mendeleev Documentation

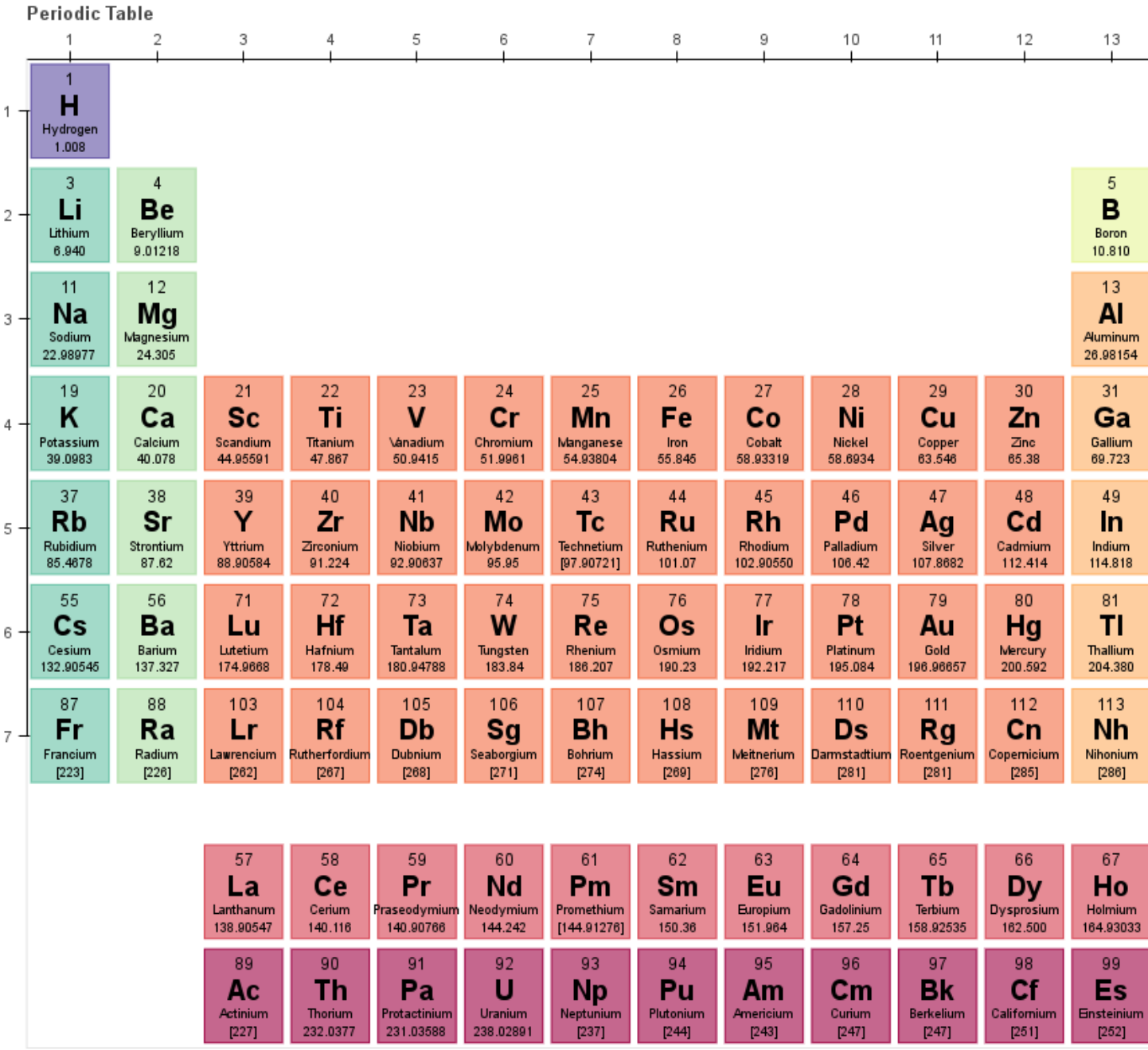
Release 0.4.3

Lukasz Mentel

Jul 16, 2018

Table of Contents

This package provides an API for accessing various properties of elements from the periodic table of elements.



CHAPTER 1

Getting started

1.1 Overview

This package provides an API for accessing various properties of elements from the periodic table of elements.

The repository is hosted on bitbucket.

1.2 Contributing

All contributions are welcome!

If you would like to suggest an improvement or report a bug or data inconsistency please consider creating an issue on bitbucket. I would be especially grateful for references to possible data updates and sources and recommendations of new data.

1.3 Citing

If you use *mendeleev* in a scientific publication, please consider citing the software as

L. M. Mentel, *mendeleev* - A Python resource for properties of chemical elements, ions and isotopes. , 2014– . Available at: https://bitbucket.org/lukaszmentel/mendeleev.

Here's the reference in the BibLaTeX format

```
@software{mendeleev2014,
   author = {Mentel, Łukasz},
   title = {{mendeleev} -- A Python resource for properties of chemical elements,␣
↪ions and isotopes},
```

(continues on next page)

(continued from previous page)

```
    url = {https://bitbucket.org/lukaszmentel/mendeleev},
    version = {0.3.6},
    date = {2014--},
}
```

or the older BibTeX format

```
@misc{mendeleev2014,
   auhor = {Mentel, Łukasz},
   title = {mendeleev} -- A Python resource for properties of chemical elements, ions␣
↪and isotopes, ver. 0.3.6},
   howpublished = {\url{https://bitbucket.org/lukaszmentel/mendeleev}},
   year  = {2014--},
}
```

1.4 Related projects

periodictable This package provides a periodic table of the elements with support for mass, density and xray/neutron scattering information.

periodic Periodic is an open source simple python API/command line script for the periodic table.

1.5 Funding

This project is supported by the RCN (The Research Council of Norway) project number 239193.

CHAPTER 2

Installation

The package can be installed using pip

```
pip install mendeleev
```

You can also install the most recent version from the repository:

```
pip install https://bitbucket.org/lukaszmentel/mendeleev/get/tip.tar.gz
```

If you use conda you can install the package from my anaconda channel by

```
conda install -c lmmentel mendeleev=0.4.3
```

CHAPTER 3

Tutorials

3.1 mendeleev tutorial

This simple tutorial will illustrate the basic capabilities of the package.

3.1.1 Basic interactive usage

Getting single elements

The simplest way of accessing the elements is importing them directly from `mendeleev` by symbols

```
In [6]: from mendeleev import Si, Fe, O
        print("Si's name: ", Si.name)
        print("Fe's atomic number:", Fe.atomic_number)
        print("O's atomic weight: ", O.atomic_weight)
```

```
Si's name:  Silicon
Fe's atomic number: 26
O's atomic weight:  15.999
```

An alternative interface to the data is through the `element` function that returns a single `Element` object or a list of `Element` object depending on the arguments.

The function can be imported directly from the `mendeleev` package

```
In [7]: from mendeleev import element
```

The `element` method accepts unique identifiers: **atomic number**, **atomic symbol** or **element's name** in English. To retrieve the entries on Silicon by symbol type

```
In [8]: si = element('Si')

In [9]: si

Out[9]: Element(
            abundance_crust=282000.0,
            abundance_sea=2.2,
```

```
        annotation='',
        atomic_number=14,
        atomic_radius=132.0,
        atomic_radius_rahm=231.99999999999997,
        atomic_volume=12.1,
        atomic_weight=28.085,
        atomic_weight_uncertainty=None,
        block='p',
        boiling_point=2628.0,
        c6=305.0,
        c6_gb=308.0,
        cas='7440-21-3',
        covalent_radius_bragg=117.0,
        covalent_radius_cordero=111.00000000000001,
        covalent_radius_pyykko=115.99999999999999,
        covalent_radius_pyykko_double=107.0,
        covalent_radius_pyykko_triple=102.0,
        covalent_radius_slater=110.00000000000001,
        cpk_color='#daa520',
        density=2.33,
        description="Metalloid element belonging to group 14 of the periodic table. It is the sec
        dipole_polarizability=37.31,
        discoverers='Jöns Berzelius',
        discovery_location='Sweden',
        discovery_year=1824,
        ec=<ElectronicConfiguration(conf="1s2 2s2 2p6 3s2 3p2")>,
        econf='[Ne] 3s2 3p2',
        electron_affinity=1.3895211,
        en_allen=11.33,
        en_ghosh=0.178503,
        en_pauling=1.9,
        evaporation_heat=383.0,
        fusion_heat=50.6,
        gas_basicity=814.1,
        geochemical_class='major',
        goldschmidt_class='litophile',
        group=<Group(symbol=IVA, name=Carbon group)>,
        group_id=14,
        heat_of_formation=450.0,
        ionic_radii=[IonicRadius(
        atomic_number=14,
        charge=4,
        coordination='IV',
        crystal_radius=40.0,
        econf='2p6',
        id=379,
        ionic_radius=26.0,
        most_reliable=True,
        origin='',
        spin='',
    ), IonicRadius(
        atomic_number=14,
        charge=4,
        coordination='VI',
        crystal_radius=54.0,
        econf='2p6',
        id=380,
        ionic_radius=40.0,
        most_reliable=True,
```

```
            origin='from r^3 vs V plots, ',
            spin='',
        )],
            is_monoisotopic=None,
            is_radioactive=False,
            isotopes=[<Isotope(Z=14, A=28, mass=27.976926535)>, <Isotope(Z=14, A=29, mass=28.9764946
            jmol_color='#f0c8a0',
            lattice_constant=5.43,
            lattice_structure='DIA',
            melting_point=1683.0,
            metallic_radius=117.0,
            metallic_radius_c12=138.0,
            molcas_gv_color='#f0c8a0',
            name='Silicon',
            name_origin='Latin: silex, silicus, (flint).',
            period=3,
            proton_affinity=837.0,
            screening_constants=[<ScreeningConstant(Z=  14, n=  1, s=s, screening=    0.4255)>, <Scre
            sources='Makes up major portion of clay, granite, quartz (SiO2), and sand. Commercial pro
            specific_heat=0.703,
            symbol='Si',
            thermal_conductivity=149.0,
            uses='Used in glass as silicon dioxide (SiO2). Silicon carbide (SiC) is one of the hardes
            vdw_radius=210.0,
            vdw_radius_alvarez=219.0,
            vdw_radius_batsanov=210.0,
            vdw_radius_bondi=210.0,
            vdw_radius_dreiding=426.99999999999994,
            vdw_radius_mm3=229.0,
            vdw_radius_rt=None,
            vdw_radius_truhlar=None,
            vdw_radius_uff=429.5,
        )
```

Similarly to access the data by atomic number or element names type

```
In [10]: al = element(13)
         print(al.name)
```

```
Aluminum
```

```
In [11]: o = element('Oxygen')
         print(o.atomic_number)
```

```
8
```

Getting list of elements

The `element` method also accepts list or tuple of identifiers and then returns a list of `Element` objects

```
In [12]: c, h, o = element(['C', 'Hydrogen', 8])
         print(c.name, h.name, o.name)
```

```
Carbon Hydrogen Oxygen
```

3.1.2 Extended attributes

Next to simple attributes returning `str`, `int` or `float`, there are extended attributes

- `oxistates`, returns a list of oxidation states

- `ionenergies`, returns a dictionary of ionization energies
- `isotopes`, returns a list of `Isotope` objects
- `ionic_radii` returns a list of `IonicRadius` objects
- `ec`, electronic configuration object

Oxidation states

`oxistates` returns a list of most common oxidation states for a given element

```
In [13]: fe = element('Fe')
         print(fe.oxistates)
```

```
[6, 3, 2, 0, -2]
```

Ionization energies

The `ionenergies` returns a dictionary with ionization energies in `eV` as values and degrees of ionization as keys

```
In [14]: o = element('O')
         o.ionenergies
```

```
Out[14]: {1: 13.618054,
          2: 35.12111,
          3: 54.93554,
          4: 77.4135,
          5: 113.8989,
          6: 138.1189,
          7: 739.32679,
          8: 871.40985}
```

Isotopes

The `isotopes` attribute returns a list of `Isotope` objects with the following attributes per isotope

- abundance
- atomic_number
- half_life
- half_life_unit
- is_radioactive
- mass
- mass_number
- mass_uncertainty

```
In [15]: print("{0:^4s} {1:^4s} {2:^10s} {3:8s} {4:6s} {5:5s}\n{6}".format("AN", "MN", "Mass", "Unc."
         for iso in fe.isotopes:
             print('{0:4d} {1:4d} {2:10.5f} {3:8.2e} {4:6.2f} {5:}'.format(
                 iso.atomic_number, iso.mass_number, iso.mass, iso.mass_uncertainty, iso.abundance *
```

```
 AN   MN      Mass     Unc.     Abu.   Rad.
-----------------------------------------
  26   54   53.93961 3.00e-06   5.85 False
  26   56   55.93494 3.00e-06  91.75 False
```

```
  26   57   56.93539 3.00e-06   2.12 False
  26   58   57.93327 3.00e-06   0.28 False
```

Ionic radii

Another composite attribute is `ionic_radii` which returns a list of `IonicRadius` object with the following attributes

- `atomic_number`, atomic number of the ion
- `charge`, charge of the ion
- `econf`, electronic configuration of the ion
- `coordination`, coordination type of the ion
- `spin`, spin state of the ion (HS or LS)
- `crystal_radius`, crystal radius in pm
- `ionic_radius`, ionic radius in pm
- `origin`, source of the data
- `most_reliable`, recommended value, (see the original paper for more information)

```
In [16]: for ir in fe.ionic_radii:
             print(ir)
```

```
charge=   2, coordination=IV   , crystal_radius=77.000, ionic_radius=63.000
charge=   2, coordination=IVSQ , crystal_radius=78.000, ionic_radius=64.000
charge=   2, coordination=VI   , crystal_radius=75.000, ionic_radius=61.000
charge=   2, coordination=VI   , crystal_radius=92.000, ionic_radius=78.000
charge=   2, coordination=VIII , crystal_radius=106.000, ionic_radius=92.000
charge=   3, coordination=IV   , crystal_radius=63.000, ionic_radius=49.000
charge=   3, coordination=V    , crystal_radius=72.000, ionic_radius=58.000
charge=   3, coordination=VI   , crystal_radius=69.000, ionic_radius=55.000
charge=   3, coordination=VI   , crystal_radius=78.500, ionic_radius=64.500
charge=   3, coordination=VIII , crystal_radius=92.000, ionic_radius=78.000
charge=   4, coordination=VI   , crystal_radius=72.500, ionic_radius=58.500
charge=   6, coordination=IV   , crystal_radius=39.000, ionic_radius=25.000
```

electronic configuration

`ec` attribute is an object from the `ElectronicConfiguration` class that has additional method for manipulating the configuration. Internally the configuration is represented as a `OrderedDict` from the `collections` module where tuples `(n, s)` (`n` is the principal quantum number and `s` is the subshell label) are used as keys and shell occupations are the values

```
In [17]: si.ec.conf
```

```
Out[17]: OrderedDict([((1, 's'), 2),
                      ((2, 's'), 2),
                      ((2, 'p'), 6),
                      ((3, 's'), 2),
                      ((3, 'p'), 2)])
```

the occupation of different subshells can be access supplying a proper key

```
In [18]: si.ec.conf[(1, 's')]
```

```
Out[18]: 2
```

to calculate the number of electrons per shell type

```
In [19]: si.ec.electrons_per_shell()
```

```
Out[19]: {'K': 2, 'L': 8, 'M': 4}
```

get the largest value of the pricipal quantum number

```
In [20]: si.ec.max_n()
```

```
Out[20]: 3
```

Some useful functions

Next to stored attributes there is a number of useful functions

```
In [21]: si = element('Si')
```

```
In [22]: # get the number of valence electrons
         si.nvalence()
```

```
Out[22]: 4
```

```
In [23]: # calculate softness for an ion
         si.softness(charge=2)
```

```
Out[23]: 0.058318712346158874
```

```
In [24]: # calcualte hardness for an ion
         si.hardness(charge=4)
```

```
Out[24]: 60.812605
```

```
In [25]: # calculate mulliken electronegativity for a neutral atom or ion
         si.en_mulliken(charge=1)
```

```
Out[25]: 8.1729225
```

```
In [26]: # calculate the effective nuclear charge for a subshell using Slater's rules
         si.zeff(n=3, o='s')
```

```
Out[26]: 4.149999999999999
```

```
In [27]: # calculate the effective nuclear charge for a subshell using Clemneti's and Raimondi's exp
         si.zeff(n=3, o='s', method='clementi')
```

```
Out[27]: 4.9032
```

Electronegativity

Currently there are 9 electronagativity scales implemented that can me accessed though the common `electronegativity` method, the scales are:

- `allen`
- `allred-rochow`
- `cottrell-sutton`
- `gordy`
- `li-xue`
- `mulliken`
- `nagle`

- pauling
- sanderson

More information can be found in the documentation.

```
In [28]: si.electronegativity(scale='pauling')
```

```
Out[28]: 1.9
```

```
In [29]: si.electronegativity(scale='allen')
```

```
Out[29]: 11.33
```

3.1.3 CLI utility

For those who work in the terminal there is a simple command line interface (CLI) for printing the information about a given element. The script name is element.py and it accepts either the symbol or name of the element as an argument and prints the data about it. For example, to print the properties of silicon type

```
In [31]: !element.py Si
```

```
   _  _  _  _       _
 _(_)(_)(_)(_)_    (_)
(_)            (_)_  _
(_)_  _  _  _   (_)(_)
  (_)(_)(_)(_)_    (_)
 _             (_)   (_)
(_)_  _  _  _(_)_ (_) _
  (_)(_)(_)(_)  (_)(_)(_)



Description
===========

  Metalloid element belonging to group 14 of the periodic table. It is
  the second most abundant element in the Earth's crust, making up 25.7%
  of it by weight. Chemically less reactive than carbon. First
  identified by Lavoisier in 1787 and first isolated in 1823 by
  Berzelius.

Properties
==========

Abundance crust                                                  282000
Abundance sea                                                      2.2
Annotation
Atomic number                                                       14
Atomic radius                                                      132
Atomic radius rahm                                                 232
Atomic volume                                                     12.1
Atomic weight                                                   28.085
Atomic weight uncertainty                                          NaN
Block                                                                p
Boiling point                                                     2628
C6                                                                 305
C6 gb                                                              308
Cas                                                        7440-21-3
Covalent radius bragg                                              117
Covalent radius cordero                                            111
```

```
Covalent radius pyykko                                                116
Covalent radius pyykko double                                         107
Covalent radius pyykko triple                                         102
Covalent radius slater                                                110
Cpk color                                                         #daa520
Density                                                              2.33
Dipole polarizability                                               37.31
Discoverers                                                 Jöns Berzelius
Discovery location                                                  Sweden
Discovery year                                                        1824
Electron affinity                                                  1.38952
Electronic configuration                                     [Ne] 3s2 3p2
En allen                                                             11.33
En ghosh                                                          0.178503
En pauling                                                            1.9
Evaporation heat                                                      383
Fusion heat                                                          50.6
Gas basicity                                                        814.1
Geochemical class                                                   major
Goldschmidt class                                                litophile
Group id                                                             14
Heat of formation                                                    450
Is monoisotopic                                                     None
Is radioactive                                                     False
Jmol color                                                        #f0c8a0
Lattice constant                                                     5.43
Lattice structure                                                     DIA
Melting point                                                        1683
Metallic radius                                                       117
Metallic radius c12                                                   138
Molcas gv color                                                   #f0c8a0
Name                                                              Silicon
Name origin                                 Latin: silex, silicus, (flint).
Period                                                                 3
Proton affinity                                                      837
Series id                                                              5
Sources                       Makes up major portion of clay, granite, quart...
Specific heat                                                        0.703
Symbol                                                                  Si
Thermal conductivity                                                  149
Uses                          Used in glass as silicon dioxide (SiO2). Silic...
Vdw radius                                                            210
Vdw radius alvarez                                                    219
Vdw radius batsanov                                                   210
Vdw radius bondi                                                      210
Vdw radius dreiding                                                   427
Vdw radius mm3                                                        229
Vdw radius rt                                                         NaN
Vdw radius truhlar                                                    NaN
Vdw radius uff                                                      429.5
```

```
In [33]: %version_information mendeleev, sqlalchemy
```

Software	Version
Python	3.6.3 64bit [GCC 7.2.0]
IPython	6.2.1
OS	Linux 4.9.0 4 amd64 x86_64 with debian 9.1
mendeleev	0.3.6
sqlalchemy	1.1.13
Wed Nov 01 14:28:45 2017 CET	

3.2 Getting tables from the database

This short tutorial explains how to retrieve full tables from the database into pandas DataFrames.

3.2.1 The following tables are available from `mendeleev`

- elements
- ionicradii
- ionizationenergies
- oxidationstates
- groups
- series
- isotopes

`mendeleev` provides a convenient function `get_table` to perform the task at hand. The function can be directly imported from `mendeleev`

```
In [4]: from mendeleev import get_table
```

To retrieve a table call the `get_table` with the table name as argument. Here we'll get probably the most important table `elements` with basis data on each element

```
In [5]: ptable = get_table('elements')
```

Now we can use pandas' capabilities to work with the data.

```
In [6]: ptable.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 118 entries, 0 to 117
Data columns (total 67 columns):
annotation                      118 non-null object
atomic_number                   118 non-null int64
atomic_radius                   88 non-null float64
atomic_volume                   91 non-null float64
block                           118 non-null object
boiling_point                   96 non-null float64
density                         95 non-null float64
description                     109 non-null object
dipole_polarizability           106 non-null float64
electron_affinity               77 non-null float64
electronic_configuration        118 non-null object
evaporation_heat                88 non-null float64
fusion_heat                     75 non-null float64
group_id                        90 non-null float64
```

```
lattice_constant                 87 non-null float64
lattice_structure                91 non-null object
melting_point                    100 non-null float64
name                             118 non-null object
period                           118 non-null int64
series_id                        118 non-null int64
specific_heat                    81 non-null float64
symbol                           118 non-null object
thermal_conductivity             66 non-null float64
vdw_radius                       103 non-null float64
covalent_radius_cordero          96 non-null float64
covalent_radius_pyykko           118 non-null float64
en_pauling                       85 non-null float64
en_allen                         71 non-null float64
jmol_color                       109 non-null object
cpk_color                        103 non-null object
proton_affinity                  32 non-null float64
gas_basicity                     32 non-null float64
heat_of_formation                89 non-null float64
c6                               43 non-null float64
covalent_radius_bragg            37 non-null float64
covalent_radius_slater           86 non-null float64
vdw_radius_bondi                 28 non-null float64
vdw_radius_truhlar               16 non-null float64
vdw_radius_rt                    9 non-null float64
vdw_radius_batsanov              65 non-null float64
vdw_radius_dreiding              21 non-null float64
vdw_radius_uff                   103 non-null float64
vdw_radius_mm3                   94 non-null float64
abundance_crust                  88 non-null float64
abundance_sea                    81 non-null float64
molcas_gv_color                  103 non-null object
en_ghosh                         103 non-null float64
vdw_radius_alvarez               94 non-null float64
c6_gb                            86 non-null float64
atomic_weight                    118 non-null float64
atomic_weight_uncertainty        74 non-null float64
is_monoisotopic                  21 non-null object
is_radioactive                   118 non-null bool
cas                              118 non-null object
atomic_radius_rahm               96 non-null float64
geochemical_class                76 non-null object
goldschmidt_class                118 non-null object
metallic_radius                  56 non-null float64
metallic_radius_c12              63 non-null float64
covalent_radius_pyykko_double    108 non-null float64
covalent_radius_pyykko_triple    80 non-null float64
discoverers                      118 non-null object
discovery_year                   105 non-null float64
discovery_location               106 non-null object
name_origin                      118 non-null object
sources                          118 non-null object
uses                             112 non-null object
dtypes: bool(1), float64(44), int64(3), object(19)
memory usage: 61.0+ KB
```

For clarity let's take only a subset of columns

```
In [7]: cols = ['atomic_number', 'symbol', 'atomic_radius', 'en_pauling', 'block', 'vdw_radius_mm3']
```

```
In [8]: ptable[cols].head()
```

```
Out[8]: atomic_number symbol  atomic_radius  en_pauling block  vdw_radius_mm3
        0              1      H            79.0        2.20     s            162.0
        1              2     He             NaN         NaN     s            153.0
        2              3     Li           155.0        0.98     s            255.0
        3              4     Be           112.0        1.57     s            223.0
        4              5      B            98.0        2.04     p            215.0
```

It is quite easy now to get descriptive statistics on the data.

```
In [9]: ptable[cols].describe()
```

```
Out[9]: atomic_number  atomic_radius  en_pauling  vdw_radius_mm3
        count     118.000000      88.000000   85.000000       94.000000
        mean       59.500000     169.397727    1.748588      248.468085
        std        34.207699      49.810108    0.634442       36.017828
        min         1.000000      79.000000    0.700000      153.000000
        25%        30.250000     137.000000    1.240000      229.000000
        50%        59.500000     160.000000    1.700000      244.000000
        75%        88.750000     181.000000    2.160000      269.250000
        max       118.000000     299.000000    3.980000      364.000000
```

3.2.2 Isotopes table

Let try and retrieve another table, namely `isotopes`

```
In [10]: isotopes = get_table('isotopes', index_col='id')
```

```
In [11]: isotopes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 406 entries, 1 to 406
Data columns (total 11 columns):
atomic_number        406 non-null int64
mass                 377 non-null float64
abundance            288 non-null float64
mass_number          406 non-null int64
mass_uncertainty     377 non-null float64
is_radioactive       406 non-null bool
half_life            121 non-null float64
half_life_unit       85 non-null object
spin                 323 non-null float64
g_factor             323 non-null float64
quadrupole_moment    320 non-null float64
dtypes: bool(1), float64(7), int64(2), object(1)
memory usage: 35.3+ KB
```

Merge the elements table with the isotopes

We can now perform SQL-like merge operation on two `DataFrames` and produce an outer join

```
In [12]: import pandas as pd
```

```
In [13]: merged = pd.merge(ptable[cols], isotopes, how='outer', on='atomic_number')
```

now we have the following columns in the `merged DataFrame`

```
In [14]: merged.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 406 entries, 0 to 405
Data columns (total 16 columns):
atomic_number        406 non-null int64
symbol               406 non-null object
atomic_radius        325 non-null float64
en_pauling           313 non-null float64
block                406 non-null object
vdw_radius_mm3       350 non-null float64
mass                 377 non-null float64
abundance            288 non-null float64
mass_number          406 non-null int64
mass_uncertainty     377 non-null float64
is_radioactive       406 non-null bool
half_life            121 non-null float64
half_life_unit       85 non-null object
spin                 323 non-null float64
g_factor             323 non-null float64
quadrupole_moment    320 non-null float64
dtypes: bool(1), float64(10), int64(2), object(3)
memory usage: 51.1+ KB
```

```
In [15]: merged.head()
```

```
Out[15]: atomic_number symbol  atomic_radius  en_pauling block  vdw_radius_mm3  \
         0              1      H           79.0         2.2     s           162.0
         1              1      H           79.0         2.2     s           162.0
         2              1      H           79.0         2.2     s           162.0
         3              2     He            NaN         NaN     s           153.0
         4              2     He            NaN         NaN     s           153.0

               mass  abundance  mass_number  mass_uncertainty  is_radioactive  \
         0  1.007825   0.999720            1      6.000000e-10           False
         1  2.014102   0.000280            2      8.000000e-10           False
         2       NaN        NaN            3               NaN            True
         3  3.016029   0.000002            3      2.000000e-08           False
         4  4.002603   0.999998            4      4.000000e-10           False

            half_life half_life_unit  spin  g_factor  quadrupole_moment
         0        NaN           None   0.5  5.585695            0.00000
         1        NaN           None   1.0  0.857438            0.00286
         2        NaN           None   0.5  5.957994            0.00000
         3        NaN           None   0.5 -4.254995            0.00000
         4        NaN           None   0.0  0.000000            0.00000
```

To display all the isotopes of Silicon

```
In [16]: merged[merged['symbol'] == 'Si']
```

```
Out[16]: atomic_number symbol  atomic_radius  en_pauling block  vdw_radius_mm3  \
         28             14     Si          132.0         1.9     p           229.0
         29             14     Si          132.0         1.9     p           229.0
         30             14     Si          132.0         1.9     p           229.0

                mass  abundance  mass_number  mass_uncertainty  is_radioactive  \
         28  27.976927    0.92191           28      3.000000e-09           False
         29  28.976495    0.04699           29      3.000000e-09           False
         30  29.973770    0.03110           30      2.000000e-08           False

            half_life half_life_unit  spin  g_factor  quadrupole_moment
         28        NaN           None   0.0   0.00000                0.0
```

```
        29         NaN           None   0.5  -1.11058                0.0
        30         NaN           None   0.0   0.00000                0.0
```

```
In [17]: %version_information mendeleev, sqlalchemy, pandas
```

Software	Version
Python	3.6.3 64bit [GCC 7.2.0]
IPython	6.2.1
OS	Linux 4.9.0 4 amd64 x86_64 with debian 9.1
mendeleev	0.3.6
sqlalchemy	1.1.13
pandas	0.20.3
Wed Nov 01 15:15:51 2017 CET	

3.3 Jupyter notebooks

A series of short tutorials is available as Jupyter notebooks that present the main functionality and provide real-life examples

- Introduction
- Gettin tables
- Plotting periodic tables

CHAPTER 4

Data

4.1 Elements

The followig data are currently available:

Name	Type	Comment	Unit	Data Source
abundance_crust	float	Abundance in the Earth's crust	mg/kg	*[21]*
abundance_sea	float	Abundance in the seas	mg/L	*[21]*
annotation	str	Annotations regarding the data		
atomic_number	int	Atomic number		
atomic_radius	float	Atomic radius	pm	
atomic_radius_rahm	float	Atomic radius by Rahm et al.	pm	*[38]*
atomic_volume	float	Atomic volume	cm3/mol	
atomic_weight	float	Atomic weight[1]		*[30][55]*
atomic_weight_uncertainty	float	Atomic weight uncertainty[1]		*[30][55]*
block	int	Block in periodic table		
boiling_point	float	Boiling temperature	K	
c6	float	C_6 dispersion coefficient in a.u.	a.u.	*[13][47]*
c6_gb	float	C_6 dispersion coefficient in a.u. (Gould & Bučko)	a.u.	*[20]*
cas	str	Chemical Abstracts Serice identifier		

Continued on next page

Table 1 – continued from previous page

Name	Type	Comment	Unit	Data Source
covalent_radius_bragg	float	Covalent radius by Bragg	pm	*[10]*
covalent_radius_cordero	float	Covalent radius by Cerdero et al.[2]	pm	*[16]*
covalent_radius_pyykko	float	Single bond covalent radius by Pyykko et al.	pm	*[36]*
covalent_radius_pyykko_double	float	Double bond covalent radius by Pyykko et al.	pm	*[35]*
covalent_radius_pyykko_triple	float	Triple bond covalent radius by Pyykko et al.	pm	*[37]*
covalent_radius_slater	float	Covalent radius by Slater	pm	*[44]*
cpk_color	str	Element color in CPK convention	HEX	*[52]*
density	float	Density at 295K	g/cm3	
description	str	Short description of the element		
dipole_polarizability	float	Dipole polarizability	a.u.	*[58]*
discoverers	str	The discoverers of the element		
discovery_location	str	The location where the element was discovered		
dipole_year	int	The year the element was discovered		
electron_affinity	float	Electron affinity[3]	eV	*[21][6]*
electrons	int	Number of electrons		
en_allen	float	Allen's scale of electronegativity[4]	eV	*[26][27]*
en_ghosh	float	Ghosh's scale of electronegativity		*[18]*
en_mulliken	float	Mulliken's scale of electronegativity	eV	*[31]*
en_pauling	float	Pauling's scale of electronegativity		*[21]*
econf	str	Ground state electron configuration		
evaporation_heat	float	Evaporation heat	kJ/mol	
fusion_heat	float	Fusion heat	kJ/mol	
gas_basicity	float	Gas basicity	kJ/mol	*[21]*
geochemical_class	str	Geochemical classification		*[50]*
goldschmidt_class	str	Goldschmidt classification		*[50][51]*

Continued on next page

Table 1 – continued from previous page

Name	Type	Comment	Unit	Data Source
group	int	Group in periodic table		
heat_of_formation	float	Heat of formation	kJ/mol	*[21]*
ionenergy	tuple	Ionization energies	eV	*[22]*
ionic_radii	list	Ionic and crystal radii in pm	pm	*[43]*
is_monoisotopic	bool	Is the element monoisotopic		
is_radioactive	bool	Is the element radioactive		
isotopes	list	Isotopes		
jmol_color	str	Element color in Jmol convention	HEX	*[56]*
lattice_constant	float	Lattice constant	Angstrom	
lattice_structure	str	Lattice structure code		
mass_number	int	Mass number (most abundant isotope)		
melting_point	float	Melting temperature	K	
mendeleev_number	int	Mendeleev's number[5]		*[34][48]*
metallic_radius	float	Single-bond metallic radius	pm	*[1]*
metallic_radius_c12	float	Metallic radius with 12 nearest neighbors	pm	*[1]*
molcas_gv_color	str	Element color in MOCAS GV convention	HEX	*[57]*
name	str	Name in English		
name_origin	str	Origin of the name		
neutrons	int	Number of neutrons (most abundant isotope)		
oxistates	list	Oxidation states		
period	int	Period in periodic table		
proton_affinity	float	Proton affinity	kJ/mol	*[21]*
protons	int	Number of protons		
sconst	float	Nuclear charge screening constants[6]		*[14][15]*
series	int	Index to chemical series		
sources	str	Sources of the element		
specific_heat	float	Specific heat @ 20 C	J/(g mol)	
symbol	str	Chemical symbol		

Continued on next page

Table 1 – continued from previous page

Name	Type	Comment	Unit	Data Source
thermal_conductivity	float	Thermal conductivity @25 C	W/(m K)	
uses	str	Applications of the element		
vdw_radius	float	Van der Waals radius	pm	*[21]*
vdw_radius_alvarez	float	Van der Waals radius according to Alvarez[7]	pm	*[5][49]*
vdw_radius_batsanov	float	Van der Waals radius according to Batsanov	pm	*[8]*
vdw_radius_bondi	float	Van der Waals radius according to Bondi	pm	*[9]*
vdw_radius_dreiding	float	Van der Waals radius from the DREIDING FF	pm	*[29]*
vdw_radius_mm3	float	Van der Waals radius from the MM3 FF	pm	*[3]*
vdw_radius_rt	float	Van der Waals radius according to Rowland and Taylor	pm	*[40]*
vdw_radius_truhlar	float	Van der Waals radius according to Truhlar	pm	*[28]*
vdw_radius_uff	float	Van der Waals radius from the UFF	pm	*[39]*

[1] **Atomic Weights**

Atomic weights and their uncertainties were retrieved mainly from ref. *[55]*. For elements whose values were given as ranges the *conventional atomic weights* from Table 3 in ref. *[30]* were taken. For radioactive elements the standard approach was adopted where the weight is taken as the mass number of the most stable isotope. The data was obtained from CIAAW page on radioactive elements. In cases where two isotopes were specified the one with the smaller standard deviation was chosen. In case of Tc and Pm relative weights of their isotopes were used, for Tc isotope 98, and for Pm isotope 145 were taken from CIAAW.

[2] **Covalent Radius by Cordero et al.**

In order to have a more homogeneous data for covalent radii taken from ref. *[16]* the values for 3 different valences for C, also the low and high spin values for Mn, Fe Co, were respectively averaged.

[3] **Electron affinity**

Electron affinities were taken from *[21]* for the elements for which the data was available. For He, Be, N, Ar and Xe affinities were taken from *[6]* where they were specified for metastable ions and therefore the values are negative.

Updates

- Electron affinity of niobium was taken from *[25]*.
- Electron affinity of cobalt was taken from *[11]*.
- Electron affinity of lead was taken from *[12]*.

[4] **Allen's configuration energies**

The values of configurational energies from refs. *[26]* and *[27]* were taken as reported in eV without converting to Pauling units.

[5] **Mendeleev numbers**

Mendeleev numbers were mostly taken from *[48]* but the range was extended to cover the whole periodic table following the prescription in the article of increasing the numbers going from top to bottom in each group and group by group from left to right in the periodic table.

[6] **Nuclear charge screening constants**

The screening constants were calculated according to the following formula

$$\sigma_{n,l,m} = Z - n \cdot \zeta_{n,l,m}$$

4.2 Isotopes

Name	Type	Comment	Unit	Data Source
abundance	float	Relative Abundance		*[54]*
g_factor	float	Nuclear g-factor[8]		*[46]*
half_life	float	Half life of the isotope		*[30]*
half_life_unit	str	Unit in which the half life is given		*[30]*
is_radioactive	bool	Is the isotope radioactive		*[53]*
mass	float	Atomic mass	Da	*[53]*
mass_number	int	Mass number of the isotope		*[53]*
mass_uncertainty	float	Uncertainty of the atomic mass		*[53]*
spin	float	Nuclear spin quantum number		
quadrupole_moment	float	Nuclear electric quadrupole moment[8]	b [100 fm^2]	*[45]*

Data Footnotes

where n is the principal quantum number, Z is the atomic number, $\sigma_{n,l,m}$ is the screening constant, $\zeta_{n,l,m}$ is the optimized exponent from *[14][15]*.

For elements Nb, Mo, Ru, Rh, Pd and Ag the exponent values corresponding to the ground state electronic configuration were taken (entries with superscript *a* in Table II in *[15]*).

For elements La, Pr, Nd and Pm two exponent were reported for 4f shell denoted 4f and 4f' in *[15]*. The value corresponding to 4f were used since according to the authors these are the dominant ones.

[7] **van der Waals radii according to Alvarez**

The bulk of the radii data was taken from Ref. *[5]*, but the radii for noble gasses were update according to the values in Ref. *[49]*.

[8] **Isotope g-factors and quadrupole moments**

The data regarding g-factors and electric quadrupole moments was parsed from easyspin webpage (accessed 25.01.2017) where additional notes are mentioned:

- Typo for Rh-103: Moment is factor of 10 too large
- 237Np, 239Pu, 243Am magnetic moment data from *[21]*, section 11-2
- In quadrupole moment data - a typo for Ac-227: sign should be +

CHAPTER 5

Electronegativities

Since electronegativity is useful concept rather than a physical observable, several scales of electronegativity exist and some of them are available in *mendeleev*. Depending on the definition of a particular scale the values are either stored directly or recomputed on demand with appropriate formulas. The following scales are stored:

- *Allen*
- *Ghosh*
- *Pauling*

Moreover there are electronegativity scales that can be computed from their respective definition and the atomic properties available in *mendeleev*:

- *Allred-Rochow*
- *Cottrell-Sutton*
- *Gordy*
- *Li and Xue*
- *Martynov and Batsanov*
- *Mulliken*
- *Nagle*
- *Sanderson*

For a short overview on electronegativity see this presentation.

All the examples shown below are for Silicon:

```
>>> from mendeleev import element
>>> Si = element('Si')
```

5.1 Allen

The electronegativity scale proposed by Allen in ref *[2]* can be defined as:

$$\chi_A = \frac{\sum_x n_x \varepsilon_x}{\sum_x n_x}$$

where: ε_x is the multiplet-averaged one-electron energy of the subshell x and n_x is the number of electrons in subshell x and the summation runs over the valence shell.

The values that are tabulated were obtained from refs. *[26]* and *[27]*.

Example:

```
>>> Si.en_allen
11.33
>>> Si.electronegativity('allen')
11.33
```

5.2 Allred and Rochow

The scale of Allred and Rochow *[4]* introduces the electronegativity as the electrostatic force exerted on the electron by the nuclear charge:

$$\chi_{AR} = \frac{e^2 Z_{\text{eff}}}{r^2}$$

where: Z_{eff} is the effective nuclear charge and r is the covalent radius.

Example:

```
>>> Si.electronegativity('allred-rochow')
0.00028240190249702736
```

5.3 Cottrell and Sutton

The scale proposed by Cottrell and Sutton *[17]* is derived from the equation:

$$\chi_{CS} = \sqrt{\frac{Z_{\text{eff}}}{r}}$$

where: Z_{eff} is the effective nuclear charge and r is the covalent radius.

Example:

```
>>> Si.electronegativity('cottrell-sutton')
0.18099342720014772
```

5.4 Ghosh

Ghosh *[18]* presented a scale of electronegativity based on the absolute radii of atoms computed as

$$\chi_{GH} = a \cdot (1/R) + b$$

where: R is the absolute atomic radius and a and b are empirical parameters.

Example:

```
>>> Si.en_ghosh
0.178503
```

5.5 Gordy

Gordy's scale *[19]* is based on the potential that measures the work necessary to achieve the charge separation, according to:

$$\chi_G = \frac{eZ_{\text{eff}}}{r}$$

where: Z_{eff} is the effective nuclear charge and r is the covalent radius.

Example:

```
>>> Si.electronegativity('gordy')
0.03275862068965517
```

5.6 Li and Xue

Li and Xue *[23][24]* proposed a scale that takes into account different valence states and coordination environment of atoms and is calculated according to the following formula:

$$\chi_{LX} = \frac{n^* \sqrt{I_j / Ry}}{r}$$

where: n^* is the effective principal quantum number, I_j is the *j*'th ionization energy in eV, Ry is the Rydberg constant in eV and r is either the crystal radius or ionic radius.

Example:

```
>>> Si.en_li_xue(charge=4)
{u'IV': 13.16033405547733, u'VI': 9.748395596649873}
>>> Si.electronegativity('li-xue', charge=4)
{u'IV': 13.16033405547733, u'VI': 9.748395596649873}
```

5.7 Martynov and Batsanov

Martynov and Batsanov *[7]* used the square root of the averaged valence ionization energy as a measure of electronegativity:

$$\chi_{MB} = \sqrt{\frac{1}{n_v} \sum_{k=1}^{n_v} I_k}$$

where: n_v is the number of valence electrons and I_k is the k th ionization potential.

Example:

```
>>> Si.en_martynov_batsanov()
5.0777041564076963
>>> Si.electronegativity(scale='martynov-batsanov')
5.0777041564076963
```

5.8 Mulliken

Mulliken scale *[31]* is defined as the arithmetic average of the ionization potential (IP) and the electron affinity (EA):

$$\chi_M = \frac{IP + EA}{2}$$

Example:

```
>>> Si.en_mulliken()
4.0758415
>>> Si.electronegativity('mulliken')
4.0758415
```

5.9 Nagle

Nagle *[32]* derived his scale from the atomic dipole polarizability:

$$\chi_N = \sqrt[3]{\frac{n}{\alpha}}$$

Example:

```
>>> Si.electronegativity('nagle')
0.47505611644667534
```

5.10 Pauling

Pauling's thermochemical scale was introduced in *[33]* as a relative scale based on electronegativity differences:

$$\chi_A - \chi_B = \sqrt{E_d(AB) - \frac{1}{2}\left[E_d(AA) + E_d(BB)\right]}$$

where: $E_d(XY)$ is the bond dissociation energy of a diatomic XY. The values available in *mendeleev* are taken from ref. *[21]*.

Example:

```
>>> Si.en_pauling
1.9
>>> Si.electronegativity('pauling')
1.9
```

5.11 Sanderson

Sanderson *[41][42]* established his scale of electronegativity based on the stability ratio:

$$\chi_S = \frac{\rho}{\rho_{\text{ng}}}$$

where: ρ is the average electron density $\rho = \frac{Z}{4\pi r^3/3}$, and ρ_{ng} is the average electron density of a hypothetical noble gas atom with charge Z.

Example:

```
>>> Si.en_sanderson()
0.3468157872145231
>>> Si.electronegativity()
0.3468157872145231
```

5.12 Bibliography

CHAPTER 6

API Reference

6.1 Accessing data

6.1.1 Elements

The easiest way to access individual elements is simply by importing them from the *mendeleev* directly using their symbols:

```
>>> from mendeleev import H, C, O, Og
>>> [x.name for x in [H, C, O, Og]]
['Hydrogen', 'Carbon', 'Oxygen', 'Oganesson']
```

An alternative method of access is through the *element()* function that returns either a single *Element* instance or a tuple of those instances depending on the input. It provides a more flexible interface since it accepts element names, atomic numbers and symbols as well as their combinations.

element(*ids*)

Based on the type of the *ids* identifier return either an *Element* object from the database, or a list of *Element* objects if the *ids* is a list or a tuple of identifiers. Valid identifiers for an element are: *name*, *symbol*, and *atomic number*.

Args: ids (str): element identifier

Raises: ValueError: when the identifier is not a list/tuple, int or str

Example: The element can be identified by symbol

```
>>> from mendeleev import element
>>> si = element('Si')
>>> si.atomic_number
14
```

by the atomic number

```
>>> al = element(13)
>>> al.name
'Aluminum'
```

or by the name

```
>>> o = element('Oxygen')
>>> o.symbol
'O'
```

Mutiple elements can be instantiated simultaneously through as combination of identifiers

```
>>> c, h, o = element(['C', 'Hydrogen', 8])
>>> print(c.name, h.name, o.name)
Carbon Hydrogen Oxygen
```

6.1.2 Tables

If you want a whole set of data you can retrieve one of the tables from the database as pandas DataFrame through the *get_table*. The following tables are available:

- *elements*
- *groups*
- *ionicradii*
- *ionizationenergies*
- *isotopes*
- *oxidationstates*
- *screeningconstants*
- *series*

get_table(*tablename*, ***kwargs*)

Return a table from the database as pandas DataFrame

Args:

tablename: str Name of the table from the database

kwargs: A dictionary of keyword arguments to pass to the *pandas.read_qsl*

Returns:

df: pandas.DataFrame Pandas DataFrame with the contents of the table

Example:

```
>>> from mendeleev import get_table
>>> df = get_table('elements')
>>> type(df)
pandas.core.frame.DataFrame
```

6.1.3 Database session and engine

For those how want to interact with the database through a layer of SQLAlchemy there are methods for getting the session or the engine:

get_session(*dbpath=None*)
Return the database session connection.

get_engine(*dbpath=None*)
Return the db engine

6.2 Classes

6.2.1 Element

class Element(***kwargs*)
Chemical element.

Attributes:

abundance_crust [float] Abundance in the earth's crust in mg/kg

abundance_sea [float] Abundance in the seas in mg/L

annotation [str] Annotations regarding the data

atomic_number [int] Atomic number

atomic_radius [float] Atomic radius in pm

atomic_radius_rahm [float] Atomic radius by Rahm et al. in pm

atomic_volume [float] Atomic volume in cm3/mol

atomic_weight [float] Relative atomic weight as the ratio of the average mass of atoms of the element to 1/12 of the mass of an atom of 12C

block [int] Block in periodic table, s, p, d, f

boiling_point [float] Boiling temperature in K

c6 [float] C_6 dispersion coefficient in a.u. from X. Chu & A. Dalgarno, J. Chem. Phys., 121(9), 4083–4088 (2004) doi:10.1063/1.1779576, and the value for Hydrogen was taken from K. T. Tang, J. M. Norbeck and P. R. Certain, J. Chem. Phys. 64, 3063 (1976), doi:10.1063/1.432569

c6_gb [float] C_6 dispersion coefficient in a.u. from Gould, T., & Bučko, T. (2016). JCTC, 12(8), 3603–3613. http://doi.org/10.1021/acs.jctc.6b00361

cas [str] Chemical Abstracts Service identifier

covalent_radius_bragg [float] Covalent radius in pm from

covalent_radius_cordero [float] Covalent radius in pm from Cordero, B., Gómez, V., Platero-Prats, A. E., Revés, M., Echeverría, J., Cremades, E., … Alvarez, S. (2008). Covalent radii revisited. Dalton Transactions, (21), 2832. doi:10.1039/b801115j

covalent_radius_pyykko [float] Single bond covalent radius in pm Pyykkö, P., & Atsumi, M. (2009). Molecular Single-Bond Covalent Radii for Elements 1-118. Chemistry - A European Journal, 15(1), 186–197. doi:10.1002/chem.200800987

covalent_radius_pyykko_double [float] Double bond covalent radius in pm from P. Pyykkö et al.

covalent_radius_pyykko_triple [float] Triple bond covalent radius in pm from P. Pyykkö et al.

covalent_radius_slater [float] Covalent radius in pm from Slater

cpk_color [str] CPK color of the atom in HEX, see http://jmol.sourceforge.net/jscolors/#color_U

density [float] Density at 295K in g/cm3

description [str] Short description of the element

dipole_polarizability [float] Dipole polarizability in atomic units from P. Schwerdtfeger "Table of experimental and calculated static dipole polarizabilities for the electronic ground states of the neutral elements (in atomic units)", February 11, 2014

discoverers: str The discoverers of the element

discovery_location: str The location where the element was discovered

discovery_year: int The year the element was discovered

electron_affinity [float] Electron affinity in eV

en_allen [float] Allen's scale of electronegativity (Configurational energy)

en_ghosh [float] Ghosh's scale of enectronegativity

en_pauling [float] Pauling's scale of electronegativity

econf [str] Ground state electron configuration

evaporation_heat [float] Evaporation heat in kJ/mol

fusion_heat [float] Fusion heat in kJ/mol

gas_basicity [Float] Gas basicity

geochemical_class [String] Geochemical classification of the elements

goldschmidt_class [String] Goldschmidt classification of the elements

group [int] Group number

group_id [Group] Group details

heat_of_formation [float] Heat of formation in kJ/mol

is_monoisotopic [bool] A flag marking if the element is monoisotopic

jmol_color [str] Color of the atom as used in Jmol, in HEX, see http://jmol.sourceforge.net/jscolors/#color_U

lattice_constant [float] Lattice constant in ang

lattice_structure [str] Lattice structure code

mass [float] Relative atomic mass. Ratio of the average mass of atoms of the element to 1/12 of the mass of an atom of 12C

mendeleev_number [int] Mendeleev number

melting_point [float] Melting temperature in K

metallic_radius [Float] Single-bond metallic radius or metallic radius, have been calculated by Pauling using interatomic distances and an equation relating such distances with bond number

metallic_radius_c12 [Float] Metallic radius obtained by Pauling with an assumed number of nearest neighbors equal to 12

molcas_gv_color [str] Color of an atom in HEX from MOLCAS GV http://www.molcas.org/GV/

name [str] Name in English

name_origin: str Origin of the name

period [int] Period in periodic table

proton_affinity [Float] Proton affinity

series [int] Index to chemical series

sources: str Sources of the element

specific_heat [float] Specific heat in J/g mol @ 20 C

symbol [str of length 1 or 2] Chemical symbol

thermal_conductivity [float] Thermal conductivity in @/m K @25 C

uses: str Uses of the element

vdw_radius [float] Van der Waals radius in pm from W. M. Haynes, Handbook of Chemistry and Physics 95th Edition, CRC Press, New York, 2014, ISBN-10: 1482208679, ISBN-13: 978-1482208672.

vdw_radius_bondi [float] Van der Waals radius according to Bondi in pm

vdw_radius_truhlar [float] Van der Waals radius according to Truhlar in pm

vdw_radius_rt [float] Van der Waals radius according to Rowland and Taylor in pm

vdw_radius_batsanov [float] Van der Waals radius according to Batsanov in pm

vdw_radius_dreiding [float] Van der Waals radius from the DREIDING force field in pm

vdw_radius_uff [float] Van der Waals radius from the UFF in pm

vdw_radius_mm3 [float] Van der Waals radius from MM3 in pm

oxistates [list] Oxidation states

ionenergy [dict] Ionization energies in eV parsed from http://physics.nist.gov/cgi-bin/ASD/ie.pl on April 13, 2015

calc_en_sanderson(*radius='covalent_radius_pyykko'*)
Sanderson electronegativity

$$\chi = \frac{AD}{AD_{\text{ng}}}$$

Args:

radius [str] Radius to use in the calcualtion

covalent_radius
Return the default covalent radius which is covalent_radius_pyykko

electronegativity(*scale='pauling'*, *charge=0*)
Calculate the electronegativity using one of the methods

Args:

scale [str] Name of the electronegativity scale, one of

- *allen*
- *allred-rochow*
- *cottrell-sutton*
- *gordy*

- *li-xue*
- *martynov-batsanov*
- *mulliken*
- *nagle*
- *pauling*
- *sanderson*

electrons
Return the number of electrons.

en_calc(*radius='covalent_radius_pyykko'*, *rpow=1*, *apow=1*, ***zeffkwargs*)
Calculate the electronegativity from a general formula

$$\chi = \left(\frac{Z_{\text{eff}}}{r^{\beta}} \right)^{\alpha}$$

where

- Z_{eff} is the effective nuclear charge
- r is the covalent radius
- α, β parameters

en_li_xue(*charge=0*, *radius='crystal_radius'*)
Calculate the electronegativity of an atom according to the definition of Li and Xue

Args:

charge [int] Charge of the ion

radius [str] Type of radius to be used in the calculation, either *crystal_radius* as recommended in the paper or *ionic_radius*

Returns:

out [dict] A dictionary with electronegativities as values and coordination string as keys or tuple of coordination and spin if the ion is LS or HS

en_martynov_batsanov()
Calculates the electronegativity value according to Martynov and Batsanov as the average of the ionization energies of the valence electrons

$$\chi_{MB} = \sqrt{\frac{1}{n_v} \sum_{k=1}^{n_v} I_k}$$

where: n_v is the number of valence electrons and I_k is the k th ionization potential.

en_mulliken(*charge=0*, *missingIsZero=False*, *useNegativeEA=False*)
Return the absolute electronegativity (Mulliken scale), calculated as

$$\chi = \frac{I + A}{2}$$

where I is the ionization energy and A is the electron affinity

hardness(*charge=0*)
Return the absolute hardness, calculated as

$$\eta = \frac{I - A}{2}$$

where I is the ionization energy and A is the electron affinity

Args:

charge: int Charge of the cation for which the hardness will be calculated

init_on_load()
Initialize the ElectronicConfiguration class as attribute of self

ionenergies
Return a dict with ionization degree as keys and ionization energies in eV as values.

mass
Return the *atomic_weight* if defined or mass number otherwise.

mass_number
Return the mass number of the most abundant natural stable isotope

mass_str()
String representation of atomic weight

neutrons
Return the number of neutrons of the most abundant natural stable isotope.

nvalence(*method=None*)
Return the number of valence electrons

oxistates
Return the oxidation states as a list of ints

protons
Return the number of protons.

sconst
Return a dict with screening constants with tuples (n, s) as keys and screening constants as values

softness(*charge=0*)
Return the absolute softness, calculated as

$$S = \frac{1}{2\eta}$$

where η is the absolute hardness

Args:

charge: int Charge of the cation for which the hardness will be calculated

zeff(*n=None*, *o=None*, *method='slater'*, *alle=False*)
Return the effective nuclear charge for `(n, s)`

Args:

method [str]

Method to calculate the screening constant, the choices are

- *slater*, for Slater's method as in Slater, J. C. (1930). Atomic Shielding Constants. Physical Review, 36(1), 57–64. doi:10.1103/PhysRev.36.57
- *clementi* for values of screening constants from Clementi, E., & Raimondi, D. L. (1963). Atomic Screening Constants from SCF Functions. The Journal of Chemical Physics, 38(11), 2686. doi:10.1063/1.1733573 and Clementi, E. (1967). Atomic Screening Constants from SCF Functions. II. Atoms with 37 to 86 Electrons. The Journal of Chemical Physics, 47(4), 1300. doi:10.1063/1.1712084

n [int] Principal quantum number

o [str] Orbital label, (s, p, d, . . .)

alle [bool] Use all the valence electrons, i.e. calculate screening for an extra electron when method='slater', if method='clementi' this option is ignored

6.2.2 IonicRadius

class IonicRadius(***kwargs*)

Effective ionic radii and crystal radii in pm retrieved from[1].

Attributes:

atomic_number [int] Atomic number

charge [int] Charge of the ion

econf [str] Electronic configuration of the ion

coordination [str] Type of coordination

spin [str] Spin state: HS - high spin, LS - low spin

crystal_radius [float] Crystal radius in pm

ionic_radius [float] Ionic radius in pm

origin [str] Source of the data

most_reliable [bool] Most reliable value (see reference)

6.2.3 IonizationEnergy

class IonizationEnergy(***kwargs*)

Ionization energy of an element

Attributes:

atomic_number [int] Atomic number

degree [int] Degree of ionization with respect to neutral atom

energy [float] Ionization energy in eV parsed from http://physics.nist.gov/cgi-bin/ASD/ie.pl on April 13, 2015

6.2.4 Isotope

class Isotope(***kwargs*)

Attributes:

abundance [float] Abundance of the isotope

atomic_number [int] Atomic number

half_life [float] Half life time

half_life_unit [str] Unit for the half life time

[1] Shannon, R. D. (1976). Revised effective ionic radii and systematic studies of interatomic distances in halides and chalcogenides. Acta Crystallographica Section A. doi:10.1107/S0567739476001551

is_radioactive [bool] A flag marking wheather the isotope is radioactive

mass [float] Mass of the isotope

mass_number [int] Mass number of the isotope

mass_uncertainty [float] Uncertainty of the mass

6.2.5 ScreeningConstant

class ScreeningConstant(***kwargs*)

Nuclear screening constants from Clementi, E., & Raimondi, D. L. (1963). Atomic Screening Constants from SCF Functions. The Journal of Chemical Physics, 38(11), 2686. doi:10.1063/1.1733573 and Clementi, E. (1967). Atomic Screening Constants from SCF Functions. II. Atoms with 37 to 86 Electrons. The Journal of Chemical Physics, 47(4), 1300. doi:10.1063/1.1712084

Attributes:

atomic_number [int] Atomic number

n [int] Principal quantum number

s [str] Subshell label, (s, p, d, . . .)

screening [float] Screening constant

6.2.6 Series

class Series(***kwargs*)

Name of the series in the periodic table.

Attributes:

name [str] Name of the series

color [str] The HEX representation of a color of the series, the colors were obtained from ColorBrewer the qualitative 10-class paired colormap

6.2.7 Group

class Group(***kwargs*)

Name of the group in the periodic table.

6.2.8 OxidationState

class OxidationState(***kwargs*)

Oxidation states of an element

Attributes:

atomic_number [int] Atomic number

oxidation_state [int] Oxidation state

6.2.9 ElectronicConfiguration

class ElectronicConfiguration(*conf=None*, *atomre=None*, *shellre=None*)
Electronic configuration handler

atomre
Regular expression for atomic symbols

conf
Return the configuration

electrons_per_shell()
Return number of electrons per shell as dict

get_largest_core()
Find the largest noble gas core possible for the current configuration and return the symbol of the corresponding noble gas element.

get_valence()
Find the valence configuration i.e. remove the largest noble gas core from the current configuration and return the result.

ionize(*n=1*)
Remove *n* electrons from and return a new *ElectronicConfiguration* object

last_subshell(*wrt='order'*)
Return the valence shell

max_l(*n*)
Return the largest value of azimutal quantum number for a given value of principal quantum number

Args:

n [int] Principal quantum number

max_n()
Return the largest value of principal quantum number for the atom

ne()
Return the number of electrons

nvalence(*block*, *method=None*)
Return the number of valence electrons

parse(*string*)
Parse a `string` with electronic configuration into an `OrderedDict` representation

shell2int()
configuration as list of tuples (n, l, e)

shellre
Regular expression for the shell

slater_screening(*n*, *o*, *alle=False*)
Calculate the screening constant using the approach introduced by Slater in Slater, J. C. (1930). Atomic Shielding Constants. Physical Review, 36(1), 57-64. doi:10.1103/PhysRev.36.57

Args:

n [int] Principal quantum number

o [str] orbtial label, (s, p, d, . . .)

alle [bool] Use all the valence electrons, i.e. calculate screening for an extra electron

sort(*inplace=True*)
Sort the occupations OD

spin_occupations()
For each subshell calculate the number of *alpha*, *beta* electrons, electron pairs and unpaired electrons

spin_only_magnetic_moment()
Return the magnetic moment insluding only spin of the electrons and not the angular momentum

to_str()
Return a string with the configuration

unpaired_electrons()
Return the number of unpaired electrons

CHAPTER 7

License

CHAPTER 8

mendeleev Changelog

8.1 Latest

- Added `mendeleev_number` attribute to elements.
- Added footnotes to the data documentation.

8.2 v0.4.0 (22-11-2017)

- The elements can now be directly imported from *mendeleev* by symbols.
- Added sphinxcontrib.bibtex extension to the docs to handle BibTeX style references to improve handling of the bibliographic entries.
- Added nbsphinx to include Jupyter Notebook tutorials in the docs.

8.3 v0.3.6 (17-09-2017)

- Added API documentation
- Corrected the sphinx configuration
- Updated the documentation

8.4 v0.3.5 (07-09-2017)

- Added a module with functions to scrape data from ciaaw.org
- Added new `Element` attributes, `name_origin`, `uses` and `sources`

- Added new `Element` attributes related to the discovery: `discoverers`, `discovery_location`, `discovery_year`

8.5 v0.3.4 (28-06-2017)

- Fixed python2.7 compatibility issue
- Added double and triple bond covalent radii from Pyykko
- Corrected minor error in the documentation
- Replaced lazy loading with eager in db queries

8.6 v0.3.3 (16-05-2017)

- Corrected the coordination of Br5+ ion in the ionic radii table

8.7 v0.3.2 (01-05-2017)

- Added `metallic_radius`
- Added Goldschmidt and geochemical classifications
- Corrected the docs configuration
- Added `cas` number attribute
- Added atomic radii by Rahm et al.
- Created a conda recipe
- Added a citation information to the readme
- Electronic configuration code was split into a separate module

8.8 v0.3.1 (25-01-2017)

- Added new properties of isotopes: `spin`, `g_factor`, `quadrupole_moment`

8.9 v0.3.0 (09-01-2017)

- Updates of the documentation and tutorials
- Added radioactive isotope half-lifes

8.10 v0.2.17 (08-01-2017)

- Extended the schema for isotopes with additional attributes and updated the values of abundancies, half lifes and mass uncertainties.
- Updates to the tutorials and docs.

8.11 v0.2.16 (06-01-2017)

- Corrected the radioactive attribute of Th, Pa and U elements.

8.12 v0.2.15 (02-01-2017)

- Patched the sphinx configuration.

8.13 v0.2.14 (02-01-2017)

- Patched typos in README.

8.14 v0.2.13 (01-01-2017)

- Updated atomic weight with the newest IUPAC and CIAAW recommendations.
- Added `is_radioactive` and `is_monoisotopic` attributes.
- Updated the docs.

8.15 v0.2.12 (21-12-2016)

- Got rid of the scipy dependency.

8.16 v0.2.11 (10-11-2016)

- Updated the names and symbols of elements 113, 115, 117, 118.
- Updated the docs.

8.17 v0.2.10 (18-10-2016)

- Added the C6 coefficients from Gould and Bucko.
- Added van der Waals radii from Alvarez.

8.18 v0.2.9 (16-10-2016)

- Added a scale of electronegativities by Ghosh.

8.19 v0.2.8 (29-08-2016)

- Updated the electron affinity of Pb and Co.
- Updates of the docs.

8.20 v0.2.7 (02-04-2016)

- Maintenance.

8.21 v0.2.6 (02-04-2016)

- Mainly maintenance updates to docs, sphinx `conf.py`, `setup.py`, requirements.

8.22 v0.2.5 (02-04-2016)

8.22.1 Features added

- Added calculation of Martynov and Batsanov scale of electronegativity in `en_martynov_batsanov` method in the `Element` class
- Added `abundance_crust` and `abundance_sea` with element abundancies in the crust and seas
- Added `molcas_gv_color` attribute with MOLCAS GV colors

8.22.2 Bugs fixed

- Restored Python 3.x compatibility

8.23 v0.2.4 (05-02-2016)

8.23.1 Features added

- Extended and corrected the documentation and Jupyter notebook tutorials on basic usage electronegativities, plotting and tables

8.23.2 Bugs fixed

- Corrected `raise` to `return` when calling `en_sanderson` from `electronegativity`
- Fixed and tested the formula for calculating the Li and Xue scale of electronegativity in `en_lie-xue`

8.24 v0.2.3 (27-01-2016)

8.24.1 Features added

- Added new vdW radii: `vdw_radius_batsanov`, `vdw_radius_bondi`, `vdw_radius_dreiding`, `vdw_radius_mm3`, `vdw_radius_rt`, `vdw_radius_truhlar`, `vdw_radius_uff`
- Added an option to plot the long (wide) version of the periodic table in `periodic_plot`

8.24.2 Bugs fixed

- Typos in the docstrings

8.25 v0.2.2 (29-11-2015)

8.25.1 Features added

- Added new covalent radii: `covalent_radius_bragg`, `covalent_radius_slater`
- Added the `c6` dispersion coefficients
- Added `gas_basicity`, `proton_affinity` and `heat_of_formation`
- Added `periodic_plot` function for producing Bokeh based plots of the periodic table
- Added `jmol_color` and `cpk_color` with different coloring schemes for atoms

8.25.2 Bug fixes

- Changed the series of elements 113, 114, 115, 116 to poor metals

8.26 v0.2.1 (26-10-2015)

8.26.1 Features added

- Extended the list of options for calculating Mulliken electronegativities in `en_mulliken`
- Added `electrons_per_shell` method
- Added a function to calculate linear interpolation of radii required for calculation of Sandersons electronegativity
- Added hybrid attributes `electrons`, `protons`, `neutrons` and `mass_number`

8.26.2 Bug fixes

- Changed the type of the `melting_point` from `str` to `float`

8.27 v0.2.0 (22-10-2015)

8.27.1 Features added

- Instead of `covalent_radius` added `covalent_radius_2008` and `covalent_radius_2009`
- Instead of `electronegativity` added `en_pauling` and `en_mulliken`
- Added a method for getting ionic radii
- Improved the method for calculating the nuclear screening constants
- Added `ElectronicConfiguration` class initialized as `Element` attribute
- Added nuclear screening constants from Clementi and Raimondi
- Added a method to calculate the absolute softness, absolute hardness and absolute electronegativity
- Added `get_table` method to retrieve the tables as `pandas DataFrames`

8.27.2 Bug fixes

- Added missing electronic configurations
- Converted ionic radii from Angstrom to pico meters

8.28 v0.1.0 (11-07-2015)

First tagged version with the initial structure of the package and first version of the database and the python interface

CHAPTER 9

Indices and tables

- genindex
- modindex
- search

Bibliography

[1] Kyle & laby tables of physical & chemical constants. (2017). 3.7.5 atomic radii. [Online; accessed 30-April-2017]. URL: http://www.kayelaby.npl.co.uk/chemistry/3_7/3_7_5.html.

[2] Leland C Allen. Electronegativity is the average one-electron energy of the valence-shell electrons in ground-state free atoms. *Journal of the American Chemical Society*, 111(25):9003–9014, 1989. doi:10.1021/ja00207a003.

[3] Norman L. Allinger, Xuefeng Zhou, and John Bergsma. Molecular mechanics parameters. *Journal of Molecular Structure: THEOCHEM*, 312(1):69–83, jan 1994. URL: http://linkinghub.elsevier.com/retrieve/pii/S0166128009800080, doi:10.1016/S0166-1280(09)80008-0.

[4] A Louis Allred and E G Rochow. A scale of electronegativity based on electrostatic force. *Journal of Inorganic and Nuclear Chemistry*, 5(4):264–268, jan 1958. URL: http://linkinghub.elsevier.com/retrieve/pii/0022190258800032, doi:10.1016/0022-1902(58)80003-2.

[5] Santiago Alvarez. A cartography of the van der Waals territories. *Dalton Transactions*, 42(24):8617, 2013. doi:10.1039/c3dt50599e.

[6] T. Andersen. Atomic negative ions: structure, dynamics and collisions. *Physics Reports*, 394(4-5):157–313, may 2004. URL: http://linkinghub.elsevier.com/retrieve/pii/S0370157304000316, doi:10.1016/j.physrep.2004.01.001.

[7] Stepan S Batsanov. Dielectric Methods of Studying the Chemical Bond and the Concept of Electronegativity. *Russian Chemical Reviews*, 51(7):684–697, jul 1982. URL: http://stacks.iop.org/0036-021X/51/i=7/a=R08?key=crossref.14f2fec4c742d81d9efd6ad10be9ac6a, doi:10.1070/RC1982v051n07ABEH002900.

[8] Stepan S Batsanov. Van der Waals radii of elements. *Inorganic materials*, 37(9):871–885, 2001. URL: http://www.springerlink.com/index/wh8425p357657518.pdf, doi:10.1023/A:1011625728803.

[9] A Bondi. van der Waals Volumes and Radii. *The Journal of Physical Chemistry*, 68(3):441–451, 1964. URL: http://pubs.acs.org/doi/abs/10.1021/j100785a001, doi:10.1021/j100785a001.

[10] W. Lawrence Bragg. The arrangement of atoms in crystals. *Philosophical Magazine*, 40(236):169–189, aug 1920. URL: http://www.tandfonline.com/doi/abs/10.1080/14786440808636111, doi:10.1080/14786440808636111.

[11] Xiaolin Chen and Chuangang Ning. Accurate electron affinity of Co and fine-structure splittings of Co\$^-\$ via slow-electron velocity-map imaging. *Physical Review A*, 93(5):052508, may 2016. URL: http://link.aps.org/doi/10.1103/PhysRevA.93.052508, doi:10.1103/PhysRevA.93.052508.

[12] Xiaolin Chen and Chuangang Ning. Accurate electron affinity of Pb and isotope shifts of binding energies of Pb-. *The Journal of Chemical Physics*, 145(8):084303, aug 2016. URL: http://scitation.aip.org/content/aip/journal/jcp/145/8/10.1063/1.4961654, doi:10.1063/1.4961654.

[13] X Chu and Alexander Dalgarno. Linear response time-dependent density functional theory for van der Waals coefficients. *The Journal of chemical physics*, 121(9):4083–8, sep 2004. URL: http://www.ncbi.nlm.nih.gov/pubmed/15332953, doi:10.1063/1.1779576.

[14] Enrico Clementi and D L Raimondi. Atomic Screening Constants from SCF Functions. *The Journal of Chemical Physics*, 38(11):2686, 1963. URL: http://scitation.aip.org/content/aip/journal/jcp/38/11/10.1063/1.1733573, doi:10.1063/1.1733573.

[15] Enrico Clementi, D L Raimondi, and William P Reinhardt. Atomic Screening Constants from SCF Functions. II. Atoms with 37 to 86 Electrons. *The Journal of Chemical Physics*, 47(4):1300, 1967. URL: http://scitation.aip.org/content/aip/journal/jcp/47/4/10.1063/1.1712084, doi:10.1063/1.1712084.

[16] Beatriz Cordero, Verónica Gómez, Ana E Platero-Prats, Marc Revés, Jorge Echeverría, Eduard Cremades, Flavia Barragán, and Santiago Alvarez. Covalent radii revisited. *Dalton Transactions*, pages 2832, 2008. URL: http://xlink.rsc.org/?DOI=b801115j, doi:10.1039/b801115j.

[17] T. L. Cottrell and L. E. Sutton. Covalency, Electrovalency and Electronegativity. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 207(1088):49–63, jun 1951. URL: http://rspa.royalsocietypublishing.org/cgi/doi/10.1098/rspa.1951.0098, doi:10.1098/rspa.1951.0098.

[18] Dulal C Ghosh. A NEW SCALE OF ELECTRONEGATIVITY BASED ON ABSOLUTE RADII OF ATOMS. *Journal of Theoretical and Computational Chemistry*, 04(01):21–33, mar 2005. URL: http://www.worldscientific.com/doi/abs/10.1142/S0219633605001556, doi:10.1142/S0219633605001556.

[19] Walter Gordy. A New Method of Determining Electronegativity from Other Atomic Properties. *Physical Review*, 69(11-12):604–607, jun 1946. URL: http://link.aps.org/doi/10.1103/PhysRev.69.604, doi:10.1103/PhysRev.69.604.

[20] Tim Gould and Tomáš Bučko. C6 Coefficients and Dipole Polarizabilities for All Atoms and Many Ions in Rows 1-6 of the Periodic Table. *Journal of Chemical Theory and Computation*, 12(8):3603–3613, aug 2016. URL: http://pubs.acs.org/doi/abs/10.1021/acs.jctc.6b00361, doi:10.1021/acs.jctc.6b00361.

[21] William M Haynes. *CRC Handbook of Chemistry and Physics*. 100 Key Points. CRC Press, London, 95th edition, 2014. ISBN 9781482208689. URL: https://books.google.no/books?id=bNDMBQAAQBAJ.

[22] A Kramida, Yu Ralchenko, J Reader, and and NIST ASD Team. Nist atomic spectra database (ver. 5.3), national institute of standards and technology, gaithersburg, md. 2015. [Online; accessed 13-April-2015]. URL: http://physics.nist.gov/asd.

[23] Keyan Li and Dongfeng Xue. Estimation of Electronegativity Values of Elements in Different Valence States. *The Journal of Physical Chemistry A*, 110(39):11332–11337, oct 2006. URL: http://pubs.acs.org/doi/abs/10.1021/jp062886k, doi:10.1021/jp062886k.

[24] KeYan Li and DongFeng Xue. New development of concept of electronegativity. *Chinese Science Bulletin*, 54(2):328–334, jan 2009. URL: http://link.springer.com/10.1007/s11434-008-0578-9, doi:10.1007/s11434-008-0578-9.

[25] Zhihong Luo, Xiaolin Chen, Jiaming Li, and Chuangang Ning. Precision measurement of the electron affinity of niobium. *Physical Review A*, 93(2):020501, feb 2016. URL: http://link.aps.org/doi/10.1103/PhysRevA.93.020501, doi:10.1103/PhysRevA.93.020501.

[26] Joseph B Mann, Terry L Meek, and Leland C Allen. Configuration Energies of the Main Group Elements. *Journal of the American Chemical Society*, 122(12):2780–2783, mar 2000. URL: http://pubs.acs.org/doi/abs/10.1021/ja992866e, doi:10.1021/ja992866e.

[27] Joseph B Mann, Terry L Meek, Eugene T Knight, Joseph F Capitani, and Leland C Allen. Configuration Energies of the d-Block Elements. *Journal of the American Chemical Society*, 122(21):5132–5137, may 2000. URL: http://pubs.acs.org/doi/abs/10.1021/ja9928677, doi:10.1021/ja9928677.

[28] Manjeera Mantina, Adam C Chamberlin, Rosendo Valero, Christopher J Cramer, and Donald G Truhlar. Consistent van der Waals Radii for the Whole Main Group. *The Journal of Physical Chemistry A*, 113(19):5806–5812, may 2009. URL: http://pubs.acs.org/doi/abs/10.1021/jp8111556, doi:10.1021/jp8111556.

[29] Stephen L. Mayo, Barry D. Olafson, and William A Goddard III. DREIDING: a generic force field for molecular simulations. *The Journal of Physical Chemistry*, 94(26):8897–8909, dec 1990. URL: http://pubs.acs.org/doi/abs/10.1021/j100389a010, doi:10.1021/j100389a010.

[30] Juris Meija, Tyler B. Coplen, Michael Berglund, Willi A. Brand, Paul De Bièvre, Manfred Gröning, Norman E. Holden, Johanna Irrgeher, Robert D. Loss, Thomas Walczyk, and Thomas Prohaska. Atomic weights of the elements 2013 (IUPAC Technical Report). *Pure and Applied Chemistry*, 88(3):265–291, jan 2016. URL: http://www.degruyter.com/view/j/pac.2016.88.issue-3/pac-2015-0305/pac-2015-0305.xml, doi:10.1515/pac-2015-0305.

[31] Robert S Mulliken. A New Electroaffinity Scale; Together with Data on Valence States and on Valence Ionization Potentials and Electron Affinities. *The Journal of Chemical Physics*, 2(11):782, 1934. URL: http://scitation.aip.org/content/aip/journal/jcp/2/11/10.1063/1.1749394, doi:10.1063/1.1749394.

[32] Jeffrey K. Nagle. Atomic polarizability and electronegativity. *Journal of the American Chemical Society*, 112(12):4741–4747, jun 1990. URL: http://pubs.acs.org/doi/abs/10.1021/ja00168a019, doi:10.1021/ja00168a019.

[33] Linus Pauling. THE NATURE OF THE CHEMICAL BOND. IV. THE ENERGY OF SINGLE BONDS AND THE RELATIVE ELECTRONEGATIVITY OF ATOMS. *Journal of the American Chemical Society*, 54(9):3570–3582, sep 1932. URL: http://pubs.acs.org/doi/abs/10.1021/ja01348a011, doi:10.1021/ja01348a011.

[34] D G Pettifor. A chemical scale for crystal-structure maps. *Solid State Communications*, 51(1):31–34, jul 1984. URL: http://www.sciencedirect.com/science/article/pii/0038109884907658, doi:10.1016/0038-1098(84)90765-8.

[35] Pekka Pyykkö and Michiko Atsumi. Molecular Double-Bond Covalent Radii for Elements Li-E112. *Chemistry - A European Journal*, 15(46):12770–12779, nov 2009. URL: http://doi.wiley.com/10.1002/chem.200901472, doi:10.1002/chem.200901472.

[36] Pekka Pyykkö and Michiko Atsumi. Molecular Single-Bond Covalent Radii for Elements 1-118. *Chemistry - A European Journal*, 15(1):186–197, jan 2009. URL: http://doi.wiley.com/10.1002/chem.200800987, doi:10.1002/chem.200800987.

[37] Pekka Pyykkö, Sebastian Riedel, and Michael Patzschke. Triple-Bond Covalent Radii. *Chemistry - A European Journal*, 11(12):3511–3520, jun 2005. URL: http://doi.wiley.com/10.1002/chem.200401299, doi:10.1002/chem.200401299.

[38] Martin Rahm, Roald Hoffmann, and N. W. Ashcroft. Atomic and Ionic Radii of Elements 1-96. *Chemistry - A European Journal*, 22(41):14625–14632, oct 2016. URL: http://doi.wiley.com/10.1002/chem.201602949, doi:10.1002/chem.201602949.

[39] A K Rappe, C. J. Casewit, K. S. Colwell, William A Goddard III, and W. M. Skiff. UFF, a full periodic table force field for molecular mechanics and molecular dynamics simulations. *Journal of the American Chemical Society*, 114(25):10024–10035, dec 1992. URL: http://pubs.acs.org/doi/abs/10.1021/ja00051a040, doi:10.1021/ja00051a040.

[40] R Scott Rowland and Robin Taylor. Intermolecular Nonbonded Contact Distances in Organic Crystal Structures: Comparison with Distances Expected from van der Waals Radii. *The Journal of Physical Chemistry*, 100(18):7384–7391, 1996. doi:10.1021/jp953141+.

[41] R T Sanderson. An Interpretation of Bond Lengths and a Classification of Bonds. *Science*, 114(2973):670–672, dec 1951. URL: http://www.sciencemag.org/cgi/doi/10.1126/science.114.2973.670, doi:10.1126/science.114.2973.670.

[42] R T Sanderson. An Explanation of Chemical Variations within Periodic Major Groups. *Journal of the American Chemical Society*, 74(19):4792–4794, oct 1952. URL: http://pubs.acs.org/doi/abs/10.1021/ja01139a020, doi:10.1021/ja01139a020.

[43] R. D. Shannon. Revised effective ionic radii and systematic studies of interatomic distances in halides and chalcogenides. *Acta Crystallographica Section A*, 32(5):751–767, 1976. doi:10.1107/S0567739476001551.

[44] John C Slater. Atomic Radii in Crystals. *The Journal of Chemical Physics*, 41(10):3199, 1964. URL: http://scitation.aip.org/content/aip/journal/jcp/41/10/10.1063/1.1725697, doi:10.1063/1.1725697.

[45] N Stone. Table of nuclear quadrupole moments, international atomic energy agency, indc(nds)-650. December 2013. URL: https://www-nds.iaea.org/publications/indc/indc-nds-0650.pdf.

[46] N Stone. Table of nuclear magnetic dipole and electric quadrupole moments, international atomic energy agency, indc(nds)-0658. February 2014. URL: https://www-nds.iaea.org/publications/indc/indc-nds-0658.pdf.

[47] K T Tang, J M Norbeck, and P R Certain. Upper and lower bounds of two- and three-body dipole, quadrupole, and octupole van der Waals coefficients for hydrogen, noble gas, and alkali atom interactions. *The Journal of Chemical Physics*, 64(7):3063, 1976. URL: http://scitation.aip.org/content/aip/journal/jcp/64/7/10.1063/1.432569, doi:10.1063/1.432569.

[48] P. Villars, K. Cenzual, J. Daams, Y. Chen, and S. Iwata. Data-driven atomic environment prediction for binaries using the Mendeleev number. *Journal of Alloys and Compounds*, 367(1-2):167–175, mar 2004. URL: http://linkinghub.elsevier.com/retrieve/pii/S0925838803008004, doi:10.1016/j.jallcom.2003.08.060.

[49] Jürgen Vogt and Santiago Alvarez. van der Waals Radii of Noble Gases. *Inorganic Chemistry*, 53(17):9260–9266, sep 2014. URL: http://pubs.acs.org/doi/abs/10.1021/ic501364h, doi:10.1021/ic501364h.

[50] W M White. *Geochemistry*. Wiley, 2013. ISBN 9781118485255. URL: https://books.google.no/books?id=QPH1nY8WztkC.

[51] Wikipedia. Goldschmidt classification — wikipedia, the free encyclopedia. [Online; accessed 30-April-2017]. URL: https://en.wikipedia.org/w/index.php?title=Goldschmidt_classification&oldid=775842423.

[52] Wikipedia. Cpk coloring — wikipedia, the free encyclopedia. 2017. [Online; accessed 5-October-2017]. URL: https://en.wikipedia.org/w/index.php?title=CPK_coloring&oldid=802098372.

[53] IUPAC-CIAAW. Atomic masses. [Online; accessed 7-January-2017]. URL: hhttp://ciaaw.org/atomic-masses.htm.

[54] IUPAC-CIAAW. Isotopic abundances. [Online; accessed 7-January-2017]. URL: http://ciaaw.org/isotopic-abundances.htm.

[55] IUPAC-CIAAW. Standard atomic weights. [Online; accessed 1-January-2017]. URL: http://www.ciaaw.org/atomic-weights.htm.

[56] Jmol Team. Jmol colors. [Online; accessed 5-October-2017]. URL: http://jmol.sourceforge.net/jscolors/#color_U.

[57] MOLCAS Team. Molcas gv colors. [Online; accessed 5-October-2017]. URL: http://www.molcas.org/GV/.

[58] Schwerdtfeger, Peter. Table of experimental and calculated static dipole polarizabilities for the electronic ground states of the neutral elements (in atomic units). 2014. URL: http://ctcp.massey.ac.nz/Tablepol2014.pdf.

Index

U

Z