XSTAR Manual

Release 2.5x

XSTAR Team

Sep 04, 2024

TABLE OF CONTENTS:

CHAPTER ONE

INTRODUCTION

1.1 What is XSTAR?

Xstar is a command-driven computer program for calculating the physical conditions and spectra of ionized and nearly-neutral gases. It has been developed for simulating astrophysical plasmas, and has been used for modeling diverse environments, such as nebulae, stellar winds, accretion disks, and interstellar and intergalactic gases. In addition, it has been benchmarked against laboratory experiments.

Xstar is intended to be flexible, and for this reason it makes simple assumptions about the geometry of the gas, the observer, and any illumination or other soure of energy. Stripped to essentials, its job may be described simply: A spherical gas shell surrounding a central source of ionizing radiation absorbs some of this radiation and reradiates it in other portions of the spectrum; XSTAR computes the effects on the gas of absorbing this energy, and the spectrum of reradiated light. Other sources of heat may be included, such as mechanical compression or expansion, or cosmic ray scattering. The user supplies the shape and strength of the incident continuum, the elemental abundances in the gas, its density or pressure, and its thickness; the code returns the ionization balance and temperature, opacity, and emitted line and continuum fluxes. In addition, Xstar includes Python tools for diagnosing the various atomic rates and internal quantities, running grids of models, and accessing and modifying the raw atomic data.

This document describes how to run xstar and interpret the output. Chapters include: how to obtain and install xstar (chapter 2), a simple introductory tutorial (chapter 3), detailed descriptions of the input parameters and output (chapters 4 and 5), descriptions of the associated tools (chapters 6 thru 8), examples and threads (chapter 10), and descriptions of the internal working and assumptions (chapters 11 thru 13). The new user is encouraged to read chapter 3 and then 4 and 5.

1.2 Acknowledgements

We would like to thank many colleagues for their contributions, including Dick McCray, Ian Stevens, Yuan-Kuen Ko, Julian Krolik, Tom Bridgman, James Peachey, Bryan Irby, Bill Pence, Randall Dannen and Abdu Zoghbi. Crucial work on the atomic data was done by Manuel Bautista, Patrick Palmeri, Claudio Mendoza, Javier Garcia, and Mike Witthoeft.

1.3 Feedback and Bug Reports

XSTAR is under active development and we appreciate bug reports, requests for additional features and general feedback from the community. Please use the HEASARC Feedback form <https://heasarc.gsfc.nasa.gov/cgi-bin/Feedback> and select "xstar" as mailing list.

Timothy Kallman, & timothy.r.kallman@nasa.gov

Ralf Ballhausen, & ralf.ballhausen@nasa.gov

CHAPTER

TWO

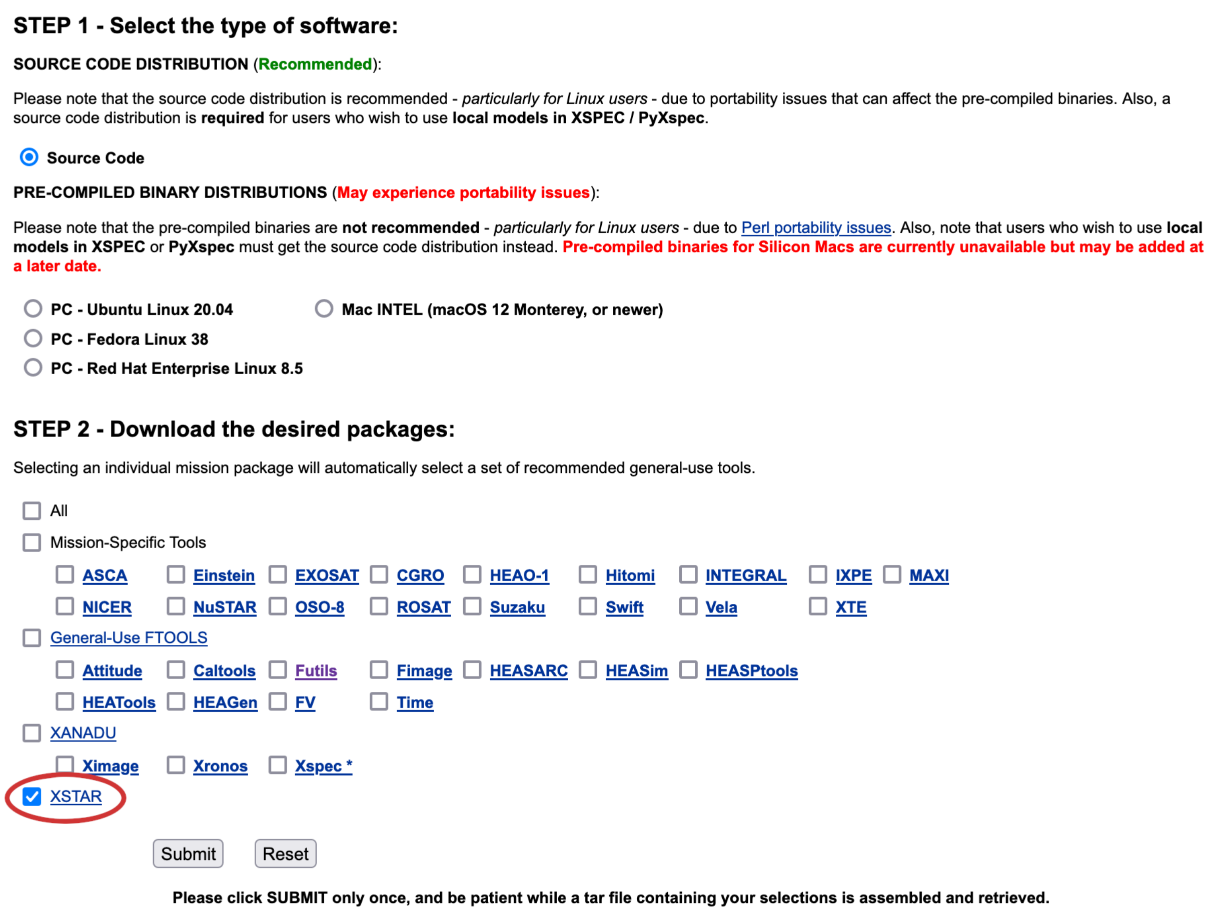
DOWNLOAD AND INSTALLATION

2.1 XSTAR as Part of the HEASOFT

XSTAR is included in the FTOOLS package of HEASoft. Instructions for download and installation of the HEASoft are available at

http://heasarc.gsfc.nasa.gov/lheasoft

Make sure to check "xstar" when downloading only selected HEASoft packages:



STEP 1 - Select the type of software:

SOURCE CODE DISTRIBUTION (Recommended):

Please note that the source code distribution is recommended - *particularly for Linux users* - due to portability issues that can affect the pre-compiled binaries. Also, a source code distribution is **required** for users who wish to use **local models in XSPEC / PyXspec**.

[✓ **Source Code**]

PRE-COMPILED BINARY DISTRIBUTIONS (May experience portability issues):

Please note that the pre-compiled binaries are **not recommended** - *particularly for Linux users* - due to Perl portability issues. Also, note that users who wish to use **local models in XSPEC** or **PyXspec** must get the source code distribution instead. **Pre-compiled binaries for Silicon Macs are currently unavailable but may be added at a later date.**

PC - Ubuntu Linux 20.04 **Mac INTEL (macOS 12 Monterey, or newer)**
PC - Fedora Linux 38
PC - Red Hat Enterprise Linux 8.5

STEP 2 - Download the desired packages:

Selecting an individual mission package will automatically select a set of recommended general-use tools.

All

Mission-Specific Tools

ASCA **Einstein** **EXOSAT** **CGRO** **HEAO-1** **Hitomi** **INTEGRAL** **IXPE** **MAXI**
NICER **NuSTAR** **OSO-8** **ROSAT** **Suzaku** **Swift** **Vela** **XTE**

General-Use FTOOLS

Attitude **Caltools** **Futils** **Fimage** **HEASARC** **HEASim** **HEASPtools**
HEATools **HEAGen** **FV** **Time**

XANADU

Ximage **Xronos** **Xspec ***

[✓ XSTAR]

Submit Reset

Please click SUBMIT only once, and be patient while a tar file containing your selections is assembled and retrieved.

We expect that the majority of users will obtain XSTAR through HEASoft.

2.2 XSTAR as a Standalone Package

The standalone version of xstar is available as a gzipped and tarred file on the XSTAR website

The source code is available along with compiled executables for several machine architectures.

The installation is the same as for the full heasoft:

http://heasarc.gsfc.nasa.gov/docs/software/lheasoft/install.html

In more detail, you follow these instructions, but obviously the tarfile is named xstar22src.tar.gz rather than what is given in the heasoft instructions, and the directory that appears when it is untarred is called heasoft, not heasoft-6.9. A condensed version of what you need to do is as follows:

2) In the directory in which you want to install the software, unpack the file you downloaded in step 1 using e.g.:

   ```
   gunzip -c xstar22src.tar.gz | tar xf -
   ```

   This will create a heasoft/ directory containing the software distribution.

3) Configure the software for your platform (necessary for both binary and source downloads):

   ```
   cd heasoft/BUILD_DIR/
   ```

   and run the main configure script, which will probe your system for libraries, header files, compilers, etc., and then generate the main Makefile.

   To produce a default configuration, the configure script may simply be invoked by (we recommend capturing the screen output from configure as below):

   ```
   ./configure >& config.out &     (C Shell variants)
   ./configure > config.out 2>&1 & (Bourne Shell variants)
   ```

4) Start the build process. We strongly recommend that you capture all output into a log file. Then, if you need to report a problem, please send us the ENTIRE log file. And since it may take some time to run (from minutes to hours, depending on the speed of your system) we recommend that you build it in the background:

   ```
   make >& build.log &     (C Shell variants)
   make > build.log 2>&1 & (Bourne Shell variants)
   ```

   To check on the build progress in real-time (if you wish) try:

   ```
   tail -f build.log
   ```

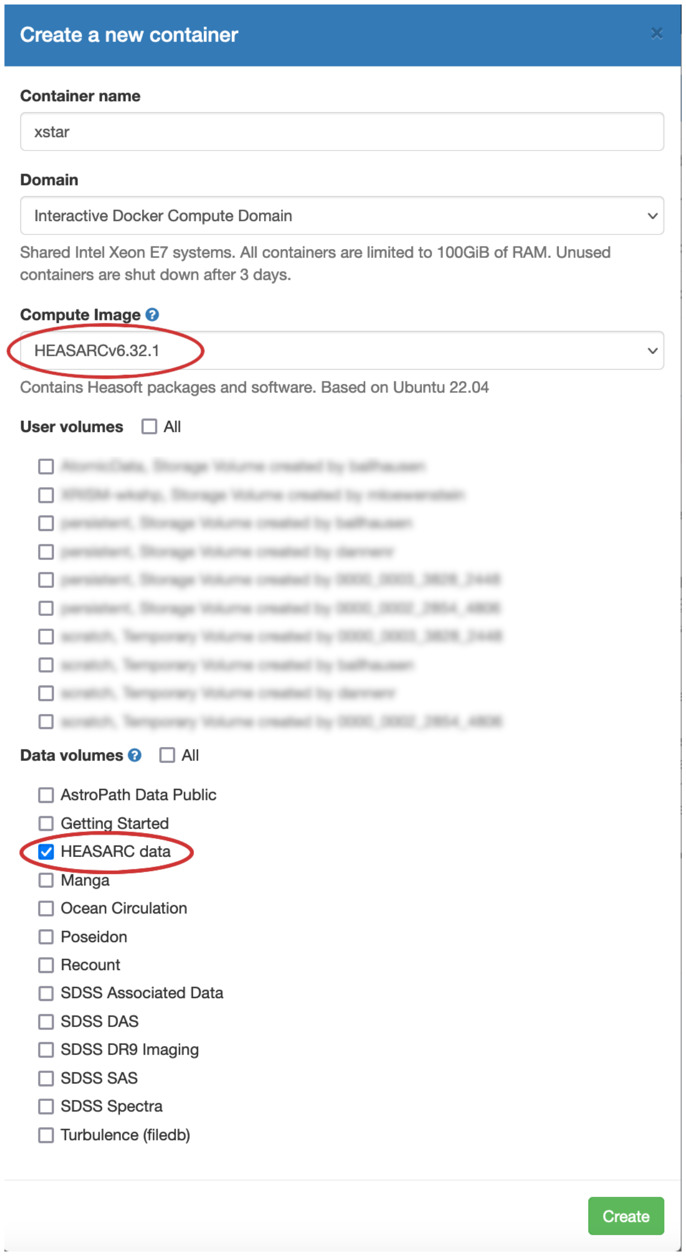
6) Perform the final installation of the executables, libraries, help files, calibration data, perl scripts, etc, by executing:

   ```
   make install >& install.log &     (C Shell variants)
   make install > install.log 2>&1 & (Bourne Shell variants)
   ```

   This will create an appropriately-named system-dependent directory, e.g. sparc-sun-solaris2.9/, either under heasoft/ or, if you specified a prefix argument to configure, in the directory you selected at that time.

2.3 XSTAR on SciServer

Users who do not wish to install HEASoft locally can run XSTAR on SciServer. Make sure to select a HEASARC computing image and mount the HEASARC Data volume.



Create a new container

Container name

xstar

Domain

Interactive Docker Compute Domain

Shared Intel Xeon E7 systems. All containers are limited to 100GiB of RAM. Unused containers are shut down after 3 days.

Compute Image

HEASARCv6.32.1

Contains Heasoft packages and software. Based on Ubuntu 22.04

User volumes ☐ All

[redacted]

Data volumes ☐ All

- ☐ AstroPath Data Public
- ☐ Getting Started
- ✓ HEASARC data
- ☐ Manga
- ☐ Ocean Circulation
- ☐ Poseidon
- ☐ Recount
- ☐ SDSS Associated Data
- ☐ SDSS DAS
- ☐ SDSS DR9 Imaging
- ☐ SDSS SAS
- ☐ SDSS Spectra
- ☐ Turbulence (filedb)

2.4 Subroutine XSTAR

This is a version of XSTAR which retains most of the functionality of the full code, but which provides a framework whereby XSTAR can be called as a subroutine from another program, or whereby XSTAR can be applied to situations which do not employ the standard assumptions concerning, e.g. the gas density distribution, geometry, or time-steady behavior. This consists of a large fortran file xstarsub.f90 containing all of the subroutines employed by the standard XSTAR, together with two primary subroutines: xstarsetup and xstarcalc. As implied by the names, xstarsetup is intended to be called once at the beginning of a calculation and handles reading of input data and initialization; xstarcalc calculates the physical conditions at one point in a photoionized gas and returns level populations, emissivities, opacities, etc. In addition there is a calling program which is intended to illustrate the use of these subroutines. Subroutine XSTAR also requires the the fits data file, atdb.fits, and must be linked to the cfitsio subroutine library. All are available via ftp from https://heasarc.gsfc.nasa.gov/FTP/software/plasma_codes/xstar/subroutine

CHAPTER

THREE

A WALKTHROUGH OF XSTAR

3.1 Spherical photoionized cloud

In this section we illustrate how to use XSTAR's interface. Once XSTAR is installed and configured, at the unix prompt, type:

```
unix> xstar
```

By invoking XSTAR with this simple command, you will be prompted for a series of physical and control parameters for the simulation.

The input parameters that must always be specified are: The model (initial) temperature, density, spectrum shape, ionizing luminosity, column density, ionization parameter, and elemental abundance table. Definitions for these and the units assumed are described in detail in chapter *User Input to XSTAR*. All input parameters have default values, selected by pressing return at the prompt, thus it is possible to simply start XSTAR as above to invoke the default model.

In this example we model a spherical, constant density cloud with a source at its center. The cloud is optically thin. The source luminosity is 10^{32} erg s^{-1}. The ionization parameter at the inner edge of the cloud is $\log(\xi) = 5$. The ionizing spectrum is a power law with energy index -1. We assume all elemental abundances are solar but exclude Li, Be, B. The total column density is 10^{19} cm^{-1}. In this example, we require thermal equilibrium by expicitely setting the hidden parameter `niter` to 99 and spherical symmetry by setting the covering fraction `cfrac` to 1.

Call XSTAR from the command line:

```
unix> xstar niter=99 cfrac=1
xstar version 2.59cj
temperature (/10**4K) (0.:1.e4) [400.] 1000.
density (cm**-3) (0.:1.e21) [1.e+4] 1E+6
spectrum type?[pow]
radiation temperature or alpha?[-1.]
luminosity (/10**38 erg/s) (0.:1.e10) [1.e-6]
column density (cm**-2) (0.:1.e25) [1.E17] 1E19
log(ionization parameter) (erg cm/s) (-10.:+10.) [5.]
abundance table[xdef]
model name[XSTAR Default] 'filled sphere'
```

Alternatively, XSTAR can called directly with command-line parameters to avoid prompting.

```
xstar cfrac=1 temperature=1000. density=1.E+6 spectrum='pow ' \
trad=-1. rlrad38=1.E-6 column=1.E+19 rlogxi=5. abundtbl='xdef' \
modelname='filled sphere' niter=99
```

The terminal output reports the progress of the XSTAR run. Each line represents a spatial zone. For each spatial zone, XSTAR prints the radius, stepsize, column density, ionization parameter, electron fraction, temperature, heating/cooling balance, optical depth in forward and reverse direction, and number of iterations required to reach thermal equilibrium.

```
xstar version 2.59cj

pass number=           1          -1
  log(r) delr/r log(N) log(xi) x_e   log(n) log(t) h-c(%) h-c(%) log(tau)
                                                               fwd    rev
  10.50 -36.00 -10.00   5.00   1.21   6.00   7.77  -0.00   0.00 -10.00 -10.00 15
  10.50 -36.00 -10.00   5.00   1.21   6.00   7.77  -0.00  -0.00 -10.00 -10.00  1
  10.62  -0.60  16.02   4.75   1.21   6.00   7.75   0.00  -0.00 -10.00 -10.00  5
  [...]
  12.98  -0.00  18.98   0.04   1.20   6.00   4.33   0.01   0.12  -2.39 -10.00 14
  13.00  -0.00  18.99   0.01   1.20   6.00   4.32  -0.01   0.12  -2.32 -10.00 13
  13.00  -0.00  19.00  -0.00   1.20   6.00   4.32  -0.01   0.12  -2.30 -10.00 13
 final print:           0
xstar: Prepping to write spectral data
xstar: Done writing spectral data
total time   1663.5743254758418
```

Upon completion, the following output files are produced:

```
unix> ls
xout_abund1.fits
xout_cont1.fits
xout_lines1.fits
xout_rrc1.fits
xout_spect1.fits
xout_step.log
```

A detailed description of the output files is given in *XSTAR output*. The ASCII file "xout_step.log" contains mostly diagnostic information. Information about ion fractions, line and continuum emission and absorption as well as spectral information is stored in several FITS files.

In this example we are interested in the total optical depth of the cloud as a function of energy. This information can be obtained from the spectral file "xout_spect1.fits". The file structure can for example be inspected with the FTOOL `fstruct`:

```
unix> fstruct xout_spect1.fits


  No. Type     EXTNAME      BITPIX Dimensions(columns)      PCOUNT  GCOUNT

0  PRIMARY                 16     0                           0    1
1  BINTABLE PARAMETERS      8     66(5) 56                    0    1

   Column Name                Format     Dims       Units     TLMIN  TLMAX
   1 index                      1I
   2 parameter                  20A
   3 value                      1E
   4 type                       10A
   5 comment                    30A

```

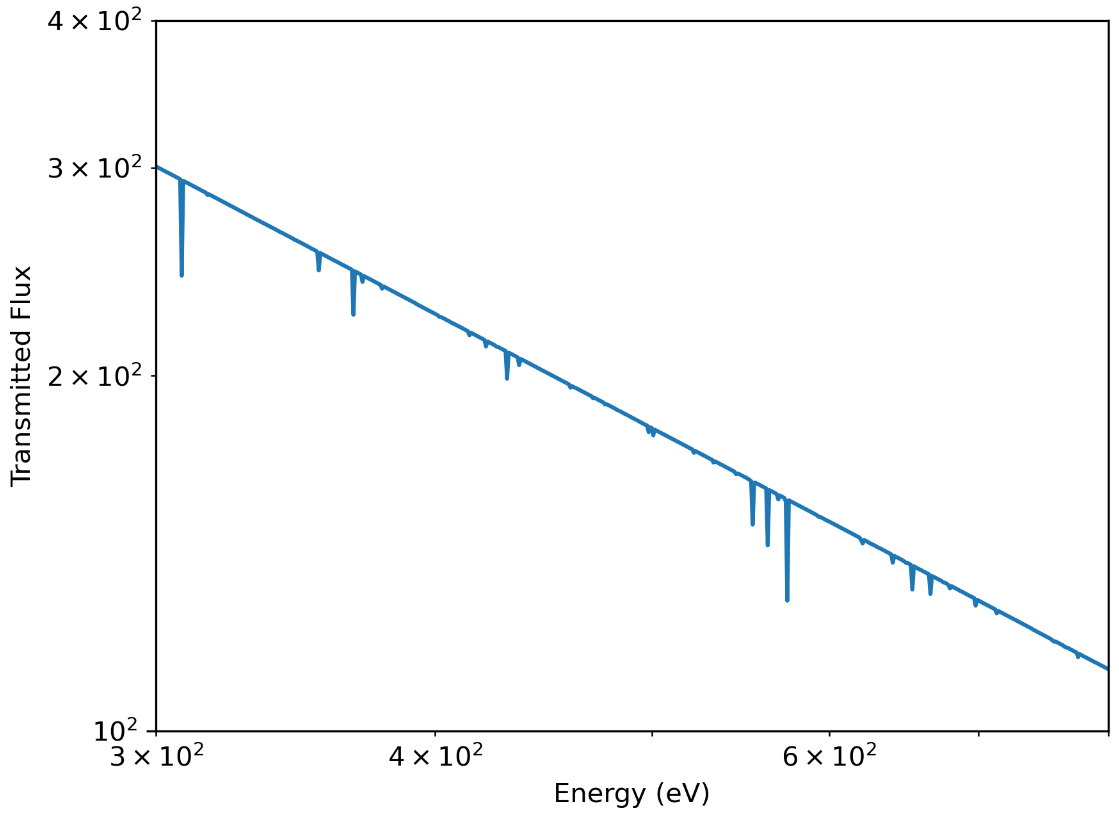
(continues on next page)

(continued from previous page)

```
2  TABLE    XSTAR_SPECTRA   8      69(5) 9999                  0    1

   Column Name                 Format     Dims       Units     TLMIN  TLMAX
   1 energy                      E13.5               eV
   2 incident                    E13.5               erg/s/erg
   3 transmitted                 E13.5               erg/s/erg
   4 emit_inward                 E13.5               erg/s/erg
   5 emit_outward                E13.5               erg/s/erg
```

FITS files can be processed with various programming languages and tools. In this example we are interested in the transmitted spectrum. Here we use Python to plot the transmitted spectrum in 300 eV–800 eV range.

```
>>> import matplotlib.pyplot as plt
>>> from astropy.io import fits
>>> import numpy as np
>>>
>>> hdul = fits.open('xout_spect1.fits')
>>> en = hdul[2].data['energy']
>>> trns = hdul[2].data['transmitted']
>>> hdul.close()
>>>
>>> plt.plot(en, trns)
>>> plt.xscale("log")
>>> plt.yscale("log")
>>> plt.xlim(300,800)
>>> plt.ylim(100, 400)
>>> plt.xlabel("Energy (eV)")
>>> plt.ylabel("Transmitted Flux")
>>> plt.show()
```


4×10^2
3×10^2
2×10^2
10^2
Transmitted Flux
3×10^2
4×10^2
6×10^2
Energy (eV)

CHAPTER

FOUR

USER INPUT TO XSTAR

4.1 User Interface and Parameter Formats

For most applications, input parameters to XSTAR are parsed as command line arguments. If invoking XSTAR from the command line without further parameters, the user is prompted for a series of input values.

```
unix> xstar
  xstar version 2.59b
  covering fraction (0.:1.) [1.]
  temperature (/10**4K) (0.:1.e4) [400.]
  constant pressure switch (1=yes, 0=no) (0:1) [0]
  pressure (dyne/cm**2) (0.:1.) [0.03]
  density (cm**-3) (0.:1.e21) [1.e+4] 1.e8
  spectrum type?[pow]
  [...]
```

The format of each line is: *description (min value : max value) [default value]*. In this example we use default parameters, except for the density which we set to 1.0E+8.

Alternatively, parameter values can be specified directly on the command line.

```
unix> xstar cfrac=0 temperature=1000
  xstar version 2.59b
  constant pressure switch (1=yes, 0=no) (0:1) [0]
  pressure (dyne/cm**2) (0.:1.) [0.03]
  density (cm**-3) (0.:1.e21) [1.0e8]
  [...]
```

Now XSTAR only starts prompting at the constant pressure switch because covering fraction (`cfrac`) and temperature (`temperature`) are already defined. A complete list of parameters is given in Table *Input Parameter Summary*. Specifying all parameters will avoid prompting entirely and is particularly useful for scripting.

Internally, input parameters to XSTAR are handled through an IRAF-style interface, XPI, developed for the FTOOLS suite of programs. This interface expects a parameter file, a simple text file, called “xstar.par” at the location the user’s `$PFILES` environment variable points to. This environment variable is set upon initialization of HEAsoft and works similar to the `PATH` variable in that it may point to several locations (usually to `$HOME/pfiles:$HEADAS/syspfiles`). On the very first call of XSTAR, a local copy of a template parameter file is created from the system-wide default parameter file..

The parameter file contains information about which parameters exist and what their description, formats, limits and default values are. Upon inspection of the parameter file will users will note that the parameter file contains several more parameters than they get prompted. We refer to these as *hidden* parameters. Whether a parameter is hidden or prompted is an attribute defined in the parameter file itself. With only few exceptions, the hidden parameters are rather technical parameters that allow for controlling the code behavior at a very fundamental level. **Changing these parameters requires a understanding of the operation of the code. Users should therefore use great caution when modifying hidden parameters.**

Another feature the user may notice is that for some of the prompted parameters, the default values get updated when the parameter is changed. The updated values are stored in the user's local copy of the parameter file. In order to undo all changes and start with a fresh set of default parameter, the user may simply delete the *local* copy of the parameter file which will lead XSTAR to revert to the system-wide default parameter file.

Obviously, the parameter file can also be used to save the complete setup of an XSTAR run for later reproducibility. However, there are certain caveats associated with this procedure. For example, the default values of hidden parameters have changed among different versions of XSTAR. Also for remote or large scale calculations, the user may want to avoid prompting completely, which also requires modifications to the parameter modes. The recommended way to save XSTAR calls is therefore in executable shell scripts that explicitly specify all prompted and other parameters of interest. For example, the following bash script will run XSTAR with default parameters without prompting.

```
#!/bin/bash
xstar cfrac=1.0 temperature=400 lcpres=0 pressure=0.03 \
density=1.e4 spectrum='pow' trad=-1. rlrad38=1.e-6 \
column=1.e17 rlogxi=5 abundtbl='xdef'
```

4.2 Input Parameter Summary

Here we list all XSTAR parameters in the order of the parameter file.

Table 1: Input Parameter Summary

Prompt string	Parameter name	Description	Default
Temperature	`temperature`	Gas Temperature (10^4 K)	400
Pressure	`pressure`	Pressure in dynes cm^{-2}	0.03
Density	`density`	Density in cm^{-3}	1.0E+04
Spectrum	`spectrum`	Define the input spectrum	pow
Spectrum File	`spectrum_file`	Name of spectrum file	spct.dat
Spectrum Units	`spectun`	0=energy, 1=photons	0
Radiation Temperature or Alpha	`trad`	10^7 K or unitless (E^α)	-1.0
Luminosity	`rlrad38`	Luminosity in 10^{38} erg s^{-1}	1.0E-06
Column Density	`column`	Column density in cm^{-2}	1.0E+17
log(ionization parameter)	`rlogxi`	log(ξ) (erg cm/s) or $log(\Xi)$	5.0
Abundance table	`abundtbl`	Name of abundance table	xdef
(Hydrogen abundance)	`habund`	relative H abundance	1.0
(Helium abundance)	`heabund`	relative He abundance	1.0
(Lithium abundance)	`liabund`	relative Li abundance	0.0
(Beryllium abundance)	`beabund`	relative Be abundance	0.0
(Boron abundance)	`babund`	relative B abundance	0.0
(Carbon abundance)	`cabund`	relative C abundance	1.0
(Nitrogen abundance)	`nabund`	relative N abundance	1.0
(Oxygen abundance)	`oabund`	relative O abundance	1.0
(Fluorind abundance)	`fabund`	relative F abundance	1.0
(Neon abundance)	`neabund`	relative Ne abundance	1.0
(Magnesium abundance)	`mgabund`	relative Mg abundance	1.0
(Aluminum abundance)	`alabund`	relative Al abundance	1.0
(Silicon abundance)	`siabund`	relative Si abundance	1.0

continues on next page

Table 1 – continued from previous page

Prompt string	Parameter name	Description	Default
(Phosphorus abundance)	`pabund`	relative P abundance	1.0
(Sulfur abundance)	`sabund`	relative S abundance	1.0
(Chlorine abundance)	`clabund`	relative Cl abundance	1.0
(Argon abundance)	`arabund`	relative Ar abundance	1.0
(Potassium abundance)	`kabund`	relative K abundance	1.0
(Scandium abundance)	`scabund`	relative Sc abundance	1.0
(Titanium abundance)	`tiabund`	relative Ti abundance	1.0
(Vanadium abundance)	`vabund`	relative V abundance	1.0
(Chromium abundance)	`crabund`	relative Cr abundance	1.0
(Manganese abundance)	`mnabund`	relative Mn abundance	1.0
(Iron abundance)	`feabund`	relative Fe abundance	1.0
(Cobalt abundance)	`coabund`	relative Co abundance	1.0
(Nickel abundance)	`niabund`	relative Ni abundance	1.0
(Zinc abundance)	`znabund`	relative Zn abundance	1.0
(Copper abundance)	`cuabund`	relative Cu Abundance	1.0
Model Name	`modelname`	Name used for this run	XSTAR Default
(Covering fraction)	`cfrac`	Covering Fraction	1.0
(Constant Pressure Switch)	`lcpres`	1=yes, 0=no	0
(nsteps)	`nsteps`	Number of steps	3
(niter)	`niter`	Number of iterations	0
(write switch)	`lwrite`	(1=yes, 0=no)	0
(print switch)	`lprint`	(1=yes, 0=no)	0
(emult)	`emult`	Courant multiplier	0.5
(taumax)	`taumax`	τ_{max} for courant step	5.0
(xeemin)	`xeemin`	minimum electron fraction	0.1
(critf)	`critf`	critical ion abundance	1.0E-07
(vturbi)	`vturbi`	turbulent velocity (km/s)	1.0
(radexp)	`radexp`	density distribution power law index	0.0
(ncn2)	`ncn2`	number of continuum bins	9999
(loopcontrol)	`loopcontrol`	loop control (0=standalone)	0
(npass)	`npass`	number of passes	1
(mode)	`mode`	Paramter Interface mode	ql

4.3 Detailed Description of XSTAR Parameters

4.3.1 Temperature (temperature)

Gas temperature in units of 10^4 K. It is important to note that in typical photoionization models, the temperature is not an independent parameter, but calculated by the code by calculating thermal equilibrium. XSTAR allows the user to specify whether thermal equilibrium is calculated or not via the `niter` parameter, which consequently affects the meaning of the temperature input parameter.

If the parameter `niter` is set to 0 then the input temperature is fixed at this value. This is be useful to calculate collision-dominated plasmas or to speed up calculations in general. *Be aware that in this case the specified plasma is likely not int thermal equilibrium.*

If the parameter `niter` is set to a non-zero value, the input temperature is used as a first guess in calculating the thermal equilibrium value.

Finally, this temperature is used to calculate the initial guess for the gas density, $n = P/(kT)$, for the constant pressure case.

The input format is a float. The default parameter value is 400.

4.3.2 Pressure (pressure)

Model pressure in dynes cm^{-2}. Note that this quantity represents the full isotropic pressure (neutral atoms + ions + electrons + trapped line radiation) instead of just the pressure due to hydrogen atoms and protons. Whether or not this quantity is held fixed is determined by the value of the constant pressure switch `lcpres`. If the pressure is constant then the appropriate definition of ionization parameter (Ξ) is adopted. In the constant density case, `lcpres=0`, then this quantity is ignored.

The input format is a float. The default value is 0.03.

4.3.3 Density (density)

Model gas density, n in cm^{-3}. This is actually the hydrogen nucleus density, so that, e.g., the total particle density in a fully-ionized plasma with solar abundances is $2.3n$. Whether or not this quantity is held fixed is determined by the value of the constant pressure switch `lcpres`. If the density is constant then the appropriate definition of ionization parameter (ξ) is adopted. The default case is constant density. Optionally, the density can be made radius-dependent. See *Radius Exponent (radexp) - hidden* for details.

The input format is a float. The default value is 1.0E+04.

4.3.4 Spectrum (spectrum)

This parameter allows to define the incident spectral energy distribution. The user can select between powerlaw, black body, and bremsstrahlung models, or alternatively provide a tabulated spectrum. *Note that numerical problems can arise if the radiation field is zero throughout a significant energy range, because then the photoionization rates for some ions may be zero, and these appear in the denominator of the equations for the ionization balance.* The parameter format is a string. The following options exist:

- **Black body spectrum**

  Black body spectrum, defined as

  $$A(\varepsilon) = \varepsilon^3/(\exp(\varepsilon/kT) - 1)$$

  where T is temperature in units of 10^7 K.

  The black body spectrum is selected by `spectrum=bbody`.

- **Bremsstrahlung spectrum**

  Thermal bremsstrahlung spectrum, including gaunt factors, but not including $e - e$ bremsstrahlung. The input parameter for this model is plasma temperature in keV.

  The bremsstrahlung spectrum is selected by `spectrum=bremss`.

- **Powerlaw spectrum**

  Simple photon power law, defined as

  $$A(\varepsilon) = \varepsilon^\alpha$$

  where α is the energy index of power law.

The user is cautioned that simple power laws can have unintended consequences owing to the fact that they are automatically extrapolated to the lowest (0.1 eV) and highest (1 MeV) energies employed in the calculation. This can cause processes such as stimulated recombination and Compton cooling to dominate the model results, and may not represent a physically realistic result. These effects can be avoided by the use of a simple file spectrum as demonstrated below.

The powerlaw spectrum is selected by `spectrum=pow`.

- **File spectrum**

  Fluxes and energies are read from a text file, with name given by the `spectrum_file` keyword. The format of the file is as follows: the first line must contain the integer number of (energy, flux) pairs; each of the the remaining lines contains one (energy, flux) pair, with energy in eV and flux in energy units. These values will be interpolated onto the energy grid used internally by XSTAR using logarithmic interpolation. *Absolute fluxes are arbitrary as XSTAR will re-normalize the spectrum according to the specified luminosity.* An example of a file which results in a ε^{-1} spectrum power law spectrum between 0.1 Ry and 1000 Ry (and zero elsewhere) is as follows:

```
006
1.0000E-03 1.E-10
1.3590E+00 1.E-10
1.3598E+00 1.E+11
1.3598E+05 1.E+06
1.3600E+05 1.E-10
2.0000E+05 1.E-10
```

  As already states, be aware that small numbers (less than 1.E-30, say), may result in undesired results owing to the limitations of many machines in the range of exponents. And remember that it is not necessary to use actual physical units in specifying the input spectrum (except to distinguish between photon and energy fluxes) since the entire spectrum is re-normalized to conform to the luminosity specified.

  The file spectrum is selected by `spectrum=file`.

The input format is a string. The default parameter value is “pow” (powerlaw).

4.3.5 Spectrum File (spectrum_file)

If the “file” option is chosen for the spectrum type, the user must provide a text file of the spectrum in the current working directory. The first line of the text file must be the number of energies listed in the table. The remaining lines are the energy channel (in eV) and the flux in units of photons $\mathrm{cm^{-2}\,s^{-1}\,erg^{-1}}$ or $\mathrm{erg\,cm^{-2}\,s^{-1}\,erg^{-1}}$ (see *Spectrum Units (spectun)*). Again, remember that it is not necessary to use actual physical units in specifying the input spectrum (except to distinguish between photon and energy fluxes) since the entire spectrum is re-normalized to conform to the luminosity specified.

The input format is string. The default parameter value is “spct.dat”.

4.3.6 Spectrum Units (spectun)

The appropriate units for the spectrum file specified above (1 = photons $\mathrm{cm^{-2}\,s^{-1}\,erg^{-1}}$, 0 = $\mathrm{erg\,cm^{-2}\,s^{-1}\,erg^{-1}}$).

New in version 221bn18 is a feature which allows reading in of table spectra in units of log 10(F_ε), where F_ε has units $\mathrm{erg\,cm^{-2}\,s^{-1}\,erg^{-1}}$. This requires that the spectun input parameter be set to 2.

The input format is an integer. The default parameter value is 0 (energy flux)

4.3.7 Radiation Temperature or Alpha (trad)

This parameter pulls double duty, used to enter the radiation temperature in units of 10^7 K in the case of a blackbody or in units of keV for bremsstrahlung input model. It also is used to input the power-law index (in energy), α, in the case of a power-law model. Note that α is defined as in $L_\varepsilon \sim \varepsilon^\alpha$ so generally α will be *less* than zero (this is the opposite of the convention used by xspec). XSTAR always works with specific luminosity L_ε in units $\mathrm{erg\,s^{-1}\,erg^{-1}}$, and never uses $\varepsilon L_\varepsilon$ or νF_ν, for example.

The input format is a float. The default parameter value is -1.

4.3.8 Luminosity (luminosity)

Model luminosity integrated between 1 and 1000 Ry in units of $10^{38}\,\mathrm{erg\,s^{-1}}$. In combination with the ionization parameter and the density, this parameter determines the geometrical size of the cloud.

The input format is a float. The default parameter value is 1.0E-06.

4.3.9 Column density (column)

Model column density, N in units of $\mathrm{cm^{-2}}$. This quantity is used in calculation of the thickness of the model slab according to

$$N = \int_{R_{\rm in}}^{R_{out}} n\,dr\ ,$$

where n is the density or an estimate based on pressure and temperature initial values. The model calculation terminates when this value is reached. For the constant density case, the calculation of the slab thickness simplifies to $R_{\max} = N/n$.

The input format is a float. The default parameter value is 1.0E+21.

4.3.10 Log of the ionization parameter (rlogxi)

Initial value of the log (base 10) of the model ionization parameter at the innermost shell. If the density is held constant, the Tarter *et al.* [49] form is used:

$$\xi = \frac{L}{nR^2}\ .$$

If the pressure is held constant, a version of the Krolik *et al.* [36] form is used:

$$\Xi = \frac{L}{4\pi c R^2 P}\ .$$

Note that this differs from the original form by using the full isotropic pressure (neutral atoms + ions + electrons + trapped line radiation) instead of just the pressure due to hydrogen atoms and protons. This quantity is used in calculating the radius of the innermost edge of the shell by inverting the parameter definition.

The input format is a float. The default parameter value is 5.

4.3.11 Abundances

Atomic abundances for elements H though Zn are entered as a predefined abundance table. Additionally the user can change the value of each element relative to this table. For abundance tables, the user can select between XSTAR default abundances (close to those defined in Grevesse *et al.* [22]) and all abundance tables implemented in xspec. The abundance table is specified as a 4 character string for the parameter `abundtbl`. The full list of tables is

- xdef (xstar default)
- angr (Anders and Grevesse [4])
- aspl (Asplund *et al.* [6])
- feld (Feldman [18])
- aneb (Anders and Ebihara [3])
- grsa (Grevesse and Sauval [23])
- wilm (Wilms *et al.* [52])
- lodd (Lodders [39])
- lpgp (Lodders *et al.* [38])
- lpgs (Lodders *et al.* [38])

All abundance tables listed below. The default parameter is "xdef".

Table 2: Abundance Tables

Z	El	xdef	angr	aspl	feld	aneb
1	H	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00
2	He	1.00E-01	9.77E-02	8.51E-02	9.77E-02	8.01E-02
3	Li	1.00E-10	1.45E-11	1.12E-11	1.26E-11	2.19E-09
4	Be	1.00E-10	1.41E-11	2.40E-11	2.51E-11	2.87E-11
5	B	1.00E-10	3.98E-10	5.01E-10	3.55E-10	8.82E-10
6	C	3.70E-04	3.63E-04	2.69E-04	3.98E-04	4.45E-04
7	N	1.10E-04	1.12E-04	6.76E-05	1.00E-04	9.12E-05
8	O	6.80E-04	8.51E-04	4.90E-04	8.51E-04	7.39E-04
9	F	3.98E-08	3.63E-08	3.63E-08	3.63E-08	3.10E-08
10	Ne	2.80E-05	1.23E-04	8.51E-05	1.29E-04	1.38E-04
11	Na	1.78E-06	2.14E-06	1.74E-06	2.14E-06	2.10E-06
12	Mg	3.50E-05	3.80E-05	3.98E-05	3.80E-05	3.95E-05
13	Al	2.45E-06	2.95E-06	2.82E-06	2.95E-06	3.12E-06
14	Si	3.50E-05	3.55E-05	3.24E-05	3.55E-05	3.68E-05
15	P	3.31E-07	2.82E-07	2.57E-07	2.82E-07	3.82E-07
16	S	1.60E-05	1.62E-05	1.32E-05	1.62E-05	1.89E-05
17	Cl	3.98E-07	3.16E-07	3.16E-07	3.16E-07	1.93E-07
18	Ar	4.50E-06	3.63E-06	2.51E-06	4.47E-06	3.82E-06
19	K	8.91E-08	1.32E-07	1.07E-07	1.32E-07	1.39E-07
20	Ca	2.10E-06	2.29E-06	2.19E-06	2.29E-06	2.25E-06
21	Sc	1.66E-09	1.26E-09	1.41E-09	1.48E-09	1.24E-09
22	Ti	1.35E-07	9.77E-08	8.91E-08	1.05E-07	8.82E-08
23	V	2.51E-08	1.00E-08	8.51E-09	1.00E-08	1.08E-08
24	Cr	7.08E-07	4.68E-07	4.37E-07	4.68E-07	4.93E-07
25	Mn	2.51E-07	2.45E-07	2.69E-07	2.45E-07	3.50E-07
26	Fe	2.50E-05	4.68E-05	3.16E-05	3.24E-05	3.31E-05

continues on next page

Table 2 – continued from previous page

Z	El	xdef	angr	aspl	feld	aneb
27	Co	1.26E-07	8.32E-08	9.77E-08	8.32E-08	8.27E-08
28	Ni	2.00E-06	1.78E-06	1.66E-06	1.78E-06	1.81E-06
29	Cu	3.16E-08	1.62E-08	1.55E-08	1.62E-08	1.89E-08
30	Zn	1.58E-08	3.98E-08	3.63E-08	3.98E-08	4.63E-08

Table 3: Abundance Tables

Z	El	grsa	wilm	lodd	lpgp	lpgs
1	H	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00
2	He	8.51E-02	9.77E-02	7.92E-02	8.41E-02	9.69E-02
3	Li	1.26E-11	0.00E+00	1.90E-09	1.26E-11	2.15E-09
4	Be	2.51E-11	0.00E+00	2.57E-11	2.40E-11	2.36E-11
5	B	3.55E-10	0.00E+00	6.03E-10	5.01E-10	7.26E-10
6	C	3.31E-04	2.40E-04	2.45E-04	2.45E-04	2.78E-04
7	N	8.32E-05	7.59E-05	6.76E-05	7.24E-05	8.19E-05
8	O	6.76E-04	4.90E-04	4.90E-04	5.37E-04	6.06E-04
9	F	3.63E-08	0.00E+00	2.88E-08	3.63E-08	3.10E-08
10	Ne	1.20E-04	8.71E-05	7.41E-05	1.12E-04	1.27E-04
11	Na	2.14E-06	1.45E-06	1.99E-06	2.00E-06	2.23E-06
12	Mg	3.80E-05	2.51E-05	3.55E-05	3.47E-05	3.98E-05
13	Al	2.95E-06	2.14E-06	2.88E-06	2.95E-06	3.27E-06
14	Si	3.55E-05	1.86E-05	3.47E-05	3.31E-05	3.86E-05
15	P	2.82E-07	2.63E-07	2.88E-07	2.88E-07	3.20E-07
16	S	2.14E-05	1.23E-05	1.55E-05	1.38E-05	1.63E-05
17	Cl	3.16E-07	1.32E-07	1.82E-07	3.16E-07	2.00E-07
18	Ar	2.51E-06	2.57E-06	3.55E-06	3.16E-06	3.58E-06
19	K	1.32E-07	0.00E+00	1.29E-07	1.32E-07	1.45E-07
20	Ca	2.29E-06	1.58E-06	2.19E-06	2.14E-06	2.33E-06
21	Sc	1.48E-09	0.00E+00	1.17E-09	1.26E-09	1.33E-09
22	Ti	1.05E-07	6.46E-08	8.32E-08	7.94E-08	9.54E-08
23	V	1.00E-08	0.00E+00	1.00E-08	1.00E-08	1.11E-08
24	Cr	4.68E-07	3.24E-07	4.47E-07	4.37E-07	5.06E-07
25	Mn	2.45E-07	2.19E-07	3.16E-07	2.34E-07	3.56E-07
26	Fe	3.16E-05	2.69E-05	2.95E-05	2.82E-05	3.27E-05
27	Co	8.32E-08	8.32E-08	8.13E-08	8.32E-08	9.07E-08
28	Ni	1.78E-06	1.12E-06	1.66E-06	1.70E-06	1.89E-06
29	Cu	1.62E-08	0.00E+00	1.82E-08	1.62E-08	2.09E-08
30	Zn	3.98E-08	0.00E+00	4.27E-08	4.17E-08	5.02E-08

In addition to specifying the abundance table, the user can also change the abundance of individual elements, relative to that table. This is particularly important for turning off individual elements. The relative element abundance parameters are `habund`, `heabund`, `liabund`, . . . , `niabund`, `cuabund`, and `znabund`.

Note that the abundance of an element *relative to hydrogen* depends both on the abundance table and the relative abundance parameter.

The input format for the abundance table is a string. The default is “xdef”.

The input format for the relative abundances are floats. The default values are 1.0 for all elements except for Li, Be, and B, which are 0.

4.3.12 Model Name (modelname)

Model name as 80 character string. The model name is recorded in all output files but does not affect the XSTAR run in any other way.

The input format is a string. The default parameter value "XSTAR Default".

4.3.13 Covering Fraction (cfrac) - *hidden*

This parameter determines whether the geometry is a complete sphere or covers only part of the continuum source. In the former case, photons escaping the cloud in the 'inward' direction are assumed to reenter the cloud at the inner edge owing to the assumption of spherical symmetry. Default is 1.0.

cfrac is solely for deciding how the escape probabilities and optical depths are calculated. It is designed to distinguish between a plane-parallel slab where both the reflected and transmitted light is of interest (or a spherical shell with many holes so that both inward and outward light escape), vs. a closed sphere in which the only radiation which is observed is emitted outward. This is illustrated below.

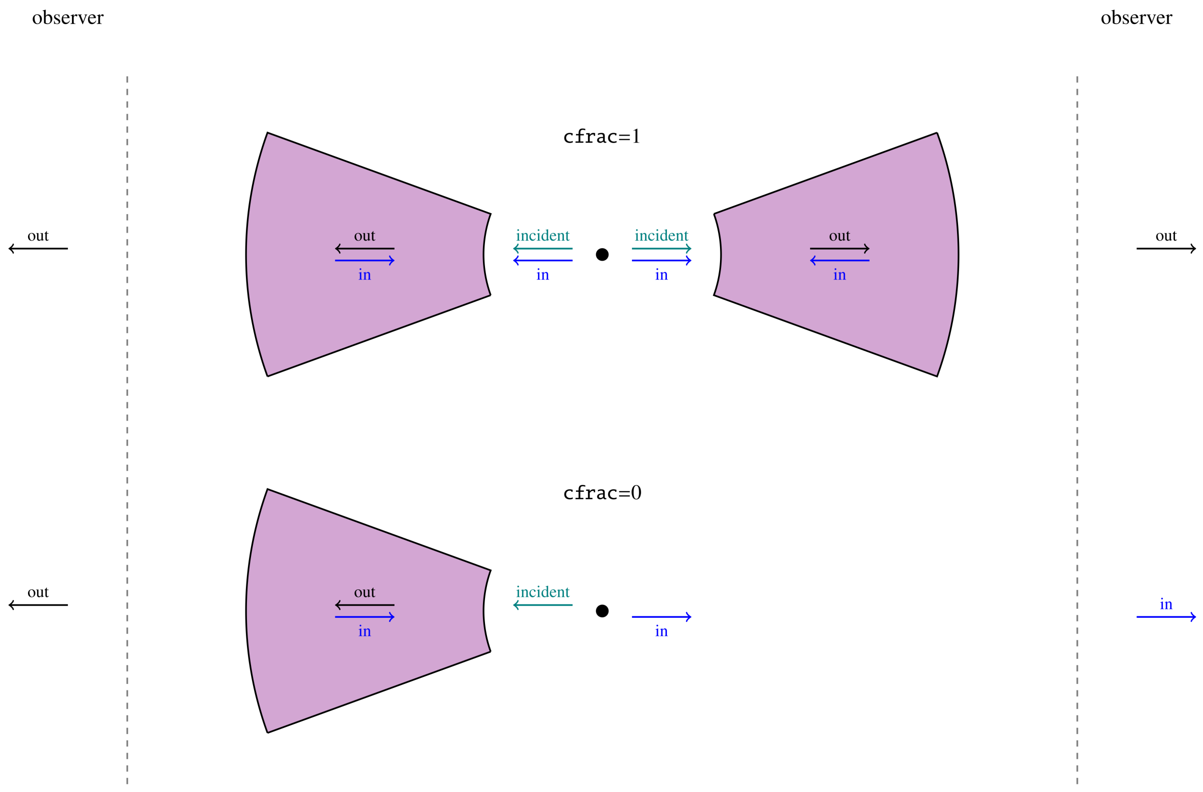


Fig. 1: Schematic illustration of the covering fraction parameter in XSTAR. For `cfrac=1` (top) all escaping inward radiation is assumed to re-enter the absorber due to spherical symmetry. No inward radiation leaves the cloud. For `cfrac=0` all radiation in inward direction that leaves the cloud is observed. This is would often be interpreted as reflection off a slab. The escaping radiation in outward direction is only mildly affected due to the internal radiation field. In particular it is *not* down-scaled by the value of `cfrac` nor diluted by unabsorbed, incident radiation.

The covering fraction does not directly affect how much total line radiation is emitted, so it is counter-intuitive in that sense. Hence it is not expected that, for example, `cfrac=0.5` would produce half as much line luminosity seen by a distant observer as `cfrac=1`. This interpretation is also different from most of the observational astrophysical literature where covering fraction usually refers to a mixing of unabsorbed and transmitted radiation. The XSTAR output separates inward and outward emitted, as well as incident and transmitted radiation so that such models can be easily constructed by the user.

In XSTAR, cfrac affects the composition of the radiation field inside the absorber. Here is how it works: XSTAR divides the radiation field into two components: inward or reflected, and outward or transmitted. At each point in the model it needs to know the escape probabilities in each of these directions in order to calculate how much of the locally emitted line radiation will escape to a distant observer, in either direction. As XSTAR progresses through the cloud it calculates the opacities and optical depth of the material at each radius. Thus at a given radius it knows accurately the optical depth, and hence the escape probability, in the inward or reflected direction. It does not know the optical depth in the outward direction, at least at first, because it has not calculated the ionization, temperature and opacity there yet. So it guesses at first, and assumes the optical depths are zero, and so the escape probability is 1. If you do a multi-pass model then it will use the optical depths from an earlier pass for the outward optical depth, and so it will be self-consistent. If `cfrac=0` xstar follows this procedure and provides the total escaping luminosities of all the lines and rrcs in the inward and outward directions.

If `cfrac=1` then xstar uses an assumption which originates from early historical photoionization models: it assumes that there is no inward or reflected radiation, and that all radiation is emitted in the outward direction. This is motivated by the assumption that the cloud is spherical, and that a sort of "Gauss's law" holds: any light emitted inward at a given radius will traverse the spherical region interior to that without interacting, and emerge at the same radius, and then have to traverse the same gas at larger radii as it would have if it were initially emitted outward. It also makes another major assumption: that the optical depth traversed in the outward direction is the same as the optical depth in the inward direction, which of course has already been calculated at a given radius as xstar proceeds outward in radius. So, if `cfrac=1`, there is no radiation escaping in the inward direction. If cfrac is between 0 and 1 then xstar does a linear interpolation between the two limits.

New in version 2.54a is the inclusion of radiative excitation. This process is discussed in more detail in section 9.D.2. The rate of radiative excitation of a given transition is calculated as described there, and then multiplied by a factor of 1-cfrac. So, if cfrac=1 the effective rate of radiative excitation is zero. If `cfrac=0` then the full rate of radiative excitation is included.

The input format is a float. The default parameter value is 1.0.

4.3.14 Constant Pressure Switch (lcpres) - *hidden*

This parameter chooses between constant density (`lcpres= 0`) and constant pressure (`lcpres=1`). This switch also affects the definition of the ionization parameter (Ξ or ξ)

The input format is an integer. The default is value is 0 (constant density).

4.3.15 Number of Steps (nsteps) - *hidden*

This parameter controls the maximum number of spatial zones used in a calculation, only in the case where the Courant condition step is larger than the size of the slab. That is, the step size is calculated as:

$$\Delta R = min(\text{emult}/\kappa_{max}(\varepsilon), R/\text{nsteps})$$

where emult is defined below, and $\kappa_{max}(\varepsilon)$ is the maximum opacity from the previous step calculation. $\kappa_{max}(\varepsilon)$ is only calculated from energies where the optical depth to the illuminated face of the cloud is less than taumax, where taumax is defined below.

The input format is an integer. The default parameter value is 2.

4.3.16 write switch (lwrite) - *hidden*

If the argument is a non-zero integer, level populations and line emissivities in the interior of the shell will be written to a FITS dataset at each spatial step for later examination or plotting. These files are named 'xo0n_detail.fits' … 'xo0n_detal4.fits', where n is the pass number. These can become quite large (hundreds of MB to several GB) for a model with many spatial zones. Various FITS file manipulation routines can be used to filter and plot the quantities in these files.

New in version 2.5: if this parameter has a value -1 no fits output is produced.

The input format is an integer. The default parameter value is 0.

4.3.17 print switch (lprint) - *hidden*

This parameter enables the printing of many quantities which are defined locally at the last spatial zone of the calculation. These include ion fractions, heating and cooling rates, line and continuum emissivities and opacitites, execution times, and level populations. They are printed to the log file "xout_step.log".

`lprint=0` results in just the 500 brightest line luminosities (sorted by luminosity) and depths, and the 500 brightest recombination continuum (sorted by luminosity) luminosities and depths to be printed to the log file at the end of the run.

`lprint=1` causes many additional quantities to be printed, including ASCII tables of total line and RRC luminosities and depths.

`lprint=2` additionally prints many useful local quantities such as ion fractions, thermal rates, level populations. Note that these local quantities are printed only at the final step of the model run.

`lprint=3` prints continuum luminosities and emissivities.

`lprint=4` the list of lines is printed, along with upper and lower levels, transition probabilities and statistical weights.

`lprint=5` prints all rates affecting level populations, ionization, heating and cooling internal to the code. Interpreting these require familiarity with the internal operation of the code.

This switch affects only the ASCII log file, xout_step.log. The standard FITS output files are unaffected by the value of lprint.

A feature added in version 2.5: if this parameter is given a value -1 only a very minimal printout is given after the final spatial zone: just the position, ionization parameter, heating and cooling rates. This option is designed to provide a streamlined printout in the case of a very large xstar2xspec run.

The input format is an integer. The default parameter value is 0.

4.3.18 Courant Multiplier (emult) - *hidden*

This is a constant factor used in calculating the spatial step size. For each radial zone, the size of the next zone is chosen to be

$$\Delta R = min(\mathrm{emult}/\kappa_{max}(\varepsilon), R/\mathrm{nsteps})$$

where emult is defined below, and $\kappa_{max}(\varepsilon)$ is the maximum opacity from the previous step calculation. $\kappa_{max}(\varepsilon)$ is only calculated from energies where the optical depth to the illuminated face of the cloud is less than taumax, where taumax is defined below. *Values outside the range 0.1–1 are unlikely to be of any practical value.*

The input format is a float. The default parameter value is 0.5.

4.3.19 Max Tau for Courant Step (taumax) - *hidden*

This quantity is used in calculating the spatial step size, as described in the previous subsection. Energy bins with continuum optical depth to the illuminated cloud face greater than taumax are not used when searching for the maximum photoelectric opacity.

The parameter format is a float. The default parameter value is 5.

4.3.20 Min Electron Abundance (xeemin) - *hidden*

This is the minimum allowed electron fractional abundance. If the electron fraction falls below this value the current pass is ended.

The input format is a float. The default paramter value is 0.1.

4.3.21 Critical Ion Fraction (critf) - *hidden*

Ions whose abundance relative to total hydrogen (H I + H II) are less than this value after the preliminary ion abundance calculation (See Chapter~ref{sec:internals}) are not included in full multilevel calculation. This parameter should be changed with caution owing to the fact that it determines the size of the matrix that xstar tries to solve in calculating the level populations.

In versions of xstar 2.1lxx and earlier this matrix had a maximum size of 2400, and an attempt to solve for more than this number of levels simultaneously would result in xstar stopping with a message “ipmat too large”.

In version 2.2 and later this limitation has been removed, and arbitrarily small values of critf can be accomodated. Also, a faster algorithm has been adopted for the multi-level calculation, so the speed advantages of large critf have been reduced. Also, the input parameter critf now refers to the fractional ion abundance (i.e. relative to the parent element) rather than the absolute (i.e. relative to H) ion abundance.

The input format is a float. The default parameter value is 1.0E-07.

4.3.22 Turbulent Velocity (vturbi) -*hidden*

This parameter is the turbulent velocity in units of $\mathrm{km\,s^{-1}}$ and allows extra line broadening to be introduced into the calculation of the synthetic spectrum. This quantity enters into the calculation of the line profile function, which is used by the line opacity and emissivity. The line profile function is a Voigt profile, which resembles a Gaussian in the Doppler core. The turbulent velocity then enters into the “sigma” of the Gaussian. The Gaussian “sigma” in velocity units is $(v_{\mathrm{turb}}^2 + v_{\mathrm{therm}}^2)^{1/2}$ where v_{therm} is the ion thermal speed.

The input format is a float. The default parameter value is 1.0.

4.3.23 Number of Passes (npass) - *hidden*

This parameter determines how many complete calculations of the temperature and ionization structure of the model shell are made. Multiple passes are needed because there is no a priori knowledge of the optical depth of the shell in all the lines and continua, and these can affect the state of the gas in the interior of the shell. During the first pass the calculation proceeds through the shell, and assumes that all optical depths from points within the shell to the far edge of the shell are 0. If an integer greater than 1 is supplied as a parameter, XSTAR performs that number of iterations through the entire calculation, setting the optical depths to the far edge at the values calculated in the previous iteration. The odd numbered passes are made from the smallest to largest radius, while the even numbered passes are made in the inward direction. The emergent spectrum is not calculated accurately during the inward passes, so npass must be odd. Multi-pass calculations substantially improve the accuracy of the predictions made for shells with finite thickness, but they are much more time consuming than single-pass calculations. They also make use of temporary unformatted datasets, named 'xout_tmp.lis', 'xout_tmp2.lis', which can become quite large.

The input format is an integer. The default parameter value is 1.

4.3.24 Number of Iterations (niter) - *hidden*

Set maximum number of iterations for thermal equilibrium and charge neutrality calculation at each spatial step. If this quantity is set to zero then a constant temperature run will result, and charge neutrality will not be calculated. If this quantity is negative, then charge neutrality will be calculated, but thermal equilibrium will not. Normal thermal equilibrium models can be calculated with `niter=99`, although the code seldom requires more than a few iterations (10 or 20 at most) to achieve thermal equilibrium under normal conditions. If niter=0 then the assumed value for the electron fraction relative to total hydrogen is 1.0.

The input format is an integer. The default parameter value is 0 (neither thermal equilibrium nor charge neutrality).

4.3.25 Radius Exponent (radexp) - *hidden*

New in version 2.2 is the ability to have the gas density variable as a power law in radius, i.e. $n = n_0(R/R_0)^{\mathrm{radexp}}$, where n_0 and R_0 are the density and radius at the cloud illuminated face. Note that this creates the possibility for calculations which do not end. This because the criteria for a model to end are either that the input column density is reached, or that the electron fraction falls below xeemin. If radexp has a value ≤ -1 then the column density integral converges only logarithmically at best, and the specified column may never be reached. If radexp ≤ -2 the local ionization parameter will increase with radius, and so the gas will not recombine and the xeemin criterion will not be met.

New in version 221bn17 is a feature which allows an array of densities to be read in. It requires that the radexp input variable be set to a number more negative than -100. Then ordered pairs of (radius, density) are read in from a file called 'density.dat'. Reading continues until the end of the file is reached. The density and radius values override the values derived from the ordinary input parameters. But execution will stop if other ending criteria are satisfied, i.e. if the model column density exceeds the input value, or the electron fraction falls below the specified minimum. The code will stop with an error if the density.dat file does not exist, or if the radius values are not monotonically increasing.

The input format is a float. The default parameter value is 0.0 (constant density).

4.3.26 Number of Continuum bins (ncn2) - *hidden*

New in version 2.2 is the option to control the number of continuum bins. Continuum bins are logarithmically spaced between 0.1 eV and 40 keV, and are calculated according to:

$$\Delta\varepsilon/\varepsilon = (40\mathrm{keV}/0.1\mathrm{eV})^{1/(0.49\mathrm{ncn2})}$$

ncn2 must be in the range between 999 and 999999. The higher value is appropriate for use in modeling X-ray grating spectra. The lower value is appropriate for models where only integrated line luminosities or ionization fractions are desired. Execution time scales approximately proportionately to ncn2.

The input format is an integer. The default parameter is 9999.

4.3.27 Loop Control (loopcontrol) - *hidden*

Used by `XSTAR2XSPEC` to track each model generated. This value should never be manipulated by the end user.

The input format is an integer. The default parameter value is 1.

4.3.28 mode - *hidden*

This parameter is used by the XPI interface and determins the behavior of the parameter input by the user, for example whether user parameters should be saved for the next call of XSTAR or whether XSTAR should start again from default parameters. This value should never be manipulated by the end user.

The input format is a string. The default parameter value is "ql" (query and learn).

CHAPTER

FIVE

XSTAR OUTPUT

5.1 The Spectral Data File: xout_spect1.fits

This ASCII FITS file contains the spectral information. The first extension lists all the user input parameters for this XSTAR run. The second extension contains the spectra *with lines binned into the continuum*. The columns are the channel energy in eV, incident, transmitted, and emitted radiation in inward and outward direction, respectively, all specific luminosities in units of 10^{38} erg s^{-1} erg^{-1}. The file format is shown below.

```
No. Type     EXTNAME      BITPIX Dimensions(columns)      PCOUNT  GCOUNT

 0  PRIMARY                 16     0                           0     1
 1  BINTABLE PARAMETERS      8     66(5) 55                    0     1

    Column Name                Format     Dims       Units     TLMIN  TLMAX
    1 index                      1I
    2 parameter                  20A
    3 value                      1E
    4 type                       10A
    5 comment                    30A

 2  TABLE    XSTAR_SPECTRA   8     69(5) 9999                  0     1

    Column Name                Format     Dims       Units     TLMIN  TLMAX
    1 energy                     E13.5                eV
    2 incident                   E13.5                erg/s/erg
    3 transmitted                E13.5                erg/s/erg
    4 emit_inward                E13.5                erg/s/erg
    5 emit_outward               E13.5                erg/s/erg
```

5.2 The Continuum File: xout_cont1.fits

This ASCII FITS file contains the continuum luminosities. The first extension lists all the user input parameters for this XSTAR run. The second extension contains the spectra *without lines binned into the continuum*. The columns are the channel energy in eV, incident, transmitted, and emitted specific luminosities in the inward and outward directions, respectively, all in units of 10^{38} erg s^{-1} erg^{-1}. The file format is the same as for xout_spect1.fits.

```
No. Type     EXTNAME      BITPIX Dimensions(columns)      PCOUNT  GCOUNT

 0  PRIMARY                 16     0                           0     1
```

(continues on next page)

(continued from previous page)

```
 1  BINTABLE PARAMETERS      8     66(5) 55                  0    1

    Column Name                Format     Dims       Units     TLMIN  TLMAX
    1 index                      1I
    2 parameter                  20A
    3 value                      1E
    4 type                       10A
    5 comment                    30A

 2  TABLE    XSTAR_SPECTRA    8     69(5) 9999                0    1

    Column Name                Format     Dims       Units     TLMIN  TLMAX
    1 energy                     E13.5                eV
    2 incident                   E13.5                erg/s/erg
    3 transmitted                E13.5                erg/s/erg
    4 emit_inward                E13.5                erg/s/erg
    5 emit_outward               E13.5                erg/s/erg
```

5.3 The Line Lumnosity File: xout_lines1.fits

This ASCII FITS file contains information about the strongest emission lines. The first extension lists all the user input parameters for this XSTAR run. The second extension lists the 600 strongest emission lines. For each line, the line index (internal reference of XSTAR and the atomic database), the ion (string), the lower and upper level (string), the wavelength (Å), the emitted luminosity in forward and backward directions (in units of 10^{38} erg s^{-1}), and optical depth in the forward and backward directions. The file format is shown below.

```
No. Type     EXTNAME      BITPIX Dimensions(columns)      PCOUNT  GCOUNT

 0  PRIMARY                 16     0                         0    1
 1  BINTABLE PARAMETERS      8     66(5) 55                  0    1

    Column Name                Format     Dims       Units     TLMIN  TLMAX
    1 index                      1I
    2 parameter                  20A
    3 value                      1E
    4 type                       10A
    5 comment                    30A

 2  TABLE    XSTAR_LINES     8     128(9) 600                0    1

    Column Name                Format     Dims       Units     TLMIN  TLMAX
    1 index                      I6
    2 ion                        A9
    3 lower_level                A20
    4 upper_level                A20
    5 wavelength                 E13.5                A
    6 emit_inward                E13.5                erg/s/10**38
    7 emit_outward               E13.5                erg/s/10**38
    8 depth_inward               E13.5
    9 depth_outward              E13.5
```

5.4 The Abundances Data File: xout_abund1.fits

This ASCII FITS file contains ion abundances as well as heating and cooling rates. Only ions with fractional abundance (relative to its parent element) greater than 10^{-10} (relative to its parent element) are recorded. The elements are ordered by increasing nuclear charge, ions by increasing free charge.

The FITS file has four extensions: spatial ionic abundances, total ionic columns and heating and cooling rates:

```
No. Type     EXTNAME      BITPIX Dimensions(columns)      PCOUNT  GCOUNT

 0  PRIMARY                 16     0                           0     1
 1  TABLE    ABUNDANCES      8     6621(473) *********         0     1
 2  TABLE    COLUMNS         8     6621(473) *********         0     1
 3  TABLE    HEATING         8     559(40) **********          0     1
 4  TABLE    COOLING         8     573(41) **********          0     1
```

The "ABUNDANCES" extension contains information about ionic abundance for each spatial zone. Each row contains the radius and size of the spatial zone, the local ionization parameter, electron fraction, density, pressure, temperature, fractional heating-cooling, followed by the abundance for each ion relative to its parent element:

```
No. Type     EXTNAME      BITPIX Dimensions(columns)      PCOUNT  GCOUNT

 1  TABLE    ABUNDANCES      8     6621(473) *********         0     1

    Column Name                Format     Dims       Units     TLMIN  TLMAX
    1 radius                     E13.5                cm
    2 delta_r                    E13.5                cm
    3 ion_parameter              E13.5                erg*cm/s
    4 x_e                        E13.5
    5 n_p                        E13.5
    6 pressure                   E13.5                dynes/cm**2
    7 temperature                E13.5                10**4 K
    8 frac_heat_error            E13.5
    9 h_i                        E13.5
   10 he_i                       E13.5
   11 he_ii                      E13.5
   12 li_i                       E13.5
   13 li_ii                      E13.5
   14 li_iii                     E13.5
   15 be_i                       E13.5
   16 be_ii                      E13.5
   17 be_iii                     E13.5
   18 be_iv                      E13.5
   [...]
```

The "COLUMNS" extension has a very similar format but contain column density for each ion in units of cm^{-2}. In principle, these are integrated abundances from the previous extension, weighted by the elemental abundance and scaled by the total column density. Note that the ion fraction of the fully stripped ion is never tabulated, for any element. The abundance of these ions may be derived from the other ions by calculating one minus the sum of the tabulated ion fractions.

The last two extensions, "HEATING" and "COOLING", list the Compton and total heating rates, and bremsstrahlung, Compton, and total cooling rates (in $\mathrm{erg\,cm^{-3}\,s^{-1}}$).

5.5 The RRC File: xout_rrc1.fits

This ASCII FITS file contains information contains about the RRCs (radiative bound-free transitions). The first extension lists all the user input parameters for this XSTAR run. The second extension contains for each RRC, the RRC index (internal reference of XSTAR and the atomic database), the ion (string), the bound level (string), the energy (eV), the emitted luminosity in forward and backward directions (in units of 10^{38} erg s^{-1}), and optical depth in the forward and backward directions.

```
No. Type     EXTNAME      BITPIX Dimensions(columns)      PCOUNT  GCOUNT

 0  PRIMARY                 16     0                          0    1
 1  BINTABLE PARAMETERS      8     66(5) 55                   0    1

    Column Name                Format     Dims       Units     TLMIN  TLMAX
    1 index                      1I
    2 parameter                  20A
    3 value                      1E
    4 type                       10A
    5 comment                    30A

 2  TABLE    XSTAR_SPECTRA   8     107(8) 6945                0    1

    Column Name                Format     Dims       Units     TLMIN  TLMAX
    1 index                      I6
    2 ion                        A9
    3 level                      A20
    4 energy                     E13.5                eV
    5 emit_outward               E13.5                erg/s
    6 emit_inward                E13.5                erg/s
    7 depth_outward              E13.5
    8 depth_inward               E13.5
```

5.6 Detailed Ionic Information: xoNN_detail.fits

This file is only produced for `lwrite=1` *!*

This ASCII FITS file contains all level populations for all spatial zones. This file is large and time-consuming to view and manipulate. For this file, the NN in the name is replaced by the pass number, a 2 digit integer. The first extension lists all the user input parameters for this XSTAR run. Each of the following extension represents a spatial zone, so the number of extensions will depend on the details of the XSTAR calculation. The spatial extension list the index (internal to the XSTAR database), the ion index (internal to the XSTAR database), the excitation energy in eV, the ion string, its atomic number, the configuration string, the ion population relative to the parent ion, the LTE population and the and the level index (relative to the ion). Other useful quantities like the radius, size, temperature, local ionization parameter etc. for each spatial zone are given in the header of each extension.

```
No. Type     EXTNAME      BITPIX Dimensions(columns)      PCOUNT  GCOUNT

 0  PRIMARY                 16     0                          0    1
 1  BINTABLE PARAMETERS      8     66(5) 55                   0    1

    Column Name                Format     Dims       Units     TLMIN  TLMAX
```

(continues on next page)

(continued from previous page)

```
    1 index                    1I
    2 parameter                20A
    3 value                    1E
    4 type                     10A
    5 comment                  30A

 2  BINTABLE XSTAR_RADIAL    8      50(9) 2590                 0    1

    Column Name                  Format     Dims       Units     TLMIN  TLMAX
    1 index                      1J
    2 ion_index                  1I
    3 e_excitation               1E                    eV
    4 ion                        8A
    5 atomic_number              1I
    6 ion_level                  20A
    7 population                 1E
    8 lte                        1E
    9 upper index                1I

 3  BINTABLE XSTAR_RADIAL    8      50(9) 2590                 0    1

    Column Name                  Format     Dims       Units     TLMIN  TLMAX
    1 index                      1J
    2 ion_index                  1I
    3 e_excitation               1E                    eV
    4 ion                        8A
    5 atomic_number              1I
    6 ion_level                  20A
    7 population                 1E
    8 lte                        1E
    9 upper index                1I

 [...]
```

5.7 Detailed Line Information: xoNN_detal2.fits

This file is only produced for `lwrite=1` *!*

This ASCII FITS file contains line emissivities for all spatial zones. This file is large and time-consuming to view and manipulate. For this file, the NN in the name is replaced by the pass number, a 2 digit integer. The first extension lists all the user input parameters for this XSTAR run. Each of the following extension represents a spatial zone, so the number of extensions will depend on the details of the XSTAR calculation. The spatial extension list the line index (internal to the XSTAR database), the wavelength (in Å), the ion string, the lower and upper level configuration strings, the inward and outward emission, the line opacity, and inward and outward optical depth. Other useful quantities like the radius, size, temperature, local ionization parameter etc. for each spatial zone are given in the header of each extension.

```
No. Type     EXTNAME      BITPIX Dimensions(columns)      PCOUNT  GCOUNT

 0  PRIMARY                 16      0                          0    1
 1  BINTABLE PARAMETERS      8      66(5) 55                   0    1
```

(continues on next page)

(continued from previous page)

```
    Column Name                Format     Dims       Units     TLMIN  TLMAX
    1 index                     1I
    2 parameter                 20A
    3 value                     1E
    4 type                      10A
    5 comment                   30A

 2  BINTABLE XSTAR_RADIAL    8     76(10)  18662              0    1

    Column Name                Format     Dims       Units     TLMIN  TLMAX
    1 index                     1J
    2 wavelength                1E                   A
    3 ion                       8A
    4 lower_level               20A
    5 upper_level               20A
    6 emis_inward               1E                   erg/cm^3/s
    7 emis_outward              1E                   erg/cm^3/s
    8 opacity                   1E                   /cm
    9 tau_in                    1E
   10 tau_out                   1E

 3  BINTABLE XSTAR_RADIAL    8     76(10)  18662              0    1

    Column Name                Format     Dims       Units     TLMIN  TLMAX
    1 index                     1J
    2 wavelength                1E                   A
    3 ion                       8A
    4 lower_level               20A
    5 upper_level               20A
    6 emis_inward               1E                   erg/cm^3/s
    7 emis_outward              1E                   erg/cm^3/s
    8 opacity                   1E                   /cm
    9 tau_in                    1E
   10 tau_out                   1E

[...]
```

5.8 Detailed RRC Information: xoNN_detal3.fits

This file is only produced for `lwrite=1` *!*

This ASCII FITS file contains all RRC emissivities and opacities for all spatial zones. This file is large and time-consuming to view and manipulate. For this file, the NN in the name is replaced by the pass number, a 2 digit integer. The first extension lists all the user input parameters for this XSTAR run. Each of the following extension represents a spatial zone, so the number of extensions will depend on the details of the XSTAR calculation. The spatial extension list the RRC and level indices (internal to the XSTAR database), the energy (in eV), the ion string, the lower and upper level configuration strings (upper is continuum), the inward and outward emission, the integrated absorption, the RRC opacity, and inward and outward optical depth. Other useful quantities like the radius, size, temperature, local ionization parameter etc. for each spatial zone are given in the header of each extension.

```
No. Type     EXTNAME      BITPIX Dimensions(columns)      PCOUNT  GCOUNT

 0  PRIMARY                16     0                           0     1
 1  BINTABLE PARAMETERS     8     66(5) 55                    0     1

    Column Name                Format     Dims       Units     TLMIN  TLMAX
    1 index                      1I
    2 parameter                  20A
    3 value                      1E
    4 type                       10A
    5 comment                    30A

 2  BINTABLE XSTAR_RADIAL    8     84(12) 2502                0     1

    Column Name                Format     Dims       Units     TLMIN  TLMAX
    1 rrc index                  1J
    2 level index                1J
    3 energy                     1E                  ev
    4 ion                        8A
    5 lower_level                20A
    6 upper_level                20A
    7 emis_inward                1E                  erg/cm^3/s
    8 emis_outward               1E                  erg/cm^3/s
    9 integrated absn            1E                  erg/cm^3/s
   10 opacity                    1E                  /cm
   11 tau_in                     1E
   12 tau_out                    1E

[...]
```

5.9 Detailed Line Information: xoNN_detal4.fits

This file is only produced for `lwrite=1` *!*

This ASCII FITS file contains binned continuum specific luminosities and emissivities. This file is large and time-consuming to view and manipulate. For this file, the NN in the name is replaced by the pass number, a 2 digit integer. The first extension lists all the user input parameters for this XSTAR run. Each of the following extension represents a spatial zone, so the number of extensions will depend on the details of the XSTAR calculation. Note that specific luminosity has units 10^{38} erg s^{-1} erg^{-1}. The values in the columns in the spatial extensions are as follows:

The first two columns are the index and energy (in eV) of each spectral channel. The next for columns are

- `zrems(1)`: the specific luminosity used internally by xstar, calculated including all absorption and emission
- `zrems(2)`: specific luminosity which includes only emission from the model gas in the inward (reflected) direction (i.e., no incident radiation included)
- `zrems(3)`: specific luminosity which includes only emission from the model gas in in the outward (transmitted) direction (i.e., no incident radiation included)
- `zrems(4)`: specific luminosity which includes only emission from the model gas in the inward (reflected) direction *but not including line emission*
- `zrems(5)`: specific luminosity which includes only emission from the model gas in in the outward (transmitted) direction *but not including line emission.*

The remaining columns contain the opacity, the emission in inward and outward directions, and the optical depths in inward and outward directions.

```
No. Type     EXTNAME      BITPIX Dimensions(columns)      PCOUNT  GCOUNT

 0  PRIMARY                 16     0                           0     1
 1  BINTABLE PARAMETERS      8     66(5) 55                    0     1

    Column Name                Format     Dims       Units     TLMIN  TLMAX
    1 index                      1I
    2 parameter                  20A
    3 value                      1E
    4 type                       10A
    5 comment                    30A

 2  BINTABLE XSTAR_RADIAL    8     48(12) 9999                 0     1

    Column Name                Format     Dims       Units     TLMIN  TLMAX
    1 index                      1J
    2 energy                     1E                  ev
    3 zrems(1)                   1E                  erg/s
    4 zrems(2)                   1E                  erg/s
    5 zrems(3)                   1E                  erg/s
    6 zrems(4)                   1E                  erg/s
    7 zrems(5)                   1E                  erg/s
    8 opacity                    1E                  /cm
    9 emis out                   1E                  erg/cm**3/s
   10 emis in                    1E                  erg/cm**3/s
   11 fwd dpth                   1E
   12 bck dpth                   1E
```

5.10 XSTAR Run Log: xout_step.log

This file is always written but the parameter `lprint` *controls the level of detail!*

In any case, this ASCII text file contains the input parameters and a log of the temperature and other useful quantities (radius, $\Delta R/R$ the fractional distance from the illuminated cloud face, column density, ionization parameter, electron fraction, proton number density, temperature, fractional heating-cooling rates, continuum optical depth at the Lyman continuum in the transmitted and reflected directions, and the number of iterations required to reach thermal equilibrium. This is the same as the information printed to the screen. In addition, at the end of a model calculation the luminosities of the 500 strongest lines and the 500 strongest RRCs are printed, sorted by luminosity, along with the energy budget: total energy absorbed, emitted in the continuum, emitted in lines, and the fractional difference between the first quantity and the sum of the latter two. Models with energy budget errors greater than a few percent should likely be rerun with smaller value of emult.

CHAPTER

SIX

XSTAR2XSPEC

6.1 Introduction

The previous chapter *XSTAR output* described the various output files that XSTAR produces for a specific set of parameters. Many users will also want to compare XSTAR results with observational data and in particular fit observed X-ray spectra with XSTAR models, i.e., vary XSTAR parameters until a good match between observed and synthetic spectra is reached. To facilitate this process, XSTAR2XSPEC was developed. XSTAR2XSPEC is a perl script which calls XSTAR multiple times and generates table models from the results of these simulations which can then be utilized for model fitting in the xspec spectral fitting program. Table models have advantages and disadvantages and the user is strongly recommended to read the following sections completely which contain several important caveats.

6.1.1 General Notes on Table Models in XSPEC

As indiviudal XSTAR runs can take anywhere between seconds and hours, calling XSTAR from within xspec to evaluate and optimize models is not feasible in many cases. An easy way around this is evaluating XSTAR on a pre-defined grid of parameters and tabulating the results of the individual calculations. This table model can then be loaded into and xspec will interpolate the model spectrum for any given parameter combination within the grid.

XSPEC has a specific table model format (see OGIP memo 92-009 for details) that allows for very efficient searching and interpolating among the tabulated spectra. XSPEC distinguishes between additve table models (`atable`), multiplicative table models (`mtable`), and exponential table models (`etable`) that behave differently regarding normalization and rebinning. Per default, XSTAR2XPEC is designed to create only `atable` and `mtable` models. Users who want to use `etable` models will need to build them from the existing `mtable` **(make thread to show this)**.

The advantage of table models is that the calculation of the table can easily be parallelized so even computationally expensive models can be produced in a reasonable amount of time. While table models are very straightforward to calculate and run very fast, there are a few things to consider when using tabulated models with observational data.

First, the number of tabulated spectra grows exponentially with the number of free parameters and linearly with the range and sampling of each parameter. In practice, this usually limits the number of free parameters to about one to four which is oftern rather small for complex physical models and requires additional assumptions or simplifications. The ranges and stepsizes for the parameter grid has to be choosen appropriately to avoid interpolation artifacts in the final model spectrum. This is especially problematic when varying abundances in absorption models.

Second, properties like spectral resolution or energy ranges need to be defined at the time of creation of the table model and cannot be changed afterwards. Therefore, a model that was calculated for a specific instrument or dataset cannot be easily tranfered to another and often needs to be recalculated from scratch.

In order to reduce table sizes, xspec table models consider two types of free parametes: interpolated and additive parameters. The idea behind additive parameters is that they are suffciently independent of all other parameters so that tabulated spectra do not need to stored for every single parameter combination. Interpolated parameters are treated as one might expect: if the model is represented by a vector $M_i(x_j)$, corresponding to the model flux at various energies ε_i

and stored at various values of the free (intepolated) parameter x_j, then for a value of the parameter not on the tabulated grid xspec calculates the model value as

$$M_i(x) = \Sigma_j M_i(x_j)\omega_j$$

where ω_j are suitably chosen weights for, e.g. linear interpolation. This is in contrast to the treatment of additive parameters: for an `mtable`, xspec needs the value of the model tabulated at only two values of the free additive parameter y, 0 and y_{max}. Then the model is calculated for some arbitrary y as

$$M_i(y) = (M_i(y_{\mathrm{max}}) - M_i(0))\,\frac{y}{y_{\mathrm{max}}} + M_i(0)$$

i.e. it is assumed that the model scales linearly with the value of the additive parameter y. This formalism was developed for emission models, where emissivities might be expected to scale linearly with elemental abundance. The priniple is illustrated below.

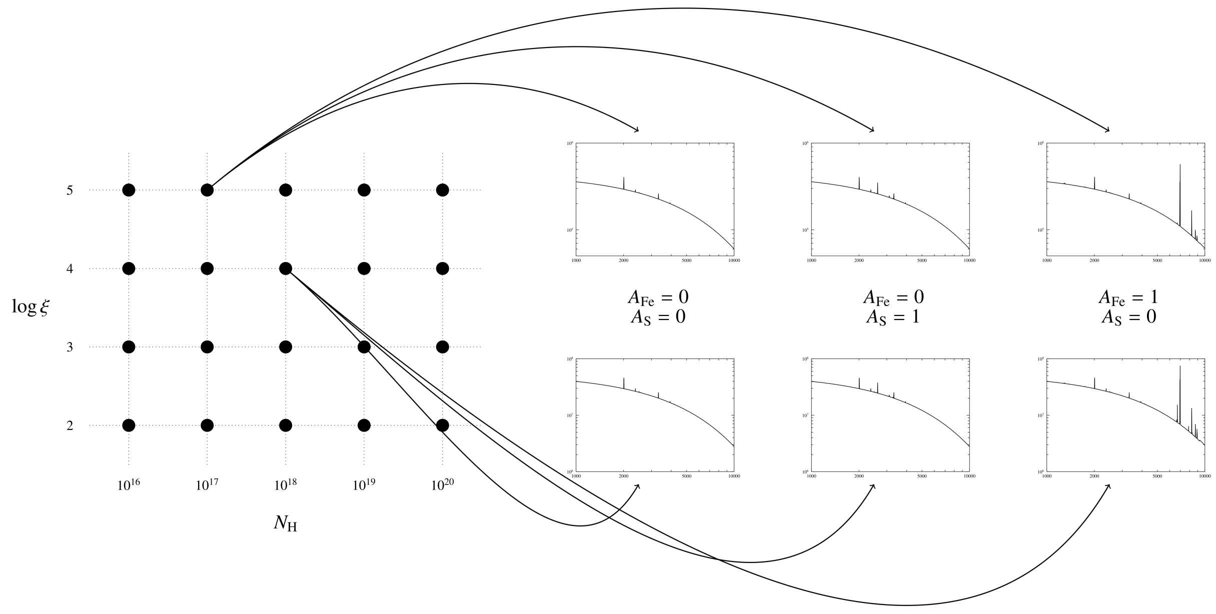


Fig. 1: Illustration of the concept of interpolated versus additive parameters. In this example $\log\xi$ and N_{H} are interpolated parameters, the iron abundance A_{Fe} and sulfur abundance A_{S} are additive parameters. At each gridpoint in the $\log\xi$-N_{H} plane, three spectra are stored. For a given combination $(\log\xi, N_{\mathrm{H}}, A_{\mathrm{S}}, A_{\mathrm{Fe}},)_i$, the spectra are interploted on the $\log\xi$-N_{H} grid and for each additive abundance. This way only a total of 60 spectra need to be stored. If all parameters were interpolated the model would contain 80 tabulated spectra.

6.1.2 Variable Abundances in Table Models

If you are using this approach to calculate absorption models (`mtables`) with variable abundances, then you will likely get totally unphysical results unless you note the following. In the case of absorption models, the value of M which xspec uses is a transmission coefficient, and this does *not* scale linearly with abundance. Rather, the transmission coefficient is related to the optical depth by:

$$M_i(y) = \exp\left(-\tau_i(y)\right)$$

and the optical depth $\tau_i(y)$ does scale (approximately) linearly with abundance. For this reason, xspec has incorporated the `etable` models, in which the model builder supplies optical depths rather than transmission coefficient, linear interpolation is used to calculate the optical depth for a given abundance, and then the transmission coefficient is calculated using the above expression. If so, users should use the `model etable{xout_etable.fits}` command in xspec instead. Entering parameter, etc., in xspec is the same for `etables` as for `mtables`.

The problem with `mtables` models is that the multiplicative factor per spectral bin is calculated as

$$M_i = M_i^0 + \Sigma_j x_j M_i^j$$

where M_i^0 is the 'zero abundance' version of the model at that ionization parameter and column, x_j is the abundance of element j, and M_i^j is the model calculated with that element set to unity (relative to cosmic). The first problem with this approach is that tranmission does not scale this way with abundance. If you double the abundance of an element, you should double the optical depth. Doubling the transmittivity has both the wrong qualitative and quantitative behavior. But an even bigger problem with this is that the whole thing, M_i, multiplies your continuum. It is almost guaranteed to bigger than 1 if some of the x_j are non-zero. So this is totally the wrong formulation to use when the abundances vary.

In the case of an `etable` the transmittivity is:

$$M_i = \exp(-(E_i^0 + \Sigma_j x_j E_i^j))$$

where $E_i^0 = -\ln(M_i^0)$ etc.

This is better, since it predicts the correct dependence on abundance, i.e. that the optical depth scales approximately linearly with abundance.

Below are six figures to illustrate these problems. All use the `mtable` or `etable` from a table model "grid18" that we will also use for other exmaples. Instructions on how to calculate this table can be found at *Running XSTAR2XSPEC*. They show the multiplicative transmission coefficient between 0.6 and 0.8 keV for a $\log(\xi) = 1$ and $N_{\rm H} = 10^{20}$ cm^{-2}.

In addition to these obvious problems, it is also important to remember that what you are trying to simulate is a 'real' photoionized gas with some abundance set $\{x_j\}$ that fits your data. What you are using to fit to your data are M_i^0, which in the case of xstar are models calculated with only hydrogen and helium, and the set of M_i^j, which are supposed to be calculated with pure element j.

The first problem is that the cooling, and therefore the temperature, depends on the element abundance in the gas, and this is not linear. The temperature couples back into the ionization balance, and that affects the opacity, transmittivity, etc.

The second problem is that, while it is straightforward to calculate a model with no elements heavier than H and He, it is not straightforward to calculate a model with pure iron, or calcium, say. That is because H and He have nice simple cooling properties at low temperature, and their opacity is simple, etc. Pure iron models at best are likely to be very different than H+He+Fe models, or from a cosmic mix. So what I do is make the M_i^j using H+He+element j in cosmic ratios. But the problem with this is that then the opacity due to H and He are present *in every :math:`M_i^j* and $E_i^j$ *, particularly at low ionization parameters. So if you sum over cosmic abundances using an etable you are including the H and He opacity with every one of the :math:`E_i^j.* At high ionization parameter ($\log(\xi) > 1$?) this should not be a problem because H and He are ionized sufficiently that they do not contribute to the opacity. But at low ionization parameter, the model spectra calculated using etables will have significantly more low energy opacity due to the multiple counting of H and He than they would in a real XSTAR model with the corresponding parameters.

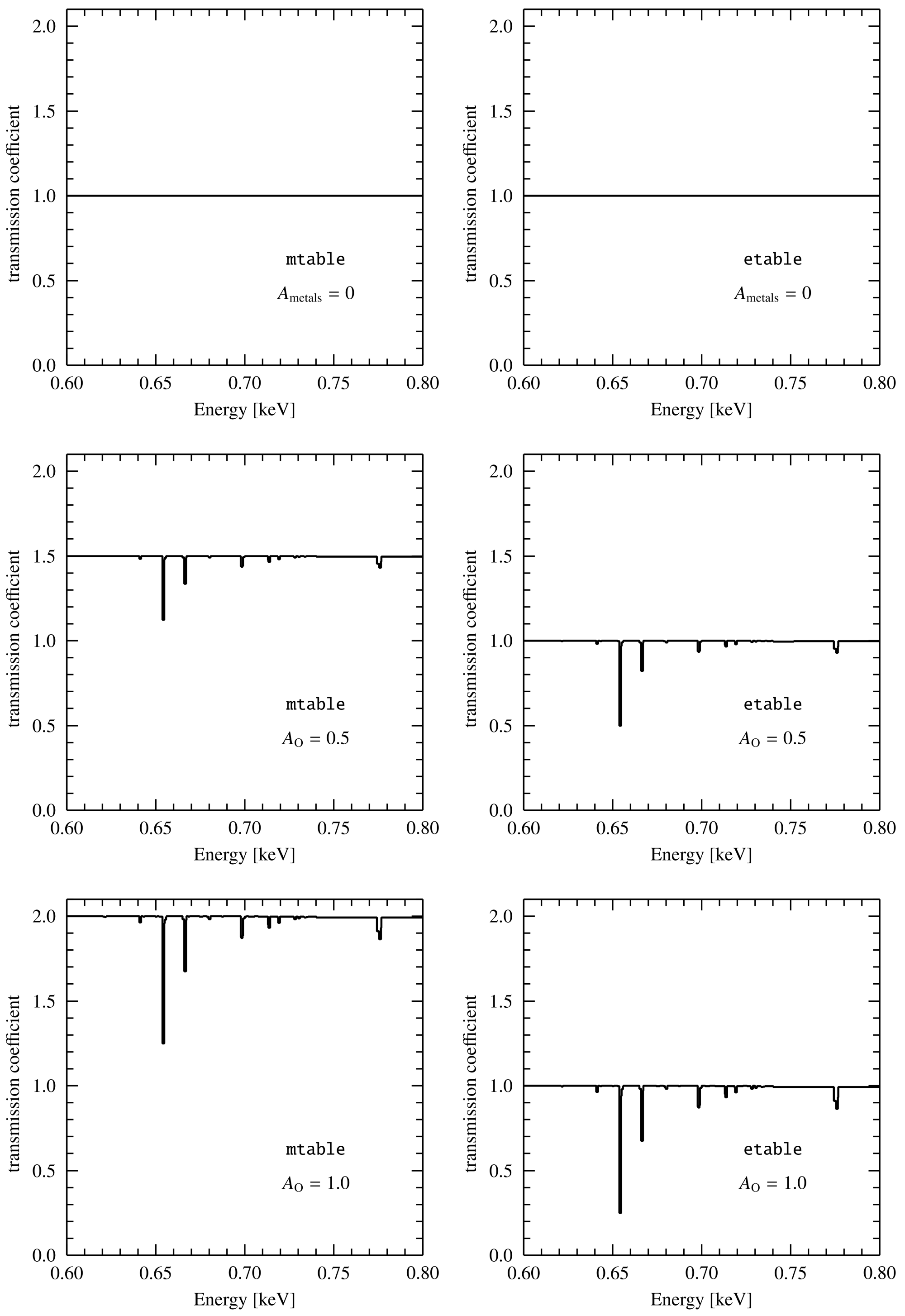


Fig. 2: Transmission coefficients for `mtable` and `etable` models, once for all metal abundances set to zero (only H and He) and once with oxygen abundance set to 0.5 solar and once to solar. The transmission coefficient should always between 0 and 1, with 1 indicating no absorption and 0 full absorption. By construction of the `mtable` model, however, the transmission coefficient becomes larger than 1, because transmission coefficients for additive parameters (like elemental abundances) are summed up. Additionally, the line depth is not calculated correctly. For `etable` table models, the optical depth τ is summed up over additive parameters and the model returns the total transmission coefficient $\exp(-\tau)$.

The strategy to use at low ionization parameters therefore is to use the `etable` to find a reasonably good fit to the spectrum with xspec, and then run xstar2xspec and make a very small table, i.e. just 1 column and 2 ionization parameters (xspec needs tables to have at least 2 entries), with constant abundances, set to the values which come from the fit. Constant abundance grids when used as `mtable` do not suffer from any of these problems.

6.1.3 Energy Resolution and Line Widths in Table Models

Users may find that turbulent velocity $v_{\rm turb}$ can make a significant difference in the results of xstar modeling of absorption spectra, and the reason is at least partially due to numerics. In calculating the energy dependent opacity xstar tries to be clever: it calculates the profile function on a very fine temporary internal energy grid and then integrates this quantity over the current energy bins. In each bin, the opacity is calculated such that the equivalent width in that bin is equal to the equivalent width that comes from the fine grid calculation. This provides a fairly accurate opacity distribution, which is better than eg. evaluating the line profile function only on grid boundaries.

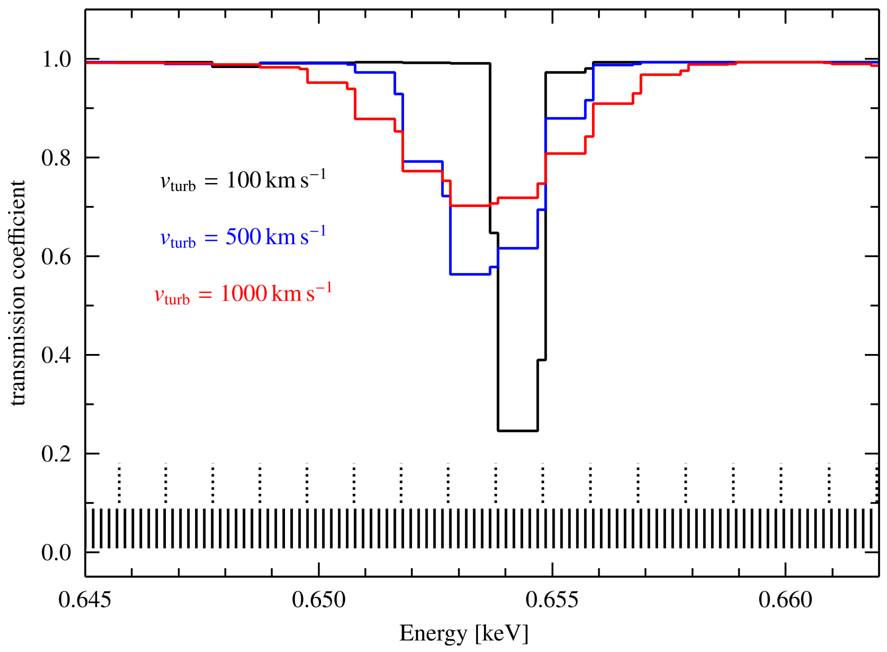


Fig. 3: Close-up of O VIII Lyα lines in `etable` models calculated for different turbulent velocities. The solid lines indicate the energy grid (that the model gets evaluatued on), the dotted lines the model grid (that the model gets calculated on).

6.1.4 Notes on Units and Normalization in Table Models

The problem with creating a flexible tool for modelling emission and absorption is that there have several free parameters affecting real spectra, including: source luminosity, distance, reprocessor column density, ionization parameter, and geometry. By geometry we mean covering fraction around the source, which affects emission, and covering fraction across our line of sight to the source, which affects absorption. With the xstar2xspec tables you should be able to model a wide range of choices for this, but there is not a unique one-to-one mapping between the values of these physical parameters and the values used in running xstar and constructing the tables.

The free parameters which can be varied when running xstar2xspec include the abundances, column density, gas density, and ionization parameter. These all have a straightforward intepretation as physical parameters. The emitter normalization and geometry are not uniquely determined, owing to the ambiguity between source luminosity and distance.

It is helpful here to be very specific, at the risk of being repetitive. The procedure followed by xstar2xspec in making tables is:

(1) Generate a sequence of command line calls to XSTAR (this is done by the tool xstinitable)

(2) Step through the calls to XSTAR

(3) Calculate each appropriate XSTAR model

(4) Take the spectrum output of the model (the file xout_spect1.fits) and convert it to the right units and append it to the fits table (xout_aout.fits, xout_ain.fits, or xout_mtable.fits). The last step is done by the tool xstar2table.

The conversion is as follows: For additive models (`atable`), xspec expects a bin-integrated spectrum in units of photons $\mathrm{s}^{-1}\,\mathrm{cm}^{-2}$. We denote this quantity F_n^{mod} for bin n. In XSTAR, however, the user specifies a bolometric source luminosity $L_{\mathrm{tot}}^{\mathrm{xstar}}$ in units $10^{38}\,\mathrm{erg\,s}^{-1}$ and XSTAR returns the specific luminosity $\mathcal{L}_\varepsilon^{\mathrm{xstar}}$ in units of $10^{38}\,\mathrm{erg\,s}^{-1}\,\mathrm{erg}^{-1}$ at each energy ε of the model grid. At distance D from the source, the observed flux in bin n is therefore

$$F_n^{\mathrm{mod}} = \frac{\mathcal{L}_\varepsilon^{\mathrm{xstar}}}{4\pi D^2}\left(\frac{\Delta\varepsilon}{\varepsilon}\right)$$

where $\Delta\varepsilon/\varepsilon$ is the fractional energy bin size. xstar has no information about the distance so assuming a standard distance of $D = 1$ kpc and renormalizing to an incident luminosity of $10^{38}\,\mathrm{erg\,s}^{-1}$ gives

$$F_n^{\mathrm{mod}} = 8.356\times10^{-7}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1} \times \left(\frac{\mathcal{L}_\varepsilon^{\mathrm{xstar}}}{10^{38}\,\mathrm{erg\,s}^{-1}\,\mathrm{erg}^{-1}}\right)\left(\frac{L_{\mathrm{tot}}^{\mathrm{xstar}}}{10^{38}\,\mathrm{erg\,s}^{-1}}\right)^{-1}\left(\frac{\Delta\varepsilon}{\varepsilon}\right)$$

The meaning of this quantity and the normalization can be better understood if we consider how xspec works in more detail. In order to calculate the model counts in bin m, xspec multiplies the F_n^{mod} vector with the response matrix A_{nm} and a normalization factor κ:

$$C_m^{mod} = \kappa\sum_n F_n^{mod} A_{nm}$$

Note that xspec will always add such a normalization factor to each additive model so that absolute values distance and intrinsic luminosity do not need to be known at the time of the model calculation in order to fit an observed spectrum. xspec will renormalize the model internally and the κ can be made a free fit parameter like any other. The normalization factor κ will also account or effects like partial covering geometries with covering fractions smaller than unity.

The two equations above illustrate how κ is connected to physical parameters such as distance and luminosity. So, for example, if your `atable` model fits to the data with a normalization of $\kappa = 1$, and you used a luminosity in creating the table, then this would imply that your data was consistent with a full shell illuminated by a luminosity of $10^{38}\,\mathrm{erg\,s}^{-1}$ at a distance of 1 kpc, or it also could be a shell illuminated by a luminosity of $10^{44}\,\mathrm{erg\,s}^{-1}$ at a distance of 1000 kpc.

For another example, let's say you know the distance to the source is 1 kpc and the luminosity is $10^{38}\,\mathrm{erg\,s}^{-1}$, but the best fit has an emitter normalization of $\kappa = 0.1$. This would suggest that rather than a full sphere, the emitter only subtends 10% of the solid angle around the source.

Obviously, the column density of the emitter is important also. If you do not know the column density, then a shell of column density $10^{19}\,\mathrm{cm}^{-2}$ illuminated by a $10^{38}\,\mathrm{erg\,s}^{-1}$ source will probably have very similar emitted X-ray spectrum to a shell of column density $10^{20}\,\mathrm{cm}^{-2}$ illuminated by a $10^{37}\,\mathrm{erg\,s}^{-1}$ source. If there are absorption features in the spectrum, then they may constrain the column density.

6.2 Running XSTAR2XSPEC

After this lengthy introduction, we now turn to how to actually use XSTAR2XSPEC. The parameter handling for XSTAR2XSPEC is designed to be as flexible as possible, in principle, limited only by the physical resources (RAM, disk space & CPU time) available on your machine. You have the choice of varying *any* of the physical parameters used as input in an XSTAR model. XSTAR2XSPEC is called from the command line very much like xstar and prompts the user in a similar way to enter parameters.

```
xstar2xspec [options]
```

The available options are currently:

- `-save`: Save the spectral FITS files, modifying the file name to include the value of the loopcontrol variable for better identification. Note this can use GBs of disk space.
- `-verbose`: Generate more diagnostic messages (applies to the XSTAR2XSPEC script only).
- `-restart`: Continues the XSTAR2XSPEC run using the previous run. Note that it does *NOT* check the integrity of the table files from the terminated run. This is the user's responsibility.

The interface for entering the prompted parameters is very similar to the one of XSTAR and should be already familiar. However, when inspecting the `$PFILES` directory, user won't find a parameter file associated with XSTAR2XSPEC. This is because XSTAR2XSPEC is a Perl script that calls a FTOOL programs that each have their own parameter files. The relevant tools are:

- **xstinitable:** This routine processes most of the input parameters and creates the parameter grid. It then produces a list of all XSTAR calls needed to calculate the full model. It also creates the (empty) FITS files that will store the model with appropriate dimensions. Changes to hidden xstar parameters need to be made through the parameter file "xstinitable.par".
- **xstar2table:** This routine takes care of the post-processing of each XSTAR run. It converts the information in the output spectral file "xout_spect1.fits" into emission spectra, absorption coeficients, optical depths etc., and fills them in into the table model. xstar2table is called every time an individual XSTAR calculation has finished.

6.3 Input Parameters

This section gives an overview over the input parameters to XSTAR2XSPEC and how they can be varied. We distinguish between parameters that are directly connected to physical properties of the photoionzed plasma and control parameters that affect the computational behavior of the code. Obviously only the physical parameters can be made variable to produce a spectral model that can be fitted to data.

6.3.1 Physical Parameters

XSTAR models are based on 21 physical parameters described in detail in Chapter *User Input to XSTAR*. Each of these parameters can be varied in different ways or held constant for all XSTAR runs. Due to the limitations of table models mentioned in earlier sections, user will usually only vary two to four interpolated parameters and possibily a few additional parameters, while all other parameters are held constant. The levels of variability for each parameter are:

- **Constant (variation type = 0):** This parameter is held constant in all the XSTAR runs.
- **Additive (variation type = 1):** The additive class of parameters provides a simple method for varying parameters that are reasonably independent of the others. For more info on how additive parameters function, see the XSPEC manual and see OGIP memo 92-009.

- **Interpolated (variation type = 2):** Interpolated parameters provide the greatest accuracy in building table models. They also require the most processing time. For each interpolated parameter, you can define some number of points between a maximum and minimum range. The placement of these intermediate points is determined by the interpolation type – linear (interpolation type = 0) or logarithmic (interpolation type = 1).

It is useful to take a look at the parameter file xstinitable.par that controls the variation of each parameter. This file is many entries so we only show a few exerpts below for illustration. While for single XSTAR runs, it is not recommended to manipulate the parameter file, for XSTAR2XSPEC it can be a convient method to configure the table model calculation.

```
[...]
density,r,a,1.E+12,1.e+4,1.E21,"density soft maximum (cm**-3)"
densitytyp,i,h,0,0,2,"density variation type"
densityint,i,a,1,0,1,"density interpolation type"
densitysof,r,a,0.,1.e+4,1.e+18,"density soft minimum"
densitynst,i,a,1,1,20,"density number of steps"
[...]
```

Here, density is held constant (densitytyp=0). All XSTAR runs will use the (“soft maximum”) density of 10^{12} cm^{-3}.

```
[...]
column,r,a,1.e+24,1.e+10,1.E25,"column density soft maximum (cm**-2)"
columntyp,i,h,2,0,2,"column density variation type"
columnint,i,a,1,0,1,"column density interpolation type"
columnsof,r,a,1.E+21,1.,1.E25,"column density soft minimum"
columnnst,i,a,7,1,20,"column density number of steps"
rlogxi,r,a,5.,-10.,+10.,"log(ionization parameter) soft maximum (erg cm/s)"
rlogxityp,i,h,2,0,2,"log(ionization parameter) variation type"
rlogxiint,i,a,0,0,1,"log(ionization parameter) interpolation type"
rlogxisof,r,a,1.,-10.0,+10.0,"log(ionization parameter) soft minimum"
rlogxinst,i,a,9,1,40,"log(ionization parameter) number of steps"
[...]
```

Here, column density and ionization parameter are *interpolated parameters* (variation type=2). The column density will be varied from 10^{21} cm^{-2} to 10^{24} cm^{-2} in seven steps. The ionization parameter will be varied from 1 to 5 in 9 steps. The column density will be varied in logarithmic steps (interpolation type=1), since it spans several orders of magnitude. The parameter `rlogxi` is already the log of the ionization parameter and is therefore sampled in linear steps (interpolation type=0).

```
[...]
feabund,r,h,1.,0.,100.,"iron abundance soft maximum"
feabundtyp,i,h,1,0,2,"iron abundance variation type"
feabundint,i,h,1,0,1,"iron abundance interpolation type"
feabundsof,r,h,0.,0.,1.,"iron abundance soft minimum"
feabundnst,i,h,1,1,20,"iron abundance number of steps"
[...]
```

In this example, the iron abundance is an *additive parameter* (variation type=1). The limits here are 0 and 1 with only one step. This means is all interpolated parameter combinations will be calculated once with the iron abundance set to 0 and once with the iron abundance set to 1.

6.3.2 Control Parameters

In addition to the parameters that define physical properties of the photoionized plasma, there is a number of non-physical parameters that control the behavior of the code. These parameters are never varied but are also included in the parameter file xstinitable.par so the user has full control over them. Additional parameters that are relevant to XSTAR2XSPEC but not to XSTAR are:

- `elow`: This parameter determines the low energy end (in eV) of the spectrum selected from the XSTAR output files.
- `ehigh`: This parameter determines the high energy end (in eV) of the spectrum selected from the XSTAR output files.

```
[...]
elow,r,h,1.0E+2,0.,5.11E+5,"energy band low end (eV)"
ehigh,r,h,2.0E+4,0.,5.11E+5,"energy band high end (eV)"
[...]
```

As an example, spectral range here is 100 eV–20 keV.

6.3.3 Runtime Estimates

In estimating the running time of XSTAR2XPEC, the key factor is the number of times XSTAR is called. Consider a run with $\mathrm{N_I}$ interpolated parameters, where interpolated parameter i is evaluated at n_i points ($1 \le i \le \mathrm{N_I}$). The total number of times XSTAR is called is then $\prod_{\mathrm{i=1}}^{\mathrm{N_I}} n_i$. However, if $\mathrm{N_A}$ additive parameters are also defined, then for each set of interpolated variables, there is one run with all the additive parameters are zero and the remaining $\mathrm{N_A}$ runs have one of the additive parameters at it's maximum value and the rest all zero. This means that the total number of times XSTAR must be run is given by

$$(\mathrm{N_A} + 1) \prod_{\mathrm{i=1}}^{\mathrm{N_I}} n_i$$

and this can provide you with a feel for how long a complete XSTAR2XSPEC run will require. As an example, if we defined 2 interpolated parameters (one evaluated at 5 points and the other evaluated at 4 points) and 8 additive parameters, XSTAR would be run a total of

$$(8 + 1) \times 5 \times 4 = 180 \text{ calls}$$

which means that if the XSTAR runs for the appropriate model range averages five minutes, it will take approximately 180×5 minutes = 900 minutes = 15 hours for a complete XSTAR2XSPEC run.

6.4 Output Files

XSTAR2XSPEC produces five output files:

(1) xstar2xspec.log is a concatenation of all the xout_step.log files from all the xstar runs called by XSTAR2XSPEC.

(2) xout_ain.fits is a FITS file containing the `atable` with the reflected emission spectrum produced by XSTAR.

(3) xout_aout.fits is FITS file containing the `atable` with the emission spectrum in the forward (transmitted) direction produced by XSTAR.

(4) xout_mtable.fits is a FITS file containing the `mtable` with the absorption spectrum in the forward (transmitted) direction produced by XSTAR.

(5) xout_etable.fits is a FITS file containing the `etable` with the absorption spectrum in the forward (transmitted) direction produced by XSTAR.

All the FITS files are formatted that they can directly loaded in xspec. Here we load the forward emission `atable` model of grid18:

```
unix~> xspec
             XSPEC version: 12.13.1
     Build Date/Time: Fri Feb  9 12:12:40 2024

XSPEC12>model atable{xout_aout.fits}

[...]

========================================================================
Model atable{xout_aout.fits}<1> Source No.: 1   Active/Off
Model Model Component  Parameter  Unit     Value
 par  comp
   1    1   template   column              1.000000E+21  +/-  0.0
   2    1   template   rlogxi              0.0           +/-  0.0
   3    1   template   cabund              1.00000      frozen
   4    1   template   nabund              1.00000      frozen
   5    1   template   oabund              1.00000      frozen
   6    1   template   neabund             1.00000      frozen
   7    1   template   mgabund             1.00000      frozen
   8    1   template   siabund             1.00000      frozen
   9    1   template   sabund              1.00000      frozen
  10    1   template   arabund             1.00000      frozen
  11    1   template   caabund             1.00000      frozen
  12    1   template   feabund             1.00000      frozen
  13    1   template   niabund             0.0          frozen
  14    1   template   z                   0.0          frozen
  15    1   template   norm                1.00000      +/-  0.0
 ______________________________________________________________________

 ***Warning: Magnitudes of parameters in model for data group 1
      have been found to differ by more than 1e+10.
      This can lead to significant roundoff errors during fit calculation.

 XSPEC12>
```

CHAPTER

SEVEN

WARMABS

An alternative method for fitting XSTAR results to observed spectra within xspec is to use the xspec “analytic” model warmabs. This actually includes several separate models: warmabs, photemis, hotabs, hotemis, and multabs. These provide xspec models for warm absorbers and photoionized emitters, and for coronal equilibrium absorbers and emitters without requiring construction of `mtables` or `etables`. The models use pre-calculated “population files” which are standard xstar output files that store information on charge state distributions, level populations, line emissivities etc. The warmabs models then calculate synthetic spectra from these files, basically skipping all the steps associated with solving the rate equations. This approach has several advantages:

First, it circumvents the intrinsic approximations associated with use of tables for absorption with variable abundances treated as multiplicative parameters (see *Variable Abundances in Table Models*). Warmabs/photemis/windabs/multabs calculate spectra using stored level populations which are then scaled using element abundances specified during the xspec session before the spectra are calculated. Therefore the approximations associated with the use of tables are avoided. (However, see the section below on the current limitations of these models).

Second, the provide the ability to use arbitrary spectral resolution, not limited by the internal xstar spectral resolution or the fixed energy grid at the time of table creation. Finally, they allow the use of turbulent broadening as a fitting parameter in xspec.

Limitations of these models are:

Most importantly, it calculates the spectrum ‘on the fly’, appropriate to the energy grid and parameters the user specifies. But it does not calculate the ionization balance self-consistently. It uses a saved file of level populations calculated for a grid of *optically thin* models calculated with a $\Gamma = 2$ *power law ionizing spectrum*. So this will not be self-consistent if your source has a very different ionizing spectrum. Also it implicitly assumes that the absorber has uniform ionization even if you specify a large column, which is not self-consistent.

It is also not blindingly fast. On a 1 GHz machine, the first time ‘mo warmabs ..’ is typed, the initial setup requires ~30 seconds. After that, each time an abundance or ionization parameter is changed, it requires 5-10 seconds to recalculate the model.

Similar to a classic tabulation the population files store charge state and level populations on a grid of ionization parameters. We expect the resolution of $\log \xi$ to be sufficient for most of the time but the user should be aware that there is an internal discretization with regard to ionization parameter and some cases may require recalculation of the population files using a smaller stepsize in $\log \xi$.

Whether photemis takes into account continuum photoexcitation or not will depend on the value of the cfrac parameter used in calculating the populations files. As with an xstar calculation, cfrac=1 ignores continuum photoexcitation and cfrac=0 includes it. The standard populations files distributed with warmabs use cfrac=0.

This package is under constant development. Some embarrassing bugs have already been found, but more may still lurk. Please contact the helpdesk with any reports or questions.

7.1 Obtaining warmabs

The warmabs package is not included as part of the standard xstar distribution within heasoft. Instead, it must be downloaded and installed separately from the FTP site. The current release can be obtained from:

https://heasarc.gsfc.nasa.gov/FTP/software/plasma_codes/xstar/warmabs247.tar.gz

The contents of the tarfile include:

- **atdbwarmabs.fits**
  xstar atomic database binary fits file. This must reside in directory pointed to by the `WARMABS_DATA` environment variable. This file has the same format as the atdb.fits file used by xstar, but is kept distinct. This is important because the structure of the populations files (described below) depend on the version of the atomic data used in the run which created them. Thus warmabs *must use the same version of this data file as was used in the calculation of the population files*. In spite of the revision process, by which we try to keep the release versions of this file used by xstar and warmabs the same, it is possible for the two to get out of sync. If this happens unpredictable, incorrect, and not necessarily fatal errors may occur. The user should beware of this potential problem.
- **coheatwarmabs.dat**
  compton heating/cooling data file. This must reside in directory pointed to by `WARMABS_DATA` environment variable. This file is identical to the file coheat.dat which is part of xstar.
- **fphotems.f**
  source code for warmabs, photemis, multabs, hotemis, and hotabs
- **lmodel_warmabs.dat, lmodel.dat**
  local model definition file needed by xspec.
- **PARAMX**
  file containing parameter statements setting array dimensions needed by fphotems.f
- **README**
  additional information

7.2 Installation

The procedure for setting up and using this model is as described in the xspec manual. You need to have the heasoft package installed on your machine, but it must be built from source. Local models cannot be installed from a binary installation.

Download and untar this directory somewhere in your user area

```
wget https://heasarc.gsfc.nasa.gov/FTP/software/plasma_codes/xstar/warmabs247.tar.gz
tar xfz warmabs247.tar.gz
```

Setup and initialize the HEADAS environment:

```
export HEADAS=</path/to/your/headas/>
source $HEADAS/headas-init.sh
```

Set the `WARMABS_DATA` enviroment variable to point to the location of the population files, atomic database and Compton heating file:

```
export WARMABS_DATA=`pwd`
```

Start xspec and compile warmabs inside xspec or just pipe the compile command to xspec.

```
echo "initpackage xstarmod lmodel.dat $WARMABS_DATA" | xspec
```

After the build is complete, the model can be loaded in xspec using the `lmod` command. The `WARMABS_DATA` environment variable needs to be defined whenever warmabs is used.

```
XSPEC12>lmod xstarmod </path/to/installation>
```

In subsequent sessions you don't neet to do the initpackage step again, just `lmod` and `WARMABS_DATA`.

7.3 warmabs

Inside of xspec, the model can be invoked by typing

```
XSPEC12> mo warmabs*pow
```

or variations on that. The input parameters include:

column
Log of the absorption column density in units of 10^{22} cm^{-}2.

rlogxi
Log of the ionization parameter. See *Log of the ionization parameter (rlogxi)* for details.

Cabund...Znabund
Elemental abundance relative to the xspec abundance table for elements C though Zn.

write_outfile
Switch to write information of the strongest 500 lines to a FITS file.

outfile_idx
Integer number to index the output FITS files. Useful if more than one warmabs component is used.

vturb
Turbulent velocity broadening. See *Turbulent Velocity (vturbi) -hidden* for details.

Redshift
Redshift applied to the whole model.

7.4 photemis

The photoionized emitter can be invoked by typing

```
XSPEC12> mo photemis
```

This model is the 'thermal' (i.e. recombination and collisional excitation) emission which comes from the same plasma used in warmabs. Note that this does may or may not include the effect of continuum photoexcitation on the level populations and line emission. Whether it does depends on the value of the cfrac parameter used in creating the populations files.

Parameters of photemis are almost identical to those of warmabs, except that instead of the column density, a normalization parameter rescales the line fluxes.

norm
The model supplies to xspec the emissivity of the gas, in units of erg cm^{-3} s^{-1}, times a factor 10^{10}. So the physical

meaning of the normalization κ, is

$$\kappa = \frac{\mathrm{EM}}{4\pi D^2} \times 10^{-10}$$

where EM is the emission measure of the gas in the source (at the ionization parameter used in the fit) and D is the distance to the source.

7.5 multabs

Multabs tries to account for line broadening by absorption by multiple discrete components rather than by turbulence or bulk flow. This model is essentially identical to warmabs in that it uses the warm absorber spectrum generated by warmabs, but initially assuming that the lines are broadened only by thermal gas motions. It then replicates these lines a fixed number of times and spreads the components over a given velocity width. The input parameters include the same parameters as warmabs, column, ionization parameter, and elemental abundances. These control the properties of the individual absorbing components, in the same way as for warmabs. In addition, the user specifies the velocity spread of the components, still called vturb, and the covering fraction, cfrac. This covering fraction can be interpreted as a covering fraction in velocity space. The number of discrete components, n_{comp} is given by

$$n_{\mathrm{comp}} = \mathtt{cfrac} \times \frac{v_{\mathrm{turb}}}{v_{\mathrm{therm}}} ,$$

where v_{turb} is input by the user and v_{therm} is the thermal line width which is determined by the equilibrium temperature (calculated by xstar). The optical depth of each component is divided by the number of components, so that the total optical depth summed over the components is independent of their number. The number of components cannot be less than 1. If it is 1 then the component will be placed at the redshifted energy of the line. If it is greater than 1, then the components will be spaced uniformly in velocity from $-v_{\mathrm{turb}}$ to $+v_{\mathrm{turb}}$ relative to the rest energy of the line. There is no restriction on the value of cfrac, so it is possible to set cfrac to some large number and thereby fill a uniform trough in velocity with the line.

`column`
: See *warmabs*.

`rlogxi`
: See *warmabs*.

`Cabund...Znabund`
: See *warmabs*.

`write_outfile`
: See *warmabs*.

`outfile_idx`
: See *warmabs*.

`vturb`
: Velocity width the individual components are spread over. The components range from $-v_{\mathrm{turb}}$ to $+v_{\mathrm{turb}}$.

`cfrac`
: Covering fraction in velocity space. Determines the number of velocity components.

`Redshift`
: See *warmabs*.

7.6 hotabs

Hotabs is the coronal analog of warmabs. In this case the free parameter determining the ionization balance is the log of the temperature in units of 10^4 K. All other parameters are the same as in warmabs.

`column`
: See *warmabs*.

`logt4`
: Log of the temperature in units of 10^4 K, i.e. `logt4` = 0 corresponds to 10^4 K and `logt4` = 3 corresponds to 10^7 K.

`Cabund...Znabund`
: See *warmabs*.

`write_outfile`
: See *warmabs*.

`outfile_idx`
: See *warmabs*.

`vturb`
: See *warmabs*.

`Redshift`
: See *warmabs*.

7.7 hotemis

Hotemis is the coronal analog of photemis. In this case the free parameter determining the ionization balance is the log of the temperature in units of 10^4 K. All other parameters are the same as in warmabs/photemis.

`norm`
: See *photemis*.

`logt4`
: Log of the temperature in units of 10^4 K, i.e. `logt4` = 0 corresponds to 10^4 K and `logt4` = 3 corresponds to 10^7 K.

`Cabund...Znabund`
: See *warmabs*.

`write_outfile`
: See *warmabs*.

`outfile_idx`
: See *warmabs*.

`vturb`
: See *warmabs*.

`Redshift`
: See *warmabs*.

7.8 Creating Your Own Popoulation Files

All the models described in this chapter rely on pre-calculated files containing ion populations and line emissivities and opacities. The default distribution contains a set of these files but the user is strongly recommended to verify wether the default distribution files are appropriate for their use case (e.g., similar spectral energy distribution, density etc). We expect that the majority of users will need to recalculate population files specific for their science case.

The population files are calculated by running xstar with setup where the spatial zones cover a wide range of ionization parameters. **SEE THREAD**

7.9 Common block 'ewout'

A feature added September 2007 is output of the strongest lines, sorted by element and ion into a common block called 'ewout' This feature is only available for the warmabs, photemis, hotemis, hotabs models (not windabs, or multabs). The contents of the common block are:

lmodtyp: identifies which model most recently put its output into the
common block. lmodtyp=1,2,3,4, where 1=hotabs, 2=hotemis, 3=warmabs, 4=photemis

newout: number of lines in the list. This is zeroed after each call
to warmabs, photemis, hotabs, etc.

lnewo: array conatining line indexes. These should correspod to the
line indexes in the ascii line lists on the xstar web page.

kdewo: character array containing the name of the ion

kdewol: character array containing the name of the lower level

kdewou: character array containing the name of the upper level

aijewo: array containing A values for the lines

flinewo: array containing f values for the lines

ggloewo: array containing statistical weights for the lower levels

ggupewo: array containing statistical weights for the upper levels

elewo: array containing the line wavelengths

tau0ewo: array containing the line center depths

tau02ewo:array containing the line depths at the energy bin nearest to line center

ewout: array containing line equivalent widths in eV, negative values correspond to emission

elout: array containing line luminosities in xstar units (erg/s/10^38)

The details of how to get at the contents of the common block are up to the user. Currently xspec does not have a mechanism to do this, but it is straightforward to write a small fortran code to call the models with suitable parameter values and print the common block from there. The calling sequence for an analytic model is described in the xspec manual. It is important to point out that the common block is overwritten at each call to one of the models, so it should be emptied by the calling program after each call to one of the models.

An example is as follows:

```
      program fphottst
c
      implicit none
```

(continues on next page)

(continued from previous page)

```
 c
     real ear(0:20000),photar(20000),photer(10000),param(30)
     integer ne,mm,ifl
     real emin,emax,dele
 c
     ne=10000
     emin=0.4
     emax=7.2
     dele=(emax/emin)**(1./float(ne-1))
     ear(0)=emin
     do mm=1,ne
       ear(mm)=ear(mm-1)*dele
       enddo
     write (6,*)ear(1),ear(ne),ear(ne/2)
     param(1)=2.
     param(2)=-4.
     param(13)=100.
     param(12)=0.
     param(3)=1.
     param(4)=1.
     param(5)=1.
     param(6)=1.
     param(7)=1.
     param(8)=1.
     param(9)=1.
     param(10)=1.
     param(11)=1.
     param(3)=0.
     call fhotabs(EAR,NE,PARAM,IFL,PHOTAR,PHOTER)
c
     call commonprint
c
     write (6,*)'after fwarmabs'
     do mm=1,ne
       write (6,*)ear(mm),photar(mm)/ear(mm)
       enddo
c
     stop
     end
     subroutine commonprint
c
     implicit none  !jg
c
     parameter (nnnl=200000)
c
     common /ewout/newout,lnewo(nnnl),kdewo(8,nnnl),
    $  kdewol(20,nnnl),kdewou(20,nnnl),aijewo(nnnl),flinewo(nnnl),
    $  ggloewo(nnnl),ggupewo(nnnl),
    $  elewo(nnnl),tau0ewo(nnnl),tau02ewo(nnnl),ewout(nnnl),
    $  elout(nnnl),lmodtyp
c
     real aijewo,flinewo,ggloewo,ggupewo,elewo,tau0ewo,tau02ewo,
```

(continues on next page)

(continued from previous page)

```
   $  ewout,elout
    integer lnewo,newout,lmodtyp
    character*1 kdewo,kdewol,kdewou
    integer kk,mm   !jg
c
     if (lmodtyp.eq.1) write (6,*)'after hotabs',newout
     if (lmodtyp.eq.2) write (6,*)'after hotemis',newout
     if (lmodtyp.eq.3) write (6,*)'after warmabs',newout
     if (lmodtyp.eq.4) write (6,*)'after photemis',newout
     write (16,*)'index, ion, wave(A), tau0, tau0grid, ew (eV),',
   $ 'lum, lev\_low, lev\_up, a\_ij, f\_ij, g\_lo, g\_up'
      do kk=1,newout
       write (16,9955)kk,lnewo(kk),(kdewo(mm,kk),mm=1,8),
   $     elewo(kk),tau0ewo(kk),tau02ewo(kk),ewout(kk),
   $     elout(kk),
   $     (kdewol(mm,kk),mm=1,20),(kdewou(mm,kk),mm=1,20),
   $     aijewo(kk),flinewo(kk),ggloewo(kk),ggupewo(kk)
       enddo
9955   format (1x,2i8,1x,8a1,5(1pe11.3),1x,2(20a1,1x),4(1pe11.3))
c
    return
    end
```

CHAPTER

EIGHT

PROBLEMS AND PITFALLS

XSTAR has been designed to be as "user-friendly" as possible while still maintaining a large amount of flexibility. However, experience has shown that it is difficult to guard against the many possible misuses of the code, and that it is impossible to generate a code which is completely free from errors or unintended features. In this chapter we list what we consider to be some of the most probable pitfalls of the use of XSTAR. This chapter should be read by any user who runs XSTAR2XSPEC or who ventures very far beyond the default parameter values of XSTAR itself.

8.1 Mtables

If you are using `mtables` with variable abundances, then you will likely get totally unphysical results. See *Variable Abundances in Table Models* for details.

8.2 Execution Time

XSTAR is designed to strike a balance between accuracy and speed, but this inevitably involves some disparity between different computing platforms. As a result, many problems of interest require large amounts of time on machines which are relatively slow, or which are heavily used for other tasks.

In an effort to avoid wasted time (and CPU cycles) we offer the following suggestions: (i) XSTAR does not attempt to calculate ionization, excitation, etc. for elements whose abundances are specified to be less than 10^{-15} relative to hydrogen. Large reductions in computation time can be achieved by zeroing the abundances of elements which are likely to be unimportant anyway, for example, calcium, argon, and nickel. (ii) For some purposes constant temperature is an adequate approximation, and is often a useful preliminary step in deciding parameter values such as column density, and require a fraction of the execution time of full thermal equilibrium models. (iii) For some purposes a low column density ($\leq 10^{18}$ cm^{-2}) will provide sufficient information. Large column densities require significantly more execution time. If large columns are needed, then execution can be speeded up by use of a large value of `emult`, or a small value of `taumax`, and by setting `npass` to 1. (iv) Reduce the number of continuum bins on input. Execution time is approximately proportional to the number of continuum bins.

The parameter files included in the source tree for both xstar and xstar2xspec are set to perform constant temperature models, in order to allow the user to become familiar with xstar without requiring large investments in computer time.

8.3 Low Density

Although all rates should extrapolate correctly at low density, the level population calculation requires the inversion of large matrices of rates. In most cases, the largest elements of the matrix are the spontaneous decay rates (A values) for allowed transitions. For highly charged ions these can exceed 10^{13} s^{-1}. At low densities the smallest rates of interest are the collisional rates and recombination rates. XSTAR attempts to avoid inverting singular matrices by discarding rows and columns whose largest elements differ from the largest element in the matrix by more than the machine precision (and then assuming the populations in the associated levels are zero). This can have the effect of producing inaccurate solutions particularly at low densities since some physically important transitions (notably recombination) may be discarded. Unfortunately, there is no clear way of automatically informing the user when this is happening. A rough rule of thumb is that densities less than approximately 1000 should be avoided when iron may be ionized beyond Fe XVII.

An indication of possible numerical problems is given by the final integer on each line of the output log file. This is the number of iterations required in order to reach thermal equilibrium and charge neutrality. Models with good convergence will typically have values of 5 or less for this quantity, except for the first step and possibly near ionization fronts. Otherwise, if this integer is large (i.e. greater than, say, 20), and if the value of heating - cooling ("h-c") is greater than 1–2%, then it is possible that the density is lower than can be treated accurately by XSTAR.

8.4 High Density

All level populations are affected by collision rates to and from the superlevels. These rates are calculated by fitting to more complete population kinetic calculations involving hundreds of levels; these fits are valid only up to (electron) densities of 10^{22} cm^{-3}. However, for densities greater than 10^{18} cm^{-3} special atomic data files should be used. Please contact us in this case.

8.5 The Energy Grid

Many of the most important components of the XSTAR calculation require numerical quadratures over energy, and these are generally carried out using straight-forward trapezoid quadratures over a fixed grid of energies. In addition, the computation is speeded by the use of a strictly logarithmic grid spacing in energy. We use 9999 energies spaced logarithmically from 0.1 eV to 20 keV. This results in a 0.12% grid spacing, corresponding to, e.g. 8.6 eV at 7 keV. This is the energy resolution of the code. Finer resolution can be implemented if larger values of ncn2 are used, up to a limit of ncn2=999999.

8.6 The Ionizing Spectrum

The ionizing spectrum has the obvious effect of creating ionization in the gas. But it also can influence the heating and cooling via Compton scattering if the gas is highly ionized. The standard ionizing spectrum options apply to all the energies in the grid. Therefore if, for example, an ε^{-1} power law is chosen the temperature in Compton equilibrium will be $kT = (\varepsilon_{max} - \varepsilon_{min})/(4\ln(\varepsilon_{max}/\varepsilon_{min}))$, and power law indeces which are greater (or less) than 1 will be influenced even more strongly by the choice of minimum (or maximum) energy. It is likely that the default choice we have made will not be the choice which is physically appropriate for the situation of interest, so the user is encouraged to input the spectral model from a file, with the appropriate cutoffs built in, in situations where power law spectra and Compton heating/cooling are important.

8.7 Energy Budget

In models where thermal equilibrium is imposed the total amount of energy absorbed from the incident radiation field should balance the total emitted energy in lines and continua. This constraint is not automatically satisfied, since the algorithm for calculating heating and cooling rates is based on the assumption that each spatial zone is at most marginally optically thick. As a check the total energy absorbed and emitted from the radiation field is printed at the end of the log file, along with the fractional error in energy conservation. An error greater than a few percent indicates an inaccuracy in the model results which may significantly compromise emission line strengths, for example. Such models should propbably be rerun with smaller values of `emult`.

CHAPTER

NINE

PYXSTAR

9.1 PyXstar Import and Setup

The PyXstar module `pyxstar` is imported with the usual Python statement leading to the setup and loading of the Xstar atomic database, which will be kept in RAM throughout the use of the module. It is also convenient to import the `math` and Matplotlib modules at this stage.

```
>>> import matplotlib.pyplot as plt
>>> import math
>>> import pyxstar as px
Xstar database has been loaded. PyXstar is ready to go.
```

9.2 Function to Access the Xstar Database

`get_data`(*ion*, *rtype*[, *llo=0*, *lup=0*, *temperature=0.0*])

Returns atomic datasets from the Xstar database for a particular ionic species.

Parameters:

`ion : character string`
The string specifies the ionic species using the Xstar notation; for instance, 'o_iii' for O III.

`rtype : integer or 2-character string`
Denotes the dataset to be retrieved:

- 13 or 'LV' : energy levels
- 4 or 'LA' : bound-bound radiative transitions
- 7 or 'PI' : photoionization cross sections
- 3 or 'EC' : electron effective collision strengths
- 41 or 'AU' : Auger transition rates

`llo : integer`
Lower level index of the ion. For `rtype` = 'LV', `llo` = 0 returns all the levels while `llo` = -1 returns the continuum. For `LA`, `PI`, `EC`, and `AU`, `llo` = 0 returns all the transitions for `llo` < `lup`.

`lup : integer`
Upper level index of the ion. For `rtype` = `LA`, `PI`, `EC`, and `AU`, `lup` = 0 returns all transitions with `lup` > `llo`.

Note

For `llo` = `lup` = 0, all transitions are listed.

`temperature :  float or list or tuple or Numpy array of floats`

Temperature array in units of 10^4 K for `rtype` ='EC'.

Returns:

`df` : Dataframe for `rtype` = 'LV', 'LA', and 'AU'

`df[0]` : Dataframe with transition identifiers for `rtype` = 'PI' and 'EC'.

`df[1]` : List of dataframes with energy tabulations of photoionization cross sections (`rtype` = 'PI') or temperature tabulations of effective collision stregths (`rtype` = 'EC').

Note

For `rtype` = 'EC' and `temperature` = 0.0, the dataframes give the origial data in Chianti format or temperature tabulations. For `temperature` != 0.0, they give effective collision strengths at the requested temperature tabulation.

Examples

Print the energy level structure of O III:

```
>>> y = px.get_data(ion='o_iii',rtype='LV')
>>> print(y)
     ion  Z  N  index           level  n  (2S+1)  L  stat_wt      energy
0   o_iii  8  6      1        2p2.3P_0  2       3  1      1.0    0.000000
1   o_iii  8  6      2        2p2.3P_1  2       3  1      3.0    0.014054
2   o_iii  8  6      3        2p2.3P_2  2       3  1      5.0    0.038022
3   o_iii  8  6      4        2p2.1D_2  2       1  2      5.0    2.512230
4   o_iii  8  6      5        2p2.1S_0  2       1  0      1.0    5.351830
..    ... .. ..    ...             ... ..     ... ..      ...         ...
74  o_iii  8  6     75    1s1.2p5.3P_1  2       3  1      3.0  575.461000
75  o_iii  8  6     76    1s1.2p5.3P_0  2       3  1      1.0  575.501000
76  o_iii  8  6     77    1s1.2p5.1P_1  2       1  1      3.0  578.680000
77  o_iii  8  6     78  superlevel_[K]  2       1  1      3.0  594.140000
78  o_iii  8  6     79       continuum  2       2  1      2.0   54.934000
..
[79 rows x 10 columns]
```

Retrieve the data for the radiative transition between levels `llo` = 3 and `lup` = 4 of O III:

```
>>> y = px.get_data(ion='o_iii',rtype='LA',llo=3,lup=4)
>>> print(y)
   lrtyp  ltyp    ion  lup  llo upper_lev lower_lev  wavelength  a_value  \
0      4    50  o_iii    4    3  2p2.1D_2  2p2.3P_2     5008.98   0.0196
  gf_value
0      0.0
```

Compute effective collision strengths for transition 1-4 in O III at three specified temperatures.

```
>>> t = ([10.,100.,1000.])
>>> y = px.get_data(ion='o_iii',rtype='EC',llo=1,lup=4,temperature=t)
Data products computed = 2
[0] Collisional transition identifiers
[1] Effective collision strengths
>>> print(y[0])
   ltyp  lrtyp  Z    ion  lup  llo upper_lev lower_lev  ttype      dE    C
0    51      3  8  o_iii    4    1  2p2.1D_2  2p2.3P_0      2  0.1847  0.4
>>> print(y[1])
[        T       ups
0    10.0  0.301366
1   100.0  0.292904
2  1000.0  0.291202]
```

9.3 Function to Compute Rates

rates(*ion*[, *rtype=0*, *llo=0*, *lup=0*, *temperature=1.0*, *density=1000.0*, *rlogxi=0.0*, *rlrad38=1.0*, *cfrac=0.0*, *lcpres=0*, *pressure=0.03*, *vturbi=1.0*, *trad=-1*, *spectrum='pow'*, *spectrum_file='cutpow.txt'*, *spectun=0*, *ncn2=999*, *run=False*])

Computes rates for different radiative and collisional atomic processes.

Parameters:

ion : character string
The string specifies the ionic species using the Xstar notation; for instance, 'o_iii' for O III.

rtype : integer or 2-character string
Denotes the dataset to be retrieved:

- 0 or 'ALL' : all rates
- 3 or 'EC' : bound-bound collisional
- 4 or 'LA' : bound-bound radiative
- 5 or 'EI' : bound-free collisional
- 7 or 'PI' : bound-free radiative
- 8 or '' : total recombination
- 13 or 'LV' : energy levels
- 14 or '' : superlevel to spect level radiative
- 15 or '' : total collisional ionization
- 41 or 'AU' : Auger transition
- 42 or '' : inner-shell photoabsorption

llo : integer
Lower level index of the ion. For `rtype` = 'LV', `llo` = 0 returns all the levels while `llo` = -1 returns the continuum. For other `rtype`, `llo` = 0 returns all the rates for `llo` < `lup`.

lup : integer
Upper level index of the ion. `lup` = 0 returns all rates with `lup` > `llo`.

Note

For `llo` = `lup` = 0, all rates for `rtype` are listed.

`temperature : float`
Electron temperature in units of 10^4 K.

`density : float`
Electron density in units of cm^{-3}.

`rlogxi : float`
Log of the ionization parameter.

`rlrad38 : float`
Luminosity in units of 10^{38} ergs/s.

`cfrac : float`
Covering fraction.

`lcpres : integer`
Constant pressure switch (1 = yes; 0 = no).

`pressure : float`
Model pressure in dynes/cm^2.

`vturbi : float`
Turbulent velocity in Km/s.

`trad : float`
Radiation temperature in 10^7 K.

`spectrum : character string`
Input spectrum type:

- 'bbody' - black body spectrum
- 'bremss' - thermal bremsstrahlung
- 'pow' - power law
- 'file

`spectrum_file : character string`
Name of file containing spectrum.

`spectun : integer`
Spectrum units (0 = energy; 1 = photon)

`ncn2 : integer`
Number of continuum bins.

`run : Boolean`
Running switch (False - allows user to specify input iteratively; True - runs function).

Returns:

`df` : Dataframe listing rates:

- `ans1`: rate for forward reaction (llo –> lup)
- `ans2`: rate for inverse reaction (lup –> llo)
- `ans3`: heating rate photon point of view for forward reaction (llo –> lup)

- `ans4`: cooling rate photon point of view for inverse reaction (lup –> llo)
- `ans5`: heating rate electron point of view for forward reaction (llo –> lup)
- `ans6}`: cooling rate electron point of view for inverse reaction (lup –> llo)

Note

For `llo` != 0 and `lup` != 0, a list with the raw data is produced. Additionally, if `rtype` = 4 or `rtype = 7`, a second dataframe is produced with opacity/emission coefficients.

Examples

Recombination rate into superlevel of O I at temperature 10^5 K, density 10^{12} cm^{-3} and `rlogxi` = -3 (`rtype` = 7, `llo` = 11, and `lup` = 19).

```
>>> y = px.rates(rtype=7,ion='o_i',llo=11,lup=19,temperature=10.0,density=1.0e12,rlogxi=-
↪3.0,run=True)
Datasets computed = 3
[0] Ionic rates
[1] Raw data
[2] Tabulation of opacity/emission coefficients
>>> y[0].iloc[0]['ans2']
1.296
```

Radiative excitation rate for Fe XXVI L alpha with `trad` =-1$ power law, `rlogxi` = 4.0, `density` = 1.0E+12 cm^{-3} (`rtype` = 4, `llo` = 1, and `lup` = 3).

```
>>> y = px.rates(lratetype=4,ion='fe_xxvi',llo=1,lup=3,density=1.0e12,trad=-1, spectrum=
↪'pow',rlogxi=4.0,temperature=10.,run=True)
Datasets computed = 3
[0] Ionic rates
[1] Raw data
[2] Tabulation of opacity/emission coefficients
>>> y[0].iloc[0]['ans1']
44.17
```

9.4 Function to Compute Plasma Statistics

pop(*atom*[, *temperature=1.0*, *density=1000.0*, *rlogxi=0.0*, *rlrad38=1.0*, *cfrac=0.0*, *lcpres=0*, *pressure=0.03*, *vturbi=1.0*, *trad=-1*, *spectrum='pow'*, *spectrum_file='cutpow.txt'*, *spectun=0*, *ncn2=999*, *run=False*])

Computes level populations and dominant rates; list of rates; ionization fractions and photoionization and recombination rates; and the total cooling and heating rates.

Parameters:

atom : character string
The string specifies the atom chemical symbol

temperature : float
Electron temperature in units of 10^4 K.

density : float
Electron density in units of cm^{-3}.

rlogxi : float
Log of the ionization parameter.

rlrad38 : float
Luminosity in units of 10^{38} ergs/s.

cfrac : float
Covering fraction.

lcpres : integer
Constant pressure switch (1 = yes; 0 = no).

pressure : float
Model pressure in dynes/cm^2.

vturbi : float
Turbulent velocity in Km/s.

trad : float
Radiation temperature in 10^7 K.

spectrum : character string
Input spectrum type:

- 'bbody' - black body spectrum
- 'bremss' - thermal bremsstrahlung
- 'pow' - power law
- 'file

spectrum_file : character string
Name of file containing spectrum.

spectun : integer
Spectrum units (0 = energy; 1 = photon)

ncn2 : integer
Number of continuum bins.

run : Boolean
Running switch (False - allows user to specify input iteratively; True - runs function).

Returns:
`df` : List of four dataframes:

`df[0]` : Level populations and dominant rates

`df[1]` : Ionic process rates

`df[2]` : Ionic fractions and ionization/recombination rates

`df[3]` : Total heating and cooling rates

Examples
Compute statistics for oxygen in a plasma at `temperature` = 10^4 K, `density` = 10^3 cm^{-3}, and `rlogxi` = 1.0.

```
>>> y = px.pop(element='o',temperature=1.0,density=1.e3,rlogxi=1.0,run=True)
Computed datasets = 4
[0] Level populations and dominant rates
[1] Process rates
```

(continues on next page)

(continued from previous page)

```
[2] Ionic fractions and ionization/recombination rates
[3] Total heating and cooling rates
>>> print(y[3])
   Z     cool_rate     heat_rate  cool_rate_epv  heat_rate_epv
0  8  9.640000e-14  1.034000e-13   7.562000e-17   6.444000e-15
```

9.5 Function to Compute Opacities and Emissivities

opak(*atom*[, *temperature=1.0*, *density=1000.0*, *rlogxi=0.0*, *rlrad38=1.0*, *cfrac=0.0*, *lcpres=0*, *pressure=0.03*, *vturbi=1.0*, *trad=-1*, *spectrum='pow'*, *spectrum_file='cutpow.txt'*, *spectun=0*, *ncn2=999*, *run=False*])

Computes line opacities and emissibities; bound-free opacities and emissivities; and an energy tabulation of opacity and emissivity

Parameters:

atom : character string
The string specifies the atom chemical symbol.

temperature : float
Electron temperature in units of 10^4 K.

density : float
Electron density in units of cm^{-3}.

rlogxi : float
Log of the ionization parameter.

rlrad38 : float
Luminosity in units of 10^{38} ergs/s.

cfrac : float
Covering fraction.

lcpres : integer
Constant pressure switch (1 = yes; 0 = no).

pressure : float
Model pressure in dynes/cm^2.

vturbi : float
Turbulent velocity in Km/s.

trad : float
Radiation temperature in 10^7 K.

spectrum : character string
Input spectrum type:

- 'bbody' - black body spectrum
- 'bremss' - thermal bremsstrahlung
- 'pow' - power law
- 'file

spectrum_file : character string
Name of file containing spectrum.

spectun : integer
Spectrum units (0 = energy; 1 = photon)

ncn2 : integer
Number of continuum bins.

run : Boolean
Running switch (False - allows user to specify input iteratively; True - runs function).

Returns:

`df` : List of three dataframes.

`df[0]` : Line opacities and emissivities

`df[1]` : Bound-free opacities and emissivities

`df[2]` : Opacity and emissivity tabulation

Examples

Compute opacities and emissivities for oxygen at `temperature` = 10^4 K, `density` = 10^3 cm^{-3}, and `rlogxi` = -2.0.

```
>>> y = px.opak(element='o',temperature=1.0,density=1.e3,rlogxi=-2.0,run=True)
Computed datasets = 3
[0] Line opacities and emissivities (erg/cm**3/sec/10**38)
[1] Bound-free opacities and emissivities (erg/cm**3/sec/10**38)
[2] Opacity and emissivity tabulation (/cm**3/sec/10**38)
```

CHAPTER

TEN

WALKTHROUGH EXAMPLES AND THREADS

10.1 XSTAR

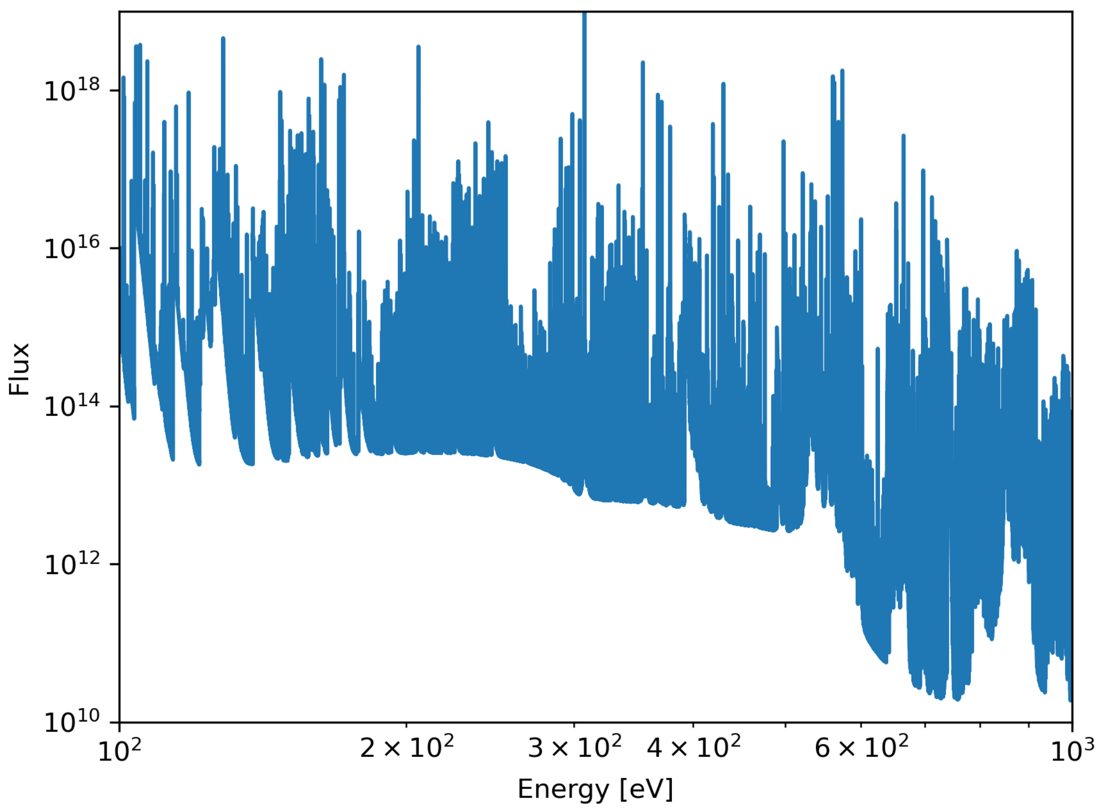
10.1.1 Broad emission line cloud and plot of simulated spectrum

In this example we model a quasar broad line cloud. This illustrates the use of constant pressure and a power law spectrum, with spectral index input in energy units. This run can take a few hours to finish. Runtime may be reduced by lowering the spectral resolution `ncn2`.

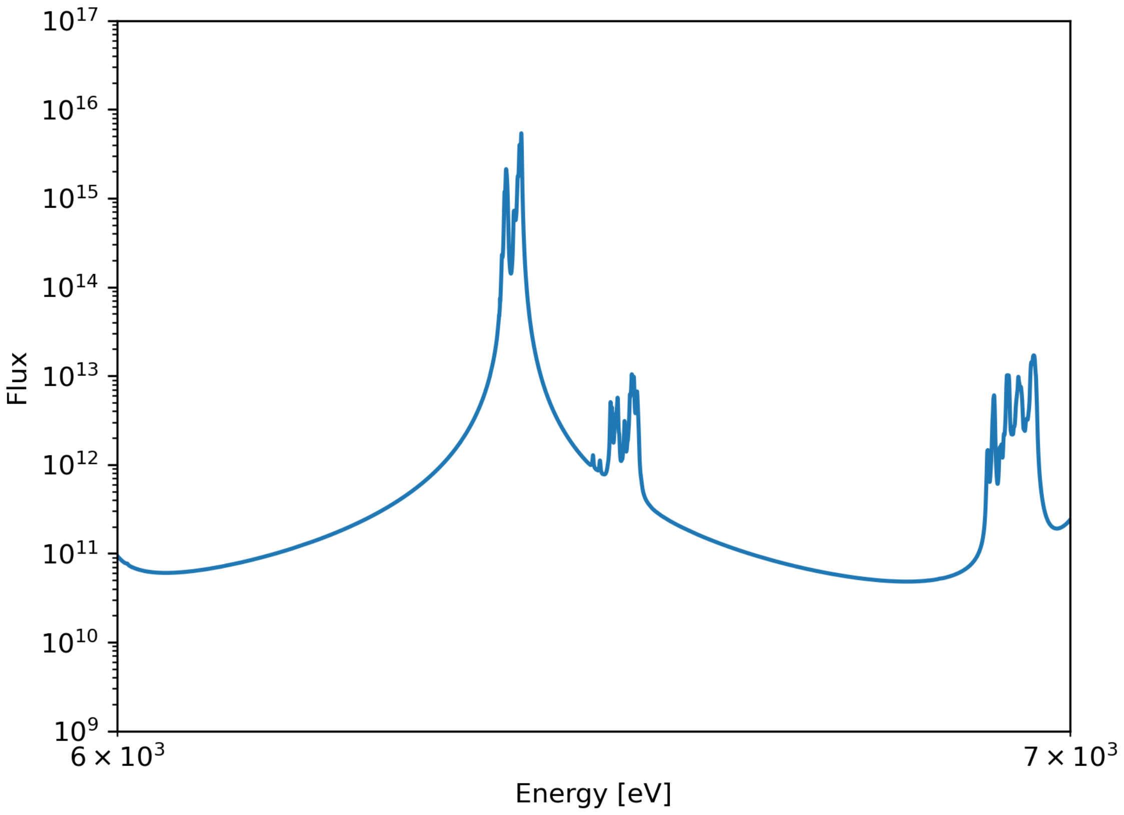
```
xstar cfrac=0. temperature=1 lcpres=1 pressure=0.03 density=1e10
spectrum='pow' trad=-0.9 rlrad38=1e8 column=1e23 rlogxi=0.2
abundtbl='xdef' modelname='quasar BLR cloud' ncn2=999999
```

We now look at the ion abundance of hydrogen and helium as a function of radius.

```
>>> import matplotlib.pyplot as plt
>>> from astropy.io import fits
>>> import numpy as np
>>>
>>> hdul= fits.open('files/xout_spect1.fits')
>>> spec=hdul[2].data
>>> hdul.close()
>>>
>>> plt.plot(spec['energy'], spec['emit_outward'])
>>> plt.xscale("log")
>>> plt.yscale("log")
>>> plt.xlabel("Energy [eV]")
>>> plt.ylabel("Flux")
>>> plt.xlim(100, 1000)
>>> plt.ylim(1e10, 1e19)
>>> plt.show()
```



```
>>> plt.plot(spec['energy'], spec['emit_outward'])
>>> plt.xscale("log")
>>> plt.yscale("log")
>>> plt.xlabel("Energy [eV]")
>>> plt.ylabel("Flux")
>>> plt.xlim(6000, 7000)
>>> plt.ylim(1e9, 1e17)
>>> plt.show()
```



10.1.2 Marginally Compton thick slab, reflection and transmission

In this example we compare reflection and transmission from a marginally Compton-thick slab. It requires the `cfrac` parameter to be set to zero. This example further illustrates a multi-pass run. We set exclude some trace elements. This run can take a few hours to finish. Runtime may be reduced by lowering the spectral resolution `ncn2` and the number of elements.

```
xstar cfrac=0. rlogxi=3 column=5e23 density=1e12 rlrad38=1e-1
temperature=1. spectrum='pow' trad=-1 abundtbl='xdef'
modelname='relf-trans' niter=99 npass=5 fabund=0 alabund=0 pabund=0
clabund=0 scabund=0 tiabund=0 vabund=0 crabund=0 mnabund=0
niabund=0 cuabund=0 znabund=0
```

```
  xstar version 2.59cj

pass number=           1          -1
  log(r) delr/r log(N) log(xi) x_e   log(n) log(t) h-c(%) h-c(%) log(tau)
                                                                 fwd    rev
  11.00 -36.00 -10.00   3.00   1.21  12.00   6.29   0.00   0.00 -10.00 -10.00 40
  11.00 -36.00 -10.00   3.00   1.21  12.00   6.29   0.00   1.09 -10.00 -10.00  1
  11.06  -0.87  22.19   2.87   1.21  12.00   5.91   0.01   1.26  -1.90 -10.00  9
  [...]
```

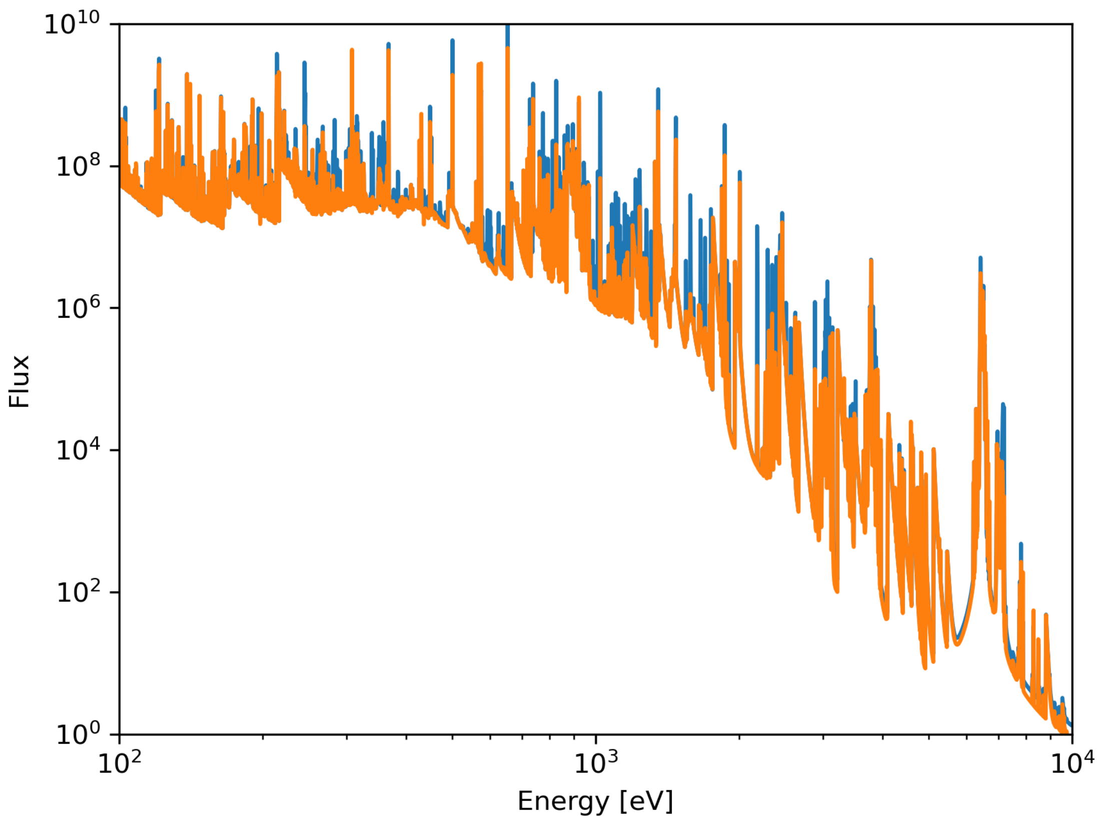
(continues on next page)

(continued from previous page)

```

pass number=           2          1
  log(r) delr/r log(N) log(xi) x_e   log(n) log(t) h-c(%) h-c(%) log(tau)
                                                              fwd    rev
  11.78  -0.08  23.70   1.45   1.10  12.00   4.01   0.02  -0.02   0.85 -10.00  9
  11.77  -0.08  23.69   1.45   1.10  12.00   4.45   0.12  -0.06   0.83  -0.11 22
  11.77  -0.08  23.69   1.45   1.10  12.00   4.43   0.19   0.22   0.82  -0.09 24
  [...]

pass number=           5         -1
  log(r) delr/r log(N) log(xi) x_e   log(n) log(t) h-c(%) h-c(%) log(tau)
                                                              fwd    rev
  11.00 -36.00 -10.00   3.00   1.21  12.00   5.84  -0.06   0.00 -10.00   0.06  9
  11.00 -36.00 -10.00   3.00   1.21  12.00   5.84  -0.06   1.87 -10.00   0.05  9
  11.06  -0.87  22.19   2.87   1.21  12.00   5.70   0.00   2.13  -1.90   0.05 10
  [...]
```

Note how the temperature change in the first and fifth pass at the same ionization parameter. Now we compare the reflected (inward emitted) and transmitted (outward emitted) spectra.

```
>>> import matplotlib.pyplot as plt
>>> from astropy.io import fits
>>> import numpy as np
>>>
>>> hdul= fits.open('files/xout_spect1.fits')
>>> spec=hdul[2].data
>>> hdul.close()
>>> plt.plot(spec['energy'], spec['emit_outward'])
>>> plt.plot(spec['energy'], spec['emit_inward'])
>>> plt.xscale("log")
>>> plt.yscale("log")
>>> plt.xlabel("Energy [eV]")
>>> plt.ylabel("Flux")
>>> plt.xlim(100, 10000)
>>> plt.ylim(1e0, 1e10)
>>>  plt.show()
```



10.1.3 Traditional H II region

In this example we model an H II region. In this case we give only the prompted values. This illustrates the use of a blackbody spectrum, with temperature given in keV, and non-solar abundances.

```
xstar cfrac=1 temperature=1 density=1e2 spectrum='bbody' trad=4e-3
rlrad38=12.7 column=1e23 rlogxi=0.15 abundtbl='xdef' cabund=0.60
nabund=0.36 oabund=0.47 neabund=1.79 mgabund=0. siabund=0.0
sabund=5.63 arabund=0.0 caabund=0 feabund=0 niabund=0 modelname='H
II region' niter=0
```

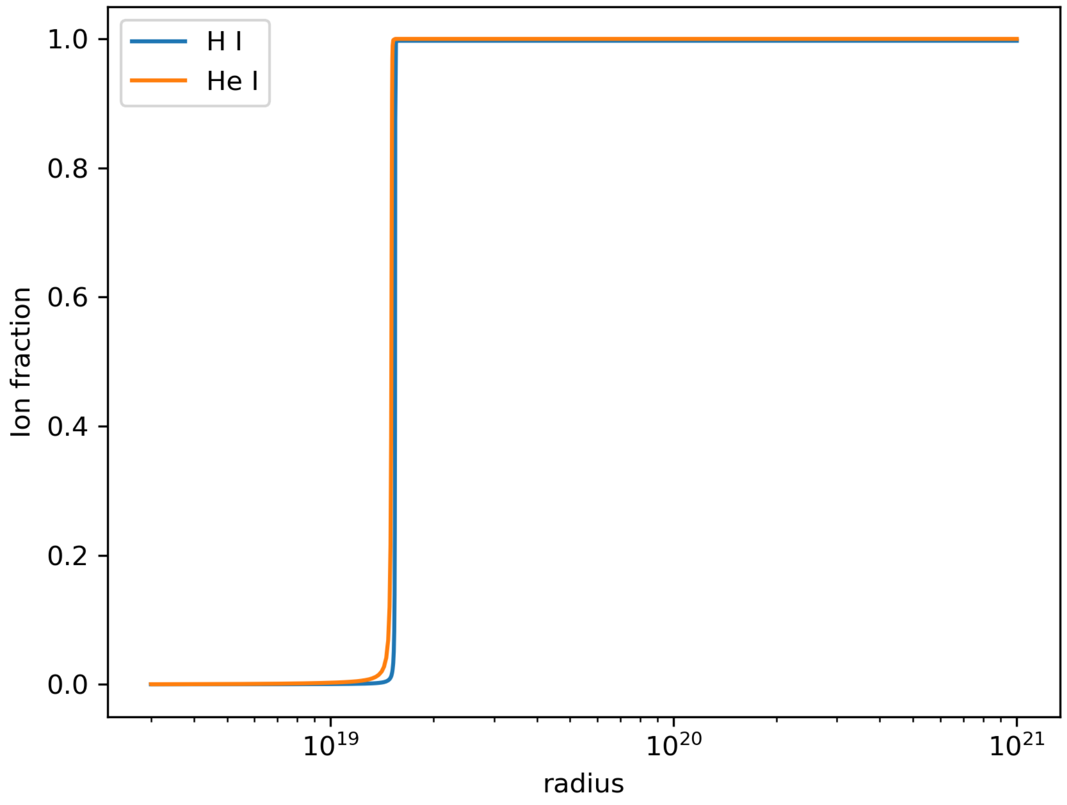
We now look at the ion abundance of hydrogen and helium as a function of radius.

```
>>> import matplotlib.pyplot as plt
>>> from astropy.io import fits
>>> import numpy as np
>>> hdul= fits.open('xout_abund1.fits')
>>> abund=hdul[1].data
>>> hdul.close()
>>> plt.plot(abund['radius'], abund['h_i'])
>>> plt.plot(abund['radius'], abund['he_i'])
>>>
>>> plt.legend(['H I', 'He I'])
```

(continues on next page)

(continued from previous page)

```
>>> plt.xscale("log")
>>> plt.xlabel("radius")
>>> plt.ylabel("Ion fraction")
>>> plt.show()
```



10.1.4 Coronal plasma

In this example we calculated the emission from a plasma in collisional ionization equilibrium at a temperature of 0.5 keV. Most importantly, we set `niter=-99` to hold the temperature constant but ensure charge neutrality. We further set the ionization parameter `rlogxi` to the minimum value of -10 so that the radiation field becomes negligible.

```
xstar cfrac=1. rlogxi=-10 column=1e18 density=1e8 rlrad38=1e-6
temperature=5.8e2 spectrum='pow' trad=-1 abundtbl='xdef'
modelname='coronal' niter=0 npass=1 fabund=0 alabund=0 pabund=0
clabund=0 scabund=0 tiabund=0 vabund=0 crabund=0 mnabund=0
niabund=0 cuabund=0 znabund=0 ncn2=99999
```

```
  xstar version 2.59cj

pass number=           1          -1
  log(r) delr/r log(N) log(xi) x_e   log(n) log(t) h-c(%) h-c(%) log(tau)
```

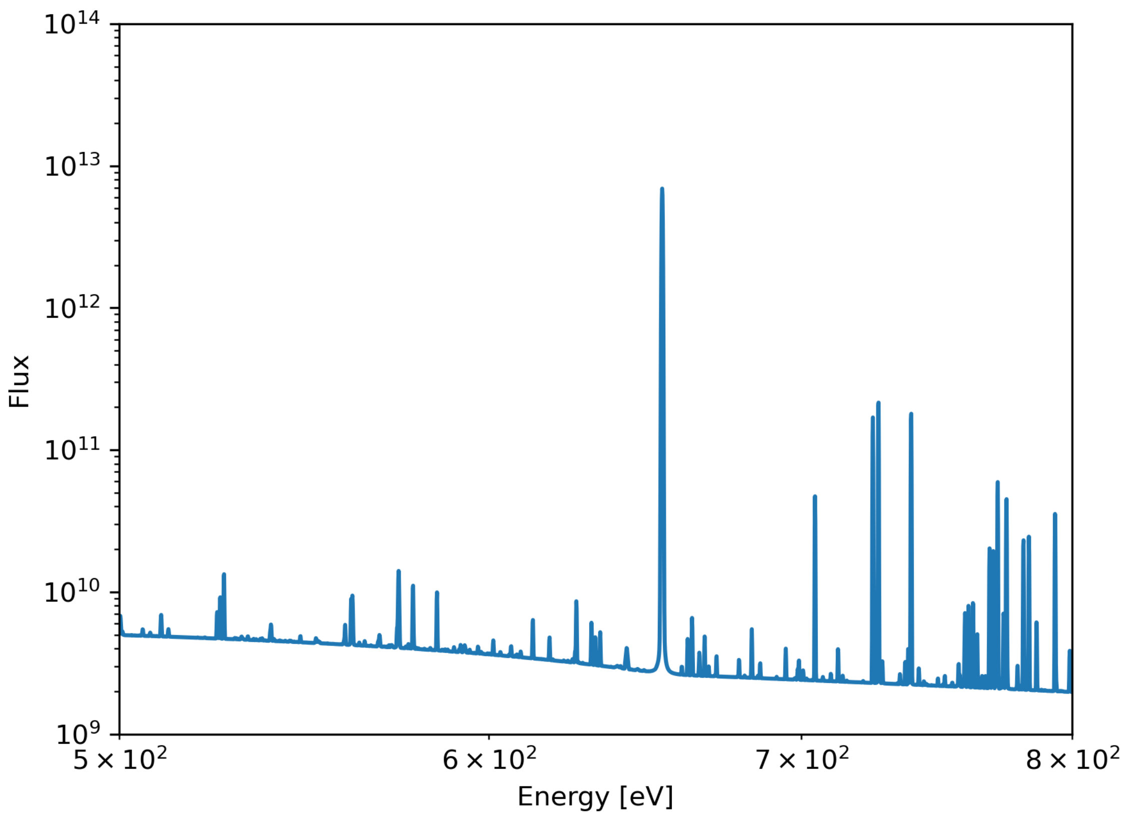
(continues on next page)

(continued from previous page)

```
                                                                      fwd     rev
  17.00 -36.00 -10.00 -10.00   1.00   8.00   6.76 -99.99   0.00 -10.00 -10.00  0
  17.00 -36.00 -10.00 -10.00   1.00   8.00   6.76 -99.99 -99.99 -10.00 -10.00  0
  17.00  -7.00  18.00 -10.00   1.00   8.00   6.76 -99.99 -99.99  -6.65 -10.00  0
 final print:           0
xstar: Prepping to write spectral data
xstar: Done writing spectral data
total time   1550.1273570153862
```

Next we plot the outward emission spectrum that features a very strong O VIII Lya line along with many Fe L lines.

```
>>> import matplotlib.pyplot as plt
>>> from astropy.io import fits
>>> import numpy as np
>>>
>>> hdul= fits.open('files/xout_spect1.fits')
>>> spec=hdul[2].data
>>>
>>> hdul.close()
>>> plt.plot(spec['energy'], spec['emit_outward'])
>>> plt.xscale("log")
>>> plt.yscale("log")
>>> plt.xlabel("Energy [eV]")
>>> plt.ylabel("Flux")
>>> plt.xlim(500, 800)
>>> plt.ylim(1e9, 1e14)
>>> plt.show()
```



10.1.5 Spectrum with line labels

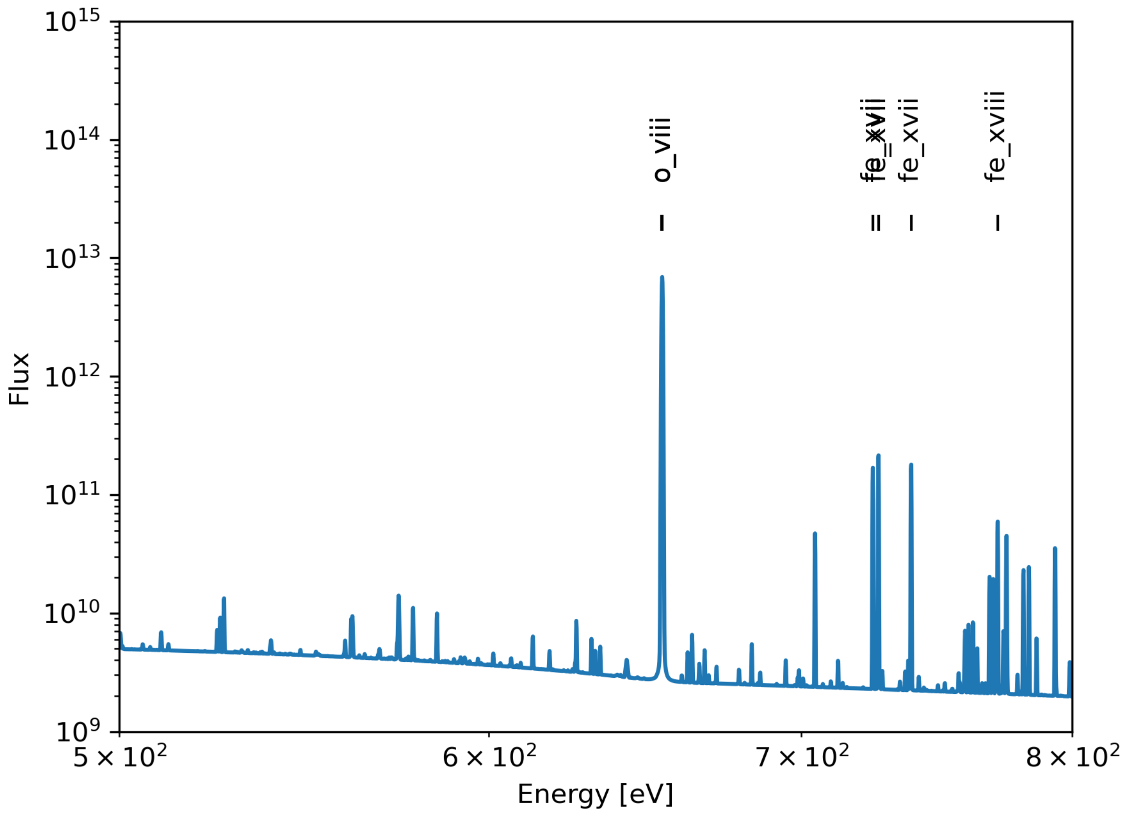
Here we show how to use the line file "xout_lines1.fits" to add line labels to a spectral plot. For this example we resuse the output files calculated in example.

```
>>> import matplotlib.pyplot as plt
>>> from astropy.io import fits
>>> import numpy as np
>>>
>>> spec=fits.open('files/xout_spect1.fits')[2].data
>>> lines=fits.open('files/xout_lines1.fits')[2].data
>>>
>>> plt.plot(spec['energy'], spec['emit_outward'])
>>> plt.xscale("log")
>>> plt.yscale("log")
>>> plt.xlabel("Energy [eV]")
>>> plt.ylabel("Flux")
>>>
>>> emin=500. # in eV
>>> emax=800. # in eV
>>>
>>> plt.xlim(emin, emax)
```

(continues on next page)

(continued from previous page)

```
>>> plt.ylim(1e9, 1e15)
>>> for ii in range(20):
>>>
>>> elin=12398./lines[ii][4] # A to eV
>>>   if emin<elin<emax:
>>>      plt.plot(elin, 2e13, '|', color='black')
>>>      plt.text ( 12398./lines[ii][4], 5e13, lines[ii][1], rotation='vertical',␣
↪horizontalalignment='center')
>>>
>>> plt.show()
```



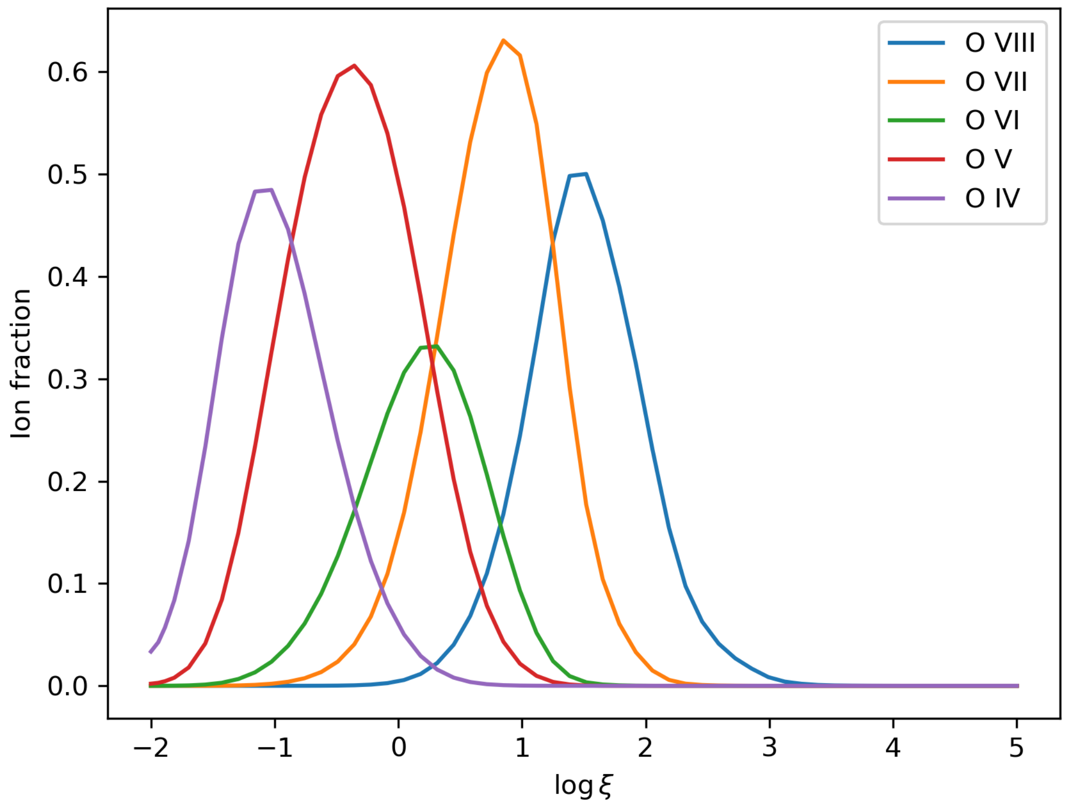
10.1.6 Charge state distribution as function of $\log\xi$

In this example we calculate the charge state distribtution of oxygen as a function of ionization parameter. For illustrative purposes we run xstar such that $\Delta R/R$ is large and the spatial zones span a wide range of $\log\xi$. For example:

```
xstar cfrac=1 temperature=1000. pressure=0.03 density=1.E+4 \
spectrum='pow ' trad=-1. rlrad38=1.E-10 column=1.E+17 \
rlogxi=5. lcpres=0 abundtbl='xdef'  modelname='filled sphere'\
niter=99 npass=1 critf=1.E-07 nsteps=6 xeemin=0.04 emult=0.5 taumax=5. \
lprint=1 ncn2=9999 radexp=0 vturb=1. \
```

FITS files can be processed with various programming languages and tools. As an example, we want to examine the charge state distribution of Oxygen as a function of distance from the central source, i.e., ionization parameter. The ion populations are stored in 'xout_abund1.fits'.

```
>>> import matplotlib.pyplot as plt
>>> from astropy.io import fits
>>> import numpy as np
>>> hdul= fits.open('xout_abund1.fits')
>>> abund=hdul[1].data
>>> hdul.close()
>>>
>>> plt.plot(abund['ion_parameter'], abund['o_viii'])
>>> plt.plot(abund['ion_parameter'], abund['o_vii'])
>>> plt.plot(abund['ion_parameter'], abund['o_vi'])
>>> plt.plot(abund['ion_parameter'], abund['o_v'])
>>> plt.plot(abund['ion_parameter'], abund['o_iv'])
>>>
>>> plt.legend(['O VIII', 'O VII', 'O VI', 'O V', 'O IV'])
>>> plt.xlabel("$\\log\\xi$")
>>> plt.ylabel("Ion fraction")
>>> plt.show()
```



10.1.7 T - $\log\xi$ curves

Here we compare heating and cooling rates as function of temperature T and ionization parameter $\log\xi$. First we create a grid of temperatures and ionization parameters.

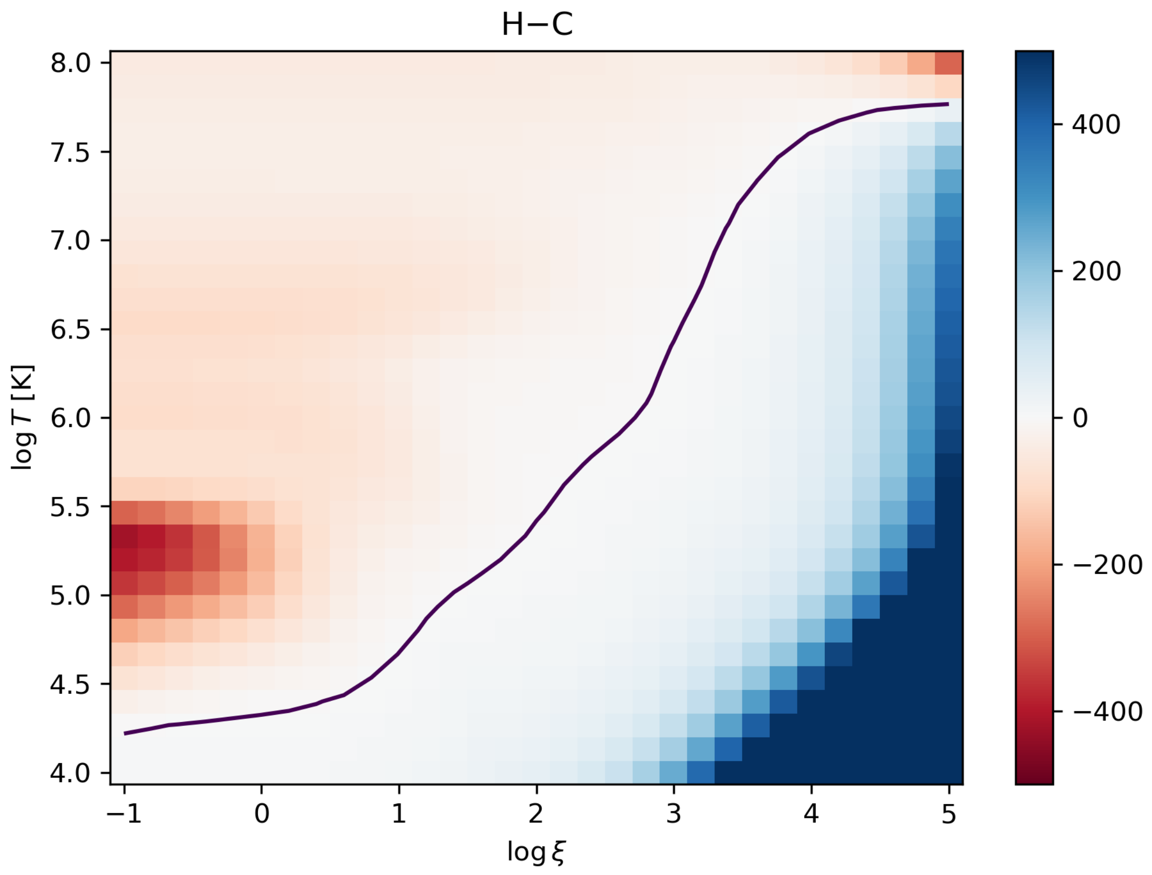
```
>>> import numpy as np
>>> import os
>>> import matplotlib.pyplot as plt
>>> from astropy.io import fits

>>> nt = 31 # Number of temperature steps
>>> nlxi = 31 # Number of logxi steps
>>> lxi = np.linspace(-1, 5, num=nlxi) # Step logxi  from 1 to 5
>>> t =  np.logspace(0, 4, num=nt) # Temperature from 1E0 to 1E4
>>> httot = np.zeros((nt, nlxi))
>>> cltot = np.zeros((nt, nlxi))
```

Then we loop over this grid and run xstar for each combination of T and $\log\xi$. This can be very time-consuming depending on the ranges and stepsize of the T- $\log\xi$ grid. Note that `niter` is set to 0 in order to keep the temperature fixed at its input value. The total heating and cooling rates can be read from the ion abundance output file.

```
>>> for ii in range(0, nt):
...       for jj in range(0, nlxi):
...         xstcmd = "xstar cfrac=1  pressure=0.03 density=1E12 trad=-1. spectrum='pow'␣
↪rlrad38=1E6 column=1E10 nsteps=3 niter=0 lwrite=0 lprint=0 lstep=0 npass=1 lcpres=0␣
↪emult=0.5 taumax=5. xeemin=0.1 critf=1E-8 vturbi=300 radexp=0 ncn2=999 modelname='t-xi
↪' abundtbl='xdef' loopcontrol=0 rlogxi={rlogxi} temperature={temp:.2E}".format(rlogxi␣
↪= lxi[jj], temp = t[ii])
...         os.system(xstcmd) # Run xstar
...         hdul = fits.open('xout_abund1.fits')
...         httot[ii,jj] = hdul[3].data['total'][0] # Read total heating rate
...         cltot[ii,jj] = hdul[4].data['total'][0] # Read total cooling rate
...         hdul.close()
...         os.system("rm xout_abund1.fits  xout_cont1.fits  xout_lines1.fits  xout_rrc1.
↪fits  xout_spect1.fits  xout_step.log") # Clean up
```

Finally we plot heating minus cooling as a 2D map and draw a contour of thermal equilibrium.

```
>>> hmc=httot-cltot
>>> fig, ax = plt.subplots(1, 1)
>>> c = ax.pcolor(lxi,np.log10(t)+4, hmc, cmap='RdBu',vmin=-500, vmax=500)
>>> ax.set_title('H$-$C')
>>> ax.set_ylabel("$\\log T$ [K]")
>>> ax.set_xlabel("$\\log\\xi$")
>>> ax.contour(lxi,np.log10(t)+4, hmc, levels=[0.0])
>>> fig.colorbar(c, ax=ax)
>>> fig.tight_layout()
>>> plt.show()
```



10.1.8 Density profiles (constant vs powerlaw)

This example illustrates the usage of the `radexp` paramter that allows for a powerlaw-like density profile. We compare `radexp=-0.5` and `radexp=0.0` in two otherwise identical runs.

First we let the density fall of as $1/\sqrt{R}$.

```
xstar cfrac=1 temperature=1000 density=1.E+8  spectrum='pow ' trad=-1. rlrad38=1.E-10␣
↪column=1.E+17 rlogxi=5. lcpres=0 abundtbl='xdef' modelname='densexp-0.5' niter=99␣
↪npass=1 critf=1.E-07 nsteps=6 xeemin=0.04 emult=0.5 taumax=5.  ncn2=9999 radexp=-0.5␣
↪vturb=1.
```

Here the density is held constant.

```
xstar cfrac=1 temperature=1000 density=1.E+8  spectrum='pow ' trad=-1. rlrad38=1.E-10␣
↪column=1.E+17 rlogxi=5. lcpres=0 abundtbl='xdef' modelname='filled sphere' niter=99␣
↪npass=1 critf=1.E-07 nsteps=6 xeemin=0.04 emult=0.5 taumax=5.  ncn2=9999 radexp=0.0␣
↪vturb=1.
```

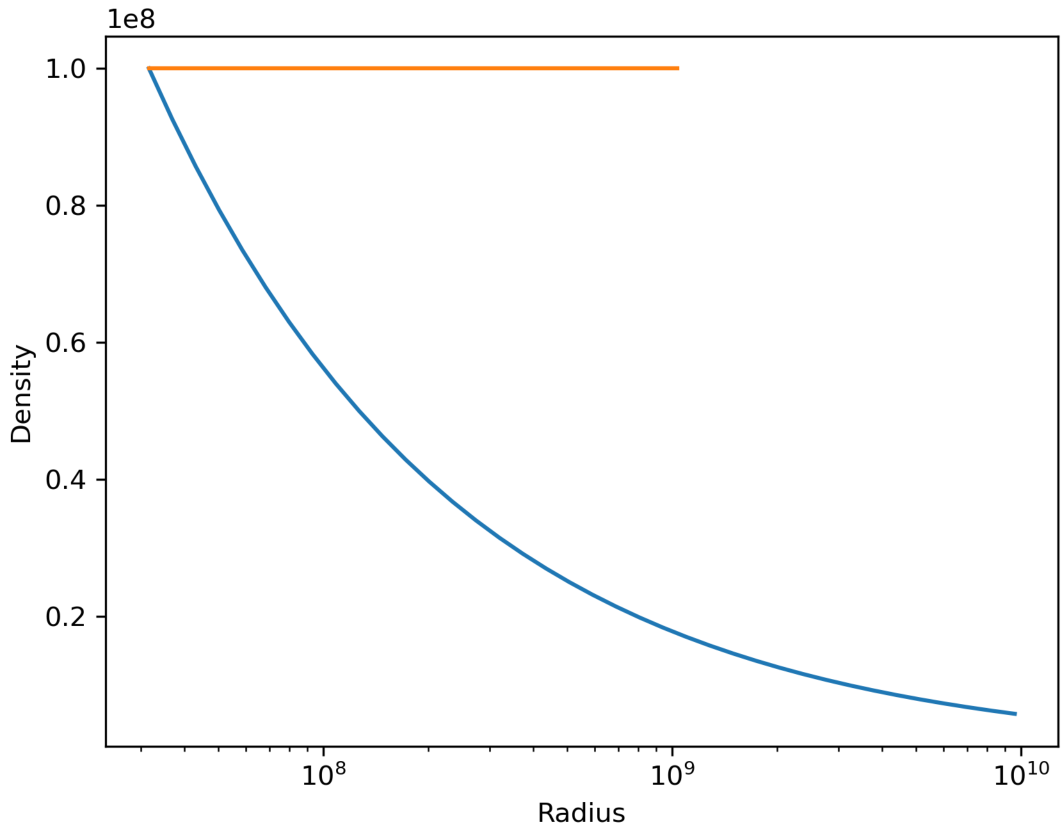
First we look at the density as a function of radius.

```
>>> import matplotlib.pyplot as plt
>>> from astropy.io import fits
```

(continues on next page)

(continued from previous page)

```
>>> import numpy as np
>>>
>>> a1=fits.open('files/0.5/xout_abund1.fits')[1].data
>>> a2=fits.open('files/0.0/xout_abund1.fits')[1].data
>>>
>>> plt.plot(a1['radius'], a1['n_p'])
>>> plt.plot(a2['radius'], a2['n_p'])
>>> plt.xscale("log")
>>> plt.xlabel("Radius")
>>> plt.ylabel("Density")
>>>
>>> plt.show();
```



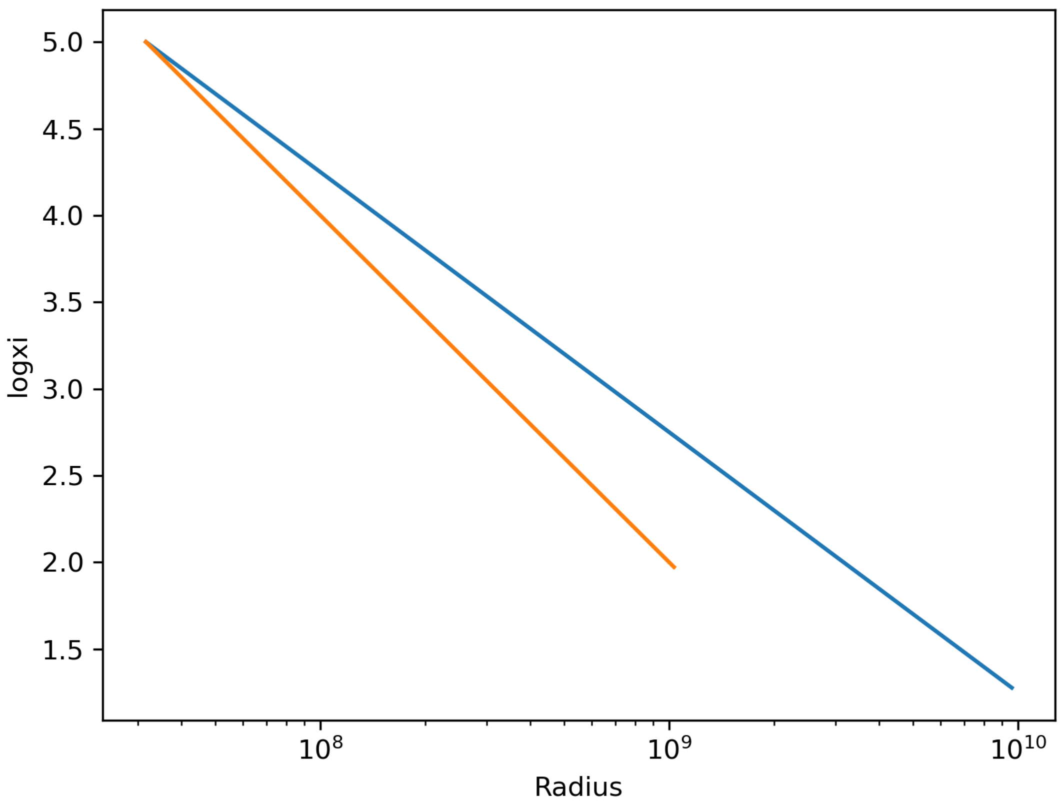
The inner radius is determined by the ionization parameter, density, and luminosity, in this example $R_{in} = 10^{17.5}$ cm. The outer radius is determined by the column density, so the two clouds have different outer radii. As the ionization structure inside the cloud is also different.

```
>>> plt.plot(a1['radius'], a1['ion_parameter'])
>>> plt.plot(a2['radius'], a2['ion_parameter'])
>>> plt.xscale("log")
>>> plt.xlabel("Radius")
>>> plt.ylabel("logxi")
>>>
```

(continues on next page)

(continued from previous page)

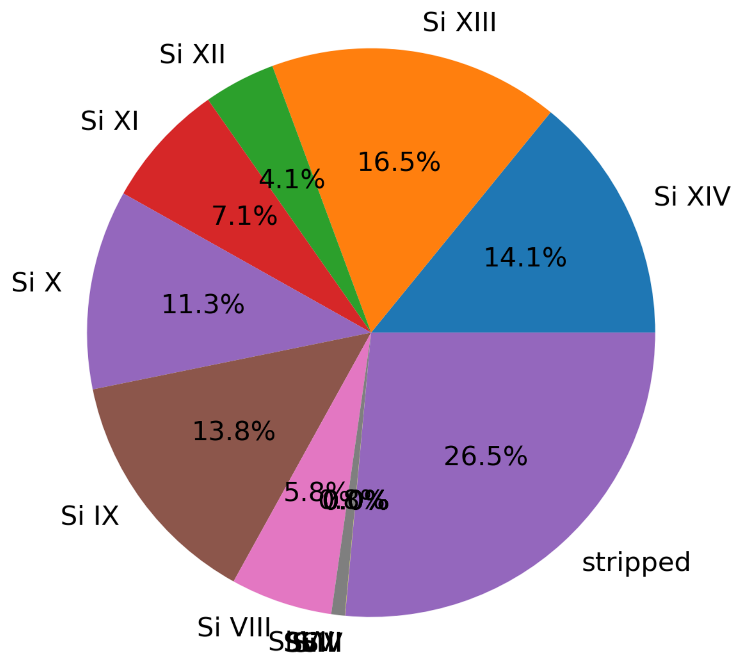
```
>>> plt.show();
```



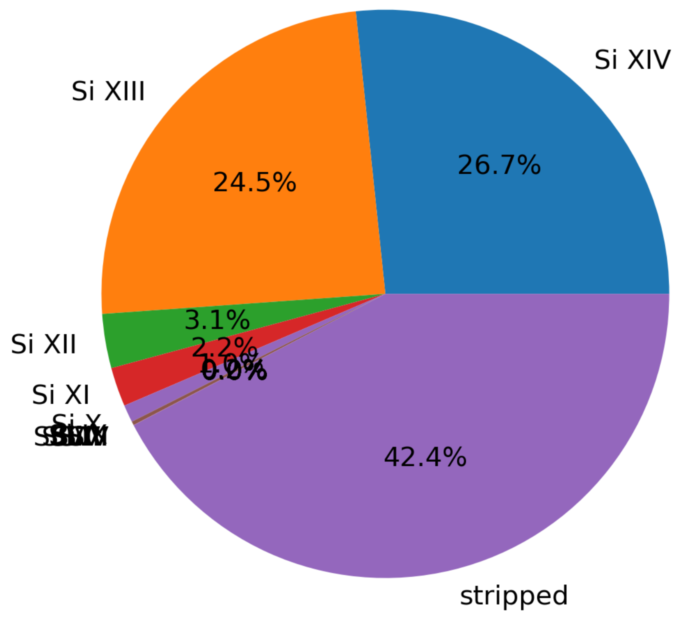
The geometrical dilution of the radiation field is proportional to $1/R^2$ while the density is $1/\sqrt{R}$. The ionization parameter falls faster for the constant density case, but does not reach values below 2. Consequently, this leads to different charge state distributions. As an example, we compare the total ion columns for silicon.

Note, that the fully ionized ions are not listed in the abundance file. Their column/fraction can be calculated from the total colum density, the elemental abundance and the sum of all other ions.

```
>>> c1= fits.open('files/0.5/xout_abund1.fits')[2].data
>>> c2= fits.open('files/0.0/xout_abund1.fits')[2].data
>>> si_bare=1e17*3.5e-5-sum([c1.si_xiv[0], c1.si_xiii[0], c1.si_xii[0], c1.si_xi[0], c1.
↪si_x[0], c1.si_ix[0], c1.si_viii[0], c1.si_vii[0], c1.si_vi[0], c1.si_v[0], c1.si_
↪iv[0], c1.si_iii[0], c1.si_ii[0], c1.si_i[0]])
>>> labels = ['Si XIV', 'Si XIII', 'Si XII', 'Si XI', 'Si X', 'Si IX', 'Si VIII',  'Si␣
↪VII', 'Si VI', 'Si V', 'Si IV', 'Si III', 'Si II', 'Si I', 'stripped']
>>> sizes = [c1.si_xiv[0], c1.si_xiii[0], c1.si_xii[0], c1.si_xi[0], c1.si_x[0], c1.si_
↪ix[0], c1.si_viii[0], c1.si_vii[0], c1.si_vi[0], c1.si_v[0], c1.si_iv[0], c1.si_iii[0],
↪ c1.si_ii[0], c1.si_i[0], si_bare]
>>>
>>> fig, ax = plt.subplots()
>>> ax.pie(sizes, labels=labels, autopct='%1.1f%%')
```



Repeating the same exercise for the constant density case gives.



10.2 XSTAR2XSPEC

10.2.1 Calculate a simple table model (grid18)

The xstar webpage contains links to pre-calculated table models that can be downloaded and directly used in xspec. While these tables are good for quicklook and preliminary analyses we generally recommend users recalculate tables for their specific data analysis need to make sure they have full control and understanding of the parameter space and guarantee the use of the most recent version of xstar. In this example we recalculate the table model 'grid18' using xstar2xspec interactively.

After initializting HEAsoft, in an empty directory, we create a copy of all required parameter files and reset `$PFILES` environment varriable. This step is optional but highly recommended.

```
$ cp $HEADAS/syspfiles/xstar.par .
$ cp $HEADAS/syspfiles/xstinitable.par .
$ cp $HEADAS/syspfiles/xstar2table.par .
$ export PFILES=`pwd`
```

Now run xstar2xspec from the command line and with the following prompted parameters.

```
 $ xstar2xspec
rm: xstinitable.lis: No such file or directory
```

(continues on next page)

(continued from previous page)

```
 rm: xstinitable.fits: No such file or directory
 rm: xstar2xspec.log: No such file or directory
 XSTAR2XSPEC: Initialize the Multiple XSTAR run...
 spectrum type? [pow]
 radiation temperature or alpha soft maximum? [-1.2] -1
 covering fraction soft maximum (0.:1.) [1.]
 density soft maximum (cm**-3) (1.e+4:1.E21) [1.E+12]
 luminosity soft maximum (/10**38 erg/s) (1.e-20:1.E10) [1.e+6]
 column density soft maximum (cm**-2) (1.e+10:1.E25) [1.e+24] 1e23
 column density interpolation type (0:1) [1]
 column density soft minimum (1.:1.E25) [1.E+21] 1e19
 column density number of steps (1:20) [7] 5
 log(ionization parameter) soft maximum (erg cm/s) (-10.:+10.) [5.] 4
 log(ionization parameter) interpolation type (0:1) [0]
 log(ionization parameter) soft minimum (-10.0:+10.0) [1.] -4
 log(ionization parameter) number of steps (1:40) [9] 17
 model name [template] XSTAR_fine_grid_1-3
 abundance table [xdef]
 XSTAR2XSPEC: Entering XSTAR2XSPEC Main Processing Loop...
 XSTAR2XSPEC: Executing Pass #1
 xstar returns: xstar returns:  xstar version 2.59cj

 pass number=           1          -1
 log(r) delr/r log(N) log(xi) x_e   log(n) log(t) h-c(%) h-c(%) log(tau)
                                                              fwd    rev
 18.00 -36.00 -10.00  -4.00   0.37  12.00   3.93  -0.24   0.00 -10.00 -10.00 78
 18.00 -36.00 -10.00  -4.00   0.37  12.00   3.93   0.22   0.88 -10.00 -10.00 32
 18.00 -12.92  17.08  -4.00   0.31  12.00   3.62   8.38   1.00  -0.30 -10.00 71
 [...]

 XSTAR2XSPEC: Executing Pass #85
 xstar returns: xstar returns:  xstar version 2.59cj

 pass number=           1          -1
 log(r) delr/r log(N) log(xi) x_e   log(n) log(t) h-c(%) h-c(%) log(tau)
                                                               fwd    rev
 14.00 -36.00 -10.00   4.00   1.21  12.00   7.59  -0.00   0.00 -10.00 -10.00 26
 14.00 -36.00 -10.00   4.00   1.21  12.00   7.59  -0.00  -1.21 -10.00 -10.00  1
 14.00  -3.00  23.00   4.00   1.21  12.00   7.59  -0.00  -1.21  -5.28 -10.00  1
final print:           0
xstar: Prepping to write spectral data
xstar: Done writing spectral data
total time   42.034025847911835
xstar2table returns: xstar2table v1.0
Compiled: Mar 15 2024 13:12:37
1) column      : Value is interpolated.
                 Logarithmically sampled at 5 points:
                 1e+19  1e+20  1e+21  1e+22  1e+23
2) rlogxi      : Value is interpolated.
                 Linearly sampled at 17 points:
                 -4  -3.5  -3  -2.5  -2  -1.5  -1  -0.5  0  0.5  1  1.5  2  2.5  3  3.5 ␣
 ↪4
```

(continues on next page)

(continued from previous page)

```

Model Control Parameters:
Spectrum Type:            pow
Spectrum File:            spect.dat
Spectrum Units:           Energy
Abundance Table:
Is redshift a parameter?: Yes
Number of Steps:          3
Number of Iterations:     99
Write Switch:             Yes
Print Switch:             Yes
Step Size Choice Switch:  0
Number of Passes:         1
Constant Pressure Switch: No
Courant multiplier (emult)     0.50
taumax for Courant step:       5.00
Minimum electron fraction:     0.10
Critical ion abundance:        0.00
Turbulent velocity (km/s):   300.00
radius exponent:               0.00
number of energy bins:    9999
Model Name:               XSTAR_fine_grid_1-3
Energy Range:                 100.00 -   20000.00 eV
SliceEnergySpectra: Selected 3414 bins in range (    99.891-     19939) eV with indices␣
↪(4452-7866).
xstar2table: Install a check of the ENERGIES extension here?
xstar2table: Does it match the previous ENERGIES extension?
xstar2table: Successful Completion
xstar2table returns: xstar2table v1.0
 Compiled: Mar 15 2024 13:12:37
1) column      : Value is interpolated.
                 Logarithmically sampled at 5 points:
                 1e+19  1e+20  1e+21  1e+22  1e+23
2) rlogxi      : Value is interpolated.
                 Linearly sampled at 17 points:
                 -4  -3.5  -3  -2.5  -2  -1.5  -1  -0.5  0  0.5  1  1.5  2  2.5  3  3.5 ␣
↪4

Model Control Parameters:
Spectrum Type:            pow
Spectrum File:            spect.dat
Spectrum Units:           Energy
Abundance Table:
Is redshift a parameter?: Yes
Number of Steps:          3
Number of Iterations:     99
Write Switch:             Yes
Print Switch:             Yes
Step Size Choice Switch:  0
Number of Passes:         1
Constant Pressure Switch: No
Courant multiplier (emult)     0.50
```

(continues on next page)

(continued from previous page)

```
taumax for Courant step:       5.00
Minimum electron fraction:     0.10
Critical ion abundance:        0.00
Turbulent velocity (km/s):   300.00
radius exponent:               0.00
number of energy bins:    9999
Model Name:               XSTAR_fine_grid_1-3
Energy Range:               100.00 -   20000.00 eV
SliceEnergySpectra: Selected 3414 bins in range (    99.891-     19939) eV with indices␣
↪(4452-7866).
xstar2table: Install a check of the ENERGIES extension here?
xstar2table: Does it match the previous ENERGIES extension?
xstar2table: Successful Completion
XSTAR2XSPEC: 0 runs remaining to process.
rm: xout_detail.fits: No such file or directory
rm: xout_step.lis: No such file or directory
XSTAR2XSPEC: Execution time:   63464.00 seconds
XSTAR2XSPEC: Processing Complete!
```

10.3 WARMABS

10.3.1 Update population files

The warmabs model uses pre-calculated ion population and auxiliary files for efficient computation of synthetic spectra. Default popoulation files are calculated for an incident spectral energy distribution (SED) of a powerlaw with index 2. For any other SEDs, the population files should be re-calculated to ensure accurate results. This example illustrates the calculation of population files from and exisiting xspec SED model.

1. Tabulate the SED in a text file

In this exmaple we use the pyxspec to tabulate the comptt model. For a given source, the incident SED will have to be obtained by fitting an appropriate unabsorbed continuum model.

The first line of the text file must be the number of energies listed in the table. The remaining lines are the energy channel (in eV) and the flux in units of photons $\mathrm{cm^{-2}\,s^{-1}\,erg^{-1}}$ or $\mathrm{erg\,cm^{-2}\,s^{-1}\,erg^{-1}}$. Here we use the latter (default units). **Note that XSTAR will appropriately renormalize the luminosity.**

```
>>> import xspec
>>> xspec.Model("comptt")
```

```
========================================================================
Model compTT<1> Source No.: 1   Active/Off
Model Model Component  Parameter  Unit     Value
par  comp
1    1   compTT     Redshift           0.0          frozen
2    1   compTT     T0         keV     0.100000     +/-  0.0
3    1   compTT     kT         keV     50.0000      +/-  0.0
4    1   compTT     taup               1.00000      +/-  0.0
5    1   compTT     approx             1.00000      frozen
```

(continues on next page)

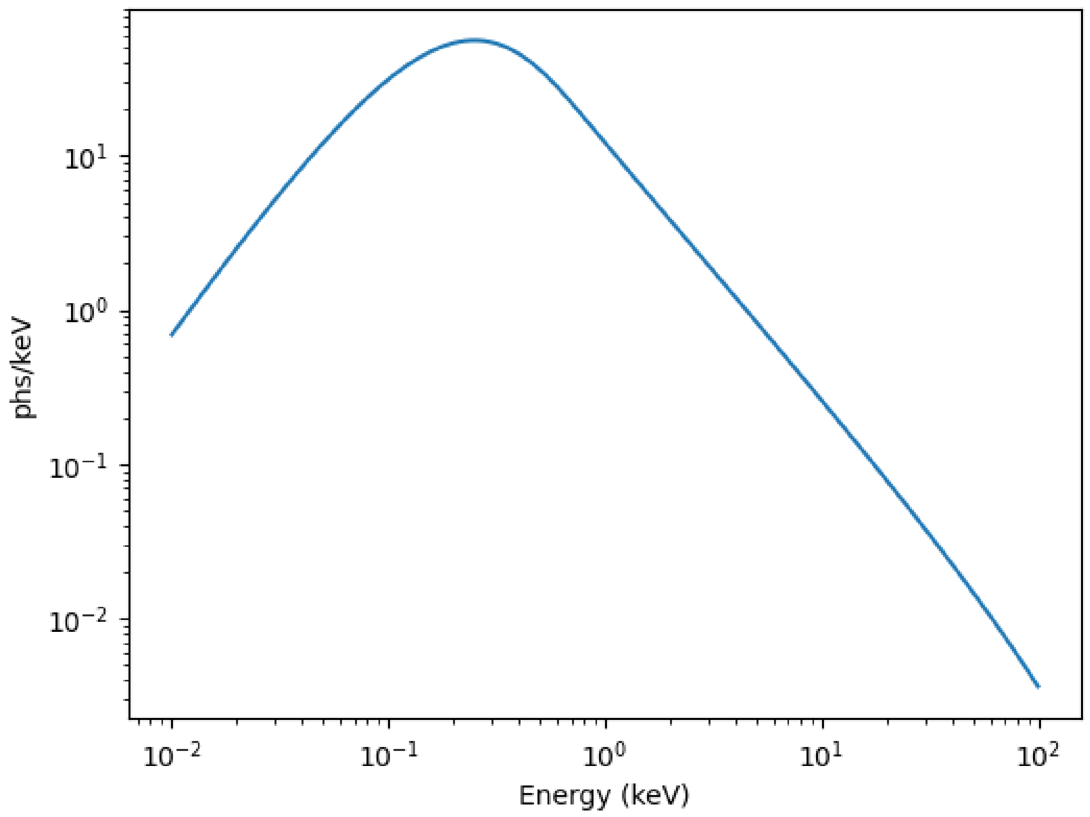
(continued from previous page)

```
6    1   compTT     norm                1.00000      +/-  0.0
________________________________________________________________________
```

```
>>> xspec.AllData.dummyrsp(.01,100.,1000,"log")
>>> xspec.plot.device = "/null"
>>> xspec.Plot("model")
>>> yVals= xspec.Plot.model(1) # Photon bin density
>>> xVals = xspec.Plot.x() # Bin center in keV
```

Inspect the photon spectrum.

```
>>> import matplotlib.pyplot as plt
>>> plt.plot(xVals, yVals)
>>> plt.xscale("log")
>>> plt.yscale("log")
>>> plt.xlabel("Energy (keV)")
>>> plt.ylabel("phs/keV")
>>> plt.show()
```



Write spectrum to file.

```
>>> of = open("comptt.txt", "w")
>>> of.write("%d\n" % len(yVals)) # First line contains number of energy bins
```

(continues on next page)

(continued from previous page)

```
>>> for ii in range(0, len(yVals)):
>>>    # energy in eV, flux in erg/cm^2/s/erg
>>>    of.write("%.4E\t%.4E\n" % (xVals[ii]*1e3, yVals[ii]*xVals[ii]))
>>> of.close()
```

```
unix> head comptt.txt
1000
1.0046E+01   6.9607E-03
1.0139E+01   7.1498E-03
1.0233E+01   7.3440E-03
1.0328E+01   7.5434E-03
1.0423E+01   7.7481E-03
1.0520E+01   7.9584E-03
1.0617E+01   8.1742E-03
1.0715E+01   8.3959E-03
1.0814E+01   8.6235E-03
```

2. Run `xstar` with this incident SED

We start in a fresh directory to prevent previous population files from being overwritten.

```
unix> mkdir popfiles
unix> cd popfiles
unix> mv <path/to/sed>/comptt.txt .
```

Now run `xstar`.

```
unix> xstar \
cfrac=1.0 \
temperature=1.0E+04 \
lcpres=0 \
pressure=3.0E-2 \
density=1.0E+4 \
spectrum='file' \
spectrum_file='comptt.txt' \
spectun=0 \
trad=-1. \
rlrad38=10.E-12 \
column=10.E+16 \
rlogxi=5.0 \
nsteps=20 \
niter=99 \
lwrite=1 \
lprint=1 \
lstep=0 \
habund=1.0E+00 \
heabund=1.0E+00 \
liabund=0.0E+00 \
beabund=0.0E+00 \
babund=0.0E+00 \
cabund=1.0E+00 \
```

(continues on next page)

(continued from previous page)

```
nabund=1.0E+00 \
oabund=1.0E+00 \
fabund=1.0E+00 \
neabund=1.0E+00 \
naabund=1.0E+00 \
mgabund=1.0E+00 \
alabund=1.0E+00 \
siabund=1.0E+00 \
pabund=1.0E+00 \
sabund=1.0E+00 \
clabund=1.0E+00 \
arabund=1.0E+00 \
kabund=1.0E+00 \
caabund=1.0E+00 \
scabund=1.0E+00 \
tiabund=1.0E+00 \
vabund=1.0E+00 \
crabund=1.0E+00 \
mnabund=1.0E+00 \
feabund=1.0E+00 \
coabund=1.0E+00 \
niabund=1.0E+00 \
cuabund=0.0E+00 \
znabund=0.0E+00 \
emult=5.0E-01 \
taumax=1.5E+01 \
xeemin=4.0E-02 \
critf=1.0E-06 \
vturbi=1.0E+00 \
npass=1.0E+00 \
modelname='custom_pops' \
loopcontrol=0
```

```
xstar version 2.59b
Loading Atomic Database...
initializng database...

pass number=           1           -1
log(r) delr/r log(N) log(xi) x_e   log(n) log(t) h-c(%) h-c(%) log(tau)
                                                        fwd    rev
9.00 -36.00 -10.00   5.00   1.21   4.00   7.79  -0.30   0.00 -10.00 -10.00  8
9.00 -36.00 -10.00   5.00   1.21   4.00   7.79  -0.30  -0.00 -10.00 -10.00  1
9.02  -1.32  11.70   4.96   1.21   4.00   7.79  -0.66  -0.00 -10.00 -10.00  1
9.04  -1.03  12.01   4.92   1.21   4.00   7.78  -0.02  -0.00 -10.00 -10.00  3
9.06  -0.87  12.20   4.87   1.21   4.00   7.78  -0.46  -0.00 -10.00 -10.00  1
9.08  -0.75  12.33   4.83   1.21   4.00   7.78  -0.94  -0.00 -10.00 -10.00  1
[...]
```

Upon completion, the following files will be created:

```
unix> ls
```

(continues on next page)

(continued from previous page)

```
comptt.txt
xo01_detal2.fits
xo01_detal4.fits
xout_cont1.fits
xout_rrc1.fits
xout_step.log
xo01_detail.fits
xo01_detal3.fits
xout_abund1.fits
xout_lines1.fits
xout_spect1.fits
xstar.par
```

Note that some of these files can be several GB in size.

3. Link atomic database and run `warmabs`

The atomic database for running `warmabs` needs to be same as used for the calculation of the population files and stored in the same location as the population files.

```
unix> ln -s $LHEA_DATA/atdb.fits atdbwarmabs.fits
unxi> ln -s $LHEA_DATA/coheat.dat coheatwarmabs.dat
```

Now reset the `$WARMABS_DATA` environment variable to point to the new population files and `warmabs` is ready to run.

```
unxi> export WARMABS_DATA=</path/to/popfiles>/popfiles
```

CHAPTER

ELEVEN

THE PHYSICS BEHIND XSTAR

11.1 Assumptions

In this section we describe the computational procedure, assumptions, free parameters, and the quantities which are calculated. Chief among the assumptions is that each model consists of a spherical gas cloud with a point source of continuum radiation at the center. Therefore it implicitly assumes spherical symmetry and radially beamed incident radiation. In principle, more complicated geometries can be mimiced by adding the local emission from various spherical sections with appropriately chosen conditions. Also important is the assumption that all physical processes affecting the state of the gas are in a steady-state, i.e. that the the timescales for variation in the gas density and illuminating radiation are long compared with timescales affecting all atomic processes and propogation of radiation within the gas. The validity of this assumption in any given situation depends on the conditions there, such as the gas density, temperature, and degree of ionization, and can be evaluated by using a model assuming steady-state and then calculating atomic rates which can (hopefully) justify the steady-state assumption a posteriori.

The primary difference between these models and atmospheric models lies in the treatment of the radiation field. In an optically thick atmosphere the state of the gas at any point in the cloud is coupled to the state of the gas in a large part of the rest of the cloud by the continuum radiation field and, in the limit of very large optical depth, can affect the excitation and ionization by suppressing radiative free-bound (recombination) transitions. We attempt to mimic some of these effects by assigning to each recombination event an escape probability, using an expression given in the following section. We also calculate the transfer of radiation by assuming that diffuse radiation emitted at each radius is directed radially outward or inward. These assumptions will be described in more detail later in this section.

A further assumption governs the treatment of the transport of radiation in spectral lines. Over a wide range of plausible situations large optical depths occur in the cores of lines of abundant ions, which may be important in cooling the gas. In treating the transfer of these photons we make the (conventional) assumption of complete redistribution in the scattering, which assumes that the transfer of the line photons occurs in a spatial region very close to the point where the photons are emitted. Therefore the line emission rates are multiplied by an escape probability using an expression given in the following section. This factor is intended to simulate the line scattering in the immediate vicinity of the emission region, and it assumes that escape from this region occurs when the photon scatters into a frequency where the optical depth is less than unity. Following escape from the local region, the line photon is assumed to be subject to absorption by continuum processes which are treated using the same 2-stream transfer equation as for the continuum.

11.2 Input

The input parameters are the source spectrum, the gas composition, and the gas density or pressure. The spectrum of the central source of radiation is described by the spectral luminosity, $L_{0\varepsilon} = Lf_\varepsilon$, where L is the total luminosity (in erg s^{-1}). The spectral function, f_ε, is normalized such that $\int_0^\infty f_\varepsilon d\varepsilon = 1$ and may be of one of a variety of types, including: Thermal bremsstrahlung, $f_\varepsilon \sim \exp(-\varepsilon/kT)$; blackbody, $f_\varepsilon \sim \varepsilon^3/[\exp(\varepsilon/kT) - 1]$; or power law, $f_\varepsilon \sim \varepsilon^\alpha$; or the user may define the form of the ionizing continuum by providing a table of energies and fluxes. The gas consists of the elements H, He, C, N, O, Ne, Mg, Si, S, Ar, Ca, Fe, and Ni with relative abundances specified by the user. The default abundances are the solar values given by Grevesse *et al.* [22].

11.3 Elementary Considerations

When the gas is optically thin, the radiation field at each radius is determined simply by geometrical dilution of the given source spectrum f_ε. Then, as shown by Tarter *et al.* [49], the state of the gas depends only on the ionization parameter $\xi = L/nR^2$, where L is the (energy) luminosity of the incident radiation integrated from 1 to 1000 Ry, n is the gas density, and R is the distance from the radiation source. This scaling law allows the results of one model calculation to be applied to a wide variety of situations. For a given choice of spectral shape this parameter is proportional to the various other customary ionization parameter definitions, i.e. $U_H = F_H/n$ [16], where F_H is the incident photon number flux above 1 Ry; $\Gamma = F_\nu(\nu_L)/(2hcn)$, where $F_\nu(\nu_L)$ is incident (energy) flux at 1 Ry; and $\Xi = L/(4\pi R^2 cnkT)$ (e.g. Krolik *et al.* [36]).

In the optically thick case, Hatchett *et al.* [26], and Kallman [30] showed that the state of the gas could be parameterized in terms of an additional parameter which is a function of the product of L and either n (the number density) or P (the pressure), depending on which quantity is held fixed. In the case n = constant, this second parameter is simply $(Ln)^{1/2}$ [40]. This parameter does not allow easy scaling of model results from value of Ln to another, since the dependence on this parameter is non-linear, but it does provide a useful indicator of which combinations of parameter values are likely to yield similar results and vice versa.

When the electron scattering optical depth, τ_e, of the cloud becomes significant, the outward-only approximation used here breaks down, and different methods of describing the radiative transfer must be used (e.g. Ross *et al.* [46]). Therefore, the range of validity of the models presented here is restricted to $\tau_e \leq 0.3$, or electron column densities $\leq 10^{24}$ cm^{-2}.

11.4 Algorithm

The construction of a model consists of the simultaneous determination of the state of the gas and the radiation field as a function of distance from the source.

11.4.1 Atomic Level Populations

The state of the gas is defined by its temperature and by the ionic level populations. As a practical matter, we maintain the distinction between the total abundance of a given ion relative to its parent element (the ion fraction or fractional abundance) and the relative populations of the various bound levels of that ion (level populations), although such distinctions are somewhat arbitrary given the presence of transitions linking non-adjacent ion stages and excited levels of adjacent ions.

Calculation of level populations proceeds in 2 steps. First, a calculation of ion fractions is performed using total ionization and recombination rates into and out of each ion analogous to those used in XSTAR v.1 (KM82). Then we eliminate ions with abundance less than a fixed fraction ϵ relative to hydrogen from further consideration. Experimentation has shown that $\epsilon = 10^{-8}$ (as parameterized by the input parameter critf) yields gas temperatures within 1% of those calculated using a larger set of ions for most situations when the density is low. This criterion leads to ion sets which can include up to 10 stages for heavy elements such as iron and nickel. We also make sure that the selected ions are all adjacent. i.e. we force the inclusion of ions which fall below our threshold if they are bracketed by ions which satisfy the abundance crterion.

The second step consists of solving the full kinetic equation matrix linking the various levels of the ions selected in step 1. We include all processes in the database which link the bound levels of any ion in our selected set with any other level, and also including the bare nucleus as the continuum level for the hydrogenic ion, if indicated. This results in a matrix with dimensionality which may be as large as 2400. The equations may be written schematically as (ratein) = (rateout) for each level. In place of the equation for the ground level of the most abundant ion we solve the number conservation constraint.

We include collisional and radiative bound-bound transitions (with continuum photoexcitation), collisional ionization, photoionization and recombination for all the levels of every ion for which the required atomic rate data is available. The effects of line scattering in all transitions are accounted for by taking into account the fact that line scattering reduces the net decay rate by repeated absorption and reemission of the line photon. An analogous procedure is used for free-bound (recombination) transitions.

11.4.2 Radiative Excitation

Excitation due to absorption of continuum photons in bound-bound transitions can dominate over other mechanisms, depending on the conditions. The importance of this has been pointed out and explored by Kinkhabwala *et al.* [35] and Band *et al.* [7]. Accurate treatment of this process requires that the geometry be treated carefully. This is because the net excitation rate is also affected by the radiation emitted by the decays of the excited levels. If every radiative excitation leads to a radiative deexcitation and emission of radiation which escapes the model volume then there is no net effect of this process. In the case of a stationary spherical model in which we are only interested in the total spectrum seen by a distant observer, and if radiative deexcitation is the only decay mechanism for the excited levels, then emission following radiative excitation and absorption will balance exactly. This is the scenario envisioned by classical nebular models, and for this reason these models did not generally include the effects of radiative excitation.

Radiative excitation has a cross section which can be much greater than the photoabsorption cross sections associated with photoionization. If so, it is important to accurately treat the spatial dependence of the absorption and depletion of the incident radiation in lines. This necessitates spatial steps which are small enough to resolve the line absorption, i.e. $\sim \kappa^{-1}$, where κ is the opacity in the line. Xstar accounts for this, but using the absorption cross section binned into the continuum grid. If the grid is coarse, then there will likely be no bin close to the centers of the strongest lines, and this process will not be accurately modeled. For this reason it is recommended that a fine energy grid be used for models where radiative excitation is of interest, and where the model cloud is optically thick in the line centers.

In order to preserve the classical result in which radiative excitation is completely ignored, xstar scales the rate for this process proportional to the value of the cfrac parameter. The rate of radiative excitation of a given transition is calculated using:

$$R_{lu} = \sigma_{line} F_{\varepsilon}$$

where :math:sigma_text{line}` is the line mean photoabsorption cross section and F_{ε} is the specific flux at the energy of the line. This rate is then multiplied by a factor of 1-cfrac. So, if cfrac=1 the effective rate of radiative excitation is zero. If cfrac=0 then the full rate of radiative excitaiton is included.

11.4.3 Atomic Levels

A large fraction of recombinations occur following cascades from a very large number of levels close to the continuum. Since explicit treatment of these levels is not feasible, we treat this process as follows (this procedure, along with detailed descriptions of other aspects of the database and the multilevel scheme are described in detail in : For every ion we choose a set of spectroscopic levels starting with the ground level, which are responsible for the identifiable emission lines and recombination continua; there are typically 10 – 50 of these for most ions, although for a few ions we include ≥ 100 such levels. In addition we include one or more superlevels and continuum levels. The continuum levels represent bound levels of more highly ionized species (in practice at most only a few such levels are of importance). The superlevel is an artificial level used to account for recombination onto the infinite series of levels that lie above the spectroscopic levels. In H and He-like ions the superlevels also account for the recombination cascades of these high lying levels onto the spectroscopic levels, and the rates for such decays are calculated by fitting to the results of population kinetic calculations for individual ions which explicitly include ≥ 1000 levels. For these isoelectronic sequences we explicitly include excited levels with a spectator electron, which give rise to satellite lines, excitation of these levels accounts for excitation-autoionization and radiative deexcitation, and recombination accounts for the dielectronic recombination process. For other iso-electronic sequences, the superlevels are assumed to decay directly to the ion's ground level, and the rates into and out of the superlevel are calculated in order to fit to the total recombination rates for the various ions []. This approach allows us to simultaneously account for the contributions of excitation, ionization, and recombination to the ion's level populations. In this way we solve ionization and excitation balance without the use of total recombination rates which is customary in many nebular calculations.

By using the approach described above and providing that every transition process accompanied by its detailed balance inverse process we insure that the level populations will naturally converge to LTE under proper conditions.

11.4.4 Thermal Equilibrium

The temperature is found by solving the equation of thermal equilibrium, which may be written schematically as (Heating) = (Cooling). This is solved simultaneously with the condition of charge conservation. We treat heating and cooling by calculating the rate of removal or addition of energy to local radiation field associated with each of the processes affecting level populations (this is in contrast to the method where these were calculated via their effects on the electron thermal bath as in KM82). Heating therefore includes photoionization heating and Compton heating. The cooling term includes radiative recombination, bremsstrahlung, and radiative deexcitation of bound levels. Cooling due to recombination and radiative deexcitation is included only for the escaping fraction, as described elsewhere in this section.

In the most highly ionized regions of our models, the dominant heating process is electron recoil following Compton scattering. In the non-relativistic approximation the net heating rate may be written (Ross 1979)

$$n_e\Gamma_e = \frac{\sigma_T}{m_e c^2}\left(\int \varepsilon J_\varepsilon d\varepsilon - 4kT\int J_\varepsilon d\varepsilon\right)(1)$$

Here σ_T is the Thomson cross section, n_e is the electron number density, T is the electron temperature, and J_ε is the local mean intensity in the radiation field. The first term in the brackets represents the heating of electrons by the X-rays, and the second term represents cooling of hot electrons by scattering with low energy photons. The treatment of Compton heating and cooling in versions prior to 2.3 were not accurate for hard spectra with significant flux above 100 keV. This has been updated in version 2.3 using rates from I. Khabibullin (private communication), based on the expressions given by Shestakov et al. (1988 JQSRT 40 577) The energy shift per scattering is calculated by interpolating in a table (coheat.dat).

The spectrum of photoelectron energies for each ion is found by convolving the radiation field, weighted by photoelectron energy, with the photoionization cross section (see, e.g., Osterbrock [43]). The integral of this quantity provides the photoelectric heating rate.

The cooling rate due to radiative recombination is calculated by explicitly evaluating the quadrature over the recombination continuum spectrum for each recombining level, weighted by the escape fraction for that transition. The bremsstrahlung cooling rate is (Osterbrock [43])

$$n_e\Gamma_e = 1.42 \times 10^{-27} T^{1/2} z^2 n_e n_z \text{ ergs cm}^{-3}\text{s}^{-1}, (2)$$

where T is the electron temperature, is the electron number density, z is the charge on the cooling ion, and n_z is the ion density.

Heating and Cooling: "Photon Point of View" vs. "Electron Point of View"

Consideration of heating and cooling depends on the definition of the energy scale. Two different natural ways to do this are the Photon Point of View and the Electron Point of View. In the former description we keep track of the destruction and creation of photons, and a heating or cooling event corresponds to the these two processes, respectively. In the Electron Point of View we keep track of energy deposited or extracted from the electron thermal bath. The two descriptions differ owing to the internal energy associated with ionization: processes which involoving bound-bound transitions have no effect on the electron thermal bath but do emit or absorb photons. Electron impact collisions (by themselves) have no effect on the photon distribution but do heat or cool the electrons. Photoionization heats the electrons with a rate proportional to the energy of the absorbed photon above threshold; it removes photons at a rate proportional to the energy of the absorbed photons above the ion ground level. Radiative recombination differs in a corresponding way between the two descriptions. Compton heating/cooling and bremsstrahlung cooling are the same in the two descriptions.

A key point is that in the 'photon point of view' calculation there is recombination cooling even at low temperature, since every recombination releases a continuum photon and also possibly line photons. Recombination is a monotonically decreasing function of temperature, and the cooling curves have this behavior at low temperature. In constrast, the 'electron point of view' cooling rates are dominated by bremsstrahlung, which goes as $T^{1/2}$, at low and high temperatures.

It is easy to show that the two descriptions are equivalent when ionization equilibrium is achieved; the quantity heating-cooling is the same. Xstar uses the photon point of view for all its calculations of thermal balance. Traditional photoionization codes use the electron point of view. For ease of comparison, xstar (starting with version 2.54) also computes the rates per ion and total rates in the electron point of view and prints them out in the ascii file.

11.5 Recombination Continuum Emission and Escape

In analogy with the line emission, recombination emission and cooling rates are calculated using the continuum level population n_∞ and the quantities calculated from the photoionization cross section and the Milne relation. The spontaneous recombination rates are given by

$$\alpha_i = (\frac{n_i}{n_{i+1} n_e})^* \int_{\varepsilon_{th}}^{\infty} \frac{d\varepsilon}{\varepsilon} \frac{\varepsilon^3}{h^3 c^2} \sigma_{pi} e^{(\varepsilon_{th}-\varepsilon)/kT} (3)$$

where $n_i{}^*$ is the LTE density of ion i, and n_e is the electron density. The continuum emissivity due to this process is given by

$$j_\varepsilon = n_{upper} n_e (\frac{n_i}{n_{i+1} n_e})^* \frac{\varepsilon^3}{h^3 c^2} \sigma_{pi} e^{(\varepsilon_{th}-\varepsilon)/kT} (4)$$

where n_{upper} is the number density of ions in the recombining level and σ_{pi} is the photoionization cross section. The cooling rate is given by the integral of this expression over energy. These rates are calculated separately for each level included in the multilevel calculation.

In order to account for the suppression of rates due to emission and reabsorption of recombination continua, we multiply the rates and emissivities by an escape fraction given by:

$$P_{esc.,cont.} = \frac{1}{1000\tau_{cont.} + 1} (5)$$

where $\tau_{\text{cont.}}$ is the optical depth at the threshold energy for the relevant transition. This factor is used to correct both the emission rate for the recombination events, and also the rates in the kinetic equations determining level populations etc, and has been found to give reasonably good fits to the results of more detailed calculations for the case of H II region models in which the Lyman continuum of hydrogen is optically thick (e.g. Harrington [25]).

11.5.1 Line Emission and Escape

Since all level populations are calculated explicitly, line emissivities and cooling rates are calculated as a straightforward product of the population of the line upper level, the spontaneous transition probability and an escape fraction.

Line optical depths may be large in some nebular situations. Photons emitted near the centers of these lines are likely to be absorbed by the transition which emitted them and reemitted at a new frequency. This line scattering will repeat many times until the photon either escapes the gas, is destroyed by continuum photoabsorption or collisional deexcitation, or is degraded into longer wavelength photons which may then escape. Our treatment of resonance line transfer is based on the assumption of complete redistribution. That is, we assume that there is no correlation of photon frequencies before and after each scattering event. This has been shown to be a good approximation for a wide variety of situations, particularly when the line profile is dominated by Doppler broadening. In this case, more accurate numerical simulations (e.g., Hummer and Rybicki [28]) have shown that line scattering is restricted to a small spatial region near the point where the photons are emitted. Line photons first scatter to a frequency such that the gas cloud is optically thin and then escape in a single long flight. The probability of escape per scattering depends on the optical depth, τ_0 at the center of the line. For $1 \leq \tau_0 \leq 10^6$, the resonant trapping is effectively local. For $\tau_0 \geq 10^6$, the lines become optically thick in the damping wings, and the line escapes as a result of diffusion in both space and frequency. Since the scattering in the Doppler core is always dominated by complete redistribution, and since most of the lines in our models are optically thin in the wings, we assume that all line scattering takes place in the emission region.

We use the following expression for escape probability (Kwan and Krolik [37]):

$$P_{esc.,line}(\tau_{line}) = \frac{1}{\tau_{line}\sqrt{\pi}(1.2+b)}(\tau_{line} \geq 1)(6)$$

$$P_{esc.,line}(\tau_{line}) = \frac{1-e^{-2\tau_{line}}}{2\tau_{line}}(\tau_{line} \leq 1)(7)$$

where

$$b = \frac{\sqrt{\log(\tau_{line})}}{1+\tau_{line}/\tau_w}(8),$$

$\tau_{\rm line}$ is the optical depth at line center, and $\tau_w = 10^5$.

The rates for line emission and the probabilities for the various resonance line escape and destruction probabilities depend on the state of the gas at each point in the cloud. The cooling function for the gas depends on the line escape probabilities, and the effects of line trapping must be incorporated in the solution for the temperature and ionization of the gas. Once the state of the gas at a given point has been determined, the emission in each line is calculated as the product of the upper level population and the corresponding net decay rate, including the suppression due to multiple scattering.

11.6 Continuum Emission

Diffuse continuum radiation is emitted by three processes: thermal bremsstrahlung, radiative recombination, and two-photon decays of metastable levels. The thermal bremsstrahlung emissivity is given by Osterbrock [43]:

$$j_\varepsilon = \frac{1}{4\pi}n_z n_e \frac{32Z^2e^4h}{3m^2c^3}\left(\frac{\pi h\nu_0}{3kT}\right)^{1/2} e^{-h\nu/kT} g_{ff}(T,Z,r)(9)$$

where T is the electron temperature, n_e is the electron abundance, Z is the charge on the most abundant ion, n_z is the abundance of that ion, and $g_{\rm ff}$ is a Gaunt factor [31]. For two photon decays, we adopt the distribution [50]:

$$H\left(\frac{\varepsilon}{\varepsilon_0}\right) = 12\left(\frac{\varepsilon}{\varepsilon_0}\right)^2\left(1-\frac{\varepsilon}{\varepsilon_0}\right)(10)$$

where ε_0 is the excitation energy.

11.6.1 Continuum Opacity and Thomson Scattering

Continuum photons can be absorbed by photoionization. This is regarded as a photon destruction process since the converse, radiative recombination, produces an 'RRC' spectrum which has a different energy dependnce than the absorption, and so the two are treated separately. Xstar includes the photoabsorption associated with every bound-free transition in the atomic database; there are $\sim 10^5$ of them.

Photons can scatter via Thomson scattering or resonance scattering in lines. Xstar currently includes resonance scattering in lines in the final spectrum seen by a distant observer (as described in the following section). Thomson scattering does not strongly affect the shape of the spectrum unless the photon energy or temperature are a significant fraction of $m_e c^2$. Furthermore, similar arguments apply to Thomson scattering as apply to the radiative excitation in section 9.D.2. In the case of a spherical stationary cloud where all the photons from the cloud are observed, Thomson scattering will have no net effect. For this reason, version of xstar prior to 2.54a omitted Thomson scattering. Beginning with verion 2.54a, Thomson scattering is included with the same multiplicative factor 1-cfrac as is used for radiative excitation. Thus, when cfrac=1 the classical results without Thomson scattering are recovered, and when cfrac=0 Thomson scattering is included in the continuum opacity. Of cours, Thomson scattering is unimportant when the cloud column is less than $\sim 10^{23}\text{cm}^{-2}$. And, when the cloud is optically thick to Thomson scattering, xstar does not adequately treat the effects of multiple scatterings.

11.6.2 Continuum Transfer

The continuum radiation field is modified primarily by photoabsorption, for which the opacity, $\kappa(\varepsilon)$, is equal to the product of the ion abundance with the total photoionization cross section, summed over all levels.

A model is constructed by dividing the cloud into a set of concentric spherical shells. The radiation field incident on the innermost shell is the source spectrum. For each shell, starting with the innermost one, the ionization and temperature structure is calculated from the local balance equations using the radiation field incident on the inner surface. The attenuation of the incident radiation field by the shell is then calculated. The diffuse radiation emitted by the cloud is calculated using an expression of the formal solution if the equation of transfer:

$$L_\varepsilon = \int_{R_{inner}}^{R_{outer}} 4\pi R^2 j_\varepsilon(R) e^{-\tau_{cont.}(R,\varepsilon)} dR (11)$$

where L_ε is the specific luminosity at the cloud boundary, $\tau_{\text{cont.}}(R, \varepsilon)$ is the optical depth from R to the boundary, and j_ε is the emissivity at the radius R. Since our models in general have two boundaries, we perform this calculation for radiation escaping at both the inner and outer cloud boundaries. This calculation is performed for each continuum energy bin, and separately for each line. In the case of the continuum, we construct a vector of emissivities, $j_\varepsilon(R)$ which includes contributions from the escaping fraction from all the levels which affect each energy. For the lines, the emissivity used in this equation is the escaping fraction for that line.

11.6.3 Radiation Field Quantities and Transfer Details

Equation (11) conceals a variety of important issues concerning the treatment of the radiation field and the values which are printed in the various output files produced by xstar. In an effort to clarify this we present here a complete description of the various radiation field quantities which are used internally to xstar, and which are output to the user. In this subsection, all radiation fields are specific luminosity, L_ε, in units erg/s/erg for the continuum, and luminosity, L_i, in units erg/s for lines. We distiguish several different radiation fields. First, the radiation field used locally by xstar for the calculation of photoionization rates and heating, we denote $L_\varepsilon^{(1)}$. This is calculated during an outward iteration using the transfer equation:

$$\frac{\mathrm{d}L_\varepsilon^{(1)}}{\mathrm{d}R} = -\kappa_{cont}(\varepsilon) L_\varepsilon^{(1)} + 4\pi R^2 j_\varepsilon(R) (12)$$

with the boundary condition that $L_\varepsilon^{(1)} = L_\varepsilon^{(inc)}$ at the inner radius of the cloud. Here $\kappa_{cont}(\varepsilon)$ and j_ε are the local continuum opacity and emissivity and $L_\varepsilon^{(inc)}$ is the incident radiation field at the inner edge of the cloud.

In addition we can define the various radiation fields of interest for use in fitting to observed data. These include the spectrum transmitted by a model, i.e. the radiation which would be observed if the incident radiation field were subject to absorption alone:

$$L_\varepsilon^{(2)} = L_\varepsilon^{(inc)} e^{-\tau_{cont}^{(tot)}(\varepsilon)} (13)$$

where $\tau_{cont}^{(tot)}(\varepsilon)$ is the total optical depth through the model cloud due to continuum photoabsorption,

$$\tau_{cont}^{(tot)}(\varepsilon) = \int_{R_{inner}}^{R_{outer}} \kappa_{cont}(\varepsilon) dR (14)$$

Also of interest is the total emitted continuum radiation in both the inward and outward directions, which is given by equations similar to (11):

$$L_\varepsilon^{(3)} = \int_{R_{inner}}^{R_{outer}} 4\pi R^2 j_\varepsilon(R) e^{-\tau_{cont}^{(in)}(\varepsilon)} P_{esc,cont.}^{(in)}(R) dR (15)$$
$$L_\varepsilon^{(4)} = \int_{R_{inner}}^{R_{outer}} 4\pi R^2 j_\varepsilon(R) e^{-\tau_{cont}^{(out)}(\varepsilon)} P_{esc,cont.}^{(out)}(R) dR (16)$$

where the escape probabilities in the inward and outward directions are $P_{esc,cont.}^{(in)}(R) = (1 - C)/2$ and $P_{esc,cont.}^{(out)}(R) = (1 + C)/2$, where C is the covering fraction, specified as an input parameter, and $\tau_{cont}^{(out)}(\varepsilon)$ and $\tau_{cont}^{(out)}(\varepsilon)$ are the continuum optical depths in the inward and outward directions.

Line luminosities are calculated separately, one at a time, according to an equation analogous to equation (12):

$$\frac{dL_i^{(1)}}{dR} = -\kappa_{cont}(\varepsilon) L_i^{(1)} + 4\pi R^2 j_i(R) P_{esc,line}^{(in)} (16)$$

$$\frac{dL_i^{(2)}}{dR} = -\kappa_{cont}(\varepsilon) L_i^{(2)} + 4\pi R^2 j_i(R) P_{esc,line}^{(out)} (17)$$

where $L_i^{(1)}$ and $L_i^{(2)}$ are the luminosities of individual lines in the inward and outward directions, respectively. The escape probabilities in the inward and outward directions are calculated using $P_{esc.,line}(\tau_{line})$ from equations (6)-(8) and $P_{esc.,line}^{(in)} = (1 - C) P_{esc.,line}(\tau_i^{(in)})$ and $P_{esc,line}^{(out)} = (1 - C) P_{esc.,line}(\tau_i^{(out)}) + C P_{esc.,line}(\tau_i^{(out)} + \tau_i^{(in)})/2$, and $\tau_i^{(in)}$ and $\tau_i^{(out)}$ are the line scattering optical depths in the inward and outward directions:

$$\tau_i^{(in)}(R) = \int_{R_{inner}}^{R} \kappa_i \mathrm{dR} (18)$$
$$\tau_i^{(out)}(R) = \int_{R}^{R_{outer}} \kappa_i \mathrm{dR} (19)$$

and κ_i is the line center opacity.

None of the continuum luminosities defined in equations (12)-(16) have the effects of lines included, either in emission or absorption. This is because lines scatter the radiation, while photoionization is true absorption. The effects of lines on the continuum can be added to the continuum for the purposes of comparing with observed spectra by binning the lines, i.e. we can calculate the binned specific luminosity and opacity:

$$L_{line,\varepsilon}^{(in)} = \Sigma_{i \ni |\varepsilon_i - \varepsilon| \leq \Delta\varepsilon} \frac{L_i^{(1)} \phi(\varepsilon - \varepsilon_i)}{\Delta\varepsilon} (20)$$
$$L_{line,\varepsilon}^{(out)} = \Sigma_{i \ni |\varepsilon_i - \varepsilon| \leq \Delta\varepsilon} \frac{L_i^{(2)} \phi(\varepsilon - \varepsilon_i)}{\Delta\varepsilon} (21)$$
$$\kappa_{line}(\varepsilon) = \Sigma_{i \ni |\varepsilon_i - \varepsilon| \leq \Delta\varepsilon} \kappa_i \phi(\varepsilon - \varepsilon_i) (22)$$

where ε and $\Delta\varepsilon$ are the energy and width, respectively, of the continuum bin closest to line i, and $\phi(\varepsilon - \varepsilon_i)$ is the profile function including the effects of broadening due to thermal Doppler motions, natural broadening, and turbulence.

Then we can define the total optical depth of the cloud

$$\tau^{(tot)}(\varepsilon) = \int_{R_{inner}}^{R_{outer}} (\kappa_{cont}(\varepsilon) + \kappa_{line}(\varepsilon))\mathrm{dR}(23)$$

and the total transmitted specific luminosity

$$L_\varepsilon^{(5)} = L_\varepsilon^{(inc)} e^{-\tau^{(tot)}(\varepsilon)}(24)$$

and the total emitted specific luminosity in the inward and outward directions:

$$L_\varepsilon^{(6)} = L_\varepsilon^{(3)} + L_{line,\varepsilon}^{(in)}(25)$$
$$L_\varepsilon^{(7)} = L_\varepsilon^{(4)} + L_{line,\varepsilon}^{(out)}(26)$$

The quantities $L_\varepsilon^{(inc)}$, $L_\varepsilon^{(5)}$, $L_\varepsilon^{(6)}$ and $L_\varepsilon^{(7)}$ are output in columns 2,3,4,5 of the file xout_spect1.fits. The quantities $L_\varepsilon^{(inc)}$ $L_\varepsilon^{(5)}$, $L_\varepsilon^{(3)}$ and $L_\varepsilon^{(4)}$ are output in columns 2,3,4,5 of the file xout_cont1.fits.

In fact, the lines should be included in the continuum which is responsible for the local ionization and heating of the gas, since they can contribute to these processes. So we define a modified version of equation (12):

$$\frac{\mathrm{d}L_\varepsilon^{(1')}}{\mathrm{d}R} = -\kappa_{cont}(\varepsilon)L_\varepsilon^{(1')} + 4\pi R^2 j_\varepsilon(R) + 4\pi R^2 \Sigma_{i\ni|\varepsilon_i-\varepsilon|\leq\Delta\varepsilon} \frac{j_i(R)P_{esc,line}^{(out)}\phi(\varepsilon-\varepsilon_i)}{\Delta\varepsilon}(27)$$

$L_\varepsilon^{(1')}$ is the quantity which is used by xstar to calculate the local ionizing flux. This is the quantity which is conserved by xstar when it calculates heating=cooling.

The quantities $L_i^{(1)}$, $L_i^{(2)}$, $\tau_i^{(in)}$ and $\tau_i^{(out)}$ are output in columns 6,7,8,9 of the file xout_lines1.fits.

The quantities which contain the continuum only, before the lines are binned and added, are printed out to the log file, xout_step.log, when the print switch lpri is set to 1 or greater. Then they are in a table following the label 'continuum luminosities'. The quantities $L_\varepsilon^{(inc)}$, $L_\varepsilon^{(1')}$, $L_\varepsilon^{(3)}$, $L_\varepsilon^{(4)}$, $\tau_{cont}^{(in)}(\varepsilon)$ and $\tau_{cont}^{(out)}(\varepsilon)$ are output in columns 3,4,5,6,7 and 8. Many other useful quantities are output to the log file when lpri=1. This includes the quantities $L_i^{(1)}$, $L_i^{(2)}$, $\tau_i^{(in)}$ and $\tau_i^{(out)}$ are in columns 2-5 following the label 'line luminosities'

11.6.4 Energy Conservation

Energy conservation is imposed as a constraint when determining the temperature in xstar when the input parameter niter is non-zero. If so, the temperature is iteratively improved until the heating and cooling rates are locally equal. This is implemented by calculating the integral over the absorbed and emitted continuum energy in a given spatial zone, and also the sum over the energy emitted in the lines. Compton heating and cooling are added analytically, since Comptonization of the radiation field is not treated. The error resulting from this procedure is tabulated in the log file 'xout_step.log' in the 8th column of the step-by-step output, in units of %.

Energy conservation locally should correspond to global energy conservation, i.e. that the total absorbed energy in the radiation field equals the total emitted energy in lines plus continuum. This is tested at each spatial zone in xstar by calculating $\int(L_\varepsilon^{(inc)} - L_\varepsilon^{(1)})d\varepsilon - \Sigma_i(L_i^{(1)} + L_i^{(2)})$. The error resulting from this procedure is tabulated in the log file 'xout_step.log' in the 9th column of the step-by-step output, in units of %.

It is important to point out that the specific luminosities in the file 'xout_spect1.fits' 'xout_cont1.fits' are not expected, in general, to show energy conservation. This is primarily because the transmitted spectra in both of these files contain the effects of binned lines. Line opacity is expected to produce a scattering event, i.e. the photon is likely to be reemitted near the same energy. This differs qualitatively from photoelectric absorption, in which an absorbed photon is likely to be reemitted at a very different energy, with an accompanying net loss or gain of energy to the electron thermal bath. Line opacity is not included in the radiative equilibrium integral used to calculate the gas temperature, and so the total absorbed energy in the radiation field $L_\varepsilon^{(5)}$ will in general not equal the emitted energy in $L_\varepsilon^{(6)} + L_\varepsilon^{(7)}$. Energy conservation can be checked using the quantities $L_\varepsilon^{(1)}$, $L_i^{(1)}$ and $L_i^{(2)}$ from the file xout_step.log.

Energy conservation checked using binned line spectra is also affected by the errors introduced by binning. This is discussed at length in the section of this manual on table models for xspec, but we emphasize here that the binned spectrum cannot be accurately integrated to derive the total line absorption or emission unless the lines are broad compared with the energy grid spacing (requiring turbulent velocities $\geq$ 500 km/s currently).

11.6.5 Algorithm

Construction of a model of an X-ray illuminated cloud consists of the simultaneous solution of the local balance equations. The radiative transfer equation is solved for both the continuum and for the lines that escape the region near the point of emission. The large number of ions in the calculation results in many ionization edges that may affect the radiation field. We solve the transfer equation on a frequency grid that includes a total of 9999 continuum grid points with even logarithmic spacing in energy from 0.1 eV to 20 keV resulting in a limiting resolution of 0.12 %, corresponding to, e.g. 8.6 eV at 7 keV. We calculate the luminosities of ~ 10000 spectral lines and solve the continuum transfer equation individually for each of these. The emissivity of each line at each point is the product of the emissivity and the local escape fraction for that line. The continuum opacity for each line is the opacity calculated for the energy bin that contains the line. This procedure is repeated for each successive shell with increasing radius.

Calculation of the escape of the diffuse radiation field depends on a knowledge of the optical depths of the cloud from any point to both the inner and outer boundaries. Since these are not known a priori we iteratively calculate the cloud structure by stepping through the radial shells at least 3 times. For the initial pass through the shells we assume that the optical depths in the outward direction are zero. This procedure is found to converge satisfactorily within 3-5 passes for most problems of interest. This procedure is tantamount to the Λ-iteration" procedure familiar from stellar atmospheres, and must suffer from the same convergence problems when applied to problems with large optical depths. These problems are reduced in our case by the use of escape probabilities rather than a full integration of the equation of transfer.

11.7 Atomic Processes

Here we summarize the most important data sources adopted for the calculations. These are discussed in greater detail, along with a description of the fitting formulas and assumptions, in .

11.7.1 Photoionization

Photoionization rates are obtained by convolving the radiation field with the photoionization cross section. Cross sections are included for all levels of every ion for a wide range of photon energies occurring in our model. The cross sections where taken from the Opacity Project [15, 48], then averaged over resonances as in Bautista *et al.* [9] and split over fine structure according to statistical weights.

For inner shell photoionization not yet available from the opacity project we use the cross sections of Verner and Yakovlev [51]. Inner shell ionization in X-ray illuminated clouds is enhanced by Auger cascades. This process can result in the ejection of up to eight extra electrons (in the case of iron) in addition to the original photoelectron. We include this effect by treating each inner shell ionization/auger event as a rate connecting the ground state of one ion with another level of an ion (in general not adjacent to the initial ion). The rates for inner shell ionization/auger processes are calculated using the relative probabilities of the various possible outcomes of an inner shell ionization event from Kaastra and Mewe [29]. These yields are multiplied by the appropriate inner shell photoionization cross section in order to calculate a rate for each inner shell ionization/Auger cascade individually. Our level scheme also includes levels with inner shell vacancies, which are populated by inner shell/Auger events. Populations of these levels are calculated in the customary way, and the decays from these levels produce inner shell flourescence lines.

11.7.2 Collisional Ionization

Ionization by electron collisions is important if the gas temperature approaches a fraction of the ionization threshold energy of the most abundant ions in the gas. For ground states we include the rates from Raymond and Smith [45] for elements other than iron, and from Arnaud and Raymond [5] for iron. Collisional ionization from excited levels may also be important to the ionization balance. We include ionization rates for all excited levels of every ion using approximate formulae by D. Sampson and coworkers [53]. 3-body recombination rates to all levels are calculated from the collisional ionization rates using the detailed balance principle.

11.7.3 Recombination

Radiative and dielectronic recombination rates to all spectroscopic levels are calculated from the photoionization cross sections using the Milne relation. We include both spontaneous and stimulated recombination caused by the illuminating radiation. Stimulated recombination by the locally emitted radiation is not treated explicitly, although its effect is taken into account in an approximate way by suppressing a fraction of the spontaneous recombinations using the escape probability described earlier in this section. Recombination onto the superlevels is calculated in order to account for the difference between the sum over all spectroscopic levels and the total ion recombination as given by Nahar and coworkers [41] where available and Aldrovandi and Pequignot [1] for species other than iron ions and ions in the H and He isoelectronic sequences. For iron we use total rates from Arnaud and Raymond [5]. For H and He-like ions the total recombination rates were calculated by Bautista *et al.* [9] and .

11.7.4 Collisional Excitation and Radiative Transition Probabilities

Collision strengths and A-values were collected from a large number of sources. Particularly important for this compilation were the CHIANTI data base [17] for X-ray and EUV lines, and the extensive R-matrix calculations by the Iron Project [27].

11.7.5 Charge Transfer

Rates for charge transfer reactions are taken from Butler *et al.* [13]. For highly charged ions, where accurate calculations do not exist, we scale the rates along isonuclear sequences, assuming that the cross section is proportional to the square of the total residual charge transfer reaction of O II with H [19].

Atomic Database

CHAPTER

TWELVE

GENERAL DESCRIPTION

The database system used by XSTAR version 2 attempts to separate, as much as possible, the numerical quantities which determine the various atomic rates from the Fortran code which actually performs the calculation. The goal is make the atomic data modular, so that new data can be adopted or tested without requiring extensive modifications to the code. The way this is done is to separate the data from the code itself, and store the data in a database which is designed specifically for use by XSTAR. The database is divided into "records", each of which corresponds to a given physical process affecting a given level or pair of levels. An example is the radiative decay of hydrogen from the 2p to the 1s level. Each record contains numerical constants needed to calculate the rate for the process, in this example simply the Einstein *A* value for the transition, together with various other associated quantities. Chief among these are two integers which describe the how the constants are to be used. The first integer is denoted the "data type", and describes the fitting formula to be used in order to calculate a rate from the constants. The second integer is the "rate type", which describes how XSTAR uses the rates calculated. The list of data types is already quite long and is expected to grow and change as new data is adopted into the database, but not all data types are used by the current database. In order to interpret the various data types, XSTAR contains one central data calculating subroutine, denoted ucalc.f, which branches to various segments of code (and calls to specialized subroutines) which are tailored to each data type. ucalc.f returns the rates in a standard form for use by the other XSTAR subroutines. It is expected that ucalc.f will require additions in order to handle new data types as they are adopted. The list of rate types is not intended to grow, since such changes could require changes to the rest of the XSTAR code structure.

The XSTAR database system can be divided into 2 parts:

First, and most important, is the ASCII file containing all the data. That is, this file contains all the numerical data and labels required for calculation of all atomic rates and resultant quantities. This includes all level excitation energies, statistical weights and spectroscopic names, all element names and abundances, all ion names, and of course all photoionization cross sections, collision rates, recombination rates, fluorescence yeilds, and line wavelengths. This file is separated into records, corresponding crudely to lines of text, although many records extend over more than one line. Each record consists of a header, followed by the data. The header currently consists of 6 integers: the data type, the rate type, a continuation flag (currently unused), the number of reals in the record, the number of integers in the record, and the numbers of characters in the record. Then follows the real data, the integer data, and the character data. The various fields within the record are separated by one or more spaces. The record is terminated with a %, and the entire database is terminated by a single line containing %%%%. Each record can currently contain up to 2000 of any of the types of constants: real, integer, or character. In the XSTAR source tree this file is named atdat.text and currently is approximately 3 GB in size.

The second part of the database is the subroutine ucalc.f. This routine, when passed the contents of a record, returns the result of the rate calculation for the corresponding process. ucalc therefore contains all of the various arithmetic expressions corresponding to rates for various physical processes. ucalc returns generally 4 real rates and two integers. The rates are: rate, inverse rate, heating rate, and cooling rate. The integers are indeces of the levels involved, lower and upper. Not all data types return all 4 rates.

12.1 Rate types

The list of rate types currently included in ucalc are as follows:

01
: Ground state ionization.

03
: Bound-bound collision.

04
: Bound-bound radiative.

05
: Bound-free collision (level).

06
: Total recombination.

07
: Bound-free radiative (level).

08
: Total recombination, forces norm.

09
: 2-photon decay.

11
: Element data.

12
: Ion data.

13
: Level data.

14
: Bound-bound radiative superlevel-spectroscopic level.

15
: Collisional ionization total rate.

40
: Bound-bound collisional superlevel-spectroscopic level.

41
: Non-radiative Auger transition.

42
: Inner shell photoabsorption followed by autoionization.

12.2 Data types

01

Radiative recombination rate coefficient of N-electron recombined ion [1, 2]: `r1` = $A_{\rm rad}$ (cm^3 s^{-1}); `r2` = η; `i1` = ion_N.

02

H^0 charge exchange rate coefficient of N-electron recombined ion [34]: `r1` = a (10^{-9} cm^3 s^{-1}); `r2` = b; `r3` = c; `r4` = d; `r5` = T_1 (K); `r6` = T_2 (K); `r7` = $\Delta E/k$ (10^4 K); `i1` = ion_N; `s1` = recombining ion identifier.

06

Data attributes of the i-th level of N-electron ion: `r1` = $E(i)$ (eV); `r2` = $(2J + 1)$; `r3` = ν (effective quantum number); `r4` = $E(\infty)$ (eV); `i1` = n; `i2` = $(2S + 1)$; `i3` = L; `i4` = Z; `i5` = i; `i6` = ion_N; `s1` = level configuration assignment.

07

Dielectronic recombination rate coefficient of N-electron recombined ion [1, 2]: `r1` = $A_{\rm di}$ (cm^3 s^{-1} K$^{3/2}$); `r2` = $B_{\rm di}$; `r3` = T_0 (K); `r4` = T_1 (K); `i1` = ion_N.

14

Ionization potential of N-electron ion: `r1` = $E(\infty)$ (eV); `i1` = $Z - N + 1$; `i2` = Z; `i3` = ion_N; `s1` = ion identifier.

22

Dielectronic recombination rate coefficient of the N-electron recombined ion [42]: `r1` = a; `r2` = b; `r3` = c; `r4` = d; `r5` = e; `r6` = f; `i1` = ion_N.

30

Total radiative recombination rate (hydrogenic) for N-electron recombined ion [21]: `i1` = Z; `i2` = ion_N.

38

Total radiative recombination rate coefficient of N-electron recombined ion [http://amdpp.phys.strath.ac.uk/tamoc/DATA/RR/]: `r1` = A(cm^3 s^{-1}); `r2` = B; `r3` = T_0 (K); `r4` = T_1 (K); `r5` = C; `r6` = T_2 (K); `i1` = Z; `i2` = $N - 1$; `i3` = M; `i4` = W; `i5` = ion_N.

39

Total dielectronic recombination rate coefficient of N-electron recombined ion [http://amdpp.phys.strath.ac.uk/tamoc/DATA/DR/]: `r1`–`rj`$_{\tt max}$ = $(C(j), j = 1, j_{\rm max})$(cm^3 s^{-1} K$^{3/2}$); `rj`$_{\tt max+1}$–`rj`$_{\tt 2*max}$ = $(T(j), j = 1, j_{\rm max})$ (K); `i1` = Z; `i2` = $N - 1$; `i3` = M; `i4` = W, `i5` = ion_N.

49

Partial photoionization cross section of i_N-th level of the N-electron ion leaving the $(N - 1)$-electron ion in the k_{N-1}-th level: `r1`–`rj`$_{\tt 2*max}$ = $(E(j), \sigma(E(j)), j = 1, j_{\rm max})$ (Energy in Ryd relative to $E(\infty)$, cross section in Mb); `i1` = n; `i2` = L; `i3` = $2J$; `i4` = Z; `i5` = k_{N-1}; `i6` = ion_{N-1}; `i7` = i_N; `i8` = ion_N.

50

Line $(k - i)$ radiation rates of N-electron ion: `r1` = λ (Å); `r2` = $gf(i, k)$; `r3` = $A(k, i)$ (s^{-1}); `i1` = i (lower level); `i2` = k (upper level); `i3` = Z; `i4` = ion_N.

51

Electron-impact effective collision strength for the $k - i$ transition of N-electron ion (CHIANTI fit [12, 17]): `r1` = ΔE (Ryd); `r2` = C; `r3`–`r7` = $(\Upsilon_{\rm red}(j), j = 1, 5)$ (reduced effective collision strength); `i1` = it (transition type); `i2` = i (lower level); `i3` = k (upper level); `i4` = Z; `i5` = ion_N.

53

TOPbase partial photoionization cross section (resonance averaged) of i_N-th level of the N-electron ion leaving the $(N - 1)$-electron ion in the k_{N-1}-th level: `r1`–`rj`$_{\tt 2*max}$ = $(E(j), \sigma(E(j)), j = 1, j_{\rm max})$ (Energy in Ryd relative to $E(\infty)$, cross section in Mb); `i1` = n; `i2` = L; `i3` = $2J$; `i4` = Z; `i5` = k_{N-1}; `i6` = ion_{N-1}; `i7` = i_N; `i8` = ion_N.

54

Radiative transition probability A_{ki} for the $k - i$ transition of N-electron ion computed by quantum defect theory (or hydrogenic): `r1` = $0.0E + 0$; `i1` = i (lower level); `i2` = k (upper level); `i3` = Z; `i5` = ion_N.

56

Electron-impact effective collision strengths for the $k - i$ transition of N-electron ion: `r1`–`rj`$_{\texttt{max}}$ = $(\log T_e(j), j = 1, j_{\max})$ (K); `rj`$_{(\texttt{max+1})}$–`rj`$_{(\texttt{2*max})}$ = $(\Upsilon(T_e(j)), j = 1, j_{\max})$ (effective collision strength); `i1` = i (lower level); `i2` = k (upper level); `i3` = Z; `i5` = ion_N.

57

Effective ion charge for i-th level of N-electron ion: `r1` = $Z_{\rm eff}$; `i1` = n; `i2` = L; `i3` = $2J$; `i4` = Z; `i5` = i; `i6` = ion_N

59

Partial photoionization cross section of i_N-th level of the N-electron ion leaving the $(N - 1)$-electron ion in the k_{N-1}-th level [51]: `r1` = $E(th)$ (eV); `r2` = $E(0)$ (eV); `r3` = $\sigma(0)$ (Mb); `r4` = $y(a)$; `r5` = P; `r6` = $y(w)$; `i1` = N; `i2` = n (shell principal quantum number); `i3` = l (orbital quantum number of the subshell); `i4` = k_{N-1}; `i5` = ion_{N-1}; `i6` = i_N; `i7` = ion_N; `s1` = shell-ion identifier.

60

Analytic fits for effective collision strengths in H-like ions [14]: `r1`–`rj`$_{\texttt{max}}$ = coefficients; `i1` = i (lower level); `i2` = k (upper level); `i3` = 1; `i8` = ion_N; `s1` = Transition.

62

Analytic fits for effective collision strengths in H-like ions [14]: `r1`–`rj`$_{\texttt{max}}$ = coefficients; `i1` = i (lower level); `i2` = k (upper level); `i3` = 1; `i8` = ion_N; `s1` = Transition.

63

Collisional transition probability C_{ik} for N-electron ion computed by quantum defect theory (or hydrogenic): `i1` = 1; `i2` = i (lower level); `i3` = k (upper level); `i4` = Z; `i5` = ion_N.

66

Fits to fine-structure collision strengths for He-like ions [32]: `r1`–`rj`$_{\texttt{max}}$ = coefficients; `i1` = i (lower level); `i2` = k (upper level); `i3` = Z; `i4` = ion_N.

67

Analytic fits for effective collision strengths in He-like ions [33]: `r1`–`rj`$_{\texttt{max}}$ = coefficients; `i1` = i (lower level); `i2` = k (upper level); `i3` = Z; `i4` = ion_N.

68

Analytic fits for effective collision strengths in He-like ions [53]: `r1`–`rj`$_{\texttt{max}}$ = coefficients; `i1` = i (lower level); `i2` = k (upper level); `i3` = Z; `i8` = ion_N.

69

Fits to LS collision strengths for He-like ions [32]: `r1`–`rj`$_{\texttt{max}}$ = coefficients; `i1` = i (lower level); `i2` = k (upper level); `i3` = Z; `i8` = ion_N.

70

Coefficients for recombination and photoionization cross sections of superlevels: `r1`–`rj`$_{\texttt{nd}}$ = $(n_e(j), j = 1, j_{nd})$; `rj`$_{\texttt{nd+1}}$–`rj`$_{\texttt{nd+nt}}$ = $(T_e(j), j = 1, j_{nt})$; `rj`$_{\texttt{nd+nt+1}}$–`rj`$_{\texttt{nd+nt+nt*nd}}$ = $((\log \alpha(j, j'), j' = 1, j'_{nd}), j = 1, j_{nt})$; `rj`$_{\texttt{nd+nt+nt*nd+1}}$ − `rj`$_{\texttt{nd+nt+nt*nd+2*nx}}$ = $(E(j), \sigma(j), j = 1, j_{nx})$; `i1` = nd; `i2` = nt; `i3` = nx; `i4` = n; `i5` = L; `i6` = $2S + 1$; `i7` = Z; `i8` = k_{N-1}; `i9` = ion_{N-1}; `i10` = i_N; `i11` = ion_N.

71

Radiative transition rates from superlevels to spectroscopic levels: `r1`–`rj`$_{\texttt{nd}}$ = $(n_e(j), j = 1, j_{nd})$; `rj`$_{\texttt{nd+1}}$–`rj`$_{\texttt{nd+nt}}$ = $(T_e(j), j = 1, j_{nt})$; `rj`$_{\texttt{nd+nt+1}}$–`rj`$_{\texttt{nd+nt+nt*nd}}$ = $((A(j, j'), j' = 1, j'_{nd}), j = 1, j_{nt})$; `r`$_{\texttt{nd+nt+nt*nd+1}}$ = λ (Å); `i1` = nd; `i2` = nt; `i3` = i (lower level); `i4` = k (upper level); `i5` = Z; `i6` = ion_N.

72

Autoionization rates for satellite levels: `r1` = $A_a(k, i)$ (s^{-1}); `r2` = $E(k)$ (eV above ionization limit); `r3` = $(2J + 1)$;

`i1` = $(2S + 1)$; `i2` = L; `i3` = k (level); `i4` = i (continuum level); `i5` = Z; `i6` = ion_N; `s1` = level configuration.

73

Fit to effective collision strengths for satellite levels of He-like ions [47]: `r1`–$\mathtt{rj}_7$ = fit coefficients; `i1` = i (lower level); `i2` = j (upper level); `i3` = Z; `i4` = ion_N.

74

Delta functions to add to photoionization cross sections to match ADF DR rates: `r1` = $E(\infty)$ (eV); `r1`–$\mathtt{rj}_\mathtt{m}$ = $(E(j), j = 1, j_m)$ (eV); $\mathtt{rj}_\mathtt{m+1}$–$\mathtt{rj}_\mathtt{2m}$ = $(f(j), j = 1, j_m)$ (cm^2); `i1` = n; `i2` = L; `i3` = $2S + 1$; `i4` = Z; `i5` = k_{N-1}; `i6` = ion_{N-1}; `i7` = i_N; `i8` = ion_N

75

Autoionization rates for Fe XXIV satellites [8]: `r1` = $A_a(k, i)$ (s^{-1}); `r2` = $E(k)$ (eV above ionization limit); `i1` = ion_N, `i2` = k_N; `i3` = ion_{N-1}; `i4` = i_{N-1}; `i5` = ion_N.

76

Two-photon radiation rate for $(k - i)$ transition of N-electron ion: `r1` = $A(k, i)$ (s^{-1}); `i1` = i (lower level); `i2` = k (upper level); `i3` = 1; `i4` = ion_N; `s1` = transition identifier.

77

Collision transition rates from superlevels to spectroscopic levels: `r1`–$\mathtt{rj}_\mathtt{nd}$ = $(n_e(j), j = 1, j_{nd})$; $\mathtt{rj}_\mathtt{nd+1}$–$\mathtt{rj}_\mathtt{nd+nt}$ = $(T_e(j), j = 1, j_{nt})$; $\mathtt{rj}_\mathtt{nd+nt+1}$–$\mathtt{rj}_\mathtt{nd+nt+nt*nd}$ = $((C(j, j'), j' = 1, j'_{nd}), j = 1, j_{nt})$ (s^{-1}); $\mathtt{rj}_\mathtt{nd+nt+nt*nd+1}$ = λ (Å); `i1` = nd; `i2` = nt; `i3` = i (lower level); `i4` = k (upper level); `i5` = Z; `i6` = ion_N.

81

Collision strengths for Fe XIX [10]: `r1` = $\Upsilon(k, i)$; `i1` = i (lower level); `i2` = k (upper level); `i3` = Z; `i4` = ion_N.

82

Decay rates for Fe UTA [24]: `r1` = λ (Å); `r2` = $E(k)$ (eV); `r3` = $gf(i, k)$; `r4` = $A_r(k, i)$ (s^{-1}); `r5` = $A_a(k, i)$ (s^{-1}); `i1` = i (lower level); `i2` = k (upper level); `i4` = ion_N.

83

Level data for Fe UTA [24]: `r1` = $E(i)$ (eV); `r2` = $(2J + 1)$; `r3` = 0.0; `r4` = 0.0; `i1` = 1; `i5` = i (level); `i6` = ion_N; `s1` = level configuration assignment.

85

Photoionization cross sections for Fe ions obtained by summation of resonances near the K edge [44]: `r1` = Z_{eff}; `r2` = E_{th} (Ryd); `r3` = f; `r4` = γ; `r5` = scaling factor; `i1` = n; `i2` = L; `i3` = $2J$; `i4` = Z; `i5` = k_{N-1}; `i6` = ion_{N-1}; `i7` = i_N; `i8` = ion_N.

86

Auger and radiative widths of k_N-th K-vacancy level: `r1` = $E(k_N)$ (eV, relative to $E(\infty)$); `r2` = $A_a(k_N)$ (s^{-1}); `r3` = $A_a(k_N, i_{N-1})$ (s^{-1}); `r4` = $A_r(k_N)$ (s^{-1}); `i1` = i_{N-1}; `i2` = k_N; `i3` = Z; `i4` = ion_{N-1}; `i5` = ion_N.

88

Photoionization cross section damped excess of i_N-th level of the N-electron ion leaving the $(N - 1)$-electron ion in superlevel_[K] k_{N-1}: `r1`–$\mathtt{rj}_\mathtt{max}$ = $(E(j), \sigma(E(j)), j = 1, j_{\mathrm{max}})$ (Energy in Ryd relative to $E(\infty)$, cross section in Mb); `i1` = n; `i2` = L; `i3` = $2J$; `i4` = Z; `i5` = k_{N-1}; `i6` = i_N; `i7` = ion_N.

91

APED line $(k - i)$ radiation rates [20]: `r1` = λ (Å); `r2` = 0.0; `r3` = $A(k, i)$ (s^{-1}); `i1` = i (lower level); `i2` = k (upper level); `i3` = Z; `i4` = ion_N.

92

APED collision strengths [20]: `r1`–$\mathtt{rj}_\mathtt{max}$ = $(T_e(j), j = 1, j_{max})$ (K); $\mathtt{rj}_\mathtt{max+1}$–$\mathtt{rj}_\mathtt{2*max}$ = $(\Upsilon(j), j = 1, j_{max})$; `i1` = 1; `i2` = i (lower level); `i3` = k (upper level); `i4` = Z; `i5` = ion_N.

95

Collisional ionization rates for N-electron ion [11]: `r1` = $E(th)$ (eV); `r2` = T_0 (K); `r3`–$\mathtt{rj}_\mathtt{max+2}$ = $(\rho(j), j = 1, j_{max})$ (effective collision strength); `i1` = i (level); `i5` = ion_N.

98

Electron-impact effective collision strengths for the $k - i$ transition of the N-electron ion (CHIANTI fit citep{1992:Burgess,1997:Dere}): `r1` $= \Delta E$ (Ryd); `r2` $= C$; `r3`–$\mathtt{rj}_{\mathtt{max+2}}$ $= (\Upsilon_{\rm red}(j), j = 1, max)$ (reduced effective collision strength); `i1` $= it$ (transition type); `i2` $= i$; `i3` $= k$; `i4` $= ion_N$.

99

Coefficients for recombination and photoionization cross sections of superlevels: `r1`–$\mathtt{rj}_{\mathtt{nd}}$ $= (n_e(j), j = 1, j_{nd})$; $\mathtt{rj}_{\mathtt{nd+1}}$–$\mathtt{rj}_{\mathtt{nd+nt}}$ $= (T_e(j), j = 1, j_{nt})$; $\mathtt{rj}_{\mathtt{nd+nt+1}}$–$\mathtt{rj}_{\mathtt{nd+nt+nt*nd}}$ $= ((\alpha(j, j'), j' = 1, j'_{nd}), j = 1, j_{nt})$; $\mathtt{rj}_{\mathtt{nd+nt+nt*nd+1}} - \mathtt{rj}_{\mathtt{nd+nt+nt*nd+2*nx}}$ $= (E(j), \sigma(j), j = 1, j_{nx})$; `i1` $= nd$; `i2` $= nt$; `i3` $= nx$; `i4` $= n$; `i5` $= L$; `i6` $= 2S + 1$; `i7` $= Z$; `i8` $= k_{N-1}$; `i9` $= ion_{N-1}$; `i10` $= i_N$; `i11` $= ion_N$.

CHAPTER

THIRTEEN

UTILITY PROGRAMS

The program which translates the ascii database file into the binary fits format used by XSTAR is called bintran.f, and is included with the XSTAR source distribution. Compilation of this program is straightforward, although it requires links to the cfitsio libraries. Execution simply requires the redirection of the input.

CHAPTER

FOURTEEN

LEVEL LABELS

New in version 2.21bh is the replacement of all level strings by a uniform system developed for the the uadb database. The following is reproduced from the uadb manual and describes the labeling system.

While level strings from any coupling scheme can be stored and retrieved from uaDB, currently it only supports searching for *LS*-coupled level strings. In order to guarantee uniqueness, level strings entered into the database must conform to the rules outlined in this appendix.

All states must have a configuration. Term-averaged or level-resolved states must also include a term string and level-resolved states must specify J. The rules for each part follow.

14.1 Configuration strings

Configurations are stored in the database using an unambiguous notation which should be familiar to most users. A configuration consists of a space-delimited list of sub-shells in standard order each having the form, *nlm*, where *nl* is the sub-shell (standard order: 1s, 2s, 2p, 3s, . . .) and *m* is the occupation number. Note that the shorthand notation of omitting *m* when unity is not used, e.g. 2s1 not 2s. Configuration strings obey the rules:

- all closed sub-shells starting with 1s and ending just prior to the first open (or last) sub-shell are not part of the configuration string,
- the first open sub-shell is always displayed even if it is empty ($m = 0$), and
- all empty sub-shells beyond the first open sub-shell are not displayed.

Some examples:

- $1s^2\,2s^2\,2p^3$ becomes 2p3,
- $1s^2\,2s^1\,2p^4$ becomes 2s1 2p4
- $1s^2\,2s^0\,2p^5$ becomes 2s0 2p5, and
- $1s^1\,2s^2\,2p^4$ becomes 1s1 2s2 2p4.

Using a list of occupation numbers as the configuration label was considered and ultimately rejected due to the impracticality of storing Rydberg levels. Consider the configuration, 1s 200p; whereas only 13 characters are needed to store this configuration in the form described above, nearly 40,000 characters are required if using a list of occupation numbers.

To get the number of electrons of a configuration takes two steps; first you need to calculate the number of electrons in the core and then add up the occupation numbers of the visible sub-shells. To get the number of electrons in the core, n_{core}, take the principal quantum number, n, and the orbital angular momentum, l of the first *open* sub-shell and apply the following expression:

$$n_{core} = \frac{1}{3}n(n-1)(2n-1) + 2l^2.$$

For a configuration of 4p5 5s2 5p1 we have $n = 4$ and $l = 1$. The above expression yields $n_{core} = 30$ and the total occupation of the visible sub-shells is 8 so this configuration has 38 electrons.

CHAPTER

FIFTEEN

TERM STRINGS

The format for term should be familiar to most users. It starts with an integer representing $2S + 1$ followed by the spectroscopic letter representing the total orbital angular momentum, L. An example is 2P where $S = 1/2$ and $L = 1$.

Level strings

To specify the total angular momentum, J, of a level-resolved state, you append the term string defined above with an underscore and the J value. If J is a half-integer then you must use fractional notation. Examples of the term and level strings include: $2P_1/2$ and $1S_0$.

CHAPTER SIXTEEN

THEORY OF OPERATIONS

16.1 XSTAR

16.1.1 Introduction

Although XSTAR is designed with the goal of maximizing the flexibility of parameter values and assumptions available to the user, there are likely to arise situations in which the standard set of input options are not sufficient. Under these circumstances the user may want to attempt to modify the source code in order to effect a particular set of assumptions, geometry, etc. Problems for which customization is more likely to be necessary include models with additional heating or cooling processes (for example adiabatic expansion cooling or cosmic ray heating) or different assumptions about line escape probability (e.g. Sobolev escape probability in a medium with a velocity gradient). If so, the internal operation of the code must be confronted. This appendix presents a summary of the code operation in order to aid in code customization.

16.1.2 Programming Philosophy

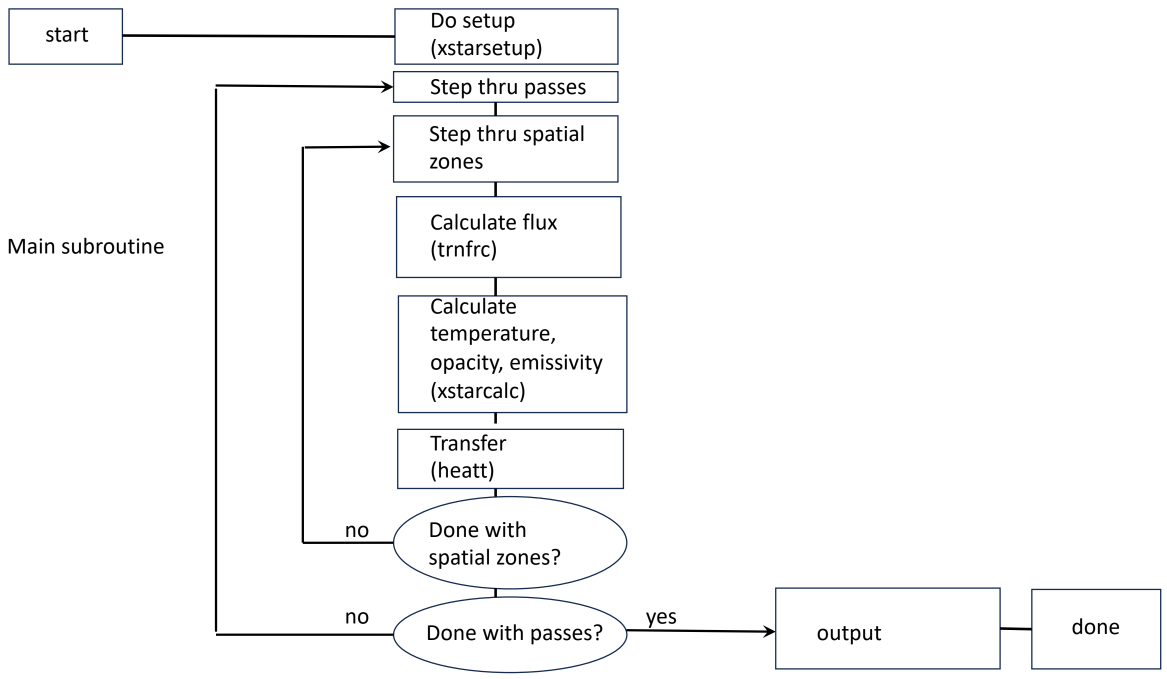
XSTAR has been written and developed over many years, and much has changed during that time. Changes include advances in the fortran language, changes in the speed and memory capacity of the available computers, new insights into the flexibility required of XSTAR, and insight into which physical processes are likely to have the greatest effect on the model results.

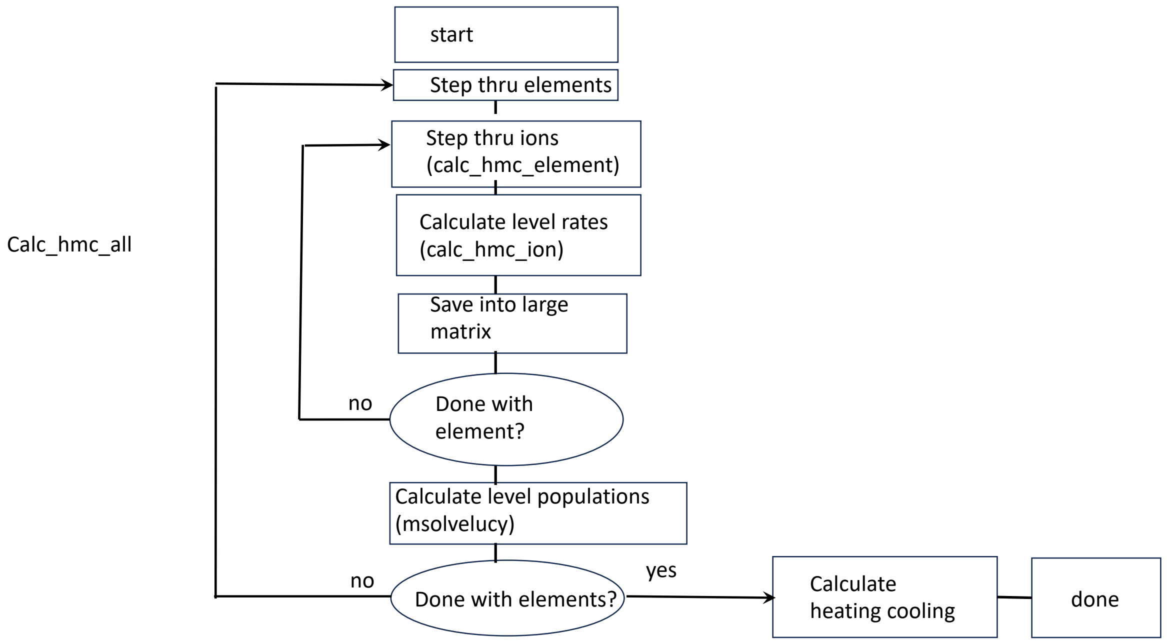
XSTAR was written in standard fortran 77, and much of the code retains the programming style associated with the older versions of the fortran language. Hard experience with moving the code from one type of machine to another has led to an attempt to avoid any machine dependent extensions to the fortran language, any word length dependent numerical constructs or any use of extended precision arithmetic (with one or two exceptions).

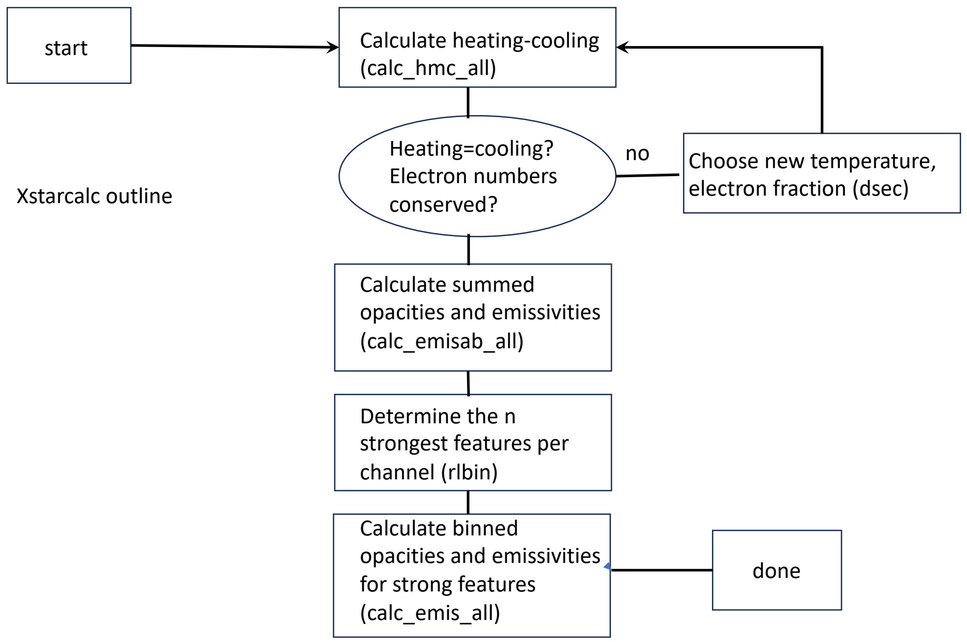
The code is structured in an attempt to be modular, and to separate the calculation of the atomic rates from the calculation of level populations, ion fractions, temperature, etc. The goal is to make it relatively easy to add or change atomic data, requiring modification of just one subroutine if a new process is added or a fitting formula is changed. The data itself is all read in from an external file, so that it can be changed without any modification to the code itself if the existing fitting formulas are unchanged.

In this section we present an overview of the code structure in the form of flowcharts. Detailed descriptions of xstar routines and data structures will be found in the accompanying document containing the doxygen documentation. We also provide flowcharts and descriptions for xstar2xspec and associated tools.

16.2 Xstar Flow Charts




start
Step thru elements
Step thru ions (calc_hmc_element)
Calc_hmc_all
Calculate level rates (calc_hmc_ion)
Save into large matrix
no
Done with element?
Calculate level populations (msolvelucy)
no
Done with elements?
yes
Calculate heating cooling
done


start
Calculate heating-cooling (calc_hmc_all)
Heating=cooling? Electron numbers conserved?
no
Choose new temperature, electron fraction (dsec)
Xstarcalc outline
Calculate summed opacities and emissivities (calc_emisab_all)
Determine the n strongest features per channel (rlbin)
Calculate binned opacities and emissivities for strong features (calc_emis_all)
done

16.3 XSTAR2XSPEC

XSTAR2XSPEC consists of three major components (in addition to XSTAR itself).

xstar2xspec
xstar2xspec is a Perl script which manages the overall program flow of XSTAR2XSPEC.

xstinitable
xstinitable is an FTOOL used in the initialization phase of XSTAR2XSPEC. It builds the FITS PARAMETER table from the input data (xstinitable.fits) and a text file (xstinitable.lis) which is basically of list of all the calls to XSTAR to generate the required spectra.

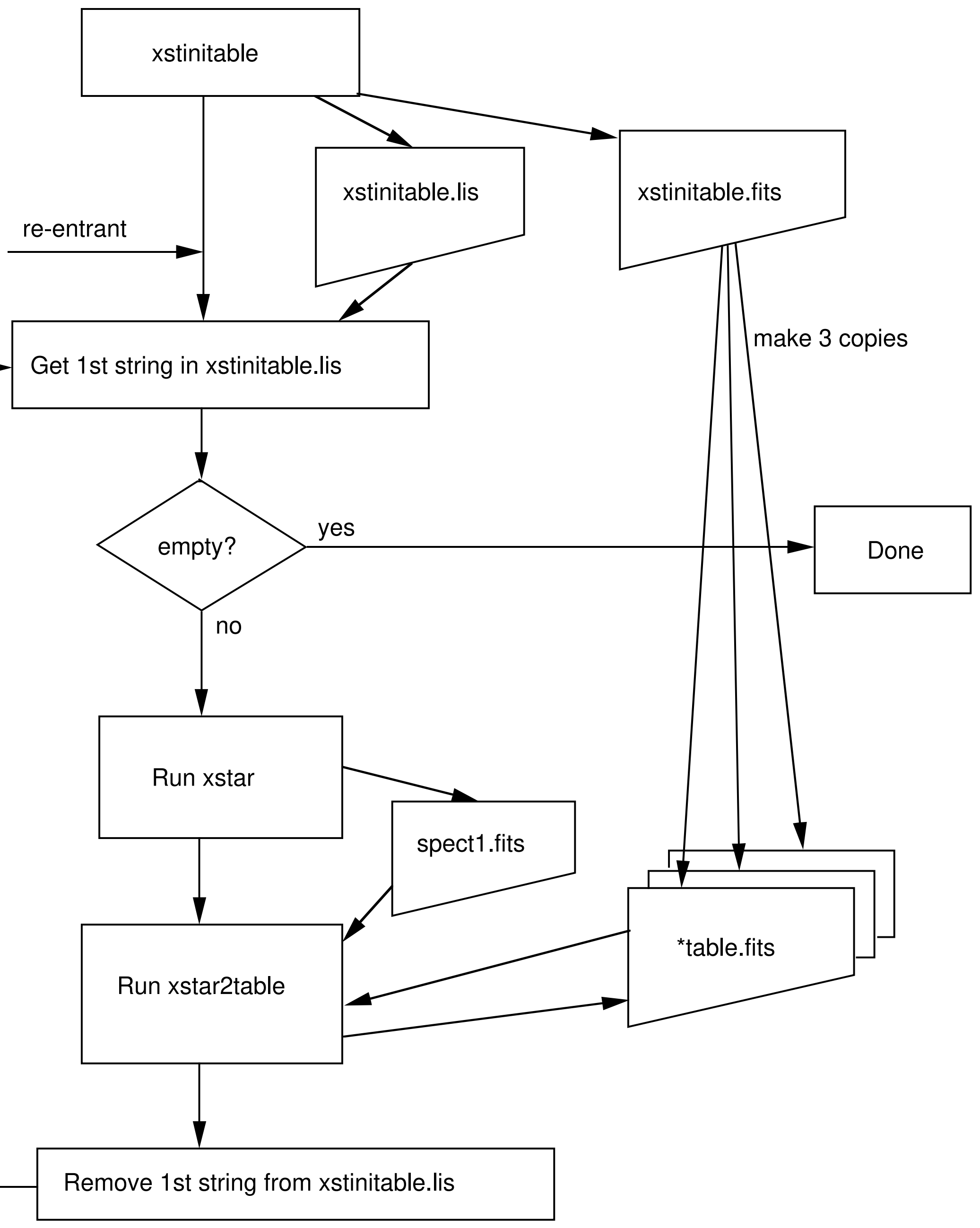
xstar2table
xstar2table is an FTOOL called after xstar in each iteration of the xstar2xspec loop.

16.3.1 xstar2xspec (script)

This is a Perl script.

One current limitation is that the file tagging in the {tt -save} option file names are limited to 9999 calls of XSTAR.

XSTAR2XSPEC



16.3.2 xstinitable

This FTOOL is written in C. It generates an initial FITS file (xstinitable.fits) with the appropriate PRIMARY and PARAMETERS extension from the atables and mtable file are build. It also generates a text file (xstinitable.lis) which contains a complete XSTAR calling command on each line. This file is processed by the XSTAR2XSPEC Perl script

Examples of changes in XSTAR that would require changes in this FTOOL (just meant as a sample, not necessarily an exhaustive list).

1. Changing the number of physical parameters in XSTAR.
2. Changing or altering the control parameters in XSTAR.

16.3.3 xstar2table

This FTOOL is also written in C. Full use is made of dynamic memory allocation. If the number of spectral channels is changed in XSTAR, this program should adapt appropriately, automatically.

Examples of changes in XSTAR that would require changes in this FTOOL (just meant as a sample, not necessarily an exhaustive list).

1. Changing the number of physical parameters in XSTAR.
2. At present, this program is written assuming that the XSTAR runs are performed sequentially and in a particular order. In fact, it checks the sequence by comparing the LASTSPEC keyword to the LOOPCONTROL variable. If, at some time in the future, the XPI interface is modified in such a way that it becomes possible to simultaneously submit multiple XSTAR runs on multiple processors, xstar2table must be modified to ignore this comparison.

CHAPTER SEVENTEEN

REVISIONS

17.1 Revision history of xstar

17.1.1 Version 2.1 (June 2000)

- Added 5 new input parameters vturbi, emult, critd, taumax, xeemin.
- Added artificial broadening of absorption lines due to turbulent velocity controlled by input parameter vturbi. If vturbi is less than the local thermal ion speed, then thermal Doppler broadening is used.
- Changed input spectrum in default parameter file to pow. Added error check for invalid input spectrum.
- Fixed error which displayed incorrect value of the constant pressure switch in output files.
- Several minor errors have been corrected in the some of the calculations of atomic rates: types 67, 51, and 70.
- Added printout of all line emissivities at each radial zone (in addition to the level populations) when the write_switch parameter is set to 1. These appear in the file named xout_detal2.fits.
- Modified the algorithm for calculating thermal equilibrium so that for temperatures less than 3030K the iteration procedure is disabled; any radius zone where the equilibrium solver finds a temperature less than this value will not calculate thermal equlibrium.
- Added 2 new output fits files to the standard output. xout_lines.fits contains the luminosities of the 100 strongest emission lines, and xout_cont.fits contains the continuum luminosities without the lines added (these two files can be combined by binning the lines suitably in order to make the contents of xout_spect1.fits).
- Added calculation and printout of LTE level populations to the quantities in xout_detail.fits.
- Improved treament of energy budget, and added printout of energy budget to xout_step.log.

Version 2.1a (December 2000)

- Fixed error in atomic rates affecting Fe XXI which caused recombination rates to be too large.
- Streamlined calculation of photoionization and recombination rate quadratures.

Version 2.1b (January 2001)

- Added printout of line and recombination cooling rates to printout in log file.

Version 2.1c (May 2001)

- Fixed error in database which resulted in too large emission in some iron K:math:*beta* fluorescence lines.

Version 2.1d (May 2001)

- Added feature which appends ion column densities in an additional extension to the xout_abund1.fits file.

Version 2.1e (June 2001)

- Extended the energy range up to 1 MeV and added relativisitic Compton heating and cooling.
- Fixed error in calculating threshold energies of some excited levels from type 53 data.
- Fixed error in subroutine which creates blackbody spectra which resulted in spuriously large fluxes at energies above 50 keV.

Version 2.1h (June 2002)

- Relativistic corrections to Compton heating and cooling have been added, using a procedure based on the work of Guilbert (1986).
- A new format for the storage of atomic data has been adopted, resulting in smaller files, and faster read times. This change should be transparent to the user, except for considerable speeding of data loading and program startup.
- A new algorithm for continuum transfer has been adopted, outward only transfer. This gives better energy conservation overall. A new column has been added in the log file output, labeled 'h-c', it tells the percent error in total energy conservation in the radiation field, i.e. total emitted - total absorbed.
- A new algorithm has been adopted for solving the statistical equilbrium, in place of LU decomposition. The algorithm involves an iterative solution to a simplified set of equations, and is described by Lucy (2001). This change should result in fast execution for models involving iron and other heavy elements, but will otherwise be transparent to the user.
- Various changes to atomic data, including increase in size of data file and addition of a new data type, for excitation of Fe XIX using data from Bhatia.
- Various minor inconsistencies and errors have been corrected, including spurious recombination emission to multiply excited levels.
- Standard output in the log file has been extended to include a list of the strongest absorption lines, and emission and absorption edges.
- Also, a new output file, xout_rrc1.fits, is created which contains a fits format list of the RRC strengths.

Version 2.1j (September 2003)

- New atomic data for iron K emission and absorption as described in Palmeri et al., 2004 A and A and references therein (see TK homepage for reprints).
- Added n=2-3 iron UTA absorption using data from FAC (Gu 2003).
- Fixed bug which limited length of spectrum file name to 8 characters.
- Fixed bug which allowed buffer containing ion fractions vs. xi to overflow when the number of spatial zones exceeded 1000. Now the limit on the number of spatial zones is 3000, and the code stops with a message when this is exceeded.
- Added accurate Voigt profile calculations for all lines in synthetic spectra.
- Fixed bug which limited length of spectrum file name to 8 characters.

Version 2.1k (May 2004)

- Added printouts of level opacitites, and level populations to final printout if the print switch is set to 2.
- Added a column to the printout of the file xout_detail.fits for the upp level index of each line.
- Repaired and streamlined the printing of the file xout_detail.fits.
- Added more informative statement when the code stops because the rate matrix overflows (ipmat too large).
- Added rate type 42: Auger decay
- Fixed arithmetic error which affected recombination rate calculation when kT $>>$ E$_{th}$
- Added new data type (85) for photoionization resonances below threshold, along with new subroutine to calculate cross subsection (PEXS.f).
- Streamlined the photoionization rate calculation (phint53)
- More accurate treatment of line damping, Voigt profiles.

Version 2.1kn3 (April 2005)

- Two bugs were found in version 2.1k in the implementation of the Voigt function when calculating line absorption and the calculation of line broadening. The Voigt function bug affected primarily lines with small damping parameters, and resulted in non-fatal numerical errors in the xstar output absorption spectrum (INFs). When xstar was called as part of xstar2xspec this resulted in fatal errors because the cfitsio routines which read the xstar output could not interpret the INFs. The line broadening bug resulted in too large absorption line depths when turbulent broadening was important. Neither of these bugs affected the temperature, ionization balance or emission spectrum. The bugs have been repaired in version 2.1kn3.
- Version 2.1kn3 also has an added feature, which is the addition of ion-by-ion heating and cooling rates as extensions to the output file xout_abund1.fits.
- Also added is the capability to set the value of niter to a negative number, which allows the solution of charge conservation without solving thermal equilibrium. As before, if niter=0 then neither charge transfer nor thermal equlibrium is calculated.

Version 2.1kn4 (April 2005)

- Fixed an error which causes the wrong inital radius to be calculated when the constant pressure is option is chosen. Also changed the units label on the ionization parameter to remove inconsistency with constant pressure case.
- Minor changes in the radiation transfer algorithm.

Version 2.1kn5 (March (?) 2006)

- Fixed bug which affected high ionization models which included nickel. This caused segmentation faults, and was caused by an incorrect data type flag in the atomic data for He-like Ni.
- Changed step size algorithm to prevent stepping beyond the column density specified in the input. This will not be accurate for constant pressure clouds in which the temperature is changing rapidly.

Version 2.1kn6 (June 2006)

- Added the effect of photoionization and heating by line photons generated elsewhere in the cloud. These are photons which have already escaped the local region close to the point of emission.
- Changed the step size computation algorithm in order to account for the process of emission. That is, the step size is now is based on the length scale for significant change of both absorption and emission.
- Fixed several errors in the atomic database, notably affecting N-like ions. These affect some of the density sensitive lines in low ionization models.
- Update to this manual, in the chapter in the Physics of xstar, describing in more detail the radiation transfer algorithm.

Version 2.1kn7 (March 2007)

- A bug has been found affecting the intensities of the He-like forbidden lines from C, N, and O at high densities.

Version 2.1kn7 (December 2007)

- The xstar database has been updated to take into account the iron M shell UTA data of Gu et al., 2006, 641, 1227. A revised database file for use with xstar21kn7 is available from the xstar website.

Version 2.1l

- represents an update to the atomic data which includes the iron and oxygen inner shell data which was presented in 2004 Ap. J. Supp. 155, 675, along with the line data for iron from Chianti 5 and features from version 2.1kn6. Recent updates include fixes to buge in the routines associated with xstar2xspec. These caused numerical problems on 64 bit machines, and also resulted in errors when large grids of models were run. Versions 2.1l, 2.1lnx, etc. have not yet been completely tested and so have not been put into the standard release.

Version 2.1ln3

- A bug has been found in version 2.1ln2 which affects the results in the paper 2004 Ap.J.Supp. 155 675. This is a bookkeeping error resulting in multiple-counting of the iron L shell cross subsection when calculating the cross subsections for the 'third row' ions, Fe I-VIII. This makes a quantitative change to the results in figures 4a and 13a. That is, it affects opacity due to iron above approximately 1 keV, only for low ionization models ($\log(\xi) < 0$). These errors have been repaired in the version 2.1ln3, and repaired versions of the figures can be found on the xstar website.

Version 2.1ln4

- Fixes to bugs in the routines associated with xstar2xspec. These caused numerical problems on 64 bit machines, and also resulted in errors when large grids of models were run (November 2007).

Version 2.1ln5

- Incorporates the revised iron UTA data from Gu et al., 2006 (December 2007).

Version 2.1ln6

- An update which implements strong typing to the fortran code. Function and results should be the same as previous versions.

Version 2.1ln7

- Contains the dielectronic recombination rates for the ions of iron calculated by Badnell 2006 Ap. J. Lett. 651, 73. These result in a qualitative change to the ionization balance of iron for log(xi)leq1.

Version 2.1kn9/v2.1ln9 (November 2008)

- Treatment of line profiles both in absorption and emission has been redone. Previously the profile function for each line was evaluated at the boundary of each energy bin. Now each energy bin contains the integrated line luminosity (or optical depth) within that bin. This will have a significant effect for lines which are narrower than the default bin spacing, which is approximately 350 km/s. This affects outputs in the binned spectrum in xout_spect1.fits.

Version v2.1ln10 (May 2009)

- An error was found in the book keeping for some inner shell photoionization cross subsections, resulting in double-counting in the opacity for some inner shell bound-free transions. These affect primarily low ionization models, and do not affect Fe or O. This has been fixed in this version of the code. This does not affect version 2.1kn9 and previous.

Version v2.1ln11 (May 2009)

- An error was found in the zeroing of one of the arrays used for zeroing an important matrix which is used in calculating level populations. This led to spurious emission in some fluorescence lines from low-medium ionization species of elements other than O or Fe, all occuring in low-medium ionization models.

17.1.2 Version v2.2.0 (November 2009)

- This version includes the following added features:
  - (i) Inclusion of all elements up to Z=30. The atomic data for the energy level structure of ions with 3 or more electrons for many of these are scaled hydrogenic and so the associated line emission must be treated with caution.
  - (ii) Inclusion of two new input parameters: the radius exponent (radexp) and the number of continuum energy bind (ncn2). These are described in the chapter on input to xstar.
  - (iii) Inclusion of the radiation scattered in resonance lines as a column in the output fits file xout_spect1.fits. This is provided in the same units of specific luminosity as the other columns.
  - (iv) Use of a new algorithm for the multilevel calculation which is considerably faster and requires less storage. Hence smaller values of critf (even 0) can be accomodated for many problems.
  - (v) The input parameter critf now refers to the fractional ion abundance (i.e. relative to the parent element) rather than the absolute (i.e. relative to H) ion abundance.
  - (vi) Minor changes have been made to some of the output formats in the ascii file xout_step.lis.
  - (vii) The atomic data for dielectronic recombination has been changed to incorporate the results from Badnell and coworkers (http://amdpp.phys.strath.ac.uk/tamoc/DATA/RR/) in place of the rates from cite{Aldrovandi1973} and cite{Arnaud1992}. This has quantitative effects on many of the results from xstar. Notable is the effect on the ionization balance of iron for ionization parameters in the range 0 $leq {rm log}(xi) leq$ 2, where the m-shell ions dominate, and where the new rates are greater than the previous ones by large factors.

Version v2.2.1 (April 2010)

- Fixes to bugs which affected the length of the name of the spectrum file used when the ‘file’ input option is specified, and which affected the operation of multi-pass runs.

Version v2.2.1bc (September 2010)

- Fix zeroing error of variable xilevt in func.
- Modifications which allow the use of data files from version 2.0 and 2.1xx.
- Updates to atomic database to include R-matrix calculations for nitrogen.
- More accurate evaluation of voigt function (greater wavelength range)for line absorption.
- Fixes to invert.f to allow iterative runs.
- Include lte level population in fits output files.
- Force evaluation of photoionization integrals even when heating sum stops changing. Include smaller Boltzmann factors in milne sum.
- Include printout of local blackbody in opacity printout (lprint=2).

- Fixed sign error in spline routine used by Burgess Tully routine.
- More accurate evaluation of Planck function.

Version v2.2.1bg (May 2011)

- Changes committed to reflect new atomic data from Mike Witthoeft for K shells of Ne, Mg, Si, S, Ca, Ar
- Modifications to codeto allow better comparisons with xstar v1.
- Changes to database to include DR from metastable levels of 3rd row iron ions.
- Fixes errors in data file for indexing of k vacancy levels in iron l shell ions.

Version v2.2.1bh (September 2011)

- Change so that explicit use of real*8 variables throughout
- Change to access of database which avoids passing large numbers of variables to reading routine. Pointers are passed instead.

Version v2.2.1bk (January 2012)

- Fix to error introduced in 221bh which allows code to modify atomic data data during calculation of data type 72.
- Fix to error in msolvelucy involving rate equation solution
- Fix to error in linopac/stpcut which led to spurious features in emission profiles during Voigt profile calculation

Version v2.2.1bn (July 2012)

- Include new Al and Ni data
- New storage for matrix of collisional-radiative rates allowing essentially no limit on number of ions which can be solved at one time.
- Add columns to xout_detal2.fits to include rrc emissivities. Put out rrc luminosity derivitives rather than raw emissivities.

Version v2.2.1bn7 (August 2012)

- changed crit in mslovelucy to 1.e-2. resurrected fac.ne.1 in heatt

Version v2.2.1bn8 (August 2012)

- added prints in init

Version v2.2.1bn10 (August 2012)

- fixed possible double counting error in pesc in func2 added ferland print, pprint(27) resurrects continuum escape probabilities

Version v2.2.1bn11 (November 2012)

- increases crith from 5.e-3 to 1.e-2.
- adds output to xout_detal3.fits of rrcs during step-by-step output
- fixes length of strings kdesc2 in ucalc to avoid compilation warnings
- brings back the chisq routine which checks statistical equilibrium
- increases the number of spatial zones which can be saved for for printout to 3999
- adopts random access io for xout_tmp files.
- resurrects the dalgarno and butler charge excchange (data type 21)
- Version v2.2.1bn13 (November 2012)
- same as bn11 but with ncn=:math:*10^6* and widths added in quadrature for shuinai

Version v2.2.1bn14 (April 2013)

- same as bn13 but allowing density to exceed 1.e+18. This represents some serious approximations, physically: there are various quantities which are tabulated vs. density, and those grids (still) end at 10^18. For example, the recombination into high n levels for each ion is lumped into one rate, for levels beyond those which are treated spectroscopically. Computationally, the recombination rate into these levels can be written $n_{\mathrm{e}} \times \alpha_{\mathrm{highn}}(n_{\mathrm{e}}, T)$. So the n_{e} which is the argument of the α_{highn} function still can't go beyond 10^18. But the n_{e} multiplier can be arbitrarily large. Similar comments apply to some other types of rates.

Version v2.2.1bn15 (July 2013)

- Changes notation: character*n –> character(n)
- Fixes error which caused spurious features in absorption line profiles: in routine stpcut: dpcrit=1.e-2 –> dpcrit=1.e-6
- Inlcudes new data on N VI level structure and collisional excitation.

Version v2.2.1bn16 (September 2013)

- changed expression for Boltzmann factor in calt57
- changed starting guess for electron fraction in dsec. Works better for low ionization cases.
- changed from use of electron fraction error to electron fraction error relative to electron fraction as quantity to be solved for in dsec. Works better for low ionization cases.
- Increased precision of expo function
- Add lte level populations to ucalc call. Calculate lte level populations before calls in func1, func2
- Allow for 200 iterations in msolvelucy instead of 100
- Calculate and print lbol in ispcg2

- Set pescv=0.5 to make rrcs optically thin always
- Changes to rates in phint53 to make rates obey lte in the limit
- Print photon occupation number in continuum printout in pprint.
- Fixed buffer size in readtbl which caused overflow and serious error during read of atomic data

Version v2.2.1bn17 (December 2013)

- Make calculation of photoionization related quantities modal, using lfpi: 1: total pi only; 2: pi + rec rates only; 3: opacities and emissivities
- Also make h-c calculation use total rates add special funcsyn, func3p and heatf for calculations of spectral quantities.
- Add profile calculation (linopac) inside of ucalc when lfpi=3; take out profile calculation from stpcut. This facitilitates calculation of contionuum photoexcitation (which is not yet included)
- Change i/o of step quantities; now includes populations, total emissivities and opacities, line emissivities,… also change savd, unsavd also change name of step quantities: xoN…M.fits where N=1,2,3,4 for various quantities and M=pass number.
- Add column, electron fraction, density as keywords in step files.
- Add comments in pprint, unify code to use the same statements when stepping thru each physical quantity: levels, lines, all data, etc.
- Move search for auger width to new rotuine deleafnd

Version v2.2.1bn18 (January 2014)

- New atomic data for K shell absorption by neutral and once- and twice- ionized stages of Ne and Mg from Gorczyca.
- Fix errors in the routine binemis which puts out binned emission lines. These errors led to spurious features in models with very high spectral resolution.
- Change to the value of the constant hc used in conversion from ev to A and back, to reflect more accurate values for constants. Old value was 12398.54, new value is 12398.41. Also change to Rydberg constant; old value was 13.598, new value is 13.605. Adoption of consistent value of proton thermal speed as 1.29e+6 cm/s at 10^4 K.
- Adoption of routine which calculates photoionization integrals (phint53) which uses interpolation and smoothing.
- Inclusion of code for calculating aped rates for collisional excitation (not yet fully implemented).
- Inclusion of Bryans rates for collisional ionization.
- Added feature which allows an array of densities to be read in. This is described in the chapter on inputs. It requires that the ‘radexp’ input variable be set to a number more negative than -100. Then ordered pairs of (radius, density) are read in from a file called ‘density.dat’. Reading continues until the end of the file is reached. The density and radius values override the values derived from the ordinary input parameters. But execution will stop if other ending criteria are satisfied, i.e. if the model column density exceeds the input value, or the electron fraction falls below the specified minimum. The code will stop with an error if the density.dat file does not exist, or if the radius values are not monotonically increasing.
- Another new feature allows reading in of table spectra in units of log10(F$_varepsilon$). This requires that the spectrum_units input parameter be set to 2.

Version v2.2.1bn19 (March 2014)

- Fix to bug which led to incorrect f value use for iron UTA lines.
- Fix errors to routine binemis and linopac associated with attempt to make routines faster: now, always use constant stepsize for internal calculation of line profile.
- change to true anders and grevesse abundances

Version v2.2.1bn20 (March 2014)

- Fix to a bug which caused the wrong damping value to be used in some cases. This occured for valence shell lines, for which the damping should be just due to the natural radiative lifetime, but for which inner shell Auger damped lines also exist for the same ions. In this case, the widths for the latter lines were incorrectly used instead of the former.

Version v2.2.1bn21 (May 2014)

- Fix to an error in implementation of Bryans collisional ionization rates.
- Fix to an error in inclusion of turbulence in implementation of iron M-shell UTA line absorption.

Version v2.2.1bn22 (September 2014)

- Photoionization integration routine now uses thresholds calculated on energy bin boundaries. This allows for better evaluation of the Milne integral, though it may affect photoionization rates in the case of very coars energy bins.
- Removed redundant subroutine phint53new.f
- Added code to print ion column densities as part of lpri=2 output
- Increased buffer size in subroutine fstepr2 which writes to xo01_detal2.fits such that table of lines is not artificially truncated.
- Add fine structure to He-like ion level and radiative decay data.

Version v2.2.1bn24 (July 2015)

- The quantities printed in the ascii file xout_step.log denoted ‘httot’ and ‘cltot’ now are the total heating and cooling respectively. In versions since 2.2.1bn19 they did not include Compton and bremsstrahlung. The criterion used to select lines when binning the spectrum used in xout_spect1.fits was changed in order to reduce execution time. Only lines with luminosities greater than 10^{-10} times the incident continuum luminosity are now included.

17.1.3 Version v2.3 (January 2016)

- Fixed error in charge transfer ionization of O I. Fixed error in compton heating-cooling which affected spectra with significant flux above 100 keV. Extended extrapolation of photoionization cross sections from 20 keV to 200 keV.

Version v2.31 (May 2016)

- An error in the treatment of continuum escape probablilities affecting two-sided models was fixed.
- Fixed an error in the N VI escited state statistical weight values which affected line opacities.

Version v2.33 (May 2016)

- An error in the calculation of ion column densities was fixed. This affected the values in the second extension to the xout_abund1.fits file, and the values printed in the xout_step.log file when the print switch is set to 1 or greater. No other quantities were affected. An error in the treatment of the N VI collisional excitation rates was also fixed.

Version v2.35 (August 2016)

- Update to type 66 (Kato and Nakazaki collision strength parameterization) to allow for more than 6 points.
- Fix to implementation of Bryans CI rates (data type 95) to include level-to-level CI.

Version v2.36 (October 2016)

- Added code to handle Chianti 2016 collisional excitation rates (data type 98) and also ad hoc treatment of inner shell collisional ionization (data type 97) from Patrick Palmerit fac calculations for Fe XXIV.
- Added new function upsiln used in this calculation.
- Fixed error in lower level statistical weight calculation for data type 95 (Bryans collisional ionization).
- Increased dimension of dummy array used in reading in atomic data in readtbl. This was filling and causing erroneous results for highest Z elements (Cu, Zn).
- Change to main xstar routine to prevent writing detailed step-by-step data unless write switch is set or unless npass>1.

Version v2.37 (October 2016)

- Added local version of getlun to handle the opening and closing of many logical unit numbers.

Version v2.38 (November 2016)

- New tests to prevent 2 photon decays from being printed as lines. Functional treatment of line list and calculation of 2 photon rates is unchanged.
- Added use of expo instead of exp in calculation of type 95 rates to prevent exponent misbehavior at low temperature.
- Fix to data for He-like ions. Mapping from ls to fine structure resulted in several errors in previous versions. In type 69 there was spurious scaling of the excitation energy along with the collision strength. Also an error in the forbidden line A value. Also an indexing error affecting the superlevel and therefore the recombination cascade.
- New feature for the write switch: if the value is -1 then no fits files are written. This speeds execution for problems where the ascii file provides sufficient information.

Version v2.38 (December 2016)

- Fix to logic error in routines calc_spline and prep_spline.
- Fix to error which omitted brems emission in spectrum calculation in previous versions, since 2.3.
- Increased size of dummy array in readtbl.
- Fix to logic and use of the function upsiln in ucalc at index 98

Version v2.39 (March 2017)

- Fix to the routine which does spline evaluations for upsilons used by chianti and atomdb (preparation for future use).
- Increase in temporary array size in routine readtbl which reads in atomic data from atdb.fits.
- Explicit inclusion of bremsstrahlung in emissivity calculation.

17.1.4 Version v2.41 (May 2017)

- Fix to type 70 calculation (recombination to pseudo-levels) for He-like ions which partially smooths the density-dependent behavior near density of 10^{10} cm^{-3}, by using quadratic interpolation rather than linear.
- Fix which passes real*4 variable to fitsio routine in fstepr4.
- Increase size of temporary array in routine reading atomic database.
- Update to treatment of dielectronic satellite emission in calt72
- Remove arrays called vsav, rates, and idrates which contained temporary saved rates in order to save memory space.
- Increase value of crit and crit2 in msolvelucy from 1.e-10 to 1.e-4.
- Add data type 96, safranova satellite emission for Fe XXIV
- Also changes to atomic database: update to Fe XXIV DR satellite emission using rates from Bautista et al. 2003; fix to the Fe XXV forbidden line (2 photon) decay; addition of direct excitation of Fe XXIV satellites using rates from atomdb; also direct excitation of Fe XXV lines using rates from atomdb.

Version v2.43 (Oct 2017)

- Minor changes to binemis to save memory
- Added tabulation of continuum-only opacity and spectrum. Reused variable previously used for scattering opacity. Scattering-only spectra are stored in variable zrems, indeces 4 and 5. New variable for continuum-only optical depth.

Version v2.44 (March 2018)

- Suppressed density dependence of type 70 recombination rates for hydrogen.

17.1.5 Version v2.51 (April 2018)

- Converted from fortran 77 to fortran 90.
- Adopted allocated temporary arrays in many routines to save memory.
- Changed atomic database from simple arrays to fortran 90 structures. No longer passed as arguments to most routines, now contained as derived data types in a module.
- Changed procedure for passing lte level populations in to fstepr2, rstepr2, the routines responsible for writing and reading rrcs.

Version v2.52 (June 2018)

- Many syntax changes to suppress compilation warnings.
- Added feature in which a value lpri=-1 suppresses almost all ascii output. This is designed to allow for large grid production without creation of unwieldy ascii files.

Version v2.53 (June 2018)

- Changed the criterion for including lines in the output to the binned spectrum to include more weak lines.

Version v2.54 (December 2018)

- A bug has been found which affects the output from xstar version 2.53, namely the file xout_spect1.fits. The outward emitted spectrum has the incident radiation field added into it. This has been repaired in version 2.54.

Version v2.54a (January 2020)

- The effects of radiative excitation by continuum pumping have now been included. See the manual section 9.D.2.
- The effects of Thomson scattering have now been included. See the manual section 9.F.3.

Version v2.56f (November 2020)

- New atomic data for elements F, Na, Al, P, K, Cl, Sc, T, V, Cr, Mn, Co, Cu, Zn based on calculations described in Mendoza et al., 2018 A&A…616A..62M and its antecedents. This results in a significant increase in the size of the atomic database and the associated data structures.
- Large scale switch from static memory to dynamically allocated memory in order to account for the growth of the data structures associated with the atomic database.
- Major restructuring and renaming of subroutines to better describe their functionality.

Version v2.57 (May 2021)

- Put calculation of fline back in.
- Avoid floating point errors in amcrs, ee1expo, huntf
- Added new quantity to printout in pprint(9)
- Repaired length incompatibilities and variable type conflicts in arguments to cfitsion routines.

Version v2.58a (July 2021)

- Trap pointers out of bounds in fstepxx, rstepxx
- Make floats output to fits tables e13.5 instead of e11.3

Version v2.58b (August 2021)

- Put back charge transfer which was erroneously removed in v2.56.
- Put back he-like fine structure in database which was erroneoudly removed in v2.57.

Version v2.58d (September 2021)

- Include a switch to take the absolute value of line wavelengths. In the database, lines with theoretical wavelengths are given negative wavelengths. These were not included in the synthetic spectrum calculation in previous versions. With the switch, called llinabs, set in the main xstar subroutine, this can be controlled. If llinabs=1 then the lines with negative wavelengths are included in the synthetic spectrum.

Version v2.58d (December 2021)

- An oscure bug was found which allowed an unitialized variable to be used in the function interpol_huntd. This has now been fixed.

17.2 Revision history of warmabs

17.2.1 Version 2.0

Version 2.02

- Inclusion of continua, both in emission and absorption.
- Equivalent widths are not calculated, and quantities analogous to the transition probability and oscillator strength are not output.
- The upper level, which may not be the ground level of the adjacent ion, is not identified.

Version 2.03

- Fix to error in normalization of voigt profile.

Version 2.04

- Rational and uniform level labels for all levels. These should now be unambiguous. A description is contained in the xstar manual.
- The interface with the commonprint common block has been updated.
- There is now a (string) variable whiche denotes whether a transition is a line or rrc/edge. Upper levels for rrc/edges are denoted 'continuum' for the ground state of the next ion, or by the appropriate level string when the upper level is not the ground state.

Version 2.06

- Consistency with xstar version 221bn. Includes

  up to date r-matrix atomic data for Ni and Al. Also includes model 'scatemis', which allows calculation of emission models for resonant-excitation dominated plasmas (optically thin).

Version 2.07

- Contents of the common block are output to fits files. A new file is created with each call to warmabs or photemis, with names like 'warmabsxxxx.fits', where xxxx is a sequential number, mod 9999. In order to implement this feature, the fphotems.f file must be edited: calls to the routine 'fitsprint' must be uncommented.

Version 2.07b

- Fixed minor errors in lmodel.dat. Shifted O I and O II L alpha to match observations.

Version 2.09

- Made consistent with xstar v221bn15. New N VI collisional excitation data.

17.2.2 Version 2.10

- Made consistent with xstar v221bn17. Changes to naming of output fits file suggested by John Houck: if the `WARMABS_OUTFILE` environment variable is set, then this name is used for the output fits file.
- Checks to make sure that the pops.fits file was created with the same atomic database as the current one, and exits if not.

Version 2.11

- O I absorption cross sections from Gorczyca et al. (2013). Fine structure for H-like ions for Z>20. Multiple errors fixed 10/08/2013.

Version 2.12

- Ne I absorption cross sections from Gorczyca et al. (2013). Fine structure for all H-like ions.

Version 2.13

- Mg I-III absorption cross sections from Gorczyca et al. (2013).
- New input parameters for warmabs, photemis, hotabs, hotemis: write_outfile is a switch controlling output of line depths/luminosities to an ascii fits file: 0=no fits files produced, 1=fits file containing lines/edges is produced, 2=fits file containing ion column densities is produced, 3=both types of fits files are produced. outfile_idx is an integer index. If the environment WARMABS_OUTPUT is not set, then the output is written to a fits file names 'warmabsxxxx.fits', where xxxx is the value of this variable. If WARMABS_OUTPUT is set, then its value is used as the name of the output file.
- Use of xstar v221bn18 routines: update to fundamental constants, adding thermal and turbulent velocities in quadrature consistently.

Version 2.14

- Extrapolation of all valence shell cross sections beyond tabulated values. This is a temporary fix to the problem of apparent 'negative edges' which appear at large column density in absorption spectra. Needed are extensions to the tabulated cross sections to higher energies.

Version 2.15

- Undid extrapolation of all valence shell cross sections beyond tabulated values because this led to spurious cross sections in some cases (He0 ground –> He+ 2p). Instead manually inserted extrapolated He0 ground –> He+ ground cross section into atomic data. Implemented cosmic abundances gotten from xspec internal tables. These can be changed using the 'abund' command, as described in the xspec manual.

Version 2.17

- When the 'write_outfile' parameter value is nonzero an additional file is produces, called warmabs_columns. This file contains the column densities of all ions from the most recent model call.

Version 2.18

- Fixed bug which affected output to the fits output files for photemis.

Version 2.19

- Fixed additional bug which affected output to the fits output files for photemis. This resulted in unphysical small wavelengths tabulated in the file for various lines, and also caused the wavelengths to change between successive calls.
- Changed the use of the write_outfile variable such that the values are as follows: 0=no fits files produced, 1=fits file containing lines/edges is produced, 2=fits file containing ion column densities is produced, 3=both types of fits files are produced.
- Increased the critical luminosity needed for photemis to add line emision to the spectrum, in order to speed execution.

17.2.3 Version 2.20

- Fixed error in Si XIV energy levels which was introduced when putting fine structure.

Version 2.21

- Added fine structure of He-like ions of C-Ni.

Version 2.22

- Added list of level indeces and ion index to the fits table output. These indeces are unlikely to change over time as new atomic data is added, so this should provide a robust way to keep track of lines and rrcs.

Version 2.23

- Removed duplications of atomic data: rrcs in Ni IX – XV and of lines in Na X and F VIII.

Version 2.24

- Changes to xstar code corresponding to xstar v2.2.1bn24.

Version 2.25

- Fix to error which led to incorrect emission in the Fe UTA.

Version 2.26

- Compatibility with xstar version 2.3. Extend photoionization extrapolation from 20 keV to 200 keV.

Version 2.27

- Compatibility with xstar version 2.31.

Version 2.29

- Compatibility with xstar version 2.38.

17.2.4 Version 2.30

- Fixed errors in binemis which resulted in natural width not being applied to emission line profiles.

Version 2.31

- Compatibility with xstar versions up to and including 2.54.

Version 2.32

- Fortran 90
- The stored model results are used for a first pass which selects only lines and rrcs (both in emission and absorption) which are strong. Then only the selected ones are calculated and binned in the synthetic spectrum. The criterion for strength is that they be in the top n in strength in the continuum bin where they peak. Currently n is set to 10. It is configurable using the 'nrank' variable in the parameter file PARAMX. A tradeoff is that full xstar outputs are needed, not just level populations. These occupy more disk space; the directory is currently 2.7Gb, about an 8-fold increase from previous versions. A consequence is that the stored files from xstar runs have fixed names. The `WARMABS_POP` environment variable is no longer needed or used. The user must run xstar and put all the output files named xo*.fits into the directory indicated by the `WARMABS_DATA` environment variable.

Version 2.34

- Various fixes to photemis and windabs.

Version 2.35

- Changes to fortran syntax to reduce warnings when compiled with -Wall and -pedantic. Also reduced the size of the array zrtmpcol.

Version 2.37

- Fix to normalization error in fphotemis continuum normalization.

BIBLIOGRAPHY

[1] S. M. V. Aldrovandi and D. Pequignot. Radiative and Dielectronic Recombination Coefficients for Complex Ions. *aap* , 25:137, May 1973.

[2] S. M. V. Aldrovandi and D. Pequignot. Erratum; Radiative and Dielectronic Recombination Coefficients for COM plex Ions. *aap* , 47:321, March 1976.

[3] E. Anders and M. Ebihara. Solar-system abundances of the elements. *gca* , 46(11):2363–2380, November 1982. doi:10.1016/0016-7037(82)90208-3.

[4] E. Anders and N. Grevesse. Abundances of the elements: Meteoritic and solar. *gca* , 53(1):197–214, January 1989. doi:10.1016/0016-7037(89)90286-X.

[5] M. Arnaud and J. Raymond. Iron Ionization and Recombination Rates and Ionization Equilibrium. *apj* , 398:394, October 1992. doi:10.1086/171864.

[6] Martin Asplund, Nicolas Grevesse, A. Jacques Sauval, and Pat Scott. The Chemical Composition of the Sun. *araa* , 47(1):481–522, September 2009. arXiv:0909.0948, doi:10.1146/annurev.astro.46.060407.145222.

[7] David L. Band, Richard I. Klein, John I. Castor, and J. K. Nash. Iron K-Shell Emission from NGC 1068. *apj* , 362:90, October 1990. doi:10.1086/169245.

[8] M. A. Bautista, C. Mendoza, T. R. Kallman, and P. Palmeri. Atomic data for the K-vacancy states of Fe XXIV. *aap* , 403:339–355, May 2003. arXiv:astro-ph/0207323, doi:10.1051/0004-6361:20030367.

[9] Manuel A. Bautista, Patrizia Romano, and Anil K. Pradhan. Resonance-averaged Photoionization Cross Sections for Astrophysical Models. *apjs* , 118(1):259–265, September 1998. arXiv:astro-ph/9712037, doi:10.1086/313132.

[10] A.K. Bhatia, J.F. Seely, and U. Feldman. Atomic data and spectral line intensities for the nitrogen isoelectronic sequence (ar xii through kr xxx). *Atomic Data and Nuclear Data Tables*, 43(1):99–143, 1989. URL: https://www.sciencedirect.com/science/article/pii/0092640X89900168, doi:https://doi.org/10.1016/0092-640X(89)90016-8.

[11] P. Bryans, N. R. Badnell, T. W. Gorczyca, J. M. Laming, W. Mitthumsiri, and D. W. Savin. Collisional Ionization Equilibrium for Optically Thin Plasmas. I. Updated Recombination Rate Coefficients for Bare through Sodium-like Ions. *apjs* , 167(2):343–356, December 2006. arXiv:astro-ph/0604363, doi:10.1086/507629.

[12] A. Burgess and J. A. Tully. On the analysis of collision strengths and rate coefficients. *aap* , 254:436–453, February 1992.

[13] S. E. Butler, T. G. Heil, and A. Dalgarno. Charge transfer of multiply charged ions with hydrogen and helium: quantal calculations. *apj* , 241:442–447, October 1980. doi:10.1086/158357.

[14] J. Callaway. Effective Collision Strengths for Hydrogen and Hydrogen-Like Ions. *Atomic Data and Nuclear Data Tables*, 57(1-2):9–20, May 1994. doi:10.1006/adnd.1994.1009.

[15] W. Cunto, C. Mendoza, F. Ochsenbein, and C. J. Zeippen. TOPbase at the CDS. *aap* , 275:L5–L8, August 1993.

[16] Kris Davidson and Hagai Netzer. The emission lines of quasars and similar objects. *Reviews of Modern Physics*, 51(4):715–766, October 1979. doi:10.1103/RevModPhys.51.715.

[17] K. P. Dere, E. Landi, H. E. Mason, B. C. Monsignori Fossi, and P. R. Young. CHIANTI - an atomic database for emission lines. \aaps , 125:149–173, October 1997. doi:10.1051/aas:1997368.

[18] U. Feldman. Elemental abundances in the upper solar atmosphere. \physscr , 46(3):202–220, September 1992. doi:10.1088/0031-8949/46/3/002.

[19] George B. Field and Gary Steigman. Charge Transfer and Ionization Equilibrium in the Interstellar Medium. \apj , 166:59, May 1971. doi:10.1086/150941.

[20] A. R. Foster, L. Ji, R. K. Smith, and N. S. Brickhouse. Updated Atomic Data and Calculations for X-Ray Spectroscopy. \apj , 756(2):128, September 2012. arXiv:1207.0576, doi:10.1088/0004-637X/756/2/128.

[21] R. J. Gould and R. K. Thakur. Atomic processes in a low-density hydrogen-helium plasma. *Annals of Physics*, 61:351–386, January 1970. doi:10.1016/0003-4916(70)90289-7.

[22] N. Grevesse, A. Noels, and A. J. Sauval. Standard Abundances. In Stephen S. Holt and George Sonneborn, editors, *Cosmic Abundances*, volume 99 of Astronomical Society of the Pacific Conference Series, 117. January 1996.

[23] N. Grevesse and A. J. Sauval. Standard Solar Composition. \ssr , 85:161–174, May 1998. doi:10.1023/A:1005161325181.

[24] Ming F. Gu, Tomer Holczer, Ehud Behar, and Steven M. Kahn. Inner-Shell Absorption Lines of Fe VI-Fe XVI: A Many-Body Perturbation Theory Approach. \apj , 641(2):1227–1232, April 2006. arXiv:astro-ph/0512410, doi:10.1086/500640.

[25] J. P. Harrington. Photoionization models. In Silvia Torres-Peimbert, editor, *Planetary Nebulae*, volume 131, 157–166. January 1989.

[26] S. Hatchett, J. Buff, and R. McCray. Transfer of X-rays through a spherically symmetric gas cloud. \apj , 206:847–860, June 1976. doi:10.1086/154448.

[27] D. G. Hummer, K. A. Berrington, W. Eissner, Anil K. Pradhan, H. E. Saraph, and J. A. Tully. Atomic data from the IRON project. I. Goals and methods. \aap , 279:298–309, November 1993.

[28] D. G. Hummer and G. Rybicki. The Formation of Spectral Lines. \araa , 9:237, January 1971. doi:10.1146/annurev.aa.09.090171.001321.

[29] J. S. Kaastra and R. Mewe. X-ray emission from thin plasmas. I - Multiple Auger ionisation and fluorescence processes for Be to Zn. \aaps , 97(2):443–482, January 1993.

[30] T. R. Kallman. Photoionization models for the winds from cataclysmic variables. \apj , 272:238–244, September 1983. doi:10.1086/161285.

[31] W. J. Karzas and Richard Latter. Detection of the Electromagnetic Radiation from Nuclear Explosions in Space. *Physical Review*, 137(5B):1369–1378, March 1965. doi:10.1103/PhysRev.137.B1369.

[32] T. Kato and S. Nakazaki. Recommended Data for Excitation Rate Coefficients of Helium Atoms and Helium-like Ions by Electron Impact. *Atomic Data and Nuclear Data Tables*, 42:313, January 1989. doi:10.1016/0092-640X(89)90010-7.

[33] F. P. Keenan and P. H. Norrington. Relative emission line strengths for Fe VII in astrophysical plasmas. \aap , 181:370–372, July 1987.

[34] J. B. Kingdon and G. J. Ferland. Rate Coefficients for Charge Transfer between Hydrogen and the First 30 Elements. \apjs , 106:205, September 1996. doi:10.1086/192335.

[35] Ali Kinkhabwala, Masao Sako, Ehud Behar, Steven M. Kahn, Frits Paerels, Albert C. Brinkman, Jelle S. Kaastra, Ming Feng Gu, and Duane A. Liedahl. XMM-Newton Reflection Grating Spectrometer Observations of Discrete Soft X-Ray Emission Features from NGC 1068. \apj , 575(2):732–746, August 2002. arXiv:astro-ph/0203290, doi:10.1086/341482.

[36] J. H. Krolik, C. F. McKee, and C. B. Tarter. Two-phase models of quasar emission line regions. *apj* , 249:422–442, October 1981. doi:10.1086/159303.

[37] J. Kwan and J. H. Krolik. The formation of emission lines in quasars and Seyfert nuclei. *apj* , 250:478–507, November 1981. doi:10.1086/159395.

[38] K. Lodders, H. Palme, and H. -P. Gail. Abundances of the Elements in the Solar System. *Landolt B&ouml;rnstein*, 4B:712, January 2009. arXiv:0901.1149, doi:10.1007/978-3-540-88055-4_34.

[39] Katharina Lodders. Solar System Abundances and Condensation Temperatures of the Elements. *apj* , 591(2):1220–1247, July 2003. doi:10.1086/375492.

[40] R. McCray, C. Wright, and S. Hatchett. Interstellar ultraviolet absorption lines and galactic X-ray sources. *apjl* , 211:L29–L33, January 1977. doi:10.1086/182336.

[41] Sultana N. Nahar. Electron-Ion Recombination Rate Coefficients, Photoionization Cross Sections, and Ionization Fractions for Astrophysically Abundant Elements. II. Oxygen Ions. *apjs* , 120(1):131–145, January 1999. doi:10.1086/313173.

[42] H. Nussbaumer and P. J. Storey. Dielectronic recombination at low temperatures. III - Recombination coefficients for Mg, Al, SI. *aaps* , 64(3):545–555, June 1986.

[43] Donald E. Osterbrock. *Astrophysics of gaseous nebulae*. W.H. Freeman & Co., San Francisco, 1974.

[44] P. Palmeri, C. Mendoza, T. R. Kallman, and M. A. Bautista. On the Structure of the Iron K Edge. *apjl* , 577(2):L119–L122, October 2002. arXiv:astro-ph/0207324, doi:10.1086/344243.

[45] J. C. Raymond and B. W. Smith. Temperature Diagnostic Line Ratios of Fe XVII. *apj* , 306:762, July 1986. doi:10.1086/164385.

[46] R. R. Ross, R. Weaver, and R. McCray. The Comptonization of iron X-ray features in compact X-ray sources. *apj* , 219:292–299, January 1978. doi:10.1086/155776.

[47] D. H. Sampson, S. J. Goett, and R. E. H. Clark. Electron-Impact Collision Strengths for Inner-Shell Excitation of Doubly Excited Levels from Singly Excited Levels in He-like Ions. *Atomic Data and Nuclear Data Tables*, 28:299, January 1983. doi:10.1016/0092-640X(83)90019-0.

[48] M. J. Seaton. The Opacity Project. In *IAU Colloq. 139: New Perspectives on Stellar Pulsation and Pulsating Variable Stars*, 231. January 1993.

[49] C. Bruce Tarter, Wallace H. Tucker, and Edwin E. Salpeter. The Interaction of X-Ray Sources with Optically Thin Environments. *apj* , 156:943, June 1969. doi:10.1086/150026.

[50] Wallace H. Tucker and Marvin Koren. Radiation from a High-Temperature Low-Density Plasma: the X-Ray Spectrum of the Solar Corona. *apj* , 168:283, September 1971. doi:10.1086/151083.

[51] D. A. Verner and D. G. Yakovlev. Analytic FITS for partial photoionization cross sections. *aaps* , 109:125–133, January 1995.

[52] J. Wilms, A. Allen, and R. McCray. On the Absorption of X-Rays in the Interstellar Medium. *apj* , 542(2):914–924, October 2000. arXiv:astro-ph/0008425, doi:10.1086/317016.

[53] Honglin Zhang and Douglas H. Sampson. Collision Rates for Excitation of Helium-like Ions with Inclusion of Resonance Effects. *apjs* , 63:487, February 1987. doi:10.1086/191171.

INDEX