xraydb

Release 4.3

Matthew Newville

2019-October-09

CONTENTS

1 Table of Contents **3**

1.1 Installation . . . 3
1.1.1 Development Version . . . 3
1.1.2 Testing . . . 3
1.1.3 Copyright, Licensing, and Re-distribution . . . 3
1.2 X-ray Periodic Table of the Elements . . . 4
1.3 Overview of Atomic and X-ray Data . . . 5
1.3.1 Elements . . . 5
1.3.2 Physical Units . . . 5
1.3.3 X-ray Edges . . . 5
1.3.4 X-ray Lines . . . 5
1.3.5 Cross Sections . . . 6
1.4 Using the XrayDB xraydb.sqlite . . . 6
1.4.1 Overall Database Schema . . . 6
1.4.2 Version Table . . . 8
1.4.3 Elements Table . . . 8
1.4.4 Xray_Levels Table . . . 8
1.4.5 Xray_Transitions Table . . . 9
1.4.6 Photoabsorption Table . . . 9
1.4.7 Scattering Table . . . 9
1.4.8 Coster_Kronig Table . . . 10
1.4.9 Waasmaier Table . . . 10
1.4.10 KeskiRahkonen_Krause Table . . . 10
1.4.11 Krause_Oliver Table . . . 11
1.4.12 Chantler Table . . . 11
1.5 Using XrayDB from Python . . . 11
1.5.1 The Python xraydb module . . . 12
1.5.2 `xraydb` functions . . . 13
1.6 References . . . 22

Bibliography **23**

Python Module Index **25**

Index **27**

XrayDB provides atomic data, characteristic X-ray energies, and X-ray cross sections for the elements in an SQLite3 database, `xraydb.sqlite`. This file can be used directly with SQLite [Hipp (2012)] using standard SQL, or from the many programming language that suppor SQLite. A Python module providing such an interface is provided here.

Because some of the components of the database hold arrays of numbers (for example, coefficients for interpolation), the arrays are stored in the database as JSON-encoded strings, and will need to be unpacked to be used.

The project began with the data from the compilation of basic atomic properties and X-ray absorption edge energies, emission energies, and absorption cross sections from [Elam, Ravel, and Sieber (2002)], who assembled data from a several sources. More data has been added from other sources. Energy widths of core holes for excited electronic levels from [Keski-Rahkonen and Krause (1974)] and [Krause and Oliver (1979)]. Elastic X-ray scattering data, $f_0(q)$ is taken from [Waasmaier and Kirfel (1995)]. Resonant scattering cross sections $f'(E)$ and $f''(E)$ and absorption cross sections from [Chantler (2000)] (as from the FFAST webpage) are also included.

In general, cross sections are in cm^2/gr, and energies are given in eV. Energy-dependent data for cross-sections are typically valid between about 250 eV to about 200,000 eV. Elements with Z=1 to 92 are supported, and some data is included for elements between Z=93 and Z=98.

- The current version of the XrayDB is 4, and the Version for the Python module is 4.3.
- See XrayDB Github Page for data sources, code, development, and issues.
- A PDF Version of this documentation is available.

CHAPTER

ONE

TABLE OF CONTENTS

1.1 Installation

The X-ray database is held in the SQLite3 file `xraydb.sqlite`. If you are looking for direct use with SQLite, you can download this from xraydb.sqlite.

If you want to use XrayDB from Python, install the XrayDB Python module (which includes the sqlite database), with:

```
pip install xraydb
```

If you are using Anaconda Python, you can also install with:

```
conda install -c gsecars xraydb
```

Depending on your system and Python installation, you may need administrative privileges to install any python library. For many linux and Mac OS X systems, you may need to use *sudo*.

Note: The Python module supports Python 3.5 and above.

1.1.1 Development Version

To work with the data sources or to add or modify data in the XrayDB, you will want to clone or download the full source code kit xrayDB on github.com which contains the current database, original source data, python module, and files for generating the Periodic Table posters. To get the latest development version, use:

```
git clone https://github.com/xraypy/XrayDB.git
```

1.1.2 Testing

There are a set of tests scripts for the Python interface that can be run with the pytest testing framework. These are located in the `python/tests` folder. These tests are automatically run as part of the development process. For any release or any master branch from the git repository, running `pytest` should run all of these tests to completion without errors or failures.

1.1.3 Copyright, Licensing, and Re-distribution

```
Public Domain

To the extent possible, the data, code, and documentation here are placed in
the public domain.  No claim of copyright is made by the authors and no
restrictions of any kind placed on the use of the work here.

The original sources of the data are mostly based on published works with
the clear intent of providing data to the general public.  Some of the
datasets have may not have clear statements of copyright or license, but
have been freely available for many years. The work here is a compilation
and reformatting of those datasets.

If you use these resources and wish to cite the original data source, please
see the documentation for more details.
```

1.2 X-ray Periodic Table of the Elements

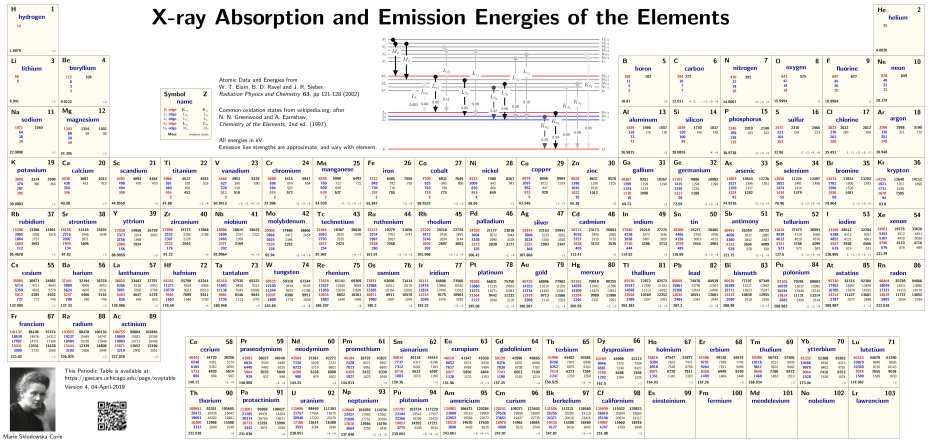
XrayDB has been used to generate X-ray Periodic Tables of the Elements.

There is a choice of two sizes, and a choice of an image of one of four prominent scientists associated with the Periodic Table and X-ray spectroscopies.

Table of X-ray Periodic Tables of the Elements The poster comes in two sizes: Large is 127.5x61 cm (about 50x24 inches) and Medium is 91x45.5 cm (about 36x17 inches).

Scientist	Large Periodic Table	Medium Periodic Table
Charles Barkla	Barkla (Large)	Barkla (Medium)
Marie Curie	Curie (Large)	Curie (Medium)
Dmitri Mendeleev	Mendeleev (Large)	Mendeleev (Medium)
Henry Moseley	Moseley (Large)	Moseley (Medium)

In general, these table look like this:



but you should definitely download the PDFs linked for high-quality results.

1.3 Overview of Atomic and X-ray Data

The data provided in XrayDB includes Atomic data and characteristic energies and cross sections for the interaction of X-rays with elements. A few definitions and conventions necessary for using this data are discussed here.

1.3.1 Elements

Most of the data resources are accessed by an elements *Atomic Symbol*. For the Python module, most methods will take *element* as the first argument, and this can either be the integer atomic number or the string for the atomic symbol.

1.3.2 Physical Units

Elemental densities are given in gr/cm^3, and molar masses are given in AMU. Unless otherwise stated, all energies are in units of eV.

1.3.3 X-ray Edges

Several resources (database tables, python methods) take either an `edge` or a `level` argument to signify a core electronic level. These are strings and must be one of the levels listed in the *Table of X-ray edge names*.

Table of X-ray Edges and Core electronic levels. The Names are the IUPAC symbols for the core electronic levels.

Name	electronic level	Name	electronic level
K	$1s$	N5	$4d_{5/2}$
L3	$2p_{3/2}$	N4	$4d_{3/2}$
L2	$2p_{1/2}$	N3	$4p_{3/2}$
L1	$2s$	N2	$4p_{1/2}$
M5	$3d_{5/2}$	N1	$4s$
M4	$3d_{3/2}$	O3	$5p_{3/2}$
M3	$3p_{3/2}$	O2	$5p_{1/2}$
M2	$3p_{1/2}$	O1	$5s$
M1	$3s$	P3	$6p_{3/2}$
N7	$4f_{7/2}$	P2	$6p_{1/2}$
N6	$4f_{5/2}$	P1	$6s$

1.3.4 X-ray Lines

Many resources (database tables or methods) take emission line arguments. These are all strings and follow the latinized version of the Siegbahn notation as indicated in the *Table of X-ray emission line names*.

Table of X-ray emission line names and the corresponding Siegbahn and IUPAC notations

Name	IUPAC	Siegbahn	Name	IUPAC	Siegbahn
Ka1	K-L3	$K\alpha_1$	Lb4	L1-M2	$L\beta_4$
Ka2	K-L2	$K\alpha_2$	Lb5	L3-O4,5	$L\beta_5$
Ka3	K-L1	$K\alpha_3$	Lb6	L3-N1	$L\beta_6$
Kb1	K-M3	$K\beta_1$	Lg1	L2-N4	$L\gamma_1$
Kb2	K-N2,3	$K\beta_2$	Lg2	L1-N2	$L\gamma_2$
Kb3	K-M2	$K\beta_3$	Lg3	L1-N3	$L\gamma_3$
Kb4	K-N4,5	$K\beta_2$	Lg6	L2-O4	$L\gamma_6$
Kb5	K-M4,5	$K\beta_3$	Ll	L3-M1	Ll
La1	L3-M5	$L\alpha_1$	Ln	L2-M1	$L\nu$
La2	L3-M4	$L\alpha_1$	Ma	M5-N6,7	$M\alpha$
Lb1	L2-M4	$L\beta_1$	Mb	M4-N6	$M\beta$
Lb2,15	L3-N4,5	$L\beta_2, L\beta_{15}$	Mg	M3-N5	$M\gamma$
Lb3	L1-M3	$L\beta_3$	Mz	M4,5-N6,7	$M\zeta$

1.3.5 Cross Sections

The photo-absorption and scattering cross sections from [Elam, Ravel, and Sieber (2002)] and [Chantler (2000)] are in cm^2/gr.

The data from [Elam, Ravel, and Sieber (2002)] is held as logarithms of energy, cross section, and logarithm of the 2nd derivative of cross section that allows for cubic spline interpolation in log-log space.

1.4 Using the XrayDB xraydb.sqlite

All the data for the X-ray database is held in the SQLite3 file `xraydb.sqlite`. To use with SQLite, this file is all you need. While many programs and languages can access SQLite files, basic usage with the `sqlite3` program (available from Windows, Mac OS X, and Linux) can be as simple as:

```
system~> sqlite3 xraydb.sqlite
sqlite> .headers on
sqlite> select * from elements where atomic_number=47;
atomic_number|element|molar_mass|density
47|Ag|107.868|10.48
```

That is, you can retrieve the data using standard SQL queries built-in to SQLite. Of course, the expectation is that you'd want to use this database within a programming environment. Currently, wrappers exist only for Python.

1.4.1 Overall Database Schema

The schema for the SQLite3 database describes where data is held in the database, and how to access it. The schema for the current version (4) looks like this:

```
Table Version (id integer primary key,
               tag text,
               date text,
               notes text);
Table elements (atomic_number integer primary key,
                element text,
                molar_mass real,
```

(continues on next page)

(continued from previous page)

```
                density real);
Table xray_levels (id integer primary key,
                   element text,
                   iupac_symbol text,
                   absorption_edge real,
                   fluorescence_yield real,
                   jump_ratio real);
Table xray_transitions (id integer primary key,
                        element text,
                        iupac_symbol text,
                        siegbahn_symbol text,
                        initial_level text,
                        final_level text,
                        emission_energy real,
                        intensity real);
Table Coster_Kronig (id integer primary key,
                     element text,
                     initial_level text,
                     final_level text,
                     transition_probability real,
                     total_transition_probability real);
Table photoabsorption (id integer primary key,
                       element text,
                       log_energy text,
                       log_photoabsorption text,
                       log_photoabsorption_spline text);
Table scattering (id integer primary key,
                  element text,
                  log_energy text,
                  log_coherent_scatter text,
                  log_coherent_scatter_spline text,
                  log_incoherent_scatter text,
                  log_incoherent_scatter_spline text);
Table Waasmaier (id integer primary key,
                 atomic_number integer,
                 element text,
                 ion text,
                 offset real,
                 scale text,
                 exponents text);
Table KeskiRahkonen_Krause (id integer primary key,
                            atomic_number integer,
                            element text,
                            edge text,
                            width float);
Table Krause_Oliver (id integer primary key,
                     atomic_number integer,
                     element text,
                     edge text,
                     width float);
Table corelevel_widths (id integer primary key,
                        atomic_number integer,
                        element text,
                        edge text,
                        width float);
Table Chantler (id integer primary key,
                element text,
```

(continues on next page)

(continued from previous page)

```
                sigma_mu real,
                mue_f2 real,
                density real,
                corr_henke float,
                corr_cl35 float,
                corr_nucl float,
                energy text,
                f1 text,
                f2 text,
                mu_photo text,
                mu_incoh text,
                mu_total text);
```

More details for each table are given below.

Note: in the tables below the type of *json array* means that arrays of numerical data are stored in the database as text of JSON-encoded arrays.

1.4.2 Version Table

The *Version* table holds data about the revisions to the database file itself. Each row represents a single revision.

DB Table of Database Versions

Column	Type	Description
id	integer	counter (primary tag)
tag	text	version name
date	text	date string
notes	text	notes on changes for version

1.4.3 Elements Table

The *elements* table holds basic data about each element. Each row represents an element.

DB Table of Basic Properties of the Elements

Column	Type	Description
atomic_number	integer	Atomic Number, Z
element	text	Atomic symbol
molar_mass	float	Atomic mass in AMU
density	float	Density of pure element (gr/cm^3)

1.4.4 Xray_Levels Table

The *xray_levels* table holds data for electronic levels of atoms. Each row represents a core electronic level.

DB Table of X-ray and core electronic levels. *fluorescence yield* gives the probability of an empty level refilling by X-ray fluorescence. The *jump ratio* is the ratio of values for photo-electric cross section (that is, from *Photoabsorption Table*) 1 eV above the absorption edge to that 1 eV below the absorption edge. See *Table of X-ray Edges*

Column	Type	Description
id	integer	Index (primary key)
element	text	Atomic symbol for element
iupac_symbol	text	IUPAC symbol for level ('K','L3',...)
absorption_edge	float	binding energy for level (eV)
fluorescence_yield	float	fluorescence yield (fraction)
jump_ratio	float	ratio of mu_photo across edge

1.4.5 Xray_Transitions Table

The *xray_transitions* table holds data for transitions between electronic levels of atoms. Each row represents a transition between two levels.

DB Table of X-ray Transitions. Both IUPAC and Siegbahn symbols are given (see *Table of X-ray emission lines*), as well as the initial and final levels. The *intensity* is the relative intensity of the transition for a given *initial level*.

Column	Type	Description
id	integer	Index (primary key)
element	text	Atomic symbol for element
iupac_symbol	text	IUPAC symbol for transition
siegbahn_symbol	text	Siegbahn symbol for transition
initial_level	text	IUPAC symbol for initial level
final_level	text	IUPAC symbol for final level
emission_energy	float	fluorescence energy (eV)
intensity	float	relative intensity for transition

1.4.6 Photoabsorption Table

The *photoabsorption* table holds data for the photo-electric absorption cross sections in cm^2/gr. Each row represents an element.

DB Table of Photoabsorption Cross Sections. JSON-encoded arrays are held for logs of energy, cross section, and cross section spline (second derivative useful for spline interpolation).

Column	Type	Description
id	integer	Index (primary key)
element	text	Atomic symbol for element
log_energy	json array	log of Energy values (eV)
log_photoabsorption	json array	log of cross section (cm^2/gr)
log_photoabsorption_spline	json array	log of cross section spline

1.4.7 Scattering Table

The *scattering* table holds data for the coherent and incoherent X-ray scattering cross sections, in cm^2/gr. Each row represents an element.

DB Table of Coherent and Incoherent Scattering Cross Sections. JSON-encoded arrays are held for logs of energy, cross section, and cross section spline (second derivative useful for spline interpolation).

Column	Type	Description
id	integer	Index (primary key)
element	text	Atomic symbol for element
log_energy	json array	log of Energy values (eV)
log_coherent_scatter	json array	log of cross section (cm^2/gr)
log_coherent_scatter_spline	json array	log of cross section spline
log_incoherent_scatter	json array	log of cross section (cm^2/gr)
log_incoherent_scatter_spline	json array	log of cross section spline

1.4.8 Coster_Kronig Table

The *Coster_Kronig* table holds data for energy levels, partial and total transition probabilities for the Coster-Kronig transitions (Auger processes in which the empty core level is filled from an electron in a higher level with the same principle quantum number). The partial probability describes direct transitions, while the total probability includes cascade effects. Each row represents a transition.

DB Table of Coster-Kronig Transitions.

Column	Type	Description
id	integer	Index (primary key)
element	text	Atomic symbol for element
initial_level	text	IUPAC symbol for initial level
final_level	text	IUPAC symbol for final level
transition_probability	float	direct transition probability
total_transition_probability	float	total transition probability

1.4.9 Waasmaier Table

The *Waasmaier* table holds data for calculating elastic X-ray scattering factors $f_0(k)$, from [Waasmaier and Kirfel (1995)]. The scattering factor is unitless, and $k = \sin(\theta)/\lambda$ where θ is the scattering angle and λ is the X-ray wavelength. available for many common ionic states for each element. Each row represents an ion.

DB Table of Elastic Scattering Cross Section Coefficients

Column	Type	Description
id	integer	Index (primary key)
atomic_number	integer	Atomic Number, Z
element	text	Atomic symbol for element
ion	text	symbol for element and ionization
offset	float	offset value
scale	json array	coefficients for calculation
exponents	json array	coefficients for calculation

1.4.10 KeskiRahkonen_Krause Table

The *KeskiRahkonen_Krause* table holds data for energy widths of the core electronic levels from [Keski-Rahkonen and Krause (1974)]. Values are in eV, and each row represents an energy level for an element.

DB Table of Core Hole Widths from Keski-Rahkonen and Krause

Column	Type	Description
id	integer	Index (primary key)
atomic_number	integer	Atomic Number, Z
element	text	Atomic symbol for element
edge	text	IUPAC symbol for energy level ('K')
width	float	width of level (eV)

1.4.11 Krause_Oliver Table

The *Krause_Oliver* table holds data for energy widths of the core electronic levels from [Krause and Oliver (1979)]. Values are in eV, and each row represents an energy level for an element.

DB Table of Core Hole Widths from Krause and Oliver

Column	Type	Description
id	integer	Index (primary key)
atomic_number	integer	Atomic Number, Z
element	text	Atomic symbol for element
edge	text	IUPAC symbol for energy level ('K')
width	float	width of level (eV)

1.4.12 Chantler Table

The *Chantler* table holds data for resonant X-ray scattering factors $f'(E)$ and $f''(E)$ as well as photo-electric absorption, coherent, and incoherent scattering factors from [Chantler (2000)]. As with other tables, scattering factors are unitless, and cross sections are in cm^2/gr. Each row represents an element.

DB Table of resonant scattering and mass attenuation coefficients from Chantler.

Column	Type	Description
id	integer	Index (primary key)
element	text	Atomic symbol for element
mue_f2	float	factor to convert mu(E) to f''(E)
density	float	atomic density (gr/cm^3)
corr_henke	float	Henke correction to f'(E)
corr_cl35	float	Cromer-Liberman correction to f'(E)
corr_nucl	float	nuclear correction to f'(E)
energy	json array	energies for interpolation
f1	json array	f'(E) (e)
f2	json array	f''(E) (e)
mu_photo	json array	photoabsorption mu(E) (cm^2/gr)
mu_incoh	json array	incoherent scattering (cm^2/gr)
mu_total	json array	total attenuation (cm^2/gr)

1.5 Using XrayDB from Python

The *python* directory contains the source code for a Python module for XrayDB. This module gives a user-friendly wrapping of the XrayDB, and automates the the conversion of data from sqlite database into Python and numpy arrays.

The module requires the *numpy*, *scipy* and *sqlalchemy* modules, all of which are readily available and can be installed with:

```
pip install xraydb
```

The current version of the Python module is 4.3, corresponding to version 4 of *xraydb.sqlite*.

1.5.1 The Python xraydb module

To use the XrayDB from Python, you can import the *xraydb* module and start using it:

```
>>> import xraydb
>>> xraydb.atomic_number('Ag')
47
#
# X-ray elastic (Thomson) scattering factors:
>>> import numpy as np
>>> q =np.linspace(0, 5, 11)
>>> xraydb.f0('Fe', q)
array([25.994603  , 11.50848469,  6.55945765,  4.71039413,  3.21048827,
       2.20939146,  1.65112769,  1.36705887,  1.21133507,  1.10155689,
       1.0035555 ])
#
# X-ray emission lines:
>>> for name, line in xraydb.xray_lines('Zn', 'K').items():
...     print(name, ' = ', line)
...
Ka3  =  XrayLine(energy=8462.8, intensity=0.000316256, initial_level='K', final_level=
↪'L1')
Ka2  =  XrayLine(energy=8614.1, intensity=0.294353, initial_level='K', final_level='L2
↪')
Ka1  =  XrayLine(energy=8637.2, intensity=0.576058, initial_level='K', final_level='L3
↪')
Kb3  =  XrayLine(energy=9567.6, intensity=0.0438347, initial_level='K', final_level=
↪'M2')
Kb1  =  XrayLine(energy=9570.4, intensity=0.0846229, initial_level='K', final_level=
↪'M3')
Kb5  =  XrayLine(energy=9648.8, intensity=0.000815698, initial_level='K', final_level=
↪'M4,5')
#
# X-ray absorption edges:
>>> xraydb.xray_edge('As', 'K')
XrayEdge(energy=11867.0, fyield=0.548989, jump_ratio=7.314)
#
# X-ray attenuation factors:
>>> as_kedge = xraydb.xray_edge('As', 'K').energy
>>> energies = np.linspace(-50, 50, 5) + as_kedge
>>> muvals = xraydb.mu_elam('As', energies)
>>> for en, mu in zip(energies, muvals):
...     print("{:.0f}   {:8.2f}".format(en, mu))
...
11817      26.07
11842      25.92
11867      25.77
11892     178.32
11917     177.38
```

Table of XrayDB function for Atomic and X-ray data for the elements

Most of these function return some element-specific property from the element symbol or atomic number. Some of the data extends to Z=98 (Cf), but some data may not be available for Z > 92 (U). Except where noted, the data comes from [Elam, Ravel, and Sieber (2002)].

xraydb functions	description
atomic_number()	atomic number from symbol
atomic_symbol()	atomic symbol from number
atomic_mass()	atomic mass
atomic_density()	density of pure element
f0()	elastic scattering factor ([Waasmaier and Kirfel (1995)])
f0_ions()	list of valid "ions" for *f0()* ([Waasmaier and Kirfel (1995)])
xray_edge()	xray edge data for a particular element and edge
xray_edges()	dictionary of all X-ray edges data for an element
xray_lines()	dictionary of all X-ray emission line data for an element
fluor_yield()	fluorescent yield for an X-ray emission line
ck_probability()	Coster-Kronig transition probability between two atomic levels
mu_elam()	absorption cross-section, photo-electric or total for an element
coherent_cross_section_elam()	coherent scattering cross-section for an element
incoherent_cross_section_elam()	incoherent scattering cross-section for an element
chantler_energies()	energies of tabulation for Chantler data ([Chantler (2000)])
f1_chantler()	$f'(E)$ anomalous scattering factor ([Chantler (2000)])
f2_chantler()	$f"(E)$ anomalous scattering factor ([Chantler (2000)])
mu_chantler()	absorption cross-section ([Chantler (2000)])
guess_edge()	guess element and edge from energy of absorption edge
chemparse()	parse a chemical formula to atomic abundances
material_mu()	absorption cross-section for a material at X-ray energies
material_mu_components()	dictionary of elemental components of *mu* for material
get_material()	get a material (name, formula, density from materials database
add_material()	add a material (name, formula, density) to local materials database
xray_delta_beta()	return anomalous index of refraction for material and energy

1.5.2 `xraydb` functions

Atomic Properties

atomic_number(*element*)
z for element name

Parameters **element** (*str*) – atomic symbol

Returns atomic number

atomic_symbol(*z*)
atomic symbol for atomic number

Parameters **z** (*int*) – atomic number

Returns atomic symbol

atomic_mass(*element*)
molar mass for an element

Parameters **element** (*int, str*) – atomic number, atomic symbol for element

Returns atomic mass, in AMU

atomic_density(*element*)
density (gr/cm^3) for common for of an element

Parameters **element** (*int, str*) – atomic number, atomic symbol for element

Returns density in gm/cm^3

Elastic Scattering Factors

f0(*ion*, *q*)
elastic X-ray scattering factor, f0(q), for an ion.

Parameters

- **ion** (*int or str*) – atomic number, atomic symbol or ionic symbol of scatterer
- **q** (*float, ndarray*) – Q value(s) for scattering

Returns scattering factor for each Q value

Notes

1. from D. Waasmaier and A. Kirfel, Acta Cryst. A51 p416 (1995) and International Tables for Crystallography, Vol. C.
2. *ion* can be of the form: 26, *Fe*, *Fe2+*. For a full list of ions use *f0_ions()*
3. elements supported are from Z = 1 to 98 ('H' to 'Cf')
4. q = sin(theta) / lambda, where theta=incident angle, lambda=X-ray wavelength

f0_ions(*element=None*)
list ion names supported in the f0() calculation from Waasmaier and Kirfel.

Parameters **element** (*None, int, str*) – scatterer

Returns list of strings for matching ion names

Notes

if element is None, all 211 ions are returned.

X-ray Edges

xray_edge(*element*, *edge*, *energy_only=False*)
get x-ray absorption edge data for an element: (energy(in eV), fluorescence yield, jump ratio)

Parameters

- **element** (*int, str*) – atomic number, atomic symbol for element
- **edge** (*str*) – iupac symbol of X-ray edge
- **energy_only** (*bool*) – whether to return only the energy [False]

Returns XrayEdge namedtuple containing (energy, fluorescence_yield, edge_jump) or float of energy

xray_edges(*element*)

get dictionary of x-ray absorption edges: energy(in eV), fluorescence yield, and jump ratio for an element.

Parameters **element** (*int, str*) – atomic number, atomic symbol for element

Returns dictionary of XrayEdge named tuples.

Notes

1. The dictionary will have keys of edge (iupac symbol) and values containing an XrayEdge namedtuple containing (energy, fluorescence_yield, edge_jump)

core_width(*element*, *edge=None*)
returns core hole width for an element and edge

Parameters

- **element** (*int or str*) – element
- **edge** (*None or str*) – edge to consider

Returns core width or list of core widths

Notes

1. if edge is None, values are return for all edges
2. Data from Krause and Oliver (1979) and Keski-Rahkonen and Krause (1974)

guess_edge(*energy, edges=['K', 'L3', 'L2', 'L1', 'M5']*)
guess an element and edge based on energy (in eV)

Parameters

- **energy** (*float*) – approximate edge energy (in eV)
- **edges** (*None or list of strings*) – edges to consider

Returns a tuple of (atomic symbol, edge) for best guess

Notes

by default, the list of edges is ['K', 'L3', 'L2', 'L1', 'M5']

X-ray Emission Lines

xray_lines(*element*, *initial_level=None*, *excitation_energy=None*)
get dictionary of X-ray emission lines of an element

Parameters

- **element** (*int, str*) – atomic number, atomic symbol for element
- **initial_level** (*None or str*) – iupac symbol of initial level
- **excitation_energy** (*None or float*) – exciation energy

Returns dict of X-ray lines with keys of siegbahn notation and values of XrayLine tuples of (energy, intensity, initial level, final level)

Notes

1. excitation energy will supercede initial_level, as it means 'all intial levels with below this energy

Exaample:

```
>>> for name, line in xraydb.xray_lines('Mn', 'K').items():
...     print(name, line)
...
Ka3 XrayLine(energy=5769.9, intensity=0.000265963, initial_level='K', final_
↪level='L1')
Ka2 XrayLine(energy=5889.1, intensity=0.293941, initial_level='K', final_
↪level='L2')
Ka1 XrayLine(energy=5900.3, intensity=0.58134, initial_level='K', final_level=
↪'L3')
Kb3 XrayLine(energy=6491.8, intensity=0.042234, initial_level='K', final_
↪level='M2')
Kb1 XrayLine(energy=6491.8, intensity=0.0815329, initial_level='K', final_
↪level='M3')
Kb5 XrayLine(energy=6537.0, intensity=0.000685981, initial_level='K', final_
↪level='M4,5')
```

fluor_yield(*element*, *edge*, *line*, *energy*)
fluorescence yield for an X-ray emission line or family of lines.

Parameters

- **element** (*int, str*) – atomic number, atomic symbol for element
- **edge** (*str*) – iupac symbol of X-ray edge
- **line** (*str*) – siegbahn notation for emission line
- **energy** (*float*) – incident X-ray energy

Returns fluorescence yield, weighted average fluorescence energy, net_probability

Examples

```
>>> xraydb.fluor_yield('Fe', 'K', 'Ka', 8000)
0.350985, 6400.752419799043, 0.874576096
```

```
>>> xraydb.fluor_yield('Fe', 'K', 'Ka', 6800)
0.0, 6400.752419799043, 0.874576096
```

```
>>> xraydb.fluor_yield('Ag', 'L3', 'La', 6000)
0.052, 2982.129655446868, 0.8618990000000000
```

See also:

xray_lines which gives the full set of emission lines ('Ka1', 'Kb3', etc) and probabilities for each of these.

ck_probability(*element*, *initial*, *final*, *total=True*)
transition probability for an element, initial, and final levels.

Parameters

- **element** (*int, str*) – atomic number, atomic symbol for element
- **initial** (*str*) – iupac symbol for initial level
- **final** (*str*) – iupac symbol for final level
- **total** (*bool*) – whether to include transitions via possible intermediate levels [True]

Returns transition probability, or 0 if transition is not allowed.

Absorption and Scattering Cross-sections

mu_elam(*element*, *energy*, *kind='total'*)
X-ray mass attenuation coefficient, mu/rho, for an element and energy or array of energies. Data is from the Elam tables.

Parameters

- **element** (*int, str*) – atomic number, atomic symbol for element
- **energy** (*float or ndarray*) – energy or array of energies
- **kind** (*str*) – type of cross-section to use, one of ('total', 'photo', 'coh', 'incoh') ['total']

Returns float value or ndarray

Notes

1. Values returned are in units of cm^2/gr
2. The default is to return total attenuation coefficient.

coherent_cross_section_elam(*element*, *energy*)
coherent scaattering cross-section for an element and energy or array of energies. Data is from the Elam tables.

Parameters

- **element** (*int, str*) – atomic number, atomic symbol for element
- **energy** (*float or ndarray*) – energy or array of energies

Returns float value or ndarray

Notes

1. Values returned are in units of cm^2/gr

incoherent_cross_section_elam(*element*, *energy*)
incoherent scaattering cross-section for an element and energy or array of energies. Data is from the Elam tables.

Parameters

- **element** (*int, str*) – atomic number, atomic symbol for element
- **energy** (*float or ndarray*) – energy or array of energies

Returns float value or ndarray

Notes

1. Values returned are in units of cm^2/gr

chantler_energies(*element*, *emin=0*, *emax=1000000000.0*)
energies at which Chantler data is tabulated for a particular element.

Parameters

- **element** (*int, str*) – atomic number, atomic symbol for element
- **emin** (*float*) – lower bound of energies (default=0)
- **emax** (*float*) – upper bound of energies (default=1.e9)

Returns ndarray of energies

Notes

energies are in eV

f1_chantler(*element*, *energy*, ***kws*)
real part of anomalous x-ray scattering factor for an element and energy or array of energies. Data is from the Chantler tables.

Parameters

- **element** (*int, str*) – atomic number, atomic symbol for element
- **energy** (*float or ndarray*) – energy or array of energies

Returns float value or ndarray

Notes

1. Values returned are in units of electrons

f2_chantler(*element*, *energy*)
imaginary part of anomalous x-ray scattering factor for an element and energy or array of energies. Data is from the Chantler tables.

Parameters

- **element** (*int, str*) – atomic number, atomic symbol for element
- **energy** (*float or ndarray*) – energy or array of energies

Returns float value or ndarray

Notes

1. Values returned are in units of electrons

mu_chantler (*element*, *energy*, *incoh=False*, *photo=False*)

X-ray mass attenuation coeficient, mu/rho, for an element and energy or array of energies. Data is from the Chantler tables.

Parameters

- **element** (*int, str*) – atomic number, atomic symbol for element
- **energy** (*float or ndarray*) – energy or array of energies
- **incoh** (*bool*) – whether to return only the incoherent contribution [False]
- **photo** (*bool*) – whether to return only the photo-electric contribution [False]

Returns float value or ndarray

Notes

1. Values returned are in units of cm^2/gr
2. The default is to return total attenuation coefficient.

Chemical and Materials calculations

xray_delta_beta (*material*, *density*, *energy*, *photo_only=False*)

anomalous components of the index of refraction for a material, using the tabulated scattering components from Chantler.

Parameters

- **material** – chemical formula ('Fe2O3', 'CaMg(CO3)2', 'La1.9Sr0.1CuO4')
- **density** – material density in g/cm^3
- **energy** – x-ray energy in eV
- **photo_only** – boolean for returning photo cross-section component only if False (default), the total cross-section is returned

Returns (delta, beta, atlen)

where delta : real part of index of refraction beta : imag part of index of refraction atlen : attenuation length in cm

These are the anomalous scattering components of the index of refraction:

n = 1 - delta - i*beta = 1 - lambda**2 * r0/(2*pi) Sum_j (n_j * fj)

Adapted from code by Yong Choi

chemparse (*formula*)
parse a chemical formula to a dictionary of elemental abundances

Parameters **formula** (*str*) – chemical formula

Returns dict of element symbol and abundance.

Examples

```
>>> from xraydb import chemparse
>>> chemparse('Mn(SO4)2(H2O)7)')
{'H': 14.0, 'S': 2.0, 'Mn': 1, 'O': 15.0}
```

```
>>> chemparse('Zn1.e-5Fe3O4')
{'Zn': 1e-05, 'Fe': 3.0, 'O': 4.0}
```

```
>>> chemparse('CO')
{'C': 1, 'O': 1}
>>> chemparse('Co')
{'Co': 1}
```

```
>>> chemparse('co')
ValueError: unrecognized element or number:
co
```

material_mu (*name*, *energy*, *density=None*, *kind='total'*)
X-ray attenuation length (in 1/cm) for a material by name or formula

Parameters

- **name** (*str*) – chemical formul or name of material from materials list.
- **energy** (*float or ndarray*) – energy or array of energies in eV
- **density** (*None or float*) – material density (gr/cm^3).
- **kind** (*str*) – 'photo' or 'total'for whether to return photo-absorption or total cross-section ['total']

Returns absorption length in 1/cm

Notes

1. material names are not case sensitive, chemical compounds are case sensitive.
2. mu_elam() is used for mu calculation.
3. if density is None and material is known, that density will be used.

Examples

```
>>> material_mu('H2O', 10000.0)
5.32986401658495
```

material_mu_components (*name*, *energy*, *density=None*, *kind='total'*)
material_mu_components: absorption coefficient (in 1/cm) for a compound

Parameters

- **name** (*str*) – chemical formul or name of material from materials list.
- **energy** (*float or ndarray*) – energy or array of energies in eV
- **density** (*None or float*) – material density (gr/cm^3).
- **kind** (*str*) – 'photo' or 'total'for whether to return photo-absorption or total cross-section ['total']

Returns

dict for constructing mu per element, with elements 'mass' (total mass), 'density', and

'elements' (list of atomic symbols for elements in material). For each element, there will be an item (atomic symbol as key) with tuple of (stoichiometric fraction, atomic mass, mu)

Examples

```
>>> xraydb.material_mu('quartz', 10000)
50.36774553547068
>>> xraydb.material_mu_components('quartz', 10000)
{'mass': 60.0843, 'density': 2.65, 'elements': ['Si', 'O'],
'Si': (1, 28.0855, 33.87943243018506), 'O': (2.0, 15.9994, 5.952824815297084)}
```

get_material (*name*)
look up material name

Parameters **name** (*str*) – name of material

Returns chemical formula, denisty of material

Examples

```
>>> xraydb.get_material('kapton')
('C22H10N2O5', 1.43)
```

add_material (*name*, *formula*, *density*)
add a material to the users local material database

Parameters

- **name** (*str*) – name of material
- **formula** (*str*) – chemical formula
- **(float** (*density*) – density

Returns None

Notes

the data will be saved to $HOME/.config/xraydb/materials.dat in the users home directory, and wiill be useful in subsequent sessions.

Examples

```
>>> xraydb.add_material('becopper', 'Cu0.98e0.02', 8.3)
```

get_xraydb()
return xraydb

1.6 References

BIBLIOGRAPHY

[Waasmaier and Kirfel (1995)] D. Waasmaier and A. Kirfel. New analytical scattering factor functions for free atoms and ions. *Acta Crystallographica A*, 51:416–431, 1995. URL: https://dx.doi.org/10.1107/S0108767394013292.

[Keski-Rahkonen and Krause (1974)] O. Keski-Rahkonen and M. O. Krause. Total and partial atomic-level widths. *Atomic Data and Nuclear Data Tables*, 14(2):139–146, 1974. URL: https://dx.doi.org/10.1016/S0092-640X(74)80020-3.

[Krause and Oliver (1979)] M. O. Krause and J. H. Oliver. Natural widths of atomic k and l levels, ka x-ray lines and several kll auger lines. *Journal of Physical and Chemical Reference Data*, 8:329, 1979. URL: https://dx.doi.org/10.1063/1.555595.

[Chantler (2000)] C. T. Chantler. Detailed tabulation of atomic form factors, photoelectric absorption and scattering cross section, and mass attenuation coefficients in the vicinity of absorption edges in the soft X-ray (Z=30-36, Z=60-89, E=0.1 keV-10 keV), addressing convergence issues of earlier work. *Journal of Physical and Chemical Reference Data*, 29(4):597–1048, JUL-AUG 2000. URL: https://dx.doi.org/10.1063/1.1321055.

[Hipp (2012)] D. R. Hipp. Sqlite. 2012. URL: https://www.sqlite.org.

[Elam, Ravel, and Sieber (2002)] W. T. Elam, B. D. Ravel, and J. R. Sieber. A new atomic database for x-ray spectroscopic calculations. *Radiation Physics and Chemistry*, 63(2):121–128, February 2002. URL: https://dx.doi.org/10.1016/S0969-806X(01)00227-4.

PYTHON MODULE INDEX

X

xraydb, 12

INDEX

A

add_material() (*in module xraydb*), 21
atomic_density() (*in module xraydb*), 14
atomic_mass() (*in module xraydb*), 14
atomic_number() (*in module xraydb*), 14
atomic_symbol() (*in module xraydb*), 14

C

chantler_energies() (*in module xraydb*), 18
chemparse() (*in module xraydb*), 19
ck_probability() (*in module xraydb*), 17
coherent_cross_section_elam() (*in module xraydb*), 17
core_width() (*in module xraydb*), 15

D

DB Table of Basic Properties of the Elements, 8
DB Table of Coherent and Incoherent Scattering Cross Sections, 9
DB Table of Core Hole Widths, 10, 11
DB Table of Coster-Kronig Transitions, 10
DB Table of Database Versions, 8
DB Table of Elastic Scattering Cross Section Coefficients, 10
DB Table of Photoabsorption Cross Sections, 9
DB Table of resonant scattering and mass attenuation coefficients from Chantler, 11
DB Table of X-ray Levels, 8
DB Table of X-ray Transitions, 9

F

f0() (*in module xraydb*), 14
f0_ions() (*in module xraydb*), 14
f1_chantler() (*in module xraydb*), 18
f2_chantler() (*in module xraydb*), 18
fluor_yield() (*in module xraydb*), 16

G

get_material() (*in module xraydb*), 21
get_xraydb() (*in module xraydb*), 22
guess_edge() (*in module xraydb*), 15

I

incoherent_cross_section_elam() (*in module xraydb*), 18

M

material_mu() (*in module xraydb*), 20
material_mu_components() (*in module xraydb*), 20
mu_chantler() (*in module xraydb*), 19
mu_elam() (*in module xraydb*), 17

T

Table of X-ray Edges, 5
Table of X-ray emission lines, 5

X

X-ray Periodic Tables, 4
xray_delta_beta() (*in module xraydb*), 19
xray_edge() (*in module xraydb*), 15
xray_edges() (*in module xraydb*), 15
xray_lines() (*in module xraydb*), 16
xraydb (*module*), 12
xraydb Python module, 12