CHIANTI

An Astrophysical Database for Emission Line Spectroscopy

CHIANTI TECHNICAL REPORT No. 15

Computing level populations

Version 0.4, 5 March 2019, Peter Young
Version 0.3, 30 January 2018, Peter Young
Version 0.2, 9 August 2017, Peter Young
Version 0.1, 28 June 2017, Peter Young

1 Overview

The calculation of level populations of an ion is perhaps the single most important part of CHIANTI. This document describes the software and methods used to perform the calculation within IDL.

Most users are recommended to use the routine `ch_pops` (Sect. 3) for calculating the populations.

2 The pop_solver routine

The workhorse of the CHIANTI IDL software is `pop_solver`, which takes the atomic data, creates the population process array, and then performs the matrix inversion to yield the level populations.

There is a two-step process in calling `pop_solver`. For example,

```
IDL> input=ch_setup_ion('o_6')
IDL> pop_solver, input, 5e5, 1e9, pop
```

The first call loads the atomic data for the ion O VI into the structure `input`. The call to `pop_solver` then uses the atomic data to compute the populations (`pop`) at the specified temperature (5×10^5 K) and electron number density (10^9 cm^{-3}).

The output `pop` is simply an array of populations for all of the ion's levels. If you need information on the identification of the levels, then you are recommended to use the routine `ch_pops` instead (Section 3).

The routine `ch_setup_ion` has a number of different options which can switch off certain processes, or modify data. Section 4 summarizes these options.

As of CHIANTI 9, new routines were added that take the atomic data from `input` and reformat them into rate matrices that form the matrix solved by `pop_solver`. More details of these "load rates" routines are given in Appendix B.

3 The ch_pops routine

A more user-friendly way of obtaining the level populations is to use the routine `ch_pops.pro`. For example:

```
IDL> pop=ch_pops('fe_13')
```

This prints the populations of the most populous levels to the screen, while information for all levels is output to the `pop` structure. The tags of the output are:

```
   DENS            DOUBLE       1.0000000e+10
   TEMP            DOUBLE           1778279.4
   LEVEL           STRUCT    -> <Anonymous> Array[749]
   RADTEMP         FLOAT           -1.00000
   RPHOT           FLOAT           -1.00000
```

```
PROTON          STRING    'yes'
VERSION         STRING    'CHIANTI 8.0.2'
DATE            STRING    'Wed Jun 28 15:36:26 2017'
SUM_MWL         INT             0
SUM_MWL_COEFFS  FLOAT        -1.00000
```

The tag `level` is a structure array, with the tags:

```
INDEX           INT             1
TERM            STRING    '3s2 3p2 3P0'
POP             DOUBLE         0.11817674
```

The level population is given in the tag `pop` and it is given as the population relative to the population of the ion as a whole, so summing the `pop` values over all levels gives 1.

By default, populations are calculated at an electron number density, $N_{\rm e}$, of 10^{10} cm^{-3} and the $T_{\rm max}$ of the ion (computed with `ch_tmax`) is used for the temperature. Keywords exist to switch off proton rates (`/noprot`) and level-resolved ionization and recombination rates (`/noionrec`) where these are available for the ion. Photon excitation and non-Maxwellian distributions can be included using the standard CHIANTI keywords (see CHIANTI User Guide).

Note that `ch_pops` is a wrapper for the older routine `show_pops`, which is itself a wrapper for calling `pop_solver`. `ch_pops` was written to standardize the input notation with other CHIANTI routines.

4 Atomic data setup (ch_setup_ion)

The default call to `ch_setup_ion` results in all of the ion's data files being loaded into the output structure. The core data-sets of energy levels, A-values, and electron collision strengths are always loaded. The secondary data-sets of proton rates and level-resolved ionization and recombination rates are loaded as long as the data files exist for the ions. They can be switched off using the /NOPROT and /NOIONREC keywords, respectively.

Photon excitation (and stimulated emission) do not have atomic data files since the rates depend on the A-values. They are switched on by the user by specifying RPHOT, the distance from the emitting source center in source radius units. The blackbody radiation temperature is set with the input RADTEMP.

For proton rates, `pop_solver` needs to know the proton density. This is computed self-consistently from the electron temperature (assumed to be the same as the proton temperature), and ion fraction file and the element abundance file. The optional inputs IONEQ_FILE and ABUND_FILE are used to specify these files. If not set, then the default files !IONEQ_FILE and !ABUND_FILE are used.

The simple call given in Sect. 2 will be sufficient for most users, but if you are calling `ch_setup_ion` from within another code and want to give users full access to the options, then the call would be:

```
input=ch_setup_ion(name,rphot=rphot,radtemp=radtemp,noprot=noprot, $
                   ioneq_file=ioneq_file,abund_file=abund_file, $
                   noionrec=noionrec)
```

A Document history

Version 0.4, 5-Mar-2019. Added Sect. B.

B The "load rates" routines

With CHIANTI 9, a significant update to the IDL software was made. Two routines that were added are `ch_load_ion_rates` and `ch_load_2ion_rates`. These routines essentially take the atomic data that was read by `ch_setup_ion` and then reformat them into atomic rates that enter into the matrix equation that is solved by `pop_solver`. Previously this part of the code was contained in `pop_solver` itself. Now `pop_solver` calls out to the load rate routines.

Generally users will not need to run the load rates routines, but here we summarize how they work.

For most ions, the procedure is simply:

```
IDL> t=[1e5,2e5,3e5]
IDL> rates=ch_load_ion_rates('o_4',t)
```

where "t" is the temperature array for which the rates are computed. The result is a structure with the following tags:

```
   N_LEVELS        LONG               204
   AA              DOUBLE    Array[204, 204]
   AAX             DOUBLE    Array[204, 204]
   PPR             DOUBLE    Array[3, 204, 204]
   QQ              DOUBLE    Array[3, 204, 204]
   TEMP            FLOAT     Array[3]
   ION_DATA        STRUCT    -> <Anonymous> Array[1]
   MULT            FLOAT     Array[204]
   SUM_MWL_COEFFS  DOUBLE    Array[3]
   SUMTST          INT              0
```

The rate matrices are: radiative decay rates (AA), photoexcitation and stimulated emission rates (AAX), proton rate coefficients (PPR), and electron rate coefficients (QQ). The ION_DATA structure is simply the atomic data structure returned by `ch_setup_ion`.

If `pop_solver` sees that the tag `ion_data.autostr` exists, then it means that the ion has autoionization data (i.e., the `.auto` file exists). This requires a special two-ion atomic model whereby the CHIANTI model for the next ionization stage is added to the ion model. An example of how this is done for O VI is as follows:

```
IDL> rates1=ch_load_ion_rates('o_6',t)
IDL> rates2=ch_load_ion_rates('o_7',t)
IDL> rates=ch_load_2ion_rates(rates1,rates2)
```

The structure `rates` is then used for creating the atomic rates matrix used by `pop_solver`. Note that this structure has an expanded set of tags:

```
N_LEVELS        LONG              972
AA              DOUBLE    Array[972, 972]
QQ              DOUBLE    Array[3, 972, 972]
AAX             DOUBLE    Array[972, 972]
PPR             DOUBLE    Array[3, 972, 972]
IONIZ           DOUBLE    Array[3, 972, 972]
RR              DOUBLE    Array[3, 972, 972]
AI              DOUBLE    Array[972, 972]
DC              DOUBLE    Array[3, 972, 972]
DR              DOUBLE    Array[3, 972, 972]
```

In particular: level-resolved ionization rate coefficients (IONIZ), level-resolved radiative recombination rate coefficients (RR), autoionization rates (AI), dielectronic capture rate coefficients (DC), and level-resolved dielectronic recombination rate coefficients (DR). As of CHIANTI 9, the IONIZ and DR matrices only contain data for transitions between the two ground states.

If you compare the `rates1` and `rates` structures then you will see they have 923 and 972 levels, respectively (as of CHIANTI 9). This is because the 49 levels of the O VII CHIANTI model have been added to the O VI model. The level populations are calculated by `pop_solver` for all 972 levels, but the array is then truncated to 923 levels so that only the O VI populations are returned. If you would like to see the populations for all 972 levels, then use the `/all_levels` keyword input to `pop_solver`.