRCAM	NIRCAM18	F322W2	3.23
NIRCAM	NIRCAM17	F323N	3.24
NIRCAM	NIRCAM19	F335M	3.36
NIRCAM	NIRCAM20	F356W	3.57
NIRCAM	NIRCAM21	F360M	3.62
NIRCAM	NIRCAM22	F405N	4.05
NIRCAM	NIRCAM23	F410M	4.08
NIRCAM	NIRCAM24	F430M	4.28
NIRCAM	NIRCAM25	F444W	4.40
NIRCAM	NIRCAM26	F460M	4.63
NIRCAM	NIRCAM27	F466N	4.65
NIRCAM	NIRCAM28	F470N	4.70
NIRCAM	NIRCAM29	F480M	4.82
NIRISS	NIRISS1	F090W	0.90
NIRISS	NIRISS2	F115W	1.15
NIRISS	NIRISS3	F140M	1.41
NIRISS	NIRISS4	F150W	1.50
NIRISS	NIRISS5	F158M	1.59
NIRISS	NIRISS6	F200W	1.98
NIRISS	NIRISS7	F277W	2.78
NIRISS	NIRISS8	F356W	3.59
NIRISS	NIRISS9	F380M	3.83
NIRISS	NIRISS10	F430M	4.29
NIRISS	NIRISS11	F444W	4.43
NIRISS	NIRISS12	F480M	4.82
PACS	PACS1	BLUE	70.
PACS	PACS2	GREEN	100.
PACS	PACS3	RED	160.
PILOT	PILOT1	PILOT240	240.
SCUBA2	SCUBA21	450um	449.
SCUBA2	SCUBA22	850um	861.
SPASS	SPASS1	–	130175.
SPIRE	SPIRE1	PSW	250.
SPIRE	SPIRE2	PMW	350.
SPIRE	SPIRE3	PLW	500.
SPM	SPM1	–	200.
SPM	SPM2	–	260.
SPM	SPM3	–	360.
SPM	SPM4	–	580.
WISE	WISE1	W1	3.4
WISE	WISE2	W2	4.6
WISE	WISE3	W3	12.
WISE	WISE4	W4	22.
WMAP	WMAP1	WMAP_W	3200.
WMAP	WMAP2	WMAP_v	4900.
WMAP	WMAP3	WMAP_Q	7300.
WMAP	WMAP4	WMAP_{Ka}	9100.
WMAP	WMAP5	WMAP_K	13000.

[#]Filter name used in the literature e.g. in a Mission Explanatory Supplement. Otherwise, a name commonly used to denote the filter.

[†]Wavelengths in this table are indicative only. The distributed `instrument_description.xcat` file lists the precise reference wavelength that is used by **DustEMWrap** for the color correction calculations.

[‡] By default, `DustEMWrap` uses averaged module A and B NIRCAM throughputs.

The available flux conventions are listed below:

- "NUINU=cste": A widely used for infrared missions. Assumes a reference spectrum $I_\nu = \tilde{I}^0_\nu \times (\nu/\nu_0)^\beta$ with $\beta = -1$. If an instrument's flux convention cannot be determined from the literature, **`DustEMWrap`** assumes this convention by default.
- "FLAMBDA=cste": The flux convention used by JWST. Assumes a reference spectrum $I_\nu = \tilde{I}^0_\nu \times (\nu/\nu_0)^\beta$, with $\beta = -2$.
- "FNU=cste": Assumes a reference spectrum $I_\nu = \tilde{I}^0_\nu \times (\nu/\nu_0)^\beta$ with $\beta = 0$.
- "IRAC': The flux convention used by the IRAC instrument on the Spitzer satellite. It is similar to the"nu*Inu=cste" convention, but the spectral response used is in electrons/photon instead of ergs/photon, leading to a slightly different formula.
- "MIPS": The flux convention used by the MIPS instrument on the Spitzer satellite [2]. The reference spectrum is a blackbody at $T = 10^4\,K$.
- "CMB": The flux convention for measurements originally provided in CMB temperature units (K_{CMB}), which have been simply transformed to specific intensity using $I_\nu = T_{CMB} \times (\mathrm{d}B_\nu/\mathrm{d}T)\,(T_{CMB})$. The reference spectrum in that case is assumed to have $I_\nu \propto (\mathrm{d}B_\nu/\mathrm{d}T)\,(T_{CMB})$.
- "HFI": This is algorithmically the same as "nu*Inu=cste" above, but specifies that the original HFI color correction **`IDL`** routine as published by the Planck consortium is used, rather than the native **`DustEMWrap`** routines.
- "NIKA2": Assumes a reference spectrum $I_\nu = \tilde{I}^0_\nu \times (\nu/\nu_0)^\beta$, with $\beta = 1.6$.

As noted above, **`DustEMWrap`** assumes a "NUINU=cste" flux convention by default, i.e. if we have not identified a publication that clearly describes an alternative convention. **We welcome any suggestions for published information about instruments and their flux conventions that we may have missed!**

10 Plug-ins

Plug-ins are a fundamental component of the **DustEMWrap** software that allow users to include information in the fitting and creation of SEDs that is not handled by the `DustEM fortran` code. One common example is fitting an observational SED with measurements at millimetre wavelengths when the user suspects that free-free and/or synchrotron emission may be present (in addition to thermal dust emission). Other examples include (i) modifying the spectral shape of the ISRF (and not just its intensity), and (ii) fitting some parts of the SED using blackbody or modified blackbody functions, rather than a physical interstellar dust model.

The current **DustEMWrap** release includes several plug-ins that may be of interest for common science applications. These are:

- `dustem_plugin_continuum`: a blackbody continuum emission component.
- `dustem_plugin_freefree`: a free-free emission component.
- `dustem_plugin_synchrotron`: a synchrotron emission component.
- `dustem_plugin_modify_dust_pol`: applies a polarization fraction and/or a polarization angle to the `DustEM` output in emission. It is needed to predict polarization fraction and angle in emission.
- `dustem_plugin_modify_dust_polx`: applies a polarization fraction and/or a polarization angle to the `DustEM` output in extinction. It is needed to predict polarization fraction and angle in extinction.
- `dustem_plugin_mbbdy`: a modified blackbody continuum emission component
- `dustem_plugin_stellar_population`: constructs an ISRF component due to stars of a user-specified spectral type(s) that contribute to heating the dust. The resulting ISRF replaces/can be added to the default `DustEM` ISRF. The current **DustEMWrap** release provides data for main-sequence stars only. Other spectral types will be implemented in future **DustEMWrap** releases.
- `dustem_plugin_modify_isrf`: constructs an ISRF component from a specified file. The resulting ISRF replaces/can be added to the default `DustEM` ISRF.

The parameters and scope of these provided plugins are described in Table 3. Multiple plug-ins can be used for a single **DustEMWrap** run.

Scopes are strings governing how the plugin function behaves with respect to the **DustEMWrap** model used to fit the data. For plugins describing processes that contribute to the observed emission and/or extinction, the plugin scope should include the prefix 'ADD' or 'REPLACE' followed by the four available modes (['_SED','_POLSED','_EXT','_POLEXT']) to instruct **DustEMWrap** how to combine the plugin with the dust model. This yields scopes like 'ADD_SED' or 'REPLACE_POLSED'. Scopes can be combined by a '+' symbol provided that they operate in both emission or in extinction. As a result, combinations such as 'ADD_SED+ADD_POLSED' can be used. The 'ADD' prefix adds a plugin's contribution to the model used to fit the data and the 'REPLACE' replaces the model by the plugin's contribution.

Among the distributed plug-ins, the plug-ins that modify the dust-heating ISRF are somewhat different. These ISRF plugins are added to the default ISRF used by `DustEM` (`=G0*Mathis`). They can be used alone or in combination. For example, if the user invokes the `dustem_plugin_stellar_population` plugin to construct a radiation field, the ISRF used by the `DustEM fortran` code will be `G0*Mathis+stellar_irsf()`, where `stellar_irsf()` is a function of the parameters set by `dustem_plugin_stellar_population`. If a user invokes the `dustem_plugin_stellar_population` and `dustem_plugin_modify_isrf` plugins simultaneously, then the ISRF used by the `DustEM fortran` will be `G0*Mathis+stellar_irsf()+my_isrf()` (where `my_isrf()` is the ISRF provided in a text file). When using these plugins, it is important to remember the general rule of **DustEMWrap** that parameters that are not explicitly specified as free/fixed will be held fixed at their default values. To suppress the Mathis component of the ISRF entirely, a user should specify `(*!dustem_params).G0` as a fixed parameter with a very

small value (e.g. 1.e-12), otherwise it will assume its default value (`G0=1`).

During a **DustEMWrap** run, the parameters of any plug-in that the user wishes to include in the fit are specified differently to dust model parameters. Plug-in parameters are accessed via strings that are included in the parameter description vector (see Section 11). The string is composed of the plug-in name, and a suffix indicating the parameter, e.g. `dustem_plugin_synchrotron_2` indicates the synchrotron intensity at 1 cm (see Table 3). Plug-in parameters can be fixed, bounded or free. This information and the initial values of any plug-in parameters are specified in the same way as for dust model parameters.

Experienced **DustEMWrap** users are welcome to write their own plug-ins. For help with writing plug-ins, please see the **DustEMWrap** Developers' Guide (or contact us at `dustemwrap@irap.omp.eu`).

Table 3: Overview of plug-ins provided in the current **DustEMWrap** release.

Plug-in Name `dustem_plugin...`	**Scope**	**Suffix**	**Description**	**Tag**	**Default Value** Value
`_continuum`	'ADD_SED'	1	Temperature of the blackbody (BB)	T_{BB} [K]	1000.
		2	Peak intensity of the BB emission	Amp	0.01
`_mbbdy`	'ADD_SED'	1	Peak intensity of the modified BB emission	Amp	0.5
		2	Temperature of the modified BB	T_{MBB} [K]	20.0
		3	Emissivity index	β	1.8
`_freefree`	'ADD_SED'	1	Ionized gas temperature	T_{gas} [K]	10000.
		2	Peak intensity of the free-free emission	Amp	1
`_synchrotron`	'ADD_SED+ADD_POLSED'	1	Spectral index of CREs	s	3
		2	Intensity at 1cm	Amp	1
		3	Polarization angle	psi	0.
		4	Polarization fraction	smallp	0.3
`_stellar_population`	'STELLAR_POPULATION'	XXX1†	Stellar radius	R_{star} [$R_\odot$]	Spec type-dependent
		XXX2	BB temperature	T_{BB} [K]	Spec type-dependent
		XXX3	Stellar distance	D [pc]	Spec type-dependent
		XXX4	Number of stars	N_{stars}	Spec type-dependent
`_modify_dust_pol`	'REPLACE_POLSED'	1	Polarization fraction	p	1.
		2	Polarization angle	Psi [deg]	0.
`_modify_dust_polx`	'REPLACE_POLEXT'	1	Polarized extinction fraction	p	1.
		2	Polarization angle in extinction	Psi [deg]	0.
`_modify_isrf`	'USER_ISRF'	1	Amplitude of spectral shape	Amp	1.

† XXX : keyword identifying spectral type and luminosity class of the stellar population. For example, to specify the distance to a population of O5V stars, the plug-in syntax is `dustem_plugin_stellar_population_O5V3`. To specify the number of B2V stars, the syntax `dustem_plugin_stellar_population_B2V4`.

11 Getting started

The current release of **DustEMWrap** includes several example routines illustrating how to run the code. These routines have filenames ending in `_example.pro`, and can be found in the `src/idl/` directory. The corresponding data files are located in the `Data/EXAMPLE_OBSDATA/` directory.

New users are encouraged to run these example routines using the default parameters, and then to explore modifying different keywords and the input data. The aim of these examples is for new users to rapidly gain confidence in running **DustEMWrap** on their own data for the most common types of scientific analysis that are possible with **DustEMWrap**. The examples do not cover all possible use cases of **DustEMWrap**, and users are encouraged to contact
The examples distributed with **DustEMWrap** V4.3 are:

- `dustem_run_example.pro` : Illustrates how to run the `DustEM fortran` code from within an IDL session and obtain several diagnostic plots of the `fortran` input/output.
- `dustem_fit_intensity_example.pro` : Illustrates how to fit an observational SED with total intensity (i.e. Stokes I only) broadband measurements.
- `dustem_fit_intensity_mbb_example.pro` : Illustrates how to fit an observational Stokes I SED using a modified black-body rather than a physical dust model.
- `dustem_fit_polarisation_example.pro` : Illustrates how to fit an observational SED with Stokes IQU measurements using an interstellar dust model that includes polarisation.
- `dustem_make_polarisation_sed_example.pro` : Illustrates how to generate an SED with Stokes IQU measurements using an interstellar dust model that includes polarisation.
- `dustem_fit_spectro_example.pro` : Illustrates how to fit an observational SED with a combination of photometric and spectrometer data.
- `dustem_fit_ext_example.pro` : Illustrates how to fit extinction measurements (Stokes I only).
- `dustem_fit_ext_pol_example.pro` : Illustrates how to fit extinction measurements (Stokes IQU).
- `dustem_fit_sed_ext_stokesi_example.pro` : Illustrates how to fit observational data that is a combination of emission and extinction measurements (Stokes I only).
- `dustem_fit_sed_ext_pol_example.pro` : Illustrates how to fit observational data that includes a combination of emission and extinction measurements (Stokes IQU).
- `dustem_stellarpopisrf_example.pro` : Illustrates how to use the stellar population ISRF plug-in.
- `dustem_myisrf_example.pro` : Illustrates how to use the modify ISRF plug-in.
- `dustem_extract_sed_example.pro` : Illustrates how to use the SED extractor helper tool.

Step-by-step instructions for running these examples are provided on the **DustEMWrap** website(`http://dustemwrap.irap.omp.eu`).

12 Modification history

- V1.2: Implemented official Planck HFI color corrections.
- V1.2: Color corrections not evaluated when re-computing derivatives with respect to fit parameters.
- V1.3: Implemented all AKARI filters (thanks to Ryou Ohsawa).
- V1.4: Added NIKA2, SCUBA2, IRS filters (thanks to Longji Bing). Made compatible with `DustEM` V4.3. Added free-free and synchrotron plugins.
- V4.3: (this release) Added JWST instrument filters. Added SABOCA, HAWCPLUS, NIKA2 and additional AKARI filters. Corrected flux conventions for WISE and MSX filters. Added polarisation capabilities. Added continuum plugins. Added ISRF-related plugins.

References

[1] F. Boulanger et al. “The dust/gas correlation at high Galactic latitude.” In: 312 (Aug. 1996), pp. 256–262.

[2] MIPS Instrument and MIPS Instrument Support Teams. *MIPS Instrument Handbook - Summary*. https://irsa.ipac.caltech.edu/data/SPITZER/docs/mips/mipsinstrumenthandbook/. 2011.

Appendices

Configuring IDL to run DustEMWrap

Some preliminary configuration is needed to run **DustEMWrap** and enable the DustEM code to run from within the **IDL** environment. This configuration is done by adding some lines to a user's idl_startup file, usually located in the user's home area. The full path to the idl_startup should be specified using the IDL_STARTUP environment variable in your .login/.bashrc (or equivalent).

To create the idl_startup file (if it does not already exist), type:
$ touch idl_startup
and modify it with an editor of your choice. Recent versions of **GDL** and **FL** will use this same configuration file. The following lines present a minimal idl_startup file to use **DustEMWrap**.

```
defsysv,'!sep','/'
defsysv,'!psep',':'

;=== Define required environment variables

;=== the top-level directory of DustEM fortran
defsysv,`!dustem_soft_dir',`/path_to_dustem_fortran_dir/dustem_fortran/'

;=== the DustEM fortran executable
defsysv,'!dustem_f90_exec',`/path_to_dustem_fortran_dir/dustem_fortran/src/dustem'

;=== the top-level directory of DustEMWrap
defsysv,`!dustem_wrap_soft_dir','/path_to_dustemwrap_dir/dustem-wrapper_idl/'

;=== specify directory architecture for DustEMWrap
defsysv,`!dustem_which','RELEASE'

;=== working directories for DustEMWrap
defsysv,'!dustem_dat','/path_to_tmpdir/dustem/'
defsysv,'!dustem_res','/path_to_tmpdir/dustem/'

;=== Path to DustEMWrap IDL routines
!path=!path+!psep+expand_path(`+'+!dustem_wrap_soft_dir)
!path=!path+!psep+expand_path(`+'+!dustem_wrap_soft_dir+`/src/idl/')
!path=!path+!psep+expand_path(`+'+!dustem_wrap_soft_dir+`/src/idl_extern/')

;===if you need to check your path, you can print it using
;print,!path
```

`DustEMWrap` Routine Descriptions

API documentation for all **`DustEMWrap`** routines is available from the **`DustEMWrap`** website (`http://dustemwrap.irap.omp.eu`).

`DustEMWrap` IDL System Variables

The system variables that may be defined by the **`DustEMWrap`** code during execution are:

- !dustemcgwin_id: struct,
- !dustemcgwin_ncmds: struct,
- !dustem_current : pointer,
- !dustem_data: pointer, contains SED (emission and extinction) data.
- !dustem_do_cc: 1/0 flag, used by **`DustEMWrap`** to control color correction calculations during model derivative calculations
- !dustem_end: 1/0 status flag that specifies whether fitting is complete
- !dustem_filters: Contains filter information. This is initialized by `dustem_filter_init.pro`
- !dustem_fit: pointer, contains SED fit parameters, such as free and fixed parameters definition, initial values, chi2 and reduced chi2 etc.
- !dustem_hcd: pointer, contains reference hydrogen column density value
- !dustem_inputs: pointer, contains `fortran` input files that should be used by default.
- !dustem_instrument_description: struct, contains instrument/filter information known to **`DustEMWrap`**.
- !dustem_iter: struct, keeps track of previous and current iteration number
- !dustem_kwords: pointer, `DustEM fortran` keywords for different grain populations in the active dust model
- !dustem_mlog: 1/0 flag, internal variable for plotting negative Stokes Q and U using pseudo-logarithmic axes
- !dustem_model: string, active interstellar dust model
- !dustem_never_do_cc: If set to 1, **`DustEMWrap`** will skip all color correction calculations. This is not recommended for scientific applications but enables a faster quick look.
- !dustem_nocatch:
- !dustem_noobj: 1/0 flag that specifies whether object-based print commands should be used.
- !dustem_params: pointer, contains current parameters of the model.
- !dustem_parinfo: pointer, contains current parameters of the model.
- !dustem_plugin: pointer, contains information about current active plugins
- !dustem_plot_range: struct, contains information about axis ranges and labelling of plots
- !dustem_previous_cc: pointer, contains the color correction information for the active instrument filters
- !dustem_redshift: float, presumed redshift of source. The plugin related to redshift conversions is still under development and not included in the current release (**`DustEMWrap`** V4.3).
- !dustem_show: pointer, contains the SED and extinction information that will be shown on plots during the minimization.

- !dustem_show_plot: 1/0 flag, specifies whether SED plots are plotted to screen during the minimization.
- !dustem_verbose: If set to 1, **DustEMWrap** is verbose, otherwise (sort of) quiet.
- !dustem_version: struct, specifies the **DustEMWrap** code version as string and float
- !dustemwrap_which: string, specifies which **DustEMWrap** code directory structure is used. This will be deprecated in the near future since all future public releases will uniquely use the directory structure corresponding to 'RELEASE'.
- !dustemwrap_which_language: string, specifies whether **DustEMWrap** code is executed from an IDL/GDL/Fawlty session.

Several of the most important **IDL** system variables used by **DustEMWrap** have non-trivial content. Their structure is described below.

- !dustem_fit:
  DATA : UNUSED SO FAR
  WAVELENGTH : Wavelengths present in the input SED `.xcat` file
  PARAM_DESCS : Free parameter(s) description
  PARAM_INIT_VALUES : Free parameter(s) initial value(s)
  PARAM_FUNC : Assign an index to each
  FIXED_PARAM_DESCS: Fixed parameter(s) description
  FIXED_PARAM_INIT_VALUES: Fixed parameter(s) value(s)
  CHI2: Current χ^2
  RCHI2: Current reduced χ^2
  CURRENT_PARAM_VALUES: Current parameter(s) value(s)
  CURRENT_PARAM_ERRORS: Current parameter(s) uncertainty value(s)

- !dustem_data: Contains SED data.
  INSTRU_NAMES : Instrument name
  FILT_NAMES : Filter names
  WAV : Wavelengths
  VALUES : Intensity values
  SIGMA : $1-\sigma$ uncertainty values

- !dustem_params: Contains current parameters of the model. NGRAINS: number of grain types
  G0 : Value of the G0 parameter
  KEYWORDS : Keywords that specify the form of the grain size distribution, and whether polarisation information is specified in the model.
  GRAINS : structures describing grain types
  ISRF : ISRF intensity array
  QABS : Qabs values for each grain type.
  CALOR : Heat capacities for each grain type
  LAMBDA : Wavelength used in emission calculations.
  SIZE : size distribution information for each grain type.
  MIX : Mixture information for each grain type.

- !dustem_filters: Contains filter information. In **DustEMWrap** V4.3, this is :
  IRAC : Spitzer IRAC filters
  MIPS : Spitzer MIPS filters
  MSX : MSX filters
  IRAS : IRAS filters
  DIRBE :DIRBE filters

SPM : Pronaos SPM filters
ISOCAM: ISOCAM filters
ISOPHOTP : ISOPHOTP filters
ISOPHOTC : ISOPHOTC filters
EFIRAS: Equivalent bandwidth for FIRAS used in [1]
ARCHEOPS :ARCHEOPS filters
HFI : Planck HFI filters
LFI : Planck LFI filters
WMAP : WMAP filters
SPIRE : Herschel SPIRE filters
PACS : Herschel PACS filters
PILOT : PILOT filters
AKARI : AKARI filters
BOLOCAM:bolocam filters
WISE : WISE filters
LABOCA : LABOCA filters
SABOCA : SABOCA filters
GISMO : GISMO filters
SPASS : SPASS filters
NIKA2 : NIKA2 filters
SCUBA2 : SCUBA2 filters
IRS : IRS filters
MIRI : JWST MIRI filters
NIRCAM : JWST NIRCam (module AB averaged) filters
NIRISS :JWST NIRISS filters

`DustEMWrap` FITS Output Description

`DustEMWrap` provides the option of saving its fitting results in a standard binary FITS file. The current release includes two helper routines, `dustem_write_fits_table.pro` and `dustem_read_fits_table.pro`, to facilitate the reading and writing of these files.

The `DustEMWrap` FITS files contain multiple extensions and have the following characteristic format:

- **Extension 1: input observed SED**. The binary FITS table includes the filter names, their representative wavelength, Stokes IQU measurements and variances. The columns corresponding to these measurements are specified in the extension header using the `TTYPEn` keywords.
- **Extension 2: best-fitting SED and fit information**. The binary FITS table includes the filter names, their representative wavelength, and the best fits for Stokes IQU. These columns are specified in the extension header using the `TTYPEn` keywords. Information about the fit, such as the ISM dust model, the free and fixed parameters included in the fit, parameter values and their starting guesses, chi2 values etc. are saved in the extension header.
- **Extension 3: model prediction for the dust emission spectra**. The model prediction for the per-grain-population and total emission spectra.
- **Extension 4: model prediction for the dust extinction curve.**