# eblur/dust Documentation

***Release 1.0***

**Lia Corrales**

October 05, 2016

Contents

The *eblur/dust* set of python modules calculate scattering absorption and scattering efficiencies for dust from the infrared to the X-ray. This code can also be used to calculate dust scattering halo images in the X-ray, in both interstellar and intergalactic (cosmological) contexts.

**First published version of this code** (released with Corrales & Paerels, 2015) http://dx.doi.org/10.5281/zenodo.15991

**Source code:** github.com/eblur/dust

**Support:** If you are having issues, please contact lia@space.mit.edu

CHAPTER 1

# Features

A number of dust grain size distributions and optical constants are provided, but they can be fully customized by the user by invoking custom objects of the approporiate class. Provided dust models include:

- A power law distribution of dust grain sizes
- Weingartner & Draine (2001) grain size distributions for Milky Way dust
- Optical constants (complex index of refraction) for 0.1 um sized graphite and astrosilicate grains
- Rayleigh-Gans scattering physics
  - Smith & Dwek (1998)
  - Mauche & Gorenstein (1986)
- Mie scattering physics using the algorithms of Bohren & Huffman (1986)
  - Converted from fortran and IDL to python

CHAPTER 2

# Installation

As of yet there is no static install version. I recommend cloning the github repo into a directory in your python path.:

```
cd /path/to/python/libraries/
git clone https://github.com/eblur/dust.git dust
```

Then be sure to add to your python path:

```
export PYTHONPATH=$PYTHONPATH:/path/to/python/libraries/dust/
```

CHAPTER 3

# Dependencies

I recommend using Anaconda to manage python distributions and packages

- Python 2.7 ()*eblur/dust* is not yet fully Python 3 compatible)
- numpy and scipy
- Astropy

CHAPTER 4

# Modules

## 4.1 astrodust.distlib

This module is for setting up dust grain populations.

### 4.1.1 sizedist

#### Classes

#### Functions

### 4.1.2 composition.cmindex

This module contains built-in complex indices of refraction typically used for astrophysical dust.

#### API for CmIndex classes

*class* **CmIndex** (object)

**ATTRIBUTES**
cmtype : string ('Drude', 'Graphite', or 'Silicate')
citation : A string containing citation to original source of optical constants
rp : either a function or scipy.interp1d object that can called,
e.g. rp(E) where E is in [keV]
ip : same as above, ip(E) where E is in [keV]

#### Class Library

#### Functions

### 4.1.3 composition.minerals

## 4.2 astrodust.extinction

This module contains libraries for calculating scattering and extinction properties of dust grains.

### 4.2.1 sigma_scat

This module combines the complex index of refraction (`cmindex`) and scattering algorithm (scatmodels) to calculate the total or differential cross-sections for a particular dust grain size distribution (`dust`).

#### Classes

The **ScatModel** object specifies what type of scattering physics will be used both through the complex index of refraction and the scattering physics algorithm.

The remaining classes contain cross-sections that are integrated over a particular dust grain size distribution.

#### Functions

The *makeScatmodel* function is a short-cut for creating a **Scatmodel** object based on two input strings (which become the *stype* and *cmtype*).

### 4.2.2 scatmodels

This module contains built-in algorithms for Mie scattering and the Rayleigh-Gans approximation.

#### API for Scattering classes

*class* **Scattering** (object)

**FUNCTIONS**
Qsca with inputs
( E : scalar or np.array [keV]
cm : cmtype object cmindex.py
a : scalar [grain size, micron] )
*returns* scalar or np.array [scattering efficiency, unitless]

Diff with inputs
( cm : cmtype object from cmindex.py
theta : scalar or np.array [angle, arcsec]
a : scalar [grain size, micron]
E : scalar or np.array [energy, keV]
** if len(E) > 1 and len(theta) > 1, then len(E) must equal len(theta)
*returns* dsigma/dOmega of values (E0,theta0), (E1,theta1) etc...

(optional)
Qext with inputs
( E : scalar or np.array [keV]
cm : cmtype object cmindex.py
a : scalar [grain size, micron] )
*returns* scalar or np.array [extinction efficiency, unitless]

Class Library

## 4.3 astrodust.halos

This module is for calculating X-ray surface brightness profiles for dust scattering halos from both Galactic and cosmological (z > 0) point sources.

### 4.3.1 Halo class

The **Halo** class is used for both Galactic and extragalactic sources and contains the attribute *htype*, which can be filled with one of two objects: either *CosmHalo* (for extragalactic sources) or *GalHalo* (for Galactic sources, see galhalo).

### 4.3.2 Halo types (*htype*)

*CosmHalo*

*GalHalo*

### 4.3.3 Sub-modules

#### galhalo

This module is for calculating X-ray scattering halos from Galactic (z=0) point sources.

One must import the **Halo** object from astrodust.halos. The **Halo** object is used for both Galactic and extragalactic sources and contains the attribute *htype*, which can be filled with one of two objects: either *CosmHalo* (for extragalactic sources) or *GalHalo* (for Galactic sources). See astrodust.halos for a description of these halo types.

##### Functions

Functions within this module take a **Halo** object as input and modifies it by updating the *htype*, *intensity*, and any other related attributes.

Intensities are calculated using numerical integration (trapezoidal method). For the semi-analytic solutions, see analytic.

#### cosmhalo

This module is for calculating X-ray scattering halos from Galactic (z=0) point sources.

One must import the **Halo** object from astrodust.halos. The **Halo** object is used for both Galactic and extragalactic sources and contains the attribute *htype*, which can be filled with one of two objects: either *CosmHalo* (for extragalactic sources) or *GalHalo* (for Galactic sources). See astrodust.halos for a description of these halo types.

##### Functions

Functions within this module take a **Halo** object as input and modifies it by updating the *htype*, *intensity*, and any other related attributes.

### cosmology

This module contains functions for calculating cosmological distances and other relevant values for the cosmhalo module.

#### Classes

#### Functions

### halodict

This module is for creating and manipulating **HaloDict** objects (a dictionary of **Halo** objects, see astrodust.halos). The keys of the **HaloDict** objects are energy values.

#### Classes

#### Functions

Some somewhat useful functions for reading and writing **HaloDict** objects

### analytic

This module contains semi-analytic functions for calculating X-ray scattering halo intensities from a power-law distribution of dust grain sizes (see Appendix of Corrales & Paerels, 2015).

When creating a **Halo** object, one must import from `halo`. The functions in this module are for Galactic sources only. (Note that galhalo contains functions for numerically integrated solutions).

#### Functions

Functions within this module modify the **Halo** object by updating the *htype* and *intensity*.

### model_halo

This module is for modeling scattering halos for Galactic sources (see galhalo) from a dust grain size distribution (astrodust.distlib) and apparent flux spectrum of a point source. It uses **HaloDict** objects from the halodict module.

#### Functions

The following two functions are helpers for integrating up the total intensity from a **HaloDict** object.

The following functions take inputs that will allow us to simulate a scattering halo intensity profile from a set of basic starting parameters.

CHAPTER 5

# Indices and tables

- genindex
- modindex
- search