Kronos

P. D. Mullen, D. Lyons, R. S. Cumbee, P. C. Stancil
Department of Physics and Astronomy,
University of Georgia, Athens, Georgia 30602

(Dated: 7 February 2017)

Kronos is a series of python and fortran codes utilizing a comprehensive charge exchange database at its backbone to generate theoretical line ratios and spectra for many single electron capture systems. Kronos' power can be found in that it is 1) very easy to use, 2) it allows for the user to specify a variety of input parameters of interest 3) it is quick, and 4) above all, it contains a fully accessible and comprehensive database of multi-channel Landau-Zener single electron capture $n\ell S$-resolved cross sections for many projectile ion/ neutral target pairs at a wide variety of collision energies. These cross sections can be used in other cascade models outside of Kronos or studied directly; in other words, for some users, Kronos may never be run and it serves the sole purpose of acting as a stand-alone charge exchange cross section database.

I. INTRODUCTION

Kronos has been a collaborative effort of the Stancil Research Group (SRG). It contains a series of python and fortran codes authored by different members of the group and contains charge exchange data computed by the Stueckelberg program set written by Mr. David Lyons and Dr. Phillip Stancil. Kronos requires little instruction for its use and has been designed to be as user-friendly as possible. Suggestions and comments on additions of features or comments on possible improvements are encouraged. Kronos is only known to be running successfully on Mac and Linux machines.

II. CONTENTS

The Kronos package contains several directories filled with various files and file types:

I. CXDatabase:

1. excitation energy files (*.nist.in)
2. **Bare Ion Projectiles**
   (a) n-resolved cross section files (*_nres.cs)
3. **Non-Bare Ion Projectiles**
   (a) $n\ell S$-resolved cross section files (*_mclz.cs, *_qmocc.cs, *_aocc.cs, etc.)
4. Cascade Files (radrathe.dat):
   (a) Contains All Transitions Considered:
      i. Photon Energies
      ii. Transition Probabilities (Einstein A Coefficients)

II. kronos.py

1. the main script to be run

III. README

1. Contains this document.

IV: Results

1. Empty until you run kronos.py
2. After Running kronos.py:
   (a) Line Ratios ASCII File (line_ratios.dat)
   (b) Line Ratios Tex File (line_ratios.tex)
   (c) Line Ratios PDF File (line_ratios.pdf)
   (d) Spectrum File (spectrum.dat)
   (e) Miscellaneous Spectrum Files (all others)

V. Source Code

1. Contains files/codes called upon by kronos.py that are mandatory for proper running.

III. REQUIREMENTS FOR KRONOS

It is worth noting that many people will not need to run Kronos; to some, it may only be a source of charge exchange cross sections (raw data can be accessed in CXDatabase directory, see above). Otherwise, see below:

Requirements:

You MUST have python scientific libraries installed on your machine. I recommend obtaining pylab from http://continuum.io/downloads. Download the pylab distribution for Python 2.7. You must also have gfortran compilers (see below):

For Linux users:

```
sudo apt-get install gfortran
```

For Mac users:

We recommend installing these compilers from the following site: http://hpc.sourceforge.net. gcc compilers are also included within this distribution. GNU gfortran 6.3.0 is known to successfully compile the code. To install GCC (gfortran included) 6.3.0 from the recommended website, (a) download the gcc-6.3-bin.tar.gz file to your Desktop, (b) remove the .gz extension (if not done automatically by your machine) with the gunzip command:

```
cd  ~/Desktop
gunzip gcc-6.3-bin.tar.gz
```

and finally, (c) install the compilers:

```
cd  ~/Desktop
sudo tar xvf gcc-6.3-bin.tar -C /
```

Finally, pdflatex is required to automatically generate pdf files from generated .tex files. You will run into errors if you do not have pdflatex installed. To avoid such errors without installing pdflatex, comment out getResults() function. However, this will prevent generating line_ratios.dat, line_ratios.tex, and line_ratios.pdf.

IV. RUNNING KRONOS

First, let's make sure that kronos.py can be run as an executable. In working directory type:

```
chmod +x kronos.py
```

Now, kronos.py should be an executable. Now, to run Kronos, in the working directory type:

```
./kronos.py --help
```

This brings up a help menu. It details how to properly run the program, i.e. we must add tags/command-line-options (-I, -Z, -A, -E, etc.). Although all of these options are listed in the help menu, we also give them in Table 1. Note that some options are required while others are not.

Term	Meaning
./kronos.py	call the script (required)
-I	specify projectile ion (required)
-Z	specify charge of projectile ion (required)
-A	specify target (required)
-E	specify collision energy in **eV/u** (required)
-D	distribution function (required for MCLZ bare ion collisions)
-R	specify FWHM resolution in eV (optional)
-P	Specify number of data points (optional)
-X	specify rmax (optional: advanced users only)
-M	specify CX method (optional: MCLZ, CTMC, MOCC, etc.)
-G	Specify "yes" or "no" to graph (optional)

Here, we make several comments about Table 1. For bare-ion collisions, only n-resolved cross sections are contained in CXDatabase for the method MCLZ. For bare-ion collisions using a method other than MCLZ, $n\ell$-resolved cross sections must be contained within the proper directory. All n-resolved cross sections must have the **same** file format. All $n\ell$ and $n\ell S$-resolved cross sections must have the **same** format. This is **critical**. *Look at* $C^{5+,6+} + H$ *data in CXDatabase for examples*.

If using MCLZ data for bare-ion collisions, distribution functions (lowe, stat, lowemod, separable, SL1) are applied to MCLZ n-resolved cross sections after running Kronos.

If one adds their own data (QMOCC, AOCC, CTMC, MCLZ, etc.) it must be placed in the proper directory within the CXDatabase with a proper file name (for example, for C^{5+} collisions with H via QMOCC: c5+h_sec_qmocc.cs). Make sure you have the same amount of commented header lines and the correct column label format.

If one does not specify a method with the -M flag, Kronos has a built in hierarchy for the data used in computation. This hierarchy is given as (in order of greatest to least preference):

QMOCC → MOCC → AOCC → CTMC → MCLZ

Kronos will warn you if the requested collision energy (-E flag) is much different than available values. No interpolation schemes are employed.

The -D flag is required for bare ion collisions. Specifying the option lowe applies the low energy distribution, lowemod applies the low energy modified distribution, separable applies the separable distribution, stat applies the statistical distribution, SL1 applies the shifted low energy distribution.

An example of running kronos.py for C^{5+} + H_2O CX collisions at 10 eV/u mimicking a detector FWHM resolution of 10 eV:

```
./kronos.py -I C -Z 5 -A H2O -E 10 -R 10
```

Here we see that we have specified all required tags but have left out several of the optional ones – see Table 1.

An example of running kronos.py for the bare ion CX collision C^{6+} + H at 1 keV/u and applying the low energy distribution to charge exchange cross section data, the terminal command should be (on all one line):

```
./kronos.py -I C -Z 6 -A H -E 1000
            -M MCLZ -D lowe
```

After running the program, one can view the results in the Results directory.