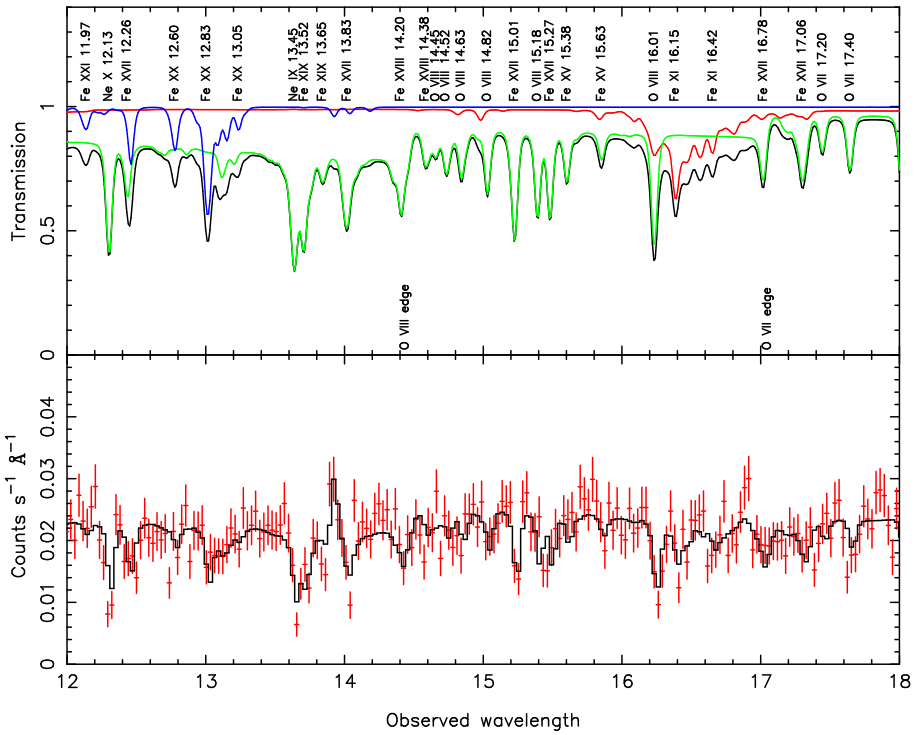

SPEX Reference Manual

Jelle Kaastra

Rolf Mewe, Ton Raassen & Jelle de Plaa

Version 3.05.00

December 19, 2018

Contents

Recent changes to the manual

Date	Version	Changes
December 2018	3.05.00	Minor updates to the manual: Updates to the installation instructions Added documentation for output feature of sector command Added documentation for the correlation output of the par show command Fixed error in rgsfluxcombine documentation Added documentation about the added Ext_rate column in spo files Updates to the pion model documentation Updated manual for the simulate command
December 2017	3.04.00	Minor updates to the manual: Updates to the pion model documentation Updates to the C-stat documentation Update to the stepcontour program documentation
November 2016	3.03.00	Minor updates to the manual: Minor updates to var, hyd, amol and pion
August 2016	3.02.00	Minor updates to the manual: Introduced SPEXACT naming and versioning
April 2016	3.01.00	Minor updates to the manual: Update of the refl model description Update of the pion model description Update of the cx model description Update of the ascdump and ions command description
January 2016	3.00.00	Full revision of the manual with a.o.: New Chapter 1, Introduction New Chapter 2, Spectral Fitting and Code Refences list updated New model descriptions added for new models

Chapter 1

Introduction

1.1 General description

This document gives a background of the use of the software package SPEctral X-ray and UV modelling and analysis, SPEX . In the last decades SRON-Utrecht has developed this software package for the computation and modelling of X-ray and EUV spectra. SPEX encompasses a number of subroutines for the computation of emergent spectra of optically thin plasmas such as stellar coronal loop structures and supernova remnants (also including transient ionization effects), photo-ionized plasmas, and optically thick plasmas. A synthetic spectra program that convolves the calculated input spectra with representative instrumental response functions and a subroutine for Differential Emission Measure modelling is available.

SPEX includes more than a million lines from 30 (from H to Zn) different chemical elements in the far UV and X-ray band. These lines are produced by excitation from electron impact, radiative and dielectronic recombination and by innershell excitation and ionization. In addition to the lines the code calculates the contributions from continuum radiation due to free-free, free-bound, and two-photon emission and includes several absorption models.

Current developments are pointed to extend the work to photoionized plasmas and charge exchange processes. This will be needed since it became evident that those processes play an important role in X-ray sources and emitting regions. Moreover, atomic physics has improved considerably during the last decade, and the advent of a new series of satellites with high sensitivity and spectral resolution like Hitomi or Athena, strongly demands the availability of spectral code with higher accuracy and more detail.

1.2 History

Starting in the early seventies (Mewe 1972) Rolf Mewe and his colleagues of the Space Research Laboratory in Utrecht started developing a spectral code for the X-ray emission of optically thin plasmas. A milestone was reached in 1985 and 1986, when two papers describing the line emission (Mewe et al. 1985) and continuum emission (Mewe et al. 1986) were published. This spectral code, often abbreviated as the Mewe-Gronenschild code, together with the code by Raymond & Smith (1977) served for many years as the basis for many X-ray spectroscopic papers.

At the beginning of the eighties the code was extended with non-equilibrium ionization balance

calculations (Gronenschild & Mewe 1982), and motivated by the work on supernova remnants done in Leiden by Fred Jansen and Jelle Kaastra vastly improved. The need for a faster code in order to be able to do spectral fitting to these non-equilibrium spectra of supernova remnants motivated us to improve the performance of the code. Then in 1992 an update of in particular the continuum calculation appeared, and this updated and faster computer code was implemented at HEASARC in the XSPEC package, at which time it was baptized as meka code, after the two main authors of this version, Mewe and Kaastra.

Around the same time (early 1992) [1]we started the SPEX project, with the intention to improve, update and extend also the line emission part of the code. In addition, we intended to make it more useful by having spectral simulation and fitting options attached to it, as well as more illustrative graphical and tabular output from the program, which helps to understand the physics of the sources being studied. Also different extensions like hydrodynamical models for supernova remnants, and differential emission measure analysis techniques were foreseen.

Figure 1.1: Mewe, Kaastra & Liedahl preparing the mekal model. Photo by Hans Nieuwenhuijzen, 1996.

In 1994 the first version of SPEX became public, with the re-written meka code as working horse in the core of the software package. A year before the ASCA satellite had been launched, and in one of the first X-ray spectra of a cluster of galaxies, the Centaurus clusters, the observed spectrum did not match neither the predicted meka calculations nor the results of other codes like the Raymond & Smith code (Fabian et al. 1994). In particular the Fe-L $4-2$ to $3-2$ blend ratio did not match the observations. Around the same time, Duane Liedahl was able to produce better results for the Fe- L complex using the HULLAC atomic code, and by joining forces an update of the meka code, now named mekal, after the three authors, was released (Mewe, Kaastra & Liedahl 1995). This code was both included in the SPEX package as well as a model in the XSPEC package. Apart form the update and extension of the iron L complex, it also contained an improvement of the $300-2000$ Å band, triggered by the EUVE spectra of stellar coronae, an update for the Fe VIII to Fe XVI ions, plus addition of DR satellites for Mg XI. Also the ionization balance for iron was updated using the results of Arnaud & Raymond (1992).

Already around 1996 we started developing version 2.0 of the code, with a transition from olf fortran77 compilers to modern fortran90, allowing much more flexibility using dynamical memory, modules etc. Also the original menu-driven structure was replaced by a command

[1]The oldest program files that we can trace back dates from 5 February 1991

structure. Stimulated by the availability of high-resolution spectra taken by Chandra and XMM-Nwton the code was gradually extended and improved, although the availability of these new data also slowed down the development due to manpower limitations.

Driven by the need to further improve on the atomic data because of the more detailed spectra that become available, in January 2016 version 3.0 of the software has been launched.

1.3 SPEXACT: The SPEX Atomic Code & Tables

The SPEX package is a spectral fitting program with integrated models that are mostly based on one atomic database and a set of routines to calculate all the atomic processes in the plasma. Although the development of the atomic database and code is performed mostly in parallel with the other SPEX development, it can sometimes be confusing which version of the atomic database was actually used in an analysis. Especially after the major update of the atomic database in SPEX 3.0 and the option to calculate models using the 'old' database and routines from SPEX 2.0, there is a need to name and version the 'core' routines of SPEX separately. This has become SPEXACT (SPEX Atomic Code & Tables). In principle, models calculated using the same SPEXACT version should produce the same results.

Please note that the SPEXACT database and routines are an integrated part of SPEX and are not distributed separately.

1.3.1 Main version number definition

SPEXACT v1: Is essentially the MEKAL model that was developed in the 1980's and is distributed with Xspec. This model is no longer developed or supported, but can be regarded as the first version of SPEXACT. It is no longer included in SPEX.

SPEXACT v2: In SPEX version 1 & 2, the original MEKAL model was extended and updated into SPEX. The version number of this SPEXACT version is the same as the version number of SPEX when it was released. For example, the SPEXACT version in SPEX version 2.04.00 is also 2.04.00. In SPEX version 3, these routines are still the default and are used in `var calc old` mode.

SPEXACT v3: These are the newly developed atomic database and corresponding routines that were officially released with SPEX version 3.00.00. This database and its routines are not (yet) the default in SPEX but can be used when the `var calc new` command is given. The second number in the SPEXACT version is the same as the SPEX version it was released in. The third number can in principle be different from the SPEX version and indicates the minor version of the database.

1.4 Changes introduced in SPEX 3.0

What is new in SPEX version 3.00 compared to SPEX 2.06? Most of the changes are actually "under the hood" of the SPEX program and are only noticed when model fits are compared, but we also provide useful new model additions. A more detailed overview of the changes in the plasma models can be found in Section 5.1. The most important changes are summarized here:

- The first version of the new atomic database prepared by Ton Raassen is included in this release. It contains hundreds of thousands of energy levels from hydrogen to zinc. The

new database is not yet enabled by default (see command `var calc new`). Using the new line database requires more memory and CPU time then before, which may not be needed in every case. Enable the new line list if you need more accuracy, for example when you fit high-resolution spectra.

- We have optimized the calculation of the radiative recombination, di-electronic recombination, two-photon, and proton excitation processes to be able to calculate a spectrum from the new atomic data within a reasonable time. The results from full calculations were approximated by functions that can be calculated much faster, which allows the fitting of complicated spectra with models containing a large range of physical parameters of the plasma.
- The photo-ionization model `pion` has been substantially improved. It is now possible to model several absorption layers with including their radiative transfer. Also the emission part is implemented. However, use with care since this model is still experimental.
- We included natural & Voigt broadening to SPEX emission models.
- We introduce a new charge exchange model (`cx`) developed by Gu et al. (2016).
- We added the `ebv` component in SPEX to model interstellar extinction.
- There are now interfaces to include your own model in SPEX . Use `com user` for additive models and `com musr` for multiplicative models. Examples of how to use this model component are shown in the revised SPEX Cookbook. It also allows you to call Xspec models from SPEX .
- We added a line emission/absorption model with a Voigt profile (`com line`) to model individual lines.
- Contour plots of χ^2 values from the `steppar` command can now be plotted using the `stepcontour` task. The `steppar` command can now export the results to an ASCII file that `stepcontour` can convert into a quick contour plot (or QDP file).
- The SPEX syntax has been slightly changed. From now on, all ranges can be written with a ':' in between to improve the consistency of the syntax. For example, changing the X axis of a plot can now be written as: `plot rx 0.1:10`. Affected commands are `plot`, `egrid`, `elim`, `dem chireg`, `step axis`, and `par ... range`. If possible, the old syntax was also kept alive. Check the manual for details.
- C-statistics, the Bryans et al. (2009) ionization balance and the proto-solar abundances by Lodders et al. (2009) are now the default in SPEX. The abundance table and the ionisation balance can now be queried using the commands `abun show` and `ibal show`, respectively.

1.4.1 Changes in version 3.01

- To optimize for calculation speed, in the new line list, one can select ions that should be calculated the old way (`ions old`) or the new way (`ions new`). The syntax is the same as the `ions ignore` and `ions use` commands.
- Added sorting option to the command `asc ter ... line` to sort lines based on emissivity.

- In version 3.00.00 (using the new line database, `var calc new`), we had also included dielectronic satellite lines of He-like ions of the type 1s2 => 2s 2p2, which actually are trielectronic recombination lines. The 2s 2p2 levels decay rapidly through radiation of a photon (2p to 1s transition) to the 1s 2s 2p level, which decays further by a similar transition to the 1s2 2s ground state of Li-like iron. Therefore, our model predicts relatively strong satellite lines from these transitions, stronger than present in corresponding EBIT spectra. We have resolved this issue by for this moment omitting transitions from these triply excited states from the calculations. Note they were/are absent in SPEX version 2.0 or all APEC versions.

- The new CIE model did not reproduce the x/y line ratio in He-like iron very well. The x & y lines are the two intercombination lines in this ion. We traced this problem down to some confusion caused by different notations for line levels. The NIST database, the Cowan code, Chianti use different notations for the same transition: some use 1s2s(3S)2p 2P1/2, others 1s2s(1S)2p 2P1/2 for the same transition (same if you compare the values of the corresponding energy levels). Some others couple the 2s and 2p electrons first, leading to P states, and then attach the 1s electron to it, leading to again different notations. This confusion led to an interchange of collisonal and radiative rates between some levels, thereby affecting the x/y line intensities. This has now been fixed.

- The innershell levels and related auger and radiative transitions for Fe XIX to Fe XXIII are added using the data from Palmeri et al. (2003a).

1.4.2 Changes in version 3.02.00

- Updated pion model (details will be described soon).

- Re-installed the tri-electronic recombination transistions (details will be described soon).

- Introduced the SPEXACT naming and versioning for the atomic code and tables in SPEX.

1.4.3 Changes in version 3.03.00

In SPEX 3.03.00:

- Fixed bug related to opening many fits files.

- Fixed bug in data merge regarding the deallocation of the derivative array.

- Fixed bug in the calculation of the average exposure time in data show.

In SPEXACT 3.03.00:

- Fixed bug in generation of atomic data files.

- Added Auger rates for O V, Ne VII, Fe XIII.

- Ionization limits added for Mg IX, Si XI, S XIII, Ar XV.

- Atomic data extended including autoionisation for Be-like ions.

- Added correction for auto-ionization for excitation and inner-shell ionisation to doubly excited levels.
- Update and bugfix for CX model (H-H collision).
- Updated ionisation balance for U16.

1.4.4 Changes in version 3.04.00

In SPEX 3.04.00:

- The ionisation balance is now by default Urdampilleta et al. (2017). The actual data did not change, but the paper was published in 2017. The command `ibal u16` still works, but gives a warning message that the actual command is `ibal u17` from now on.
- Fixed bug in stepcontour program related to logarithmic grids.

In SPEXACT 3.04.00:

- Fixed bugs in the CIE, NEIJ, CX and PION models.
- Included radiative recombination cooling following Mao et al. (2017).
- Updates of PION model.
- Updated Auger rates for Be-like to C-like Fe.
- Ni XXI of the O-like added and some Fe ions extended.
- Now we multiply the emission measure by n_e/n_H in the pion model. This will result in typically 20% more emission from the pion model.
- Added the tmod option to the pion model (see 4.29), allowing to set the temperature and not solve for energy balance (useful for hot stars, the WHIM etc.).
- Allowed to use multiple solutions in the pion model (fmod and soln parameters).
- Allowed for external heating source in the pion model (exth parameter).
- Extended the pion model with all elements from Z=1 to Z=30.
- Fe XX levels modified to Nist5.
- Bug fix: Now we use the proper argument 2 (y) in the call to the integrated Voigt function sivf. The Lorentzian component is now 2x narrower than before, which does affect the CIE emission spectra. It does not affect "old" xabs, hot calculations etc.
- Extended data for Fe XXI, Fe XXII, and Fe XXIII.

1.4.5 Changes in version 3.05.00

In SPEX 3.05.00:

- Bug in sector copy fixed.
- Fixed additional page problem in postscript output of plot.
- Introduced W-statistics (not recommended for use).
- Xabsinput can now handle SEDs with more than 1024 bins.
- New version of trafo that adds the Ext_rate column to a .spo file. This column is necessary to properly simulate a spectrum when there is a background spectrum present. The Ext_rate column shows the ratio of the backscales of the source and background spectrum.
- Simulate command syntax changed.
- When calculating errors of multiple parameters with one error command, parameters of the best fit found across all parameters are written to `spex_lower_chi.com`, if a better minimum was found.
- Spectra dumped with `plot adum` now contain a 'NO' to separate the spectra.
- The sector command can now dump a model to a text file that can be read in by the SPEX file model. This could be helpful when one wants to model the background without evaluating the model each time.
- Added correlation information in the output of the `par show` command.
- Added the instrument normalisation to the output of `par write`.
- Replaced a few proprietary math routines with open source alternatives.
- Applied GPL license to SPEX and prepared source code for publication.
- Added first version of SPEX tests to the source code directory.
- Fixed bug in the check of the number of free instrument normalisations in the model.

In SPEXACT 3.05.00:

- Bug and stability fixes for the pion model.
- Bugfix in the temperature grid of the cooling-flow model. Mass deposition rates are now consistent with other cooling-flow models.
- Stability issue with the free-bound calculation resolved.
- Fixed small issue in the charge exchange model for Fe XXIII.

1.5 Document structure

This document has been divided in the following sections:

- Chapter 1, is an introduction of SPEX Reference Manual contents.
- Chapter 2, is a brief explanation of spectral fitting philosphy, the initial information needed and code used in SPEX .
- Chapter 3, lists the main syntax used in SPEX .
- Chapter 4, contains an overview of all spectral components and models.
- Chapter 5, explains additional spectral models.
- Chapter 6, describes the tool used for plotting.
- Chapter 7, lists the auxiliary programs needed in some cases for using SPEX .
- Chapter 8, includes guidelines about how to construct a proper response matrix .
- Chapter 9, gives some instructions to test SPEX after the installation.
- Chapter 10, contains the acknowlegements.

1.6 Additional information and documents

1.6.1 SPEX Cookbook: Examples

In addition to the current manual, we have updated the SPEX cookbook, which contains several useful hints and examples of how to analyse X-ray spectra. The HTML version of the cookbook can be found at `http://var.sron.nl/SPEX-doc/cookbook` or find it in the manual section of the SPEX web site: http://www.sron.nl/spex

1.6.2 Physical background of spectral models

Apart from Chapter 5 of this manual, other books and documents that describe the physical backgrounds are also available.

The fundamentals of the SPEX code are summarized in SRON/SPEX/TRPB01, which can be downloaded from the SPEX website: `https://www.sron.nl/astrophysics-spex/physics`. The documents below describe the basics of the main emission processes in the older MEKAL code and the basics of multi-temperature analysis:

- Ionization and Energy balance in Collisionally and Photoionized Plasmas (SRON/SPEX/TRPB02)
- Continuum Radiation Processes (SRON/SPEX/TRPB03)
- Line Excitation Processes (SRON/SPEX/TRPB04)
- Differential Emission Measure Analysis (SRON/SPEX/TRPB05)

Updates on the physics and application of these older codes can be found in the books Kaastra (2008) and Kaastra & Paerels (2011).

The official SPEX website is:
http://www.sron.nl/spex

Chapter 2

Spectral Fitting

2.1 Spectral Modelling

To infer relevant physical parameters from observed X-ray spectra, physical models are needed. The models need to calculate the expected X-ray spectrum based on physical parameters, like the electron temperature, emission measure and density distributions, ion and elemental abundances, mass motions, and the nature of the ambient radiation field. Unfortunately, the physical models cannot be compared directly to the measured spectra, because the instruments used to obtain the spectra change the spectrum for example by adding background components (instrument noise), change the continuum shape due to the sensitivity of the mirror (effective area), and blur spectral lines due to the instruments spectral resolution.

To take these instrumental components into account, the usual procedure is to apply a forward modelling technique by convolving theoretical model spectra with the instrumental response and to vary the model parameters in order to optimize the fit of the model to the observational data. A common approach is to consider first a simplified plasma model for the X-ray source, neglecting much of the complexity of the temperature and density structure and of the effects of opacity, and to synthesize such models into successively more sophisticated approximations of the source model.

In Figure 2.1 we give in a processing flow diagram schematically the process of spectral modelling for the case of optically thin coronal plasmas. The synthetic spectra program is fed with input parameters from the spectral model (atomic data for ionization and line and continuum excitation), the instrumental model, and from the assumed plasma model for the source. The synthetic spectra code generates spectra which can be compared to the observations and tested by means of statistical fitting procedures.

2.2 Sectors and regions

2.2.1 Introduction

In many cases an observer analysis the observed spectrum of a single X-ray source. There are however situations with more complex geometries.

Example 1: An extended source, where the spectrum may be extracted from different regions of the detector, but where these spectra need to be analysed simultaneously due to the overlap

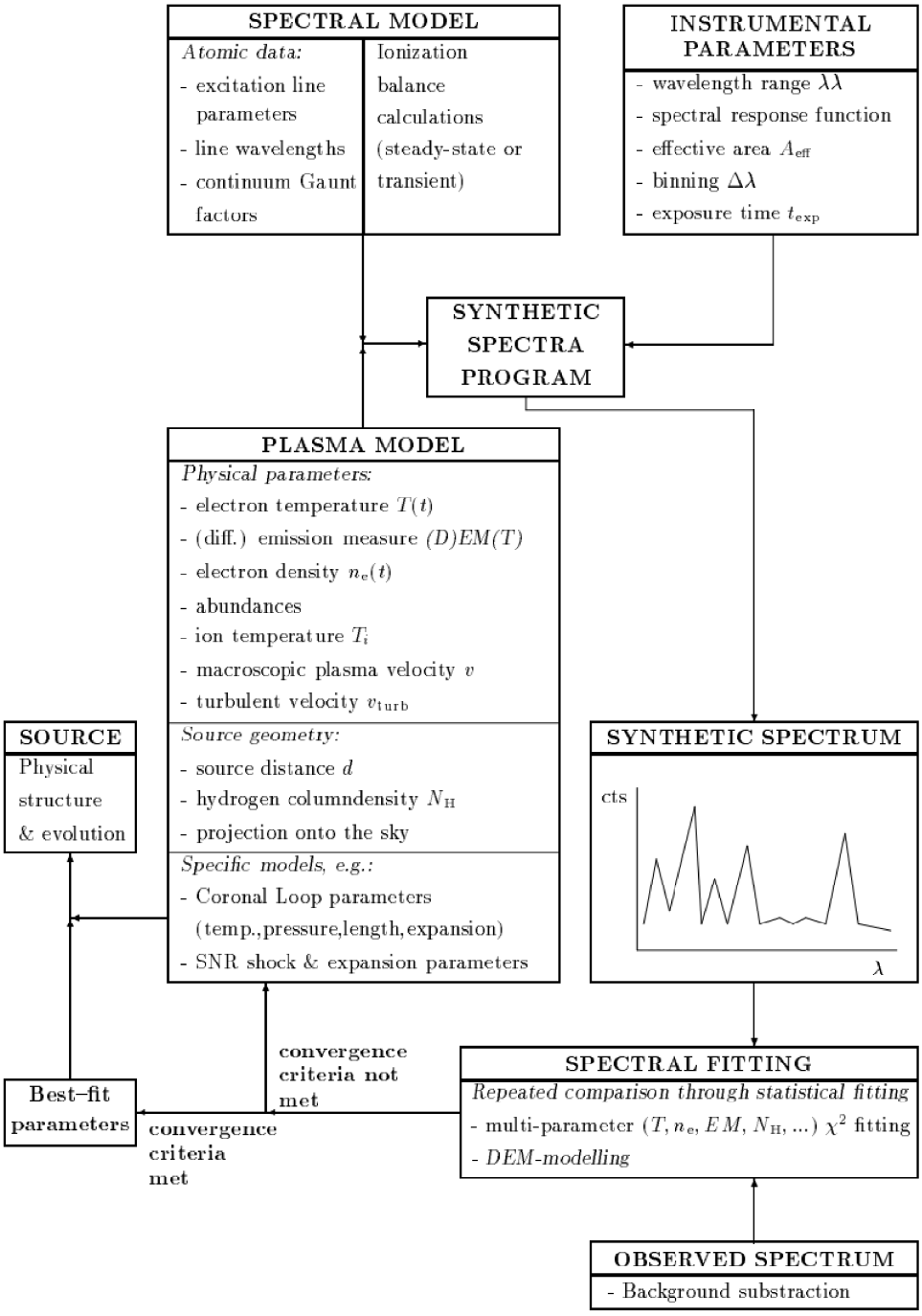

Figure 2.1: Processing Flow Diagram for Spectral Modelling of optically thin plasmas (from Mewe 1992).

in point-spread function from one region to the other. This situation is e.g. encountered in the analysis of cluster data with ASCA or BeppoSAX.

Example 2: For the RGS detector of XMM-Newton, the actual data-space in the dispersion direction is actually two-dimensional: the position z where a photon lands on the detector and its energy or pulse height E as measured with the CCD detector. X-ray sources that are extended in the direction of the dispersion axis ϕ are characterised by spectra that are a function of both the energy E and off-axis angle ϕ. The sky photon distribution as a function of (ϕ,E) is then mapped onto the (z,E)-plane. By defining appropriate regions in both planes and evaluating the correct (overlapping) responses, one may analyse extended sources.

Example 3: One may also fit simultaneously several time-dependent spectra using the same response, e.g. data obtained during a stellar flare.

It is relatively easy to model all these situations (provided that the instrument is understood sufficiently, of course), as we show below.

2.2.2 Sky sectors

First, the relevant part of the sky is subdivided into sectors, each sector corresponding to a particular region of the source, for example a circular annulus centered around the core of a cluster, or an arbitrarily shaped piece of a supernova remnant, etc.

A sector may also be a point-like region on the sky. For example if there is a bright point source superimposed upon the diffuse emission of the cluster, we can define two sectors: an extended sector for the cluster emission, and a point-like sector for the point source. Both sectors might even overlap, as this example shows!

Another example: the two nearby components of the close binary α Centauri observed with the XMM -Newton instruments, with overlapping point-spread-functions of both components. In that case we would have two point-like sky sectors, each sector corresponding to one of the double star's components.

The model spectrum for each sky sector may and will be different in general. For example, in the case of an AGN superimposed upon a cluster of galaxies, one might model the spectrum of the point-like AGN sector using a power law, and the spectrum from the surrounding cluster emission using a thermal plasma model.

2.2.3 Detector regions

The observed count rate spectra are extracted in practice in different regions of the detector. It is necessary here to distinguish clearly the (sky) sectors and (detector) regions. A detector region for the XMM EPIC camera would be for example a rectangular box, spanning a certain number of pixels in the x- and y-directions. It may also be a circular or annular extraction region centered around a particular pixel of the detector, or whatever spatial filter is desired. For the XMM RGS it could be a specific "banana" part of the detector-coordinate CCD pulse-height plane (z,E).

Note that the detector regions need not to coincide with the sky sectors, neither should their number to be equal! A good example of this is again the example of an AGN superimposed upon a cluster of galaxies. The sky sector corresponding to the AGN is simply a point, while, for a finite instrumental psf, its extraction region at the detector is for example a circular region centered around the pixel corresponding to the sky position of the source.

Also, one could observe the same source with a number of different instruments and analyse the data simultaneously. In this case one would have only one sky sector but more detector regions, namely one for each participating instrument.

2.3 Different types of spectral components

In a spectral model SPEX uses three different types of components, called *additive*, *multiplicative*, and *hybrid* components respectively. Additive components have a normalisation that determines the flux level. Multiplicative components operate on additive components. The pion model is a hybrid model that models both the absorption and emission from an intervening slab of plasma. A delta line or a power law are typical examples of additive components. Interstellar absorption is a typical example of a multiplicative component.

The redshift component is treated as a multiplicative component, since it operates on additive components.

Additive components can be divided into two classes: simple components (like power law, blackbody etc.) and plasma components, that use our atomic code. For the plasma components it is possible to plot or list specific properties, while for the simple models this is not applicable.

Multiplicative components can be divided into 3 classes. First, there are the absorption-type components, like interstellar absorption. These components simply are an energy-dependent multiplication of the original source spectrum. SPEX has both simple absorption components as well as absorption components based upon our plasma code. The second class consists of shifts: redshift (either due to Doppler shifts or due to cosmology) is the prototype of this. The third class consists of convolution-type operators. An example is Gaussian velocity broadening.

For more information about the currently defined spectral components in SPEX, see chapter 4.

Chapter 3

Syntax overview

3.1 Introduction

This chapter contains a brief explanation of the most relevant syntax used in SPEX. Each subchapter is divided in an overview, an explanation of the use of syntax followed by some practical examples.

3.2 Abundance: standard abundances

Overview

For the plasma models, a default set of abundances is used. All abundances are calculated relative to those standard values. The current default abundance set are the proto-Solar abundances of Lodders et al. (2009). Note that in older versions of SPEX, Anders & Grevesse (1989) was the default. In particular, we recommend to use the proto-Solar (= Solar system) abundances for most applications, as the Solar photospheric abundance has been affected by nucleosynthesis (burning of H to He) and settlement in the Sun during its lifetime. The following abundances (Table 3.1) can be used in SPEX:

Table 3.1: Standard abundance sets

Abbrevation	Reference
reset	default (= Lodders et al. 2009)
ag	Anders & Grevesse (1989)
allen	Allen (1973)
ra	Ross & Aller (1976)
grevesse	Grevesse et al. (1992)
gs	Grevesse & Sauval (1998))
lodders	Lodders proto-Solar (Lodders 2003)
solar	Lodders Solar photospheric (Lodders 2003))

For the case of Grevesse & Sauval (1998) we adopted their meteoritic values (in general more accurate than the photospheric values, but in most cases consistent), except for He (slightly enhanced in the solar photosphere due to element migration at the bottom of the convection zone), C, N and O (largely escaped from meteorites) Ne and Ar (noble gases).

In Table 3.2 we show the values of the standard abundances. They are expressed in logarithmic units, with hydrogen by definition 12.0. For Allen (1973) the value for boron is just an upper limit.

Table 3.2: Abundances for the standard sets

Z	elem	Default	AG	Allen	RA	Grevesse	GS	Lodders	solar
1	H	12.000	12.00	12.00	12.00	12.00	12.00	12.00	12.00
2	He	10.987	10.99	10.93	10.80	10.97	10.99	10.98	10.90
3	Li	3.331	1.16	0.70	1.00	1.16	3.31	3.35	3.28
4	Be	1.373	1.15	1.10	1.15	1.15	1.42	1.48	1.41
5	B	2.860	2.6	<3.0	2.11	2.6	2.79	2.85	2.78
6	C	8.443	8.56	8.52	8.62	8.55	8.52	8.46	8.39
7	N	7.912	8.05	7.96	7.94	7.97	7.92	7.90	7.83
8	O	8.782	8.93	8.82	8.84	8.87	8.83	8.76	8.69
9	F	4.491	4.56	4.60	4.56	4.56	4.48	4.53	4.46
10	Ne	8.103	8.09	7.92	7.47	8.08	8.08	7.95	7.87
11	Na	6.347	6.33	6.25	6.28	6.33	6.32	6.37	6.30
12	Mg	7.599	7.58	7.42	7.59	7.58	7.58	7.62	7.55
13	Al	6.513	6.47	6.39	6.52	6.47	6.49	6.54	6.46
14	Si	7.586	7.55	7.52	7.65	7.55	7.56	7.61	7.54
15	P	5.505	5.45	5.52	5.50	5.45	5.56	5.54	5.46
16	S	7.210	7.21	7.20	7.20	7.21	7.20	7.26	7.19
17	Cl	5.299	5.5	5.60	5.50	5.5	5.28	5.33	5.26
18	Ar	6.553	6.56	6.90	6.01	6.52	6.40	6.62	6.55
19	K	5.161	5.12	4.95	5.16	5.12	5.13	5.18	5.11
20	Ca	6.367	6.36	6.30	6.35	6.36	6.35	6.41	6.34
21	Sc	3.123	1.10	1.22	1.04	3.20	3.10	3.15	3.07
22	Ti	4.979	4.99	5.13	5.05	5.02	4.94	5.00	4.92
23	V	4.042	4.00	4.40	4.02	4.00	4.02	4.07	4.00
24	Cr	5.703	5.67	5.85	5.71	5.67	5.69	5.72	5.65
25	Mn	5.551	5.39	5.40	5.42	5.39	5.53	5.58	5.50
26	Fe	7.514	7.67	7.60	7.50	7.51	7.50	7.54	7.47
27	Co	4.957	4.92	5.10	4.90	4.92	4.91	4.98	4.91
28	Ni	6.276	6.25	6.30	6.28	6.25	6.25	6.29	6.22
29	Cu	4.319	4.21	4.50	4.06	4.21	4.29	4.34	4.26
30	Zn	4.700	4.60	4.20	4.45	4.60	4.67	4.70	4.63

WARNING: *For Allen (1973) the value for boron is just an upper limit.*

The current active solar abundance table can be shown using the command `abundance show`.

Syntax

The following syntax rules apply:

`abundance #a` - Set the standard abundances to the values of reference #a in the table above.
`abundance show` - Show the currently active abundance table.

Examples

`abundance gs` - change the standard abundances to the set of Grevesse & Sauval (1998)
`abundance reset` - reset the abundances to the standard set
`abundance show` - show the currently active abundance table

3.3 Ascdump: ascii output of plasma properties

Overview

One of the drivers in developing SPEX is the desire to be able to get insight into the astrophysics of X-ray sources, beyond merely deriving a set of best-fit parameters like temperature or abundances. The general user might be interested to know ionic concentrations, recombination rates etc. In order to facilitate this SPEX contains options for ascii-output.

Ascii-output of plasma properties can be obtained for any spectral component that uses the basic plasma code of SPEX; for all other components (like power law spectra, gaussian lines, etc.) this sophistication is not needed and therefore not included. There is a choice of properties that can be displayed, and the user can either display these properties on the screen or write it to file.

The possible output types are listed below. Depending on the specific spectral model, not all types are allowed for each spectral component. The keyword in front of each item is the one that should be used for the appropriate syntax.

plas: basic plasma properties like temperature, electron density etc.

abun: elemental abundances and average charge per element.

icon: ion concentrations, both with respect to Hydrogen and the relevant elemental abundance.

rate: total ionization, recombination and charge-transfer rates specified per ion.

rion: ionization rates per atomic subshell, specified according to the different contributing processes.

pop: the occupation numbers as well as upwards/downwards loss and gain rates to all quantum levels included.

lev: the contributions to the pouplation of the energy levels by various processes: positive for gain, negative for loss

elex: the collisional excitation and de-excitation rates for each level, due to collisions with electrons.

prex: the collisional excitation and de-excitation rates for each level, due to collisions with protons.

rad: the radiative transition rates from each level.

two: the two-photon emission transition rates from each level.

grid: the energy and wavelength grid used in the last evaluation of the spectrum.

clin: the continuum, line and total spectrum for each energy bin for the last plasma layer of the model.

line: the line energy and wavelength, as well as the total line emission (photons/s) for each line contributing to the spectrum, for the last plasma layer of the model. Also given is the natural line width and the Doppler broadening (including thermal and turbulent broadening), expressed as a FWHM in keV. Optionally, the results can be sorted according to various columns as follows (first description, between brackets the acronym): energy (ener), wavelength (wav), ion (ion), line power (powe), natural line width (wid).

con: list of the ions that contribute to the free-free, free-bound and two-photon continuum emission, followed by the free-free, free-bound, two-photon and total continuum spectrum, for the last plasma layer of the model.

tcl: the continuum, line and total spectrum for each energy bin added for all plasma layers of the model.

tlin: the line energy and wavelength, as well as the total line emission (photons/s) for each line contributing to the spectrum, added for all plasma layers of the model.

tcon: list of the ions that contribute to the free-free, free-bound and two-photon continuum emission, followed by the free-free, free-bound, two-photon and total continuum spectrum, added for all plasma layers of the model.

cnts: the number of counts produced by each line. Needs an instrument to be defined before.

nei: the history of ionisation parameter and temperature in NEI calculations.

snr: hydrodynamical and other properties of the supernova remnant (only for supernova remnant models such as Sedov, Chevalier etc.).

heat: plasma heating rates (only for photoionized plasmas).

ebal: the energy balance contributions of each layer (only for photoionized plasmas).

dem: the emission measure distribution (for the pdem model)

col: the ionic column densities for the hot, pion, slab, xabs and warm models

tran: the transmission and equivalent width of absorption lines and absorption edges for the hot, pion, slab, xabs and warm models. Optionally, the results (lines only) can be sorted according to various columns as follows (first description, between brackets the acronym): energy (ener), wavelength (wav), ion (ion), optical depth at line center (tau), equivalent width in keV (ewk), equivalent width in Å (ewa), Voigt a parameter (avo).

warm: the column densities, effective ionization parameters and temperatures of the *warm* model

Syntax

The following syntax rules apply for ascii output:

`ascdump terminal #i1 #i2 #a` - Dump the output for sky sector #i1 and component #i2 to the terminal screen; the type of output is described by the parameter #a which is listed in the table above.
`ascdump file #a1 #i1 #i2 #a2` - As above, but output written to a file with its name given by the parameter #a1. The suffix ".asc" will be appended automatically to this filename.
`ascdump terminal #i1 #i2 #a1 sort #a2` - Dump the output for sky sector #i1 and component #i2 to the terminal screen; the type of output is described by the parameter #a1 which is listed in the table above; the results are sorted according to parameter #a2. Sorting is only possible for the "line" and "tran" options; see there for allowed entries.
`ascdump file #a1 #i1 #i2 #a2` - As above, but output written to a file with its name given by the parameter #a1. The suffix ".asc" will be appended automatically to this filename.

WARNING: *Any existing files with the same name will be overwritten.* WARNING: *Sorting only possible for the line and tran options.*

Examples

`ascdump terminal 3 2 icon` - dumps the ion concentrations of component 2 of sky sector 3 to the terminal screen.
`ascdump file mydump 3 2 icon` - dumps the ion concentrations of component 2 of sky sector 3 to a file named mydump.asc.
`ascdump terminal 3 2 line sort pow` - dumps the emission line power of component 2 of sky sector 3 to the terminal screen, sorted according to line strength.

3.4 Bin: rebin the spectrum

Overview

This command rebins the data (thus both the spectrum file and the response file) in a manner as described in Sect. 3.7. The range to be rebinned can be specified either as a channel range (no units required) or in eiter any of the following units: keV, eV, Rydberg, Joule, Hertz, Å, nanometer, with the following abbrevations: kev, ev, ryd, j, hz, ang, nm.

Syntax

The following syntax rules apply:

`bin #r #i` - This is the simplest syntax allowed. One needs to give the range, #r, over at least the input data channels one wants to rebin. If one wants to rebin the whole input file the range must be at least the whole range over data channels (but a greater number is also allowed). #i is then the factor by which the data will be rebinned.
`bin [instrument #i1] [region #i2] #r #i` - Here one can also specify the instrument and region to be used in the binning. This syntax is necessary if multiple instruments or regions are

used in the data input.
`bin [instrument #i1] [region #i2] #r #i [unit #a]` - In addition to the above here one can also specify the units in which the binning range is given. The units can be eV, Å, or any of the other units specified above.

Examples

`bin 1:10000 10` - Rebins the input data channel 1:10000 by a factor of 10.
`bin instrument 1 1:10000 10` - Rebins the data from the first instrument as above.
`bin 1:40 10 unit a` - Rebins the input data between 1 and 40 Å by a factor of 10.

3.5 Calculate: evaluate the spectrum

Overview

This command evaluates the current model spectrum. When one or more instruments are present, it also calculates the model folded through the instrument. Whenever the user has modified the model or its parameters manually, and wants to plot the spectrum or display model parameters like the flux in a given energy band, this command should be executed first (otherwise the spectrum is not updated). On the other hand, if a spectral fit is done (by typing the "fit" command) the spectrum will be updated automatically and the calculate command needs not to be given.

Syntax

The following syntax rules apply:
`calc` - Evaluates the spectral model.

Examples

`calc` - Evaluates the spectral model.

3.6 Comp: create, delete and relate spectral components

Overview

In fitting or evaluating a spectrum, one needs to build up a model made out of at least 1 component. This set of commands can create a new component in the model, as well as delete any component. Usually we distinguish two types of spectral components in SPEX.
The *additive* components correspond to emission components, such as a power law, a Gaussian emission line, a collisional ionization equilibrium (CIE) component, etc.

The second class (dubbed here *multiplicative* components for ease) consists of operations to be applied to the additive components. Examples are truly multiplicative operations, such as the Galactic absorption, where the model spectrum of any additive component should be multiplied by the transmission of the absorbing interstellar medium, warm absorbers etc. Other operations contained in this class are redshifts, convolutions with certain velocity profiles, etc.

The user needs to define in SPEX which multiplicative component should be applied to which additive components, and in which order. The order is important as operations are not always communative. This setting is also done with this component command.

If multiple sectors are present in the spectral model or response matrix (see Sect. 2.2) then one has to specify the spectral components and their relation for each sector. The possible components to the model are listed and described in Sect. 4.

Note that the order that you define the components is not important. However, for each sector, the components are numbered starting from 1, and these numbers should be used when relating the multiplicative components to the additive components.

If you want to see the model components and the way they are related, type "model show".

WARNING: *If in any of the commands as listed above you omit the sector number or sector number range, the operation will be done for all sectors that are present. For example, having 3 sectors, the "comp pow" command will define a power law component for each of the three sectors. If you only want to define/delete/relate the component for one sector, you should specify the sector number(s). In the very common case that you have only one sector you can always omit the sector numbers.*

WARNING: *After deleting a component, all components are re-numbered! So if you have components 1,2,3 for example as pow, cie, gaus, and you type "comp del 2", you are left with 1=pow, 2=gaus.*

Syntax

The following syntax rules apply:

`comp [#i:] #a` - Creates a component #a as part of the model for the (optional) sector range #i:
`comp delete [#i1:] #i2:` - Deletes the components with number from range #i2: for sector range (optional) #i1. See also the warning above
`comp relation [#i1:] #i2: #i3,...,#in` - Apply multiplicative components #i3, . . . , #in (numbers) in this order, to the additive components given in the range #i2: of sectors in the range #i1 (optional). Note that the multiplicative components must be separated by a ","

Examples

`comp pow` - Creates a power-law component for modeling the spectrum for all sectors that are present.
`comp 2 pow` - Same as above, but now the component is created for sector 2.
`comp 4:6 pow` - Create the power law for sectors 4, 5 and 6
`com abs` - Creates a Morrison & McCammon absorption component.
`comp delete 2` - Deletes the second created component. For example, if you have 1 = pow, 2 = cie and 3 = gaus, this command delets the cie component and renumbers 1 = pow, 2 = gaus

`comp del 1:2` - In the example above, this will delete the pow and cie components and renumbers now 1 = gaus
`comp del 4:6 2` - If the above three component model (pow, cie, gaus) would be defined for 8 sectors (numbered 1–8), then this command deletes the cie component (nr. 2) for sectors 4–6 only.
`comp rel 1 2` - Apply component 2 to component 1. For example, if you have defined before with "comp pow" and "comp "abs" a power law and galactic absorption, the this command tells you to apply component 2 (abs) to component 1 (pow).
`comp rel 1 5,3,4` - Taking component 1 a power law (pow), component 3 a redshift operation (reds), component 4 galactic absorption (abs) and component 5 a warm absorber (warm), this command has the effect that the power law spectrum is multiplied first by the transmission of the warm absorber (5=warm), then redshifted (3=reds) and finally multiplied by the transmission of our galaxy (4=abs). Note that the order is always from the source to the observer!
`comp rel 1:2 5,3,4` - Taking component 1 a power law (pow), component 2 a gaussian line (gaus), and 3–5 as above, this model applies multiplicative components 5, 3 and 4 (in that orer) to the emission spectra of both component 1 (pow) and 2 (cie).
`comp rel 7:8 1:2 5,3,4` - As above, but only for sectors 7 and 8 (if those are defined).
`comp rel 3 0` - Remove all relations from component 3.

3.7 Data: read response file and spectrum

Overview

In order to fit an observed spectrum, SPEX needs a spectral data file and a response matrix. These data are stored in FITS format, tailored for SPEX (see section 8.2). The data files need not necessarily be located in the same directory, one can also give a pathname plus filename in this command.

WARNING: *Filenames should be entered in the data command without their extension. For the files response.res and spectrum.spo, the data command would look like:* `data response spectrum`.

Syntax

The following syntax rules apply:
`data #a1 #a2` - Read response matrix #a1 and spectrum #a2
`data delete instrument #i` - Remove instrument #i from the data set
`data merge sum #i:` - Merge instruments in range #i: to a single spectrum and response matrix, by adding the data and matrices
`data merge aver #i:` - Merge instruments in range #i: to a single spectrum and response matrix, by averaging the data and matrices
`data save #i #a [overwrite]` - Save data #a from instrument #i with the option to overwrite the existent file. SPEX automatically tags the .spo extension to the given file name. No response file is saved.
`data show` - Shows the data input given, as well as the count (rates) for source and background, the energy range for which there is data the response groups and data channels. Also the integration time and the standard plotting options are given.

Examples

`data mosresp mosspec` - read a spectral response file named mosresp.res and the corresponding spectral file mosspec.spo. Hint, although 2 different names are used here for ease of understanding, it is eased if the spectrum and response file have the same name, with the appropriate extension.
`data delete instrument 1` - delete the first instrument
`data merge aver 3:5` - merge instruments 3–5 into a single new instrument 3 (replacing the old instrument 3), by averaging the spectra. Spectra 3–5 could be spectra taken with the same instrument at different epochs.
`data merge sum 1:2` - add spectra of instruments 1–2 into a single new instrument 1 (replacing the old instrument 1), by adding the spectra. Useful for example in combining XMM-Newton MOS1 and MOS2 spectra.
`data save 1 mydata` - Saves the data from instrument 1 in the working directory under the filename of mydata.spo
`data /mydir/data/mosresp /mydir/data/mosspec` - read the spectrum and response from the directory /mydir/data/

3.8 DEM: differential emission measure analysis

Overview

SPEX offers the opportunity to do a differential emission measure analysis. This is an effective way to model multi-temperature plasmas in the case of continuous temperature distributions or a large number of discrete temperature components.

The spectral model can only have one additive component: the DEM component that corresponds to a multi-temperature structure. There are no restrictions to the number of multiplicative components. For a description of the DEM analysis method see document SRON/SPEX/TRPB05 (in the documentation for version 1.0 of SPEX), Mewe et al. (1994), Mewe et al. (1995) and Kaastra et al. (1996b).

SPEX has 5 different dem analysis methods implemented, as listed shortly below. We refer to the above papers for more details.

1. reg – Regularization method (minimizes second order derivative of the DEM; advantage: produces error bars; disadvantage: needs fine-tuning with a regularization parameter and can produce negative emission measures.

2. clean – Clean method: uses the clean method that is widely used in radio astronomy. Useful for "spiky emission measure distributions.

3. poly – Polynomial method: approximates the DEM by a polynomial in the $\log T - - \log Y$ plane, where Y is the emission measure. Works well if the DEM is smooth.

4. mult – Multi-temperature method: tries to fit the DEM to the sum of Gaussian components as a function of $\log T$. Good for discrete and slightly broadened components, but may not

always converge.

5. gene – Genetic algorithm: using a genetic algorithm try to find the best solution. Advantage: rather robust for avoiding local subminima.Disadvantage: may require a lot of cpu time, and each run produces slightly different results (due to randomization).

In practice to use the DEM methods the user should do the following steps:

1. Read and prepare the spectral data that should be analysed
2. Define the dem model with the "comp dem" command
3. Define any multiplicative models (absorption, redshifts, etc.) that should be applied to the additive model
4. Define the parameters of the dem model: number of temperature bins, temperature range, abundances etc.
5. give the "dem lib" command to create a library of isothermal spectra.
6. do the dem method of choice (each one of the five methods outlined above)
7. For different abundances or parameters of any of the spectral components, first the "dem lib" command must be re-issued!

Syntax

The following syntax rules apply:

`dem lib` - Create the basic DEM library of isothermal spectra
`dem reg auto` - Do DEM analysis using the regularization method, using an automatic search of the optimum regularization parameter. It determines the regularisation parameter R in such a way that $\chi^2(R) = \chi^2(0)[1 + s\sqrt{2/(n - n_{\rm T}}]$ where the scaling factor $s = 1$, n is the number of spectral bins in the data set and $n_{\rm T}$ is the number of temperature components in the DEM library.
`dem reg auto #r` - As above, but for the scaling factor s set to #r.
`dem reg #r` - Do DEM analysis using the regularization method, using a fixed regularization parameter R =#r.
`dem chireg #r1:#r2 #i` - Do a grid search over the regularization parameter R, with #i steps and R distributed logarithmically between #r1 and #r2. Useful to scan the $\chi^2(R)$ curve whenever it is complicated and to see how much "penalty' (negative DEM values) there are for each value of R.
`dem clean` - Do DEM analysis using the clean method
`dem poly #i` - Do DEM analysis using the polynomial method, where #i is the degree of the polynomial
`dem mult #i` - Do DEM analysis using the multi-temperature method, where #i is the number of broad components
`dem gene #i1 #i2` - Do DEM analysis using the genetic algorithm, using a population size given by #i1 (maximum value 1024) and #i2 is the number of generations (no limit, in practice after ∼100 generations not much change in the solution. Experiment with these numbers for your practical case.

`dem read #a` - Read a DEM distribution from a file named #a which automatically gets the extension ".dem". It is an ascii file with at least two columns, the first column is the temperature in keV and the second column the differential emission measure, in units of 10^{64} $\mathrm{m}^{-3}\,\mathrm{keV}^{-1}$. The maximum number of data points in this file is 8192. Temperature should be in increasing order. The data will be interpolated to match the temperature grid defined in the dem model (which is set by the user).
`dem save #a` - Save the DEM to a file #a with extension ".dem". The same format as above is used for the file. A third column has the corresponding error bars on the DEM as determined by the DEM method used (not always relevant or well defined, exept for the regularization method).
`dem smooth #r` - Smoothes a DEM previously determined by any DEM method using a block filter/ Here #r is the full width of the filter expressed in $^{10}\log T$. Note that this smoothing will in principle worsen the χ^2 of the solution, but it is sometimes useful to "wash out" some residual noise in the DEM distribution, preserving total emission measure.

Examples

`dem lib` - create the DEM library
`dem reg auto` - use the automatic regularization method
`dem reg 10.` - use a regularization parameter of $R = 10$ in the regularization method
`dem chireg 1.e-5:1.e5 11` - do a grid search using 11 regularisation parameters R given by 10^{-5}, 10^{-4}, 0.001, 0.01, 0.1, 1, 10, 100, 1000, 10^4, 10^5.
`dem clean` - use the clean method
`dem poly 7` - use a 7th degree polynomial method
`dem gene 512 128` - use the genetic algorithm with a population of 512 and 128 generations
`dem save mydem` - save the current dem on a file named mydem.dem
`dem read modeldem` - read the dem from a file named modeldem.dem
`dem smooth 0.3` - smooth the DEM to a temperature width of 0.3 in $^{10}\log T$ (approximately a factor of 2 in temperature range).

Recommended citation: Mewe et al. (1995).

3.9 Distance: set the source distance

Overview

One of the main principles of SPEX is that spectral models are in principle calculated at the location of the X-ray source. Once the spectrum has been evaluated, the flux received at Earth can be calculated. In order to do that, the distance of the source must be set.

SPEX allows for the simultaneous analysis of multiple sky sectors. In each sector, a different spectral model might be set up, including a different distance. For example, a foreground object that coincides partially with the primary X-ray source has a different distance value.

The user can specify the distance in a number of different units. Allowed distance units are shown in the table below.

The default unit of 10^{22} m is internally used in all calculations in SPEX. The reason is that with

Table 3.3: SPEX distance units

Abbrevation	Unit
spex	internal SPEX units of 10^{22} m (this is the default)
m	meter
au	Astronomical Unit, $1.49597892\,10^{11}$ m
ly	lightyear, $9.46073047\,10^{15}$ m
pc	parsec, $3.085678\,10^{16}$ m
kpc	kpc, kiloparsec, $3.085678\,10^{19}$ m
mpc	Mpc, Megaparsec, $3.085678\,10^{22}$ m
z	redshift units for the given cosmological parameters
cz	recession velocity in km/s for the given cosmological parameters

this scaling all calculations ranging from solar flares to clusters of galaxies can be done with single precision arithmetic, without causing underflow or overflow. For the last two units (z and cz), it is necessary to specify a cosmological model. Currently this model is simply described by H_0, Ω_m (matter density), Ω_Λ (cosmological constant related density), and Ω_r (radiation density). At startup, the values are:

H_0: 70 km/s/Mpc ,

Ω_m: 0.3 ,

Ω_Λ: 0.7 ,

Ω_r: 0.0

i.e. a flat model with cosmological constant. However, the user can specify other values of the cosmological parameters. Note that the distance is in this case the luminosity distance.

Note that the previous defaults for SPEX ($H_0 = 50$, $q_0 = 0.5$) can be obtained by putting $H_0 = 50$, $\Omega_m = 1$, $\Omega_\Lambda = 0$ and $\Omega_r = 0$.

WARNING: *when H_0 or any of the Ω is changed, the luminosity distance will not change, but the equivalent redshift of the source is adjusted. For example, setting the distance first to z=1 with the default H_0=70 km/s/Mpc results into a distance of $2.039\,10^{26}$ m. When H_0 is then changed to 100 km/s/Mpc, the distance is still $2.168\,10^{26}$ m, but the redshift is adjusted to 1.3342.*

WARNING: *In the output also the light travel time is given. This should not be confused with the (luminosity) distance in light years, which is simply calculated from the luminosity distance in m!*

Syntax

The following syntax rules apply to setting the distance:

`distance [sector #i:] #r [#a]` - set the distance to the value #r in the unit #a. This optional distance unit may be omittted. In that case it is assumed that the distance unit is the default SPEX unit of 10^{22} m. The distance is set for the sky sector range #i:. When the optional sector range is omitted, the distance is set for all sectors.
`distance show` - displays the distance in various units for all sectors.
`distance h0 #r` - sets the Hubble constant H_0 to the value #r.

`distance om #r` - sets the Ω_m parameter to the value #r.
`distance ol #r` - sets the Ω_Λ parameter to the value #r.
`distance or #r` - sets the Ω_r parameter to the value #r.

Examples

`distance 2` - sets the distance to 2 default units, i.e. to 2E22 m.
`distance 12.0 pc` - sets the distance for all sectors to 12 pc.
`distance sector 3 0.03 z` - sets the distance for sector 3 to a redshift of 0.03.
`distance sector 2 : 4 50 ly` - sets the distance for sectors 2-4 to 50 lightyear.
`distance h0 50.` - sets the Hubble constant to 50 km/s/Mpc.
`distance om 0.27` - sets the matter density parameter Ω_m to 0.27
`distance show` - displays the distances for all sectors, see the example below for the output format.

```
SPEX> di 100 mpc
 Distances assuming H0 =  70.0 km/s/Mpc, Omega_m = 0.300 Omega_Lambda = 0.700 Omega_r = 0.000
Sector       m      A.U.        ly        pc       kpc       Mpc  redshift        cz   age(yr)
-------------------------------------------------------------------------------------------------
   1 3.086E+24 2.063E+13 3.262E+08 1.000E+08 1.000E+05  100.0000    0.0229    6878.7 3.152E+08
-------------------------------------------------------------------------------------------------
```

3.10 Egrid: define model energy grids

Overview

SPEX operates essentially in two modes: with an observational data set (read using the data commands), or without data, i.e. theoretical model spectra. In the first case, the energy grid neede to evaluate the spectra is taken directly from the data set. In the second case, the user can choose his own energy grid.

The energy grid can be a linear grid, a logarithmic grid or an arbitrary grid read from an ascii-file. It is also possible to save the current energy grid, whatever that may be. In case of a linear or logarithmic grid, the lower and upper limit, as well as the number of bins or the step size must be given.

The following units can be used for the energy or wavelength: keV (the default), eV, Ryd, J, Hz, Å, nm.

When the energy grid is read or written from an ascii-file, the file must have the extension ".egr", and contains the bin boundaries in keV, starting from the lower limit of the first bin and ending with the upper limit for the last bin. Thus, the file has 1 entry more than the number of bins! In general, the energy grid must be increasing in energy and it is not allowed that two neighbouring boundaries have the same value.

Finally, the default energy grid at startup of SPEX is a logarithmic grid between 0.001 and 100 keV, with 8192 energy bins.

Syntax

The following syntax rules apply:

`egrid lin #r1:#r2 #i [#a]` - Create a linear energy grid between #r1 and #r2, in units given by #a (as listed above). If no unit is given, it is assumed that the limits are in keV. The number of energy bins is given by #i

`egrid lin #r1:#r2 step #r3 [#a]` - as above, but do not prescribe the number of bins, but the bin width #r3. In case the difference between upper and lower energy is not a multiple of the bin width, the upper boundary of the last bin will be taken as close as possible to the upper boundary (but cannot be equal to it).
`egrid log #r1:#r2 #i [#a]` - Create a logarithmic energy grid between #r1 and #r2, in units given by #a (as listed above). If no unit is given, it is assumed that the limits are in keV. The number of energy bins is given by #i
`egrid log #r1:#r2 step #r3 [#a]` - as above, but do not prescribe the number of bins, but the bin width (in log E) #r3.
`egrid read #a` - Read the energy grid from file #a.egr
`egrid save #a` - Save the current energy grid to file #a.egr

WARNING: *The lower limit of the energy grid must be positive, and the upper limit must always be larger than the lower limit.*

Examples

`egrid lin 5:38 step 0.02 a` - create a linear energy grid between $5-38$ Å, with a step size of 0.02 Å.
`egrid log 2:10 1000` - create a logarithmic energy grid with 1000 bins between $2-10$ keV.
`egrid log 2:10 1000 ev` - create a logarithmic energy grid with 1000 bins between $0.002-0.010$ keV.
`egrid read mygrid` - read the energy grid from the file mygrid.egr.
`egrid save mygrid` - save the current energy grid to file mygrid.egr.

3.11 Elim: set flux energy limits

Overview

SPEX offers the opportunity to calculate the model flux in a given energy interval for the current spectral model. This information can be viewed when the model parameters are shown (see section 3.22).

For each additive component of the model, the following quantities are listed:

the observed photon number flux (photons $\mathrm{m}^{-2}\,\mathrm{s}^{-1}$) at earth, including any effects of galactic absorption etc.

the observed energy flux ($\mathrm{W}\,\mathrm{m}^{-2}$) at earth, including any effects of galactic absorption etc.

the intrinsic number of photons emitted at the source, not diluted by any absorption etc., in photons s^{-1}.

the intrinsic luminosity emitted at the source, not diluted by any absorption etc., in W.

The following units can be used to designate the energy or wavelength range: keV (the default), eV, Ryd, J, Hz, Å, nm.

Syntax

The following syntax rules apply:

`elim #r1:#r2 [#a]` - Determine the flux between #r1 #r2, in units given by #a, and as listed above. The default at startup is $2-10$ keV. If no unit is given, it is assumed that the limits are in keV.

WARNING: *When new units or limits are chosen, the spectrum must be re-evaluated (e.g. by giving the "calc" comand) in order to determine the new fluxes.*

WARNING: *The lower limit must be positive, and the upper limit must always be larger than the lower limit.*

Examples

`elim 0.5:4.5` - give fluxes between 0.5 and 4.5 keV
`elim 500:4500 ev` - give fluxes between 500 and 4500 eV (same result as above)
`elim 5:38 a` - give fluxes in the 5 to 38 Å wavelength band

3.12 Error: Calculate the errors of the fitted parameters

Overview

This command calculates the error on a certain parameter or parameter range, if that parameter is free (thawn). Standard the 1σ error is calculated, which is equivalent to a 68 % confidence level. So $\Delta\chi^2$ is equal to 1 for a single parameter error. The $\Delta\chi^2$ value can be set, such that for instance 90 % errors are determined (see Table 3.4 for $\Delta\chi^2$ values with their corresponding confidence levels).

SPEX determines the error bounds by iteratively modifying the parameter of interest and calculating χ^2 as a function of the parameter. During this process the other free parameters of the model may vary. The iteration stops when $\chi^2 = \chi^2_{\rm min} + \Delta\chi^2$, where $\Delta\chi^2$ is a parameter that can be set separately. The iteration steps are displayed. It is advised to check them, because sometimes the fit at a trial parameter converges to a different solution branch, therefore creating a discontinuous jump in χ^2. In those situations it is better to find the error bounds by varying the search parameter by hand.

Note that SPEX remembers the parameter range for which you did your lat error search. This saves you often typing in sector numbers or component numbers if you keep the same spectral component for your next error search.

If the error search reveals a new minimum in χ^2 space that was not found in the fit, SPEXwill save the parameters of this new minimum in the file `spex_lower_chi.com`. After the error search these parameters can be set through the command `log exe spex_lower_chi`, in order to direct the model to the new minimum. Note that in the file `spex_lower_chi.com` the parameter for which the error was calculated is "frozen".

WARNING: *A parameter must have the status "thawn" in order to be able to determine its errors.*

WARNING: *The first time that you use the error command, you need to provide the sector number before it is remembered by SPEX. For example,* `error 1 1 norm`

Syntax

The following syntax rules apply:

`error [[[#i1:] #i2:] #a:]` - Determine the error bars for the parameters specified by the sector range #i1: (optional), component range #i2 (optional) and parameter range #a: (optional). If not specified, the range for the last call will be used. On startup, this is the first parameter of the first component of the first sector.
`error dchi #r` - This command changes the $\Delta\chi^2$, to the value #r. Default at startup and recommended value to use is 1, for other confidence levels see Table 3.4.
`error start #r` - This command gives an initial guess of the error bar, from where to start searching the relevant error. This can be helpful for determining the errors on normalization parameters, as otherwise SPEX may from a rather small value. To return to the initial situation, put #r=0 (automatic error search).

Table 3.4: $\Delta\chi^2$ as a function of confidence level, P, and degrees of freedom, ν.

	ν					
P	1	2	3	4	5	6
68.3%	1.00	2.30	3.53	4.72	5.89	7.04
90%	2.71	4.61	6.25	7.78	9.24	10.6
95.4%	4.00	6.17	8.02	9.70	11.3	12.8
99%	6.63	9.21	11.3	13.3	15.1	16.8
99.99%	15.1	18.4	21.1	13.5	25.7	27.8

Examples

`error norm` - Find the error for the normalization of the current component
`error 2:3 norm:gamm` - determines the error on all free parameters between "norm" and "gamm" for components 2:3
`error start 0.01` - Start calculating the error beginning with an initial guess of 0.01
`error dchi 2.71` - Calculate from now onwards the 90 % error, for 1 degree of freedom. (Not recommended, use standard 68 % errors instead!)

3.13 Fit: spectral fitting

Overview

With this command one fits the spectral model to the data. Only parameters that are thawn are changed during the fitting process. Options allow you to do a weighted fit (according to the model or the data), have the fit parameters and plot printed and updated during the fit, and limit the number of iterations done.

At the moment SPEX uses two types of fit statistic, χ^2 and C-stat. We first treat the χ^2 statistic, because historically that has been most widely used. However, in the present version of SPEX C-stat is the default because in the far majority of the cases it gives more robust results.

Chi-squared fitting

First we make a few remarks about proper data weighting. χ^2 is usually calculated as the sum over all data bins i of $(N_i - s_i)^2/\sigma_i^2$, i.e.

$$\chi^2 = \sum_{i=1}^{n} \frac{(N_i - s_i)^2}{\sigma_i^2}, \tag{3.1}$$

where N_i is the observed number of source plus background counts, s_i the expected number of source plus background counts of the fitted model, and for Poissonian statistics usually one takes $\sigma_i^2 = N_i$. Take care that the spectral bins contain sufficient counts (either source or background), recommended is e.g. to use at least ∼10 counts per bin. If this is not the case, first rebin the data set whenever you have a "continuum" spectrum. For line spectra you cannot do this of course without loosing important information! Note however that this method has inaccuracies if N_i is less than ∼100.

Wheaton et al. (1995) have shown that the classical χ^2 method becomes inaccurate for spectra with less than ∼ 100 counts per bin. This is *not* due to the approximation of the Poisson statistic by a normal distribution, but due to using the *observed* number of counts N_i as weights in the calculation of

χ^2. Wheaton et al. (1995) showed that the problem can be resolved by using instead $\sigma_i^2 = s_i$, i.e. the *expected* number of counts from the best fit model.

The option "fit weight model" allows to use these modified weights. By selecting it, the expected number of counts (both source plus background) of the current spectral model is used onwards in calculating the fit statistic. Wheaton et al. (1995) suggest to do the following 3-step process, which we also recommend to the user of SPEX who uses this option:

1. first fit the spectrum using the data errors as weights (the default of SPEX).
2. After completing this fit, select the "fit weight model" option and do again a fit
3. then repeat this step once more by again selecting "fit weight model" in order to replace s_i of the first step by s_i of the second step in the weights. The result should now have been converged (under the assumption that the fitted model gives a reasonable description of the data, if your spectral model is way off you are in trouble anyway!).

C-stat

There is yet another option to try for spectral fitting with low count rate statistics and that is maximum likelyhood fitting. It can be shown that a good alternative to χ^2 in that limit is

$$C = 2\sum_{i=1}^{n} s_i - N_i + N_i \ln(N_i/s_i). \tag{3.2}$$

This is strictly valid in the limit of Poissonian statistics. If you have a background subtracted spectrum, take care that the subtracted number of background counts is properly stored in the spectral data file, so that raw number of counts can be reconstructed.

This statistic was originally proposed in some other form by Cash (1979) and in the present form sometimes attributed to Castor. However, it appears that it was already introduced and well explained by Baker & Cousins (1984).

WARNING: *Note that for a spectrum with many counts per bin $C \rightarrow \chi^2$, but if the predicted number of counts per bin is small, the expected value for C can be substantially smaller than the number of bins n.*

To help the user to see if a C-value corresponds to an acceptable fit, SPEX gives, after spectral fitting, the expected value of C and its r.m.s. spread, based on the best-fit model. Both quantities are simply determined by adding the expected contributions and their variances over all bins. See Kaastra (2017) for more details.

The expected value C_e for C in a bin i and its variance C_v are given by:

$$C_\mathrm{e} = 2\sum_{k=0}^{\infty} P_k(\mu)(\mu - k + k\ln(k/\mu)), \tag{3.3}$$

$$S_\mathrm{v} = 4\sum_{k=0}^{\infty} P_k(\mu)(\mu - k + k\ln(k/\mu))^2, \tag{3.4}$$

$$C_\mathrm{v} = S_\mathrm{v} - C_\mathrm{e}^2, \tag{3.5}$$

with $P_k(\mu)$ the Poisson distribution:

$$P_k(\mu) = \mathrm{e}^{-\mu}\mu^k/k! \tag{3.6}$$

and μ the expected number of counts. We show both quantities in Fig. 3.1.

WARNING: *For a proper use of C-stat, it is needed that the background (if present) is also a model for the background, not a scaled background observation. Unfortunately, the fast majority of instrument software packeges provide spectra with such a scaled (and therefore noisy) background). By experimenting it can be shown that in situations where the source is (much) weaker that the subtracted background, this*

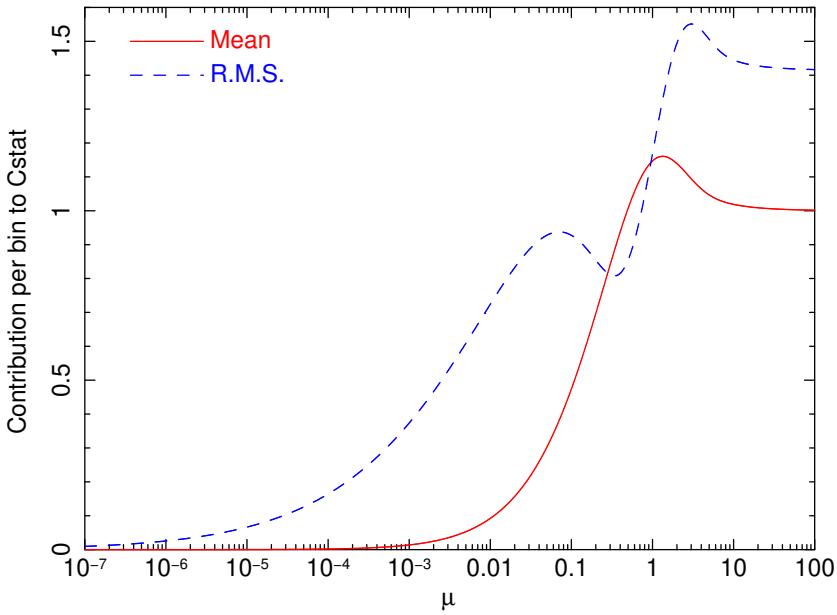

Figure 3.1: Expected value of the contribution per bin to C, and its r.m.s. uncertainty, as a function of the mean expected number of counts μ.

can give bias in the fitted flux (it will be over-estimated). Rebinning the spectrum resolves the problem (because it is some kind of smoothing) but at the expense of spectral resolution. This is undesired. We therefore offer an auxiliary program called backfilter *that can filter the subtracted background. If works on a .spo file and creates an improved .spo file. See the documentation of* backfilter *for more details.*

W-stat

The above problem is mitigated in the Xspec package by introducing the so-called W-statistic. See the Xspec manual for more details. We provide here the option to fit using W-stat for compatibility reasons, WARNING: *but we do not recommend to use it*, but instead use C-stat with background filtering (see above) where needed.

The W-stat first calculates a background estimate for each bin using maximum likelihood techniques. This background depends on the number of counts in the background region, the number of counts in the source region, the predicted number of source counts from the spectral model, and the exposure times of the source region and background region (or, equivalently, incorporating any background area scaling ratio). Using these background estimates, W-stat is then evaluated, and this can be used in the algorithm to find the best-fit set of source parameters.

The Xspec manual notes that for weak sources it can generate an obviously wrong best fit", and they advice to rebin to at least one count per bin to mitigate. This however may degrade the spectral resolution too much. Moreover, for a simple case (blackbody fit to an isolated neutron star), we found that the fitting procedure can show non-monotonous behaviour of W-stat versus iteration, with annoying oscillatory behaviour. Also, a full fit with error search of that spectrum required four times more model evaluations compared with C-stat fitting with filtered background.

Syntax

The following syntax rules apply:

`fit` - Execute a spectral fit to the data.
`fit print #i` - Printing the intermediate results during the fitting to the screen for every n-th step,

with n=#i (most useful for $n = 1$). Default value: 0 which implies no printing of intermediate steps.
`fit iter #i` - Stop the fitting process after #i iterations, regardless convergence or not. This is useful to get a first impression for very cpu-intensive models. To return to the default stop criterion, type fit iter 0.
`fit weight model` - Use the current spectral model as a basis for the statistical weight in all subsequent spectral fitting.
`fit weight data` - Use the errors in the spectral data file as a basis for the statistical weight in all subsequent spectral fitting. This is the default at the start of SPEX.
`fit method classical` - Use the classical Levenberg-Marquardt minimisation as the fitting method.
`fit statistic chi2` - Use the χ^2 statistic for the minimisation.
`fit statistic cstat` - Use the C-statistics for the minimisation. This is the default at start-up.
`fit statistic wstat` - Use the W-statistics for the minimisation.

Examples

`fit` - Performs a spectral fit. At the end the list of best fit parameters is printed, and if there is a plot this will be updated.
`fit print 1` - If followed by the above fit command, the intermediate fit results are printed to the screen, and the plot of spectrum, model or residuals is updated (provided a plot is selected).
`fit iter 10` - Stop the after 10 iterations or earlier if convergence is reached before ten iterations are completed.
`fit iter 0` - Stop fitting only after full convergence (default).
`fit weight model` - Instead of using the data for the statistical weights in the fit, use the current model.
`fit weight data` - Use the data instead for the statistical weights in the fit.
`fit method clas` - Use the classical Leveberg-Marquardt method to find minima. At this moment the only option.
`fit statistic chi2` - Switch from C-statistics to χ^2.
`fit statistic cstat` - Switch back to C-statistics.

3.14 Ibal: set type of ionisation balance

Overview

For the plasma models, different ionisation balance calculations are possible.

Currently, the default set is Arnaud & Rothenflug (1985) for H, He, C, N, O, Ne, Na, Mg, Al, Si, S, Ar, Ca and Ni, with Arnaud & Raymond (1992)) for Fe. Table 3.5 lists the possible options.

Table 3.5: Ionisation balance modes

Abbrevation	Reference
reset	default (=u17)
ar85	Arnaud & Rothenflug (1985)
ar92	Arnaud & Raymond (1992) for Fe, Arnaud & Rothenflug (1985) for the other elements
bryans09	Bryans et al. (2009)
u17	Urdampilleta et al. (2017) (default)

Syntax

The following syntax rules apply:

`ibal #a` - Set the ionisation balance to set #a with #a in the table above.

`ibal show` - Show the currently active ionisation balance.

Examples

`ibal reset` - Take the standard ionisation balance
`ibal ar85` - Take the Arnaud & Rothenflug ionisation balance

3.15 Ignore: ignoring part of the spectrum

Overview

If one wants to ignore part of a data set in fitting the input model, as well as in plotting this command should be used. The spectral range one wants to ignore can be specified as a range in data channels or a range in wavelength or energy. Note that the first number in the range must always be smaller or equal to the second number given. If multiple instruments are used, one must specify the instrument as well. If the data set contains multiple regions, one must specify the region as well. So per instrument/region one needs to specify which range to ignore. The standard unit chosen for the range of data to ignore is data channels. To undo ignore, see the use command (see section 3.31).

The range to be ignored can be specified either as a channel range (no units required) or in either any of the following units: keV, eV, Rydberg, Joule, Hertz, Å, nanometer, with the following abbrevations: kev, ev, ryd, j, hz, ang, nm.

Syntax

The following syntax rules apply:

`ignore [instrument #i1] [region #i2] #r` - Ignore a certain range #r given in data channels of instrument #i1 and region #i2. The instrument and region need to be specified if there are more than 1 data sets in one instrument data set or if there are more than 1 data set from different instruments.
`ignore [instrument #i1] [region #i2] #r unit #a` - Same as the above, but now one also specifies the unites #a in which the range #r of data points to be ignored are given. The units can be either Å(ang) or (k)eV.

Examples

`ignore 1000:1500` - Ignores data channels 1000 till 1500.
`ignore region 1 1000:1500` - Ignore data channels 1000 till 1500 for region 1.
`ignore instrument 1 region 1 1000:1500` - Ignores the data channels 1000 till 1500 of region 1 from the first instrument.
`ignore instrument 1 region 1 1:8 unit ang` - Same as the above example, but now the range is specified in units of Å instead of in data channels.
`ignore 1:8 unit ang` - Ignores the data from 1 to 8 Å, only works if there is only one instrument and one region included in the data sets.

3.16 Ion: select ions for the plasma models

Overview

For the plasma models, it is possible to include or exclude specific groups of ions from the line calulations. This is helpful if a better physical understanding of the (atomic) physics behind the spectrum is requested.

Currently these settings *only* affect the line emission; in the calculation of the ionisation balance as well as the continuum always all ions are taken into account (unless of course the abundance is put to zero).

A new quicklook mode is introduced in SPEX 3.0. This mode can greatly reduce computation time by excluding the atomic levels of outer shells that barely affect the obtained spectrum. The maximum quantum numbers n and l of Hydrogen-like ions are provided in Table 3.6.

WARNING: *This mode will not work for CX model, since electron captured by charge exchange usually populate the outer shells.*

Table 3.6: Preset maximum n and l for quicklook mode (H-like)

Ion	max. n	max. l	Ion	max. n	max. l	Ion	max. n	max. l
C VI	16	3	N VII	16	3	O VIII	16	5
F IX	2	1	Ne X	16	4	Na XI	9	2
Mg XII	16	4	Al XIII	9	2	Si XIV	16	4
P XV	5	1	S XVI	16	4	Cl XVII	4	1
Ar XVIII	13	2	K XIX	4	1	Ca XX	9	2
Sc XXI	2	1	Ti XXII	4	1	V XXIII	2	1
Cr XXIV	5	1	Mn XXV	4	1	Fe XXVI	16	4
Co XXVII	2	1	Ni XXVIII	8	2	Cu XXIX	2	1
Zn XXX	2	1						

Syntax

The following syntax rules apply:

`ions show` - Display the list of ions currently taken into account
`ions use all` - Use all possible ions in the line spectrum
`ions use iso #i:` - Use ions of the isoelectronic sequences indicated by #i: in the line spectrum
`ions use z #i:` - Use ions with the atomic numbers indicated by #i: in the line spectrum
`ions use ion #i1 #i2:` - Use ions with the atomic number indicated by #i1 and ionisation stage indicated by #i2: in the line spectrum
`ions ignore all` - Ignore all possible ions in the line spectrum
`ions ignore iso #i:` - Ignore ions of the isoelectronic sequences indicated by #i: in the line spectrum
`ions ignore z #i:` - Ignore ions with the atomic numbers indicated by #i: in the line spectrum
`ions ignore ion #i1 #i2:` - Ignore ions with the atomic number indicated by #i1 and ionisation stage indicated by #i2: in the line spectrum
`ions nmax all #i:` - Set maximum n for all ions
`ions nmax iso #i1: #i2` - Set maximum n to #i2 for isoelectronic sequence indicated by #i1
`ions nmax z #i1: #i2` - Set maximum n to #i2 for atomic number indicated by #i1
`ions nmax ion #i1 #i2: #i3` - Set maximum n to #i3 for atomic number indicated by #i1 and ionisation stage indicated by #i2.
`ions lmax all #i:` - Set maximum l for all ions
`ions lmax iso #i1: #i2` - Set maximum l to #i2 for isoelectronic sequence indicated by #i1
`ions lmax z #i1: #i2` - Set maximum l to #i2 for atomic number indicated by #i1
`ions lmax ion #i1 #i2: #i3` - Set maximum l to #i3 for atomic number indicated by #i1 and ioni-

sation stage indicated by #i2.
`ions old all` - Force the old calculation for all ions
`ions old iso #i1:` - Force the old calculation for the isoelectronic sequence indicated by #i1
`ions old z #i1:` - Force the old calculation for atomic number indicated by #i1
`ions old ion #i1 #i2:` - Force the old calculation for atomic number indicated by #i1 and ionisation stage indicated by #i2.
`ions new all` - Force the new calculation for all ions
`ions new iso #i1:` - Force the new calculation for the isoelectronic sequence indicated by #i1
`ions new z #i1:` - Force the new calculation for atomic number indicated by #i1
`ions new ion #i1 #i2:` - Force the new calculation for atomic number indicated by #i1 and ionisation stage indicated by #i2.

Examples

`ions ignore all` - Do not take any line calculation into account
`ions use iso 3` - Use ions from the $Z = 3$ (Li) isoelectronic sequence
`ions use iso 1:2` - Use ions from the H-like and He-like isoelectronic sequences
`ions ignore z 26` - Ignore all iron ($Z = 26$) ions
`ions use ion 6 5:6` - Use C V to C VI
`ions show` - Display the list of ions that are used
`ions ql` - Quicklook mode on
`ions old ion 6 6` - Use old calculation for C VI
`ions nmax ion 26 25 5` - Set maximum principal quantum number for Fe XXV to $n = 5$.
`ions lmax ion 26 25 3` - Set maximum angular momentum quantum number for Fe XXV to $\ell = 3$.

3.17 Log: Making and using command files

Overview

In many circumstances a user of SPEX wants to repeat his analysis for a different set of parameters. For example, after having analysed the spectrum of source A, a similar spectral model and analysis could be tried on source B. In order to facilitate such analysis, SPEX offers the opportunity to save the commands that were used in a session to an ascii-file. This ascii-file in turn can then be read by SPEX to execute the same list of commands again, or the file may be edited by hand.

The command files can be nested. Thus, at any line of the command file the user can invoke the execution of another command file, which is executed completely before execution with the current command file is resumed. Using nested command files may help to keep complex analyses manageable, and allow the user easy modification of the commands.

In order to facilitate the readability of the command files, the user can put comment lines in the command files. Comment lines are recognized by the first character, that must be #. Also blank lines are allowed in the command file, in order to enhance (human) readability.

Saving commands to file

After the command `log save #a` is given on the SPEX command line (`#a` is the file name), all the following commands entered by the user (NOT the ones before) are stored on the command file until the `log close out` command is given. Exceptions are the commands read from a nested command file (it is not necessary to save these commands, since they are already in the nested command file). Also, help calls and the command to open the file ("log save #a") are not stored.

All commands are expanded to the full keywords before they are saved in the file. However, for execution this is not important, since the interpreter can read both abbrevated and full keywords.

Saving the commands to the command file can be stopped by giving the command `log close save`. The file with the saved commands is closed and remains at the specified path. SPEX will automatically append the extension ".com" to the filename.

Saving output to file

It is also possible to store all the output that is printed on the screen to a file. This is usefull for long sessions or for computer systems with a limited screen buffer. The output saved this way could be inspected later by (other programs of) the user. It is also useful if SPEX is run in a kind of batch-mode. The command to save the output is `log out #a`, where `#a` should be the filename without extension. SPEX will automatically append the extension ".out" to the filename.

Executing commands from file

ASCII files with the .com extension containing SPEX commands can be executed in SPEX using the command `log execute #a`, where `#a` stands for the file name without the ".com" extension. When a command file is read and the end of file is reached, the text "Normal end of command file encountered" is printed on the screen, and execution of the calling command file is resumed, or if the command file was opened from the terminal, control is passed over to the terminal again.

For example, the user may have a command file named *run* which does his entire analysis. This command file might start with the line "log exe mydata" that will run the command file *mydata* that contains all information regarding to the data sets read, further data selection or binning etc. This could be followed by a second line in *run* like "log exe mymodel" that runs the command file *mymodel* which could contain the set-up for the spectral model and/or parameters. Also, often used plot settings (e.g. stacking of different plot types) could easily placed in separate command files.

Syntax

The following syntax rules apply for command files:

`log exe #a` - Execute the commands from the file #a. The suffix ".com" will be automatically appended to this filename.
`log save #a [overwrite] [append]` - Store all subsequent commands on the file #a. The suffix ".com" will be automatically appended to this filename. The optional argument "overwrite" will allow to overwrite an already existing file with the same name. The argument "append" indicates that if the file already exists, the new commands will be appended at the end of this file.
`log close save` - Close the current command file where commands are stored. No further commands will be written to this file.
`log out #a [overwrite] [append]` - Store all subsequent screen output on the file #a. The suffix ".out" will be automatically appended to this filename. The optional argument "overwrite" will allow to overwrite an already existing file with the same name. The argument "append" indicates that if the file already exists, the new output will be appended at the end of this file.
`log close output` - Close the current ascii file where screen output is stored. No further output will be written to this file.

Examples

`log save myrun` - writes all subsequent commands to a new file named "myrun.com". However, in case the file already exists, nothing is written but the user gets a warning instead.
`log save myrun append` - as above, but appends it to an existing file

`log save myrun overwrite` - as above, but now overwrites without warning any already existing file with the same name.
`log close save` - close the file where commands are stored.
`log exe myrun` - executes the commands in file myrun.com.
`log output myrun` - writes all subsequent output to file myrun.out.
`log close output` - closes the file above.

3.18 Menu: Menu settings

Overview

When command lines are typed, it frequently happens that often the first keywords are identical for several subsequent lines. This may happen for example when a plot is edited. SPEX offers a shortcut to this by using the menu command.

Syntax

The following syntax rules apply:

`menu none` - Quit the current menu text settings (i.e., return to the default spex prompt.
`menu text #a` - For all following commands, the text string #a will be appended automatically before the following commands

Examples

`menu text plot` - All following commants will get the "plot" keyword put in front of them. So if the next commant would be "plot dev xs" it is sufficient to type "dev xs" instead.
`menu none` - Return to the normal SPEX prompt.
`menu text "par 1 2"` - All following commands wil get the "par 1 2" keywords put in front of them. The next command could be "t val 4.", which will be expanded to the full "par 1 2 t val 4." to set the temperature of sector 1, component 2 to 4 keV. Note that here the text has three keywords (par, 1, 2) and hence it has to be put between "", to indicate that it is a single text string. If there is only one keyword, these "" are not necessary.

3.19 Model: show the current spectral model

Overview

This commands prints the current spectral model, for each sector, to the screen. The model is the set of spectral components that is used, including all additive and multiplicative components. For all additive components, it shows in which order the multiplicative components are applied to the additive (emitted) components. See Sect. 3.6 for more details.

Syntax

The following syntax rules apply:

`model show` - Prints the model for all sectors to the screen.
`model show #i` - Prints the model for sector #i to the screen.

Examples

`model show 2` - Prints the model for the second sector

3.20 Multiply: scaling of the response matrix

Overview

This command multiplies (a component of) the response matrix by a constant.

WARNING: *If this command is repeated for the same component then the original response matrix is changed by the multiplication of the constants. For example, after multiplying the response matrix by a factor of 2, the original matrix is recovered by multiplying the intermediate result by 0.5.*

WARNING: *The instrument number must be given in this command even if you use only a single instrument.*

Syntax

The following syntax rules apply:

`multiply #i1 [component #i2] #r` - Multiplies the response matrix of component #i2 of instrument #i1 by a constant #r.

Examples

`multiply 1 3.5` - Multiplies the response matrix from instrument 1 by a factor of 3.5.
`multiply 1 component 2 3.5` - Multiplies the second component of instrument 1 by the constant 3.5.

3.21 Obin: optimal rebinning of the data

Overview

This command rebins (a part of) the data (thus both the spectrum and the response) to the optimal bin size given the statistics of the source as well as the instrumental resolution. This is recommended to do in **all** cases, in order to avoid oversampling of the data. The theory and algorithms used for this rebinning are described in detail in Chap. 8. A simple cartoon of this is: binning to 1/3 of the FWHM, but the factor of 1/3 depends weakly upon the local count rate at the given energy and the number of resolution elements. The better the statistics, the smaller the bin size.

Syntax

The following syntax rules apply:

`obin #i1:` - Simplest command allowed. #i1: is the range in data channels over which the binning needs to take place.
`obin #r1: #i: unit #a` - The same command as above, except that now the ranges over which the data is to be binned (#r1:) are specified in units (#a) different from data channels. These units can be eV, keV, Å, as well as in units of Rydberg (ryd), Joules (j), Hertz (hz) and nanometers (nm).
`obin [instrument #i1:] [region #i2:] #i3:` - Here #i3: is the same as #i1: in the first command. However, here one can specify the instrument range #i1: and the region range #i2: as well, so

that the binning is done only for one given data set.
`obin [instrument #i1:] [region #i2:] #r1: [unit #a]` - This command is the same as the above, except that here one can specify the range over which the binning should occur in the units specified by #a. These units can be eV, Å, keV, as well as in units of Rydberg (ryd), Joules (j), Hertz (hz) and nanometers (nm).

Examples

`obin 1:10000` - Optimally bins the data channels 1:10000.
`obin 1:4000 unit ev` - Does the same as the above, but now the data range to be binned is given in eV, from 1−4000 eV, instead of in data channels.
`obin instrument 1 region 1 1:19 unit a` - Bins the data from instrument 1 and region 1 between 1 and 19 Å.

3.22 Par: Input and output of model parameters

Overview

This command is used as an interface to set or display the parameters of the spectral model. Each model parameter (like temperature T, abundance, normalization etc.) has a set of attributes, namely its value, status, range and coupling. This is illustrated by the following example. Assume we have a spectral model consisting of a thermal plasma. Consider the silicon abundance (acronym in the model: 14). It can have a value (for example 2.0, meaning twice the solar abundance). Its status is a logical variable, true (thawn) if the parameter can be adjusted during the spectral fitting proces or false (frozen) if it should be kept fixed during the spectral fit. The range is the allowed range that the parameter can have during the spectral fit, for example 0–1000 (minimum – maximum values). This is useful to set to constrain a spectral search to a priori "reasonable" values (e.g. abundances should be non-negative) or "safe" values (SPEX could crash for negative temperatures for example). Finally the coupling allows you to couple parameters, for example if the Si abundance is coupled to the Fe abundance, a change in the Fe abundance (either defined by the user or in a spectral fitting process) will result automatically in an adjustment of the Si abundance. This is a useful property in practical cases where for example a spectrum has strong Fe lines and weaker lines from other species, but definitely non-solar abundances. In this case, one may fit the Fe abundance, but the statistics for the other elements may not be sufficient to determine their abundance accurately; however a priori insight might lead you to think that it is the global metallicity that is enhanced, and therefore you would expect the Si abundance to be enhanced in the same way as Fe.

With the par command, you can set for each parameter individually or for a range of parameters in the same command, their attributes (value, status, range and coupling). Also, you can display the parameters on the screen or write them on a SPEX command file, which allows you to start another run with the current set of parameters.

When setting parameters, you can also specify the sector (range) and component (range). For your first call, it is assumed that you refer to sector 1, component 1, first parameter. In all subsequent calls of the parameter command, it is assumed that you refer to the same sector(s) and component(s) as your last call, unless specified differently. This is illustrated by the following example. Suppose you have the following model: power law (component 1), blackbody (component 2) and RGS line broadening (lpro, component 3). If you type in the commands in that order, this is what happens:

- par val 2 – sets the norm (= first parameter) of the power law to value 2
- par gam val 3 – sets the photon index of the power law to value 3
- par 2 t val 2.5 – sets the temperature of the blackbody to 2.5 keV

- par norm val 10 – sets the norm of the blackbody to 10
- par 1:2 norm v 5 – sets the norm of both the PL and BB to 5
- par val 4.2 – sets the norm of both the PL and BB to 4.2
- par 3 file avalue myprofile.dat – sets for the LPRO component the file name with the broadening kernel to myprofile.dat. Note that the command 'value' changes here to 'avalue' to indicate that the parameter is an ascii string.

Instrument normalisations can be thawned by entering the instrument number as a negative sector number. For example, freeing the instrument normalisation of instrument 2 is done with the command `par -2 1 norm stat thawn`. The second value in the command (1) is the region number. Therefore, freeing the normalisation of the third region of the fourth instrument is done with the command: `par -4 3 norm stat thawn`.

Syntax

The following syntax rules apply:

`par [#i1:] [#i2:] [#a1:] avalue #a2` - Assign the value #a2 to the parameters specified by the (range of) name(s) #a1 of component range #i2: of sectors #i1:. The sector (range), component (range) and parameter name (range) are optional. If #a2 should be an empty text string, specify the value "none" (without quotes, and all four characters in lowercase). If they are not specified, the sector(s), component(s) and parameter(s) of the last call are used. This command containing the word "avalue" holds for input of text strings. For the input of real numbers "avalue" is replaced by "value" and #a2.
`par [#i1:] [#i2:] [#a:] value #r` - Assign the value #r to the parameters specified by the (range of) name(s) #a of component range #i2: of sectors #i1:. The sector (range), component (range) and parameter name (range) are optional. If not specified, the sector(s), component(s) and parameter(s) of the last call are used.
`par [#i1:] [#i2:] [#a:] status #l` - As above but for the status of a (range of) parameters; #l = T (true, thawn) means a parameter that can be adjusted during a spectral fit, #l = F (false, froze, fixed) means a parameter that will remain fixed during the spectral fitting.
`par [#i1:] [#i2:] [#a:] range #r1:#r2` - As above but fore the allowed range of the parameter. #r1 denotes the lower limit and #r2 denotes the upper limit. Both should be specified.
`par [#i1:] [#i2:] [#a1:] couple [#i3:] [#i4:] #a2:` - Here #i1:, #i2:, #a1: are the sector(s), component(s) and parameter(s) that are to be coupled. The parameter (s) to which they are coupled are specified by #a2: and the optional #i3:, #i4:. If ranges are used, take care that the number of sectors, components and parameters on the right and left match.
`par [#i1:] [#i2:] [#a1:] couple [#i3:] [#i4:] #a2: factor #r` - As above, but couple using a scaling factor #r.
`par [#i1:] [#i2:] #a: decouple` - Decouples the parameter(s) #a: of (optional) components #i2: of of (optional) sector(s) #i1:. Inverse operation of the couple command.
`par show [free]` - Shows on the display screen all parameters of all components. If the optional keyword free is given, shows only parameters with fit status "T" (true, thawn), i.e., the free parameters in a spectral fit.
`par show corr #l` - Display the correlations between parameters after fitting if this flag is true (default)
`par write #a [overwrite]` - Writes all parameters to a SPEX command file #a. #a should be specified without extension, SPEX automatically adds the .com extension. If the optional overwrite command is given, then the file #a will be overwritten with the new values.

Examples

`par val 10` - Sets the value of the current parameter to 10.
`par t val 4` - Sets the parameter named "t" to 4.

`par 06:28 value 0.3` - Sets the parameters named 06:28 (for example the abundances of C ($Z = 6$), N ($Z = 7$), ... Ni ($Z = 28$)) to the value 0.3.
`par 2 nh val 0.01` - Sets the parameter named "nh" of component 2 to the value 0.01
`par 1 1 norm value 1E8` - Sets parameter named "norm" of component 1 of sector 1 to the value 10^8.
`par file avalue myfile.with.data` - sets the ascii-type parameter named "file" to the value "myfile.with.data" (without quotes).
`par file avalue none` - sets the ascii-type parameter named "file" to the value "" (i.e. an empty string)
`par status frozen` - Sets the fit status of the current parameter to "frozen" (fixed).
`par 1 3 t stat thawn` - Specifies that parameter "t" of the third component of the model for sector 1 should be left free (thawn) during spectral fitting.
`par 2 3 gamm range 1.6:1.9` - Limit parameter "gamm" of component 3 of sector 2 to the range 1.6 – 1.9.
`par norm range -1E8 1E8` - Set the range for the parameter "norm" between -10^8 and $+10^8$. This command is necessary if for example the model is a delta line used to mimick an absorption line, which normally has a default minimum value of 0. (for an emission line).
`par 1 1 norm couple 2 1 norm` - Couples the norm of the first component for the first sector to the norm of the first component of the model for the second sector.
`par 1 1 norm couple 2 1 norm factor 3` - The same command as the above, but now the norm of the first component in the model for sector 1 is 3 times the norm of the first component of the model for sector 2. For example, if the norm of the 1st component of sector 2 gets the value 40, then the norm of the 1st component of sector 1 will automatically be updated to a value of $3 \times 40 = 120$.
`par 3:5 02:30 couple 1 02:30` - Couples the abundances of He–Zn of components 3, 4 and 5 to the He–Zn abundances of component 1.
`par norm decouple` - Decouples the norm of the current component from whatever it was coupled to.
`par -2 1 norm stat thawn` - Free the instrument normalisation of instrument 2 and region 1.
`par show` - Shows all the parameters of the current model for all sectors and how they are coupled (if applicable). For each parameter it shows the value, status, range and coupling information, as well as info on its units etc. It also shows the fluxes and restframe luminosities of the additive components, photon flux (phot/m**2/s) and energy flux (W/m**2) are the values observed at Earth (including any transmission effects like Galactic absorption or intrinsic absoprtion that have been taken into account), and the nr. of photons (photons/s) and luminosity (W) are all as emitted by the source, without any attenuation.
`par show free` - As above, but only displays the parameters that have the status thawn.
`par write mypar overwrite` - SPEX writes all parameters for the model to a file named mypar.com. Any previously existing file mypar.com is overwritten.
`par write mypar` - Same command as above, but now a new file is created. If there already exists a file with the same filename SPEX will give an error message.

3.23 Plot: Plotting data and models

Overview

The plot command cause the plot to be (re)drawn on the graphics device. Multiple graphics devices can be defined in one SPEX session. For example, a plot can be sent to both a postscript and a xs device.

A user can also set the number of plot frames in the currently active device(s), e.g. the data and model can be displayed in one frame while the residuals can be displayed in a frame below. The user has to specify what is to be plotted in in each frame.

In each plot frame, there can be different sets that can be specified by the "set" keyword in the plot command. For example, for the plot type data, each set corresponds to the spectrum of an instrument or region; with this option it is therefore possible to give different instruments different colours or plot symbols, etc., or just to omit one of the instruments from the plot.

The plot command is also used to select the axis plot units and scales as well as character commands like font style, line weights, plot colours, etc. Finally, the plot command can also be used to dump the data to an ascii file.

WARNING: *To make a plot, always start with a "plot device" command to open any plot device you wish. Next select a plot type using "plot type". After this you can give any plot commands to modify or display the plot.*

WARNING: *Some of the more fancy plotting options, like adding labels to a plot, are broken. To make more sophisticated plots, it is advisable to save the plot to a QDP file with 'plot adum' and adapt them through QDP (Ftools). We intent to update the plotting system in a future version of SPEX.*

Syntax

The following syntax rules apply for plot:

`plot` - (Re)draws the plot on the graphics device. Take care to open at least one device first (with a "plot dev" command)
`plot frame new` - Creates a new additional plot frame, where an other plot type can be displayed (or the same plot type but with different units). Take care to position the viewport of the new and old frames to the desired locations.
`plot frame #i` - Sets frame number #i to be the currently active frame. After this command has been issued, all plot commands will only affect the currently active frame.
`plot frame delete #i` - Deletes frame number #i.
`plot type #a` - Specifies what is to be plotted in the selected frame. #a can be any of the plot types specified in Sect. 6.2.
`plot device #a1 [#a2]` - Selects a graphics device #a1 and optional file name #a2 (in the case of a postscript or gif device). File extensions (for example .ps) should be specified. For valid devices, see Sect. 6.1.
`plot close #i` - Close graphics device number #i. Note, always close a postscript device before quiting /spex, otherwise the contents may be corrupted.
`plot hlan` - Make a hardcopy in landscape orientation and send it to the standard printer. Use is made of the unix command "lp -c filename".
`plot hpor` - As above, but for portrait orientation
`plot x lin` - Plot the x-axis on a linear scale
`plot x log` - Plot the x-axis on a log scale
`plot y lin` - Plot the y-axis on a linear scale
`plot y log` - Plot the y-axis on a log scale
`plot y mix #r1 #r2` - Plot the y-axis on a mixed linear/logarithmic scale; for y-values below #r1 the data will be plotted linear, and for y-values above #r1 the data will be plotted logarithmically; the linear part occupies the fraction of the total frame height given by #r2. See also Sect. 6.8.2.
`plot z lin` - Plot the z-axis on a linear scale. The z-axis is defined for two-dimensional plots like contour plots.
`plot z log` - As above, but using a log scale.
`plot ux #a` - Set the plot units on the x-axis, #a can be Å, keV, eV, or whatever unit that is allowed. The allowed values depend upon the plot type. See Sect. 6.8.1 for allowed units for each plot type.
`plot ux vel #r #a` - Plot the x-axis in velocity units ($\mathrm{km\,s^{-1}}$), with reference energy provided as #r if #a is "keV" or reference wavelength provided as #r if #a is "Ang". Note that the same zero scale will be set for all plot windows.
`plot uy #a` - Same as above, but for the y-axis
`plot uz #a` - Same as above, but for the z-axis
`plot rx #r1:#r2` - set the plot range on the x-axis from #r1 to #r2
`plot ry #r1:#r2` - Same as above, but for the y-axis
`plot rz #r1:#r2` - Same as above, but for the z-axis
`plot view default #l` - For #l is true, the default viewport is used. For #l is false, the default view-

port is overruled and you can specify your own viewport limits with the "set view x" and "set view y" commands. Useful for stacking different plot frames.
`plot view x #r1:#r2` - Set the x viewport range from #r1 to #r2, where #r1 and #r2 must be in the range 0–1.
`plot view y #r1:#r2` - Same as above, but for the y-range.
`plot view back #i` - Set the viewport background colour to value #i. See Sect. 6.3 for the allowed plot colours.
`plot view transp #l` - If true, set the viewport background to to transparent, the plot frame will be shown with the background colour selected by the "plot view back" command. Default is false.
`plot box disp #l` - Display a box around the viewport, true/false
`plot box edge #a #l` - If true (default), display the edges of the plot box, #a should be one of the following keywords: top, bottom, left, right. Note that whether you see the edges or not also depends on the settings defined by the "plot box disp" command.
`plot box axis x #l` - Plots an x-axis with tickmarks at the line $y = 0$. You only see this of course if your y-range includes the value 0.
`plot box axis y #l` - As above, but for the y-axis
`plot box grid x #l` - Plot a grid for the x-axis
`plot box grid y #l` - Plot a grid for the y-axis
`plot box numlab bottom #l` - Show the tick number labels at the bottom of the plot, true/false
`plot box numlab top #l` - Same as above, but for the top
`plot box numlab left #l` - Same as above, but for the left
`plot box numlab right #l` - Same as above, but for the right
`plot box numlab vert #l` - Plot the tick number labels on the y-axis vertically or horizontally, set #l to true for vertical tnumbers and false for horizontal numbers.
`plot box numlab xscal #i` - Way to denote the numerical numbers along the x-axis. Three values are allowed: 0=automatic, 1=forced decimal labeling, 2=forced exponential labeling. Default is 0.
`plot box numlab yscal #i` - As above, but for the y-axis
`plot box tick invert x #l` - Draw the tick marks on the x-axis on the inside or outside the box, set #l to true for outside and false for inside (default).
`plot box tick invert y #l` - Same as above, but for the y-axis.
`plot box tick minor x #l` - Draw minor tick marks on the x-axis, true/false
`plot box tick minor y #l` - Same as above, but for the y-axis
`plot box tick major x #l` - Draw major tick marks on the x-axis, true/false
`plot box tick major y #l` - Same as above, but for the y-axis
`plot box tick distance x #r` - Set the distance between the major/labelled tick marks on the x-axis to #r
`plot box tick distance y #r` - Same as above, but for the y-axis
`plot box tick subdiv x #i` - Draw #i minor tick marks between each major tick mark on the x-axis
`plot box tick subdiv y #i` - Same as above, but for the y-axis
`plot box col #i` - Set the box colour to colour number #i. See Sect. 6.3 for the allowed plot colours.
`plot box lt #i` - Set the box line type to #i. See Sect. 6.4 for allowed line types.
`plot box lw #i` - Set the box line weight to number #i. See Sect. 6.4 for more about line weights.
`plot box fh #r` - Set the box font height to number #i.
`plot box font #i` - Set the box font to number #i. See Sect. 6.5 for more details about text fonts.
`plot cap #a disp #l` - If #l is true, display the caption (default). For more about captions see Sect. 6.6. Here and below, #a can be x, y, z, id, lt or ut.
`plot cap #a col #i` - Plot caption #a in colour number #i. See Sect. 6.3 for valid colour indices.
`plot cap #a back #i` - Set the background colour for caption #a to #i.
`plot cap #a lw #i` - Set the font line weight for caption #a to #i.
`plot cap #a fh #r` - Set the font height for caption #a to value #r.
`plot cap #a font #i` - Set the font type for caption #a to #i.
`plot cap #a1 text #a2` - Set the text for caption #a1 to #a2. Note that the caption text should be put between quotion marks, like "best fit results" if you want to see the text: best fit results.
`plot cap #a1 side #a2` - Plot caption #a1 (which can be x, y, z, id, lt, ut) at the side of the frame

specified by #a2, which may stand for t (top), b (bottom), lh (left, horizontal), rh (right, horizontal), lv (left, vertical) and rv (right, vertical).
`plot cap #a off #r` - Offset caption #a by #r from the edge of the viewport, where #r is in units of the character height. Enter negative values to write inside the viewport.
`plot cap #a coord #r` - Plot caption #a along the specified edge of the viewport, in units of the viewport length, where $0.0 \leq \#r \leq 1.0$.
`plot cap #a fjust #r` - Controls justification of the caption #a parallel to the specified edge of the viewport. If #r = 0.0, the left hand of #a will be placed at the position specified by "coord" above; if #r = 0.5, the center of the string will be placed at "coord", if #r = 1.0 the right hand of #a will be placed at "coord". Other values can be used but are less useful.
`plot string new #r1 #r2 #a` - Plot a new string with text as specified in #a at x=#r1 and y = #r2. See Sect. 6.5 for more deatils about text strings. Also do not forget to put #a between "" if it consists of more than one word (i.e., if it contains spaces).
`plot string del #i:` - Delete string numbers specified by the range #i from the plot.
`plot string #i: disp #l` - If true (default), display the strings specified by the range #i:.
`plot string #i: text #a` - Change the text of strings #i: to #a
`plot string #i1: col #i2` - Set the colours of strings #i1: to #i2
`plot string #i1: back #i2` - Set the background colour for the strings #i1: to the value #i2.
`plot string #i1: lw #i2` - Set the line weight of strings #i1: to #i2.
`plot string #i: fh #r` - Set the font height of strings #i to #r.
`plot string #i1: font #i2` - Set the font style of strings #i1 to #i2.
`plot string #i: x #r` - Set the x position of strings #i: to #r.
`plot string #i: y #r` - Set the y position of string #i: to #r.
`plot string #i: angle #r` - Set the angle of strings #i: to #r.
`plot string #i: fjust #r` - Controls justification of the strings #i: parallel to the specified edge of the viewport. If #r = 0.0, the left hand of the strings will be placed at the position specified by "x y" above; if #r = 0.5, the center of the strings will be placed at "x y", if #r = 1.0 the right hand of #i: will be placed at "x y". Other values can be used but are less useful.
`plot string #i: box #l` - If #l is true, plot a box around the text strings #i:. The default value is false (no box).
`plot string #i1: box lt #i2` - Set the line style of the box around the strings #i1: to the value #i2.
`plot string #i1: box lw #i2` - As above, but for the line weight specified by #i2.
`plot string #i1: box col #i2` - As above, but for the colour index for the box specified by #i2.
`plot set #i:` - Selects data set numbers as specified by #i:. Afterwards most plot commands will only affect data sets #i:
`plot set all` - Selects all data sets that are present. All subsequent plot commands will be executed for all data sets.
`plot line disp #l` - If #l is true, plots a connecting line through the data points, (default is false).
`plot line col #i` - Set the colour of the connecting line to #i.
`plot line lt #i` - Set the line style of the connecting line to #i.
`plot line lw #i` - Set the line weight of the connecting line to #i.
`plot line his #l` - If #l is true, plot the connecting line in histogram format (default is true).
`plot elin disp #l` - If #l is true, plots a connecting line through the end points of the error bars, (default depends upon the plot type).
`plot elin col #i` - Set the colour of the connecting line through the end points of the error bars to #i.
`plot elin lt #i` - Set the line style of the connecting line through the end points of the error bars to #i.
`plot elin lw #i` - Set the line weight of the connecting line through the end points of the error bars to #i.
`plot elin his #l` - If #l is true, plot the connecting line through the end points of the error bars in histogram format (default is true).
`plot model disp #l` - If #l is true, plot the current model corresponding to the relevant data set (default is true).
`plot model col #i` - Set the colour of the model to number #i.

`plot model lt #i` - Set the line style of the model to number #i.
`plot model lw #i` - Set the line weight of the model to number #i.
`plot model his #l` - If #l is true, plot the model in histogram format (default is true).
`plot back disp #l` - If #l is true, plot the subtracted background (default is true).
`plot back col #i` - Set the colour of the subtracted background to number #i.
`plot back lt #i` - Set the line style of the subtracted background to number #i.
`plot back lw #i` - Set the line weight of the subtracted background to number #i.
`plot back his #l` - If true, plot the subtracted background in histogram format (default is true).
`plot fill disp #l` - If #l is true, fill the curve below the model with the colour specified by the next command or the default colour.
`plot fill col #i` - Change the filling colour to #i.
`plot fill lt #i` - Change the line type of the filling lines to #i.
`plot fill lw #i` - Change the line weight of the filling lines to #i.
`plot fill style #i` - Change the style of the filling lines to the value #i. Here #i has values between 1–4, with the following meaning: 1 = solid filling (default), 2 = outline, 3 = hatched, 4 = cross-hatched.
`plot fill angle #r` - Set the angle for the filling lines for hatched filling. Default is 45 degrees.
`plot fill sep #r` - Set the distance between the filling lines for hatched filling. The unit spacing is 1 % of the smaller of the height or width of the viewing surface. This should not be zero.
`plot fill phase #r` - The phase between the hatch lines that fill the area.
`plot data disp #l` - If #l is true, display the data.
`plot data errx #l` - If #l is true, display the error bars in the x-direction.
`plot data erry #l` - If #l is true, display the error bars in the y-direction.
`plot data col #i` - Give the data colour index #i.
`plot data lt #i` - Give the data line style #i.
`plot data lw #i` - Give the data line weight #i.
`plot data fh #r` - Give the symbols for the data font height #r.
`plot data symbol #i` - Plot the data with symbol number #i. For symbol numbers, see Sect. 6.7.
`plot adum #a [overwrite] [append]` - Dump the data and model in the plot in an ascii file with filename #a. The extension ".qdp" will automatically be appended. Note that the data will be written as they are, i.e. if you have a logarithmic x-axis or y-axis, the logs of the plotted quantities will be written. If you want to replot your data later with for example the qdp package, take care that you plot the data in SPEX on a lin-lin frame before you execute the "plot adum" command. Also note that the data will be written in the units that were specified in the plot (energy, wavelength or whatever is applicable. If the optional "append" keyword is present, the data will be appended to any existing file with the name #a; if the optional "overwrite" keyword is present, any pre-existing file with the name #a will be overwritten by the new data.

Examples

`plot device xs` - Open the graphic device xs (xserver)
`plot device ps myplot.ps` - Select a postscript device connected to the file name myplot.ps
`plot type data` - Plot the data on the selected graphics device(s)
`plot ux angstrom` - Set the x-axis plot units to Å
`plot uy angstrom` - Set the y-axis plot units to Counts/s/Å
`plot frame new` - Open a new frame in the selected graphics device(s)
`plot frame 2` - Go to the 2nd frame, all plot commands will now only affect frame 2
`plot type chi` - Plot the residuals in frame 2
`plot uy rel` - Set the y-axis plot units in frame 2 to (Observed - Model)/Model
`plot view default f` - Set the default viewport keyword to false so that new user viewport values can be specified for frame 2
`plot view y 0.2:0.8` - Set the y viewport limits of frame 2 from 0.2 to 0.8 of the full device window
`plot cap id disp f` - Do not display the id caption of frame 2
`plot cap ut disp f` - Do not display the upper top caption of frame 2

`plot cap lt disp f` - Do not display the lower top caption of frame 2
`plot ux a` - Set the x-axis plot units of frame 2 to Å
`plot ux 21.602 ang` - Plot the x-axis as velocity in km $^{-1}$ relative to a wavelength of 21.602 Å.
`plot ry -1:1` - Set the y-axis plot range of frame 2 to between a lower limit of -1 and an upper limit of 1
`plot frame 1` - Go to frame 1
`plot view default f` - Set the default viewport keyword to false so that new user viewport values can be specified for frame 1
`plot view x 0.25:0.75` - Set the x viewport limits of frame 1 from 0.25 to 0.75 of the full device window
`plot de cps filename.ps` - Open a colour postscript graphics device and write the output file to filename.ps
`plot` - Redraw the plot on all frames and devices
`plot close 2` - Close device number 2, which is the postscript device in this case

3.24 Quit: finish the program

The quit option exits the execution of SPEX, closes the open plot-devices and scratch files (if any) and, if requested outputs the cpu-time statistics.

Syntax

The following syntax rule applies:

`quit` - quit the program as described above.

3.25 Sector: creating, copying and deleting of a sector

Overview

This allows one to create, delete, copy, and show the number of sectors, used for the analysis of the data. Sectors are designed to allow modeling of different sources of radiation with its own spectral model. This way, common components can be fit simultaneously, while the observed spectra are of a different origin. One can create a sector for another (unresolved) source in the spectrum, for modeling particle background, or for spectra extracted in a different time interval for a time variable source. See for more details about sectors and regions section 2.2.

For doing spectral fitting of data sets, the sectors need to be specified in the response matrix of the data: the response file should tell which sector number corresponds to a given part of the matrix.

The sector command features also a spectral dump mode (adump) that writes the model spectrum to an ascii file, formatted such that it is suited for the SPEX file model. The first line of the output file is an integer showing the number of bins, and the following lines show the energy bin centroid and the luminosity in 10^{44} ph/s/keV.

Syntax

The following syntax rules apply:

`sector new` - Creates a new sector, which can have its own model.
`sector show` - Gives the number of sectors that are currently used.
`sector copy #i` - Copies the model for sector #i to a new sector.
`sector delete #i` - Deletes sector #i.

`sector adump #i #a overwrite` - Writes the model spectrum of sector number #i to an ASCII file with name #a.

Examples

`sector new` - Creates a new sector.
`sector copy 2` - Creates a new sector, with the same spectral model as used in sector 2. This can be useful if the spectra of the different sectors are very similar in composition.
`sector delete 3` - Deletes sector number 3.
`sector adump 1 model.txt` - Dumps the spectrum in sector 1 to model.txt

3.26 Shiftplot: shift the plotted spectrum for display purposes

Overview

This command shifts the observed spectrum as well as the model spectrum by adding a constant or by multiplying a constant, as far as plots are concerned. The true data files do not change, so the shift is only for representational purposes.

There are basically two options, indicated by the mode variable *plotshift*: For *plotshift*=1, a constant *shift* is added to the plotted spectrum of a given part of the data, for *plotshift*=2 the plotted spectrum is multiplied by the constant *shift*.

The multiplicative constant shift (*plotshift*=2) is generally preferred for log-log plots, while for linear plots a constant additive shift (*plotshift*=1) is preferred.

WARNING: *In the case of addition (plotshift=1), the addition constant is given in counts/s. This thus leads to a different (energy-dependent) constant being added to the plotted spectrum if the units are not in counts/s. For the multiplicative case this is of course not the case.*

Syntax

The following syntax rules apply:

`shiftplot #i #r` - #i indicates whether a constant should be added (#i=1) or multiplied (#i=2) to the spectrum and model. #r is then the constant to be added or multiplied by.

`shiftplot [instrument #i1:] [region #i2:] #i3 #r` - Shift the plot using a factor #r. Here #i3 (= *plotshift*) determines whether the constant is added (*plotshift=1*) or multiplied (*plotshift=2*). The optional instrument range #i1: can be used to select a single instrument or range of instruments for which the shift needs to be applied, and the optional region range #i2: the region for these instruments.

Examples

`shiftplot 1 2.0` - Adds 2 counts/s to the plotted spectrum. Since no instrument or region is specified, this applies for all spectra.
`shiftplot 2 10.0` - Multiplies the plotted spectrum by a factor of 10.0
`shiftplot instrument 1 region 2 1 2` - Adds 2 counts/s to the plotted spectrum for the data from instrument 1 and region 2.
`shiftplot region 2:3 1 2` - Adds 2 counts/s to the plotted spectrum for the data for all instruments and regions 2–3.

3.27 Simulate: Simulation of data

Overview

This command is used for spectral simulations. The user should start with a spectral model and a spectral data set (both matrix and spectrum). After giving the "simulate" command, the observed spectrum will be replaced by the simulated spectrum in SPEX. Note that the original spectrum file (with the .spo extension) is not overwritten by this command, so no fear to destroy your input data!

Different options exist and several parameters can be set:

- Instrument, region: the instrument(s) and region(s) for which the simulation should be done (i.e., if you have more instruments you can do the simulation only for one or a few instruments)
- time: set the exposure time t (s) of the source spectrum as well as the background error scale factor f_b. This last option allows you to see what happens if for example the background would have a ten times higher accuracy (for $f_b = 0.1$).
- syserr: add a systematic error to both the source and background spectrum. An alternative way to introduce systematic errors is of course to use the syserr command (Sect. 3.29). Take care not to set the systematic errors twice, and remember that rebinning your spectrum later will reduce the systematic errors, as these will be added in quadrature to the statistical errors. So first rebin and then add systematics!
- noise: either randomize your data or just calculate the expected values
- bnoise: randomize your background model (generally not recommended to do)

WARNING: *A response matrix and spectrum of the region and the instrument you want to simulate are necessary, because SPEX needs the response matrix as well as the background to be subtracted for the simulations.*

WARNING: *When you include systematic errors in the simulation (by putting the "syserr" to non-zero values), you cannot use anymore Poissonian statistics hence the C-stat for fitting, but you have to use the "fit meth chi" to use Gaussian errors and χ^2-fitting, with all thr disadvantages of that.*

WARNING: *When you use bnoise=true, your subtracted background (the scaled background from the background region) will be randomized, and the pure C-stat cannot be used; the W-stat can be used as alternative but has serious drawbacks and is not recommended to be used).*

WARNING: *(obsolete) If your background is taken from the same observation as your source, and you multiply the original exposure time with a factor of S, you should put f_b to $S^{-0.5}$, reflecting the fact that with increasing source statistics the background statistics also improves. This is the case for an imaging observation where a part of the image is used to determine the background. If instead you use a deep field to subtract the background, then the exposure time of your background will probably not change and you can safely put $f_b = 1$ for any exposure time t.*

WARNING: *(obsolete) If your subtracted background (one of the columns in the .spo file) is derived from a low statistics Poissonian variable (for example a measured count rate with few counts per channel), then scaling the background is slightly inaccurate as it takes the observed instead of expected number of background counts as the starting point for the simulation.*

Syntax

Note that only the "simulate" keyword followed by a number (the exposure time will do the actual simulation. All other syntax rules just set some parameters. The following syntax rules apply:

`simulate #r` - Does the simulation, with #r the exposure time t in seconds.
`simulate instrument #i1` - Specify the instrument (range) to be used in the simulation. Default values

are 1 (just the first instrument).
`simulate region #i1` - Specify the region (range) to be used in the simulation. Default values are 1 (just the first region). For simulating everything you have, you can put this range to a large value: the simulation will simply ignore non-existent regions. If you use complex settings, like only region 3 for instrument 1 and region 2 for instrument 2, you may have to run the simulation separately for each entity.
`simulate syserr #r1 #r2` - Specify the systematic errors as a fraction of the source and background spectrum, respectively; both should be specified together. Default values are 0.
`simulate noise #l` - If #l is true, Poissonian noise will be added (this is the default).
`simulate bnoise #l` - If #l is false, no Poissonian noise will be added to the model background (this is the default).

Examples

`simulate 10000.` - This simulates a new spectrum/dataset with 10 000 s exposure time.
`simulate noise f` - Set simulation flag to simulate without Poissonian noise. The nominal error bars will still plotted.
`simulate syserr 0.1 0.2` - Set simulation for a systematic error of 10 % of the source spectrum and 20 % of the subtracted background spectrum added in quadrature.
`simulate instrument 2:4` - Set simulation for a spectrum for instruments 2–4 only
`simulate region 2` - Set simulation for only region 2 (of every instrument involved

3.28 Step: Grid search for spectral fits

Overview

A grid search can be performed of χ^2 versus 1, 2, 3 or 4 parameters. The minimum, maximum and number of steps for each parameter may be adjusted. Steps may be linear or logarithmic. For each set of parameters, a spectral fit is made, using the last spectral fit executed before this command as the starting point. For each step, the parameters and χ^2 are displayed. This option is useful in case of doubt about the position of the best fit in the parameter space, or in cases where the usual error search is complicated. Further it can be useful incases of complicated correlations between parameters.

WARNING: *Take care to do a spectral fit first!*

WARNING: *Beware of the cpu-time if you have a fine grid or many dimensions!*

Syntax

The following syntax rules apply:

`step dimension #i` - Define the number of axes for the search (between 1–4).
`step axis #i1 parameter [[#i2] #i3] #a range #r1:#r2 n #i4` - Set for axis #i1, optional sector #i2, optional component #i3 and parameter with name #a the range to the number specified by #r1 and #r2, with a number of steps n given by #i4. If $n > 0$, a linear mesh between #r1 and #r2 will be taken, for $n < 0$, a logarithmic grid will be taken.
`step` - Do the step search. Take care to do first the "step dimension" command, followed by as many "step axis" commands as you entered the number of dimensions.
`step file example` - Do the step search and also write the results to an .stp file used by the stepcontour program (see 7.7).

WARNING: *The step file command will overwrite existing .stp files with the same name by default.*

Examples

Below we give a worked out example for a CIE model where we took as free parameters the normalization "norm", temperature "t", Si abundance "14" and Fe abundance "26". To do a 3-dimensional grid search on the temperature (logarithmic between 0.1 and 10 keV with 21 steps), Fe abundace (linear between 0.0 and 1.0 with 11 steps) and Si abundance (linear between 0.0 and 2.0 with 5 steps, i.e. values 0.0, 0.4, 0.8, 1.2, 1.6 and 2.0) the following commands should be issued:

`fit` - Do not forget to do first a fit
`step dimension 3` - We will do a three-dimensional grid search
`step axis 1 parameter 1 1 t range 0.1:10.0 n -21` - The logarithmic grid for the first axis, temperature; note the − before the 21!
`step axis 2 par 26 range 0:1 n 11` - The grid for axis 2, Fe abundance
`step axis 3 par 14 range 0:2 n 5` - Idem for the 3rd axis, Silicon
`step` - Now do the grid search. Output is in the form of ascii data on your screen (and/or output file if you opened one).
`step file example` - Grid search with output to your screen and .stp output file for use with the step-contour task.

3.29 Syserr: systematic errors

Overview

This command calculates a new error, adding the systematic error of the source and the background to the Poissonian error in quadrature. One must specify both the systematic error as a fraction of the source spectrum as well as of the subtracted background spectrum. The total of these fractions can either be less or greater than 1.

WARNING: *This command mixes two fundamentally different types of errors: statistical (random fluctuations) and systematic (offset). The resulting uncertainties are unjustly treated as being statistical, which can lead to wrong results when the systematic offsets are substantial. Syserr should therefore be used with extreme caution.*
WARNING: *One should first rebin the data, before running syserr. Run syserr however before fitting the data or finding errors on the fit.*
WARNING: *Running syserr multiple times will increase the error every time. If the input to syserr is wrong one should restart SPEX and rerun syserr with the correct values to calculate the total error correctly.*

Syntax

The following syntax rules apply:

`syserr #i: #r1 #r2` - The shortest version of this command. #i: is the range in data channels for which the systematic error is to be calculated and added (in quadrature) to the Poissonian error. #r1 is then the the relative systematic error due to the source and #r2 the relative systematic error due to the background.
`syserr [instrument #i1:] [region #i2:] #i3: #r1 #r2` - In this syntax one can also specify the instrument and the region one wants to calculate the combined error for. Both can be ranges as well. #i3: has the same role as #i: in the above command, and #r1 and #r2 are the same as above.
`syserr [instrument #i1:] [region #i2:] #i3: #r1 #r2 [unit #a]` - Exact same command as above, except that now the data range (#i3:) for which the errors are to be calculated are given in units different than data channels. These units can be Å (ang), eV (ev), keV (kev), Rydbergs (ryd), Joules (j),

Hertz (hz) and nanometers (nm). This is the most general command.

Examples

`syserr 1:100000 0.3 0.5` - Calculates the combined Poissonian and systematic error for data channels 1:100000, where the fraction of the systematic error of the source is 0.3 and the background is 0.5.
`syserr 0:2000 0.3 0.5 unit ev` - The same as the above command, expect that now the error calculation is performed between 0 and 2000 eV instead of data channels.
`syserr instrument 2 region 1 0:2000 0.3 0.5 unit ev` - The same as the above command, but now the error calculation is only performed for the data set from the second instrument and the first region thereof.

3.30 System: call system executables

Overview

Sometimes it can be handy if SPEX interacts with the computer system, for example if you run it in command mode. You might want to check the existence of certain file, or run other programs to produce output for you, and depending on that output you want to continue SPEX.

Therefore there is an option to execute any shell type commands on your machine, using the fortran "call system" subroutine.

Another useful goody is the possibility to stop SPEX automatically if you find some condition to occur; this might be useful for example if you have a program running that calls SPEX, and depending on the outcome of SPEX you might want to terminate the execution. This is achieved in SPEX by testing for the existence of a file with a given filename; if the file exists, SPEX stops immediately execution and terminates; if the file does not exist, SPEX continues normally.

Syntax

The following syntax rules apply:

`system exe #a` - execute the command #a on your UNIX/linux shell
`system stop #a` - stop SPEX if the file #a exists

Examples

`system exe "ls -l"` - give a listing of all file names with length in the current directory
`system exe "myfortranprogram"` - execute the fortran program with name "myfortranprogram"
`system stop testfile` - stop SPEX if the file with name testfile exists; if this file does not exist, continue with SPEX

3.31 Use: reuse part of the spectrum

Overview

This command can only be used after the ignore command was used to block out part of the data set in the fitting. The command re-includes thus (part of) the data set for fitting. As it undoes what the ignore

command did (see section 3.15) the syntax and use of both are very similar. Again one can/must specify the instrument and region for one wants the data to be re-included. Further can one specify the units chosen to give the range of data points to be included. The standard unit is that of data channels and does not need to be specified. The data points re-included with use will automatically also be plotted again.

The range to be reused can be specified either as a channel range (no units required) or in eiter any of the following units: keV, eV, Rydberg, Joule, Hertz, Å, nanometer, with the following abbrevations: kev, ev, ryd, j, hz, ang, nm. Note that spectra that were binned before using the bin command are not binned anymore after the spectrum is ignored and reused again. In this case, the bin command should be applied again to restore the desired spectral binning.

Syntax

The following syntax rules apply:

`use [instrument #i1] [region #i2] #r` - Use the range #r in fitting the data. The instrument #i1 and the region #i2 must be specified if either there are more than 1 instrument or more than 1 region used in the data sets.
`use [instrument #i1] [region #i2] #r unit #a` - Same as the above, but now the units #a in which the range #r of data points to be used is specified in any of the units mentioned above.

Examples

`use 1000:1500` - Re-include the data channels 1000:1500 for fitting the data.
`use instrument 1 region 1 1000:1500` - Re-include the data channels 1000:1500 of the first instrument and the first region in the data set for fitting the data and plotting.
`use instrument 1 region 1 1:1.5 unit kev` - Same as the above, but now the unites are specified as keV instead of in data channels.
`use 1000:1500 unit ev` - Re-iclude the data points between 1000 and 1500 eV for fitting and plotting. Note that in this case the input data should come from only one instrument and should contain no regions.

3.32 Var: various settings for the plasma models

3.32.1 Overview

For the plasma models, there are several quantities that have useful default values but can be adjusted manually for specific purposes. We list them below.

Free-bound emission

Usually the freebound emission takes most of the computing time for the plasma models. This is because it needs to be evaluated for all energies, ions and atomic sublevels that are relevant. In rder to reduce the computational burden, there is a parameter `gfbacc` in SPEX that is approximately the accuracy level for the free-bound calculation. The default value is 10^{-3}. If this quantity is higher, less shells are taken into account and the computation proceeds faster (but less accurate). The users are advised not to change this parameter unless they are knowing what they do!

Line emission contributions

By default, all possible line emission processes are taken into account in the plasma models. For testing purposes, it is possible to include or exclude specific contributions. These are listed below in Table 3.7.

Table 3.7: Possible line emission processes

Abbrevation	Process
ex	electron excitation
rr	radiative recombination
dr	dielectronic recombination
ds	dielectronic recombination satellites
ii	inner shell ionisation

Doppler broadening

By default, thermal Doppler broadening is taken into account. However, the user can put this off for testing purposes. The options for the broadening are:

1. No broadening at all
2. Only Doppler broadening (default)
3. Only natural broadening (works only for *var calc new*)
4. Doppler and natural broadening, Voigt profiles (best physical representation, works only for *var calc new*)

Atomic data

The user can choose between the "old" Mekal code (the current default) and updated calculations for the ions for which this is possible.

WARNING: *The updated atomic database that can be used through the command "var calc new" is still being developed and incomplete. Using this database is therefore not yet recommended.*

Mekal code

We have made several minor improvements to the original Mekal plasma model. These improvements are included by default. However, it is possible to discard some of these improvements by going back to the old code. The improvements are classified as follows (with the appropriate syntax word added here).

Wav: wavelength corrections according to the work of Phillips et al. (1999), based upon Solar flare spectra; in addition, the 1s-np wavelengths of Ni XXVII and Ni XXVIII have been improved.

Fe17: The strongest Fe XVII lines are corrected in order to agree with the rates as calculated by Doron & Behar (2002).

Update: several minor corrections: the Si IX C7 line has been deleted; the Si VIII N6 line now only has the 319.83 Å line strength instead of the total triplet strength; the Ni XVIII and Fe XVI Na1A and NA1B lines plus their satellites have been deleted.

3.32.2 Syntax

The following syntax rules apply:

`var gacc #r` - Set the accuracy `gfbacc` for free-bound emission. Default is 10^{-3}, maximum value 1 and minimum value 0. Do not change if you do not know what it does.
`var gacc reset` - Reset `gfbacc` to its default value.
`var line #a #l` - For process #a (where #a is one of the abbreviations in Table 3.7) the process is allowed (if #l is true) or disabled (if #l is false). By default, all processes are allowed.
`var line reset` - Enable all line emission processes
`var line show` - Show the status of the line emission processses
`var doppler #i` - Line broadening, see the four allowed values in the above description
`var calc old` - Use the old Mekal code
`var calc new` - Use the new updated atomic data (for SPEX version 3.0 and higher)
`var newmekal wav #l` - if true (the default), use the updated wavelengths for the Mekal code
`var newmekal fe17 #l` - if true (the default), use the updated Fe XVII calculations for the Mekal code
`var newmekal update #l` - if true (the default), use the updates for some lines for the Mekal code
`var newmekal all #l` - if true (default), use all the above three corrections for the Mekal code
`var ibalmaxw #l` - if true use multi-Maxwellians (if relevant) for both the ionisation balance and the spectrum (default); if false, only use it for the spectrum.

3.32.3 Examples

`var gacc 0.01` - Set the accuracy `gfbacc` for free-bound emission.to 0.01
`var gacc reset` - Reset the accuracy `gfbacc` for free-bound emission.to its default value of 0.001
`var line ex f` - Exclude electron excitation
`var line ds t` - Include dielectronic satellites
`var line reset` - Include all line emission processes
`var line show` - Show status of all line emission proceses
`var doppler f` - Do not use thermal Doppler bvroadening
`var calc new` - Use the new atomic data (EXPERIMENTAL)
`var newmekal wav f` - Use the original Mekal wavelengths instead
`var newmekal fe17 t` - Use the updated Fe XVII calculations
`var newmekal all f` - Go back to the full old Mekal code
`var newmekal all t` - Take the full updated Mekal code
`var ibalmaxw f` - Do not use Multi-Maxwellians for the ionisation balance

3.33 Vbin: variable rebinning of the data

Overview

This command rebins (a part of) the data (thus both the spectrum and the response) such that the signal to noise ratio in that part of the spectrum is at least a given constant. Thus instead of taking a fixed number of data channels for binning, the number of data channels binned is here variable as is the bin width. On top of specifying the S/N ratio, you also specify the minimal bin width. This is useful for rebinning low count data, which have no strong or sharp features or lines in it.

WARNING: *For high resolution spectra with sharp and strong lines, this binning can lead to very wrong results. In this case either the emission lines or the continuum (in case of absorption lines) have a much higher signal to noise ratio than the other component. As a result this function will try to bin the line and continuum together, resulting in a smoothing out of the lines.*

WARNING: *It is not always guaranteed that the minimum signal-to-noise ratio is obtained in all channels. This is an effect of the applied algorithm. Channels with the highest S/N ratio and neighboring bins are merged until sufficient S/N ratio is obtained. This process is continued for the remaining number of bins.*

At the end of the process a few bins with a low S/N ratio will remain. These are merged with their neighbors, resulting in a possibly lower S/N ratio for that bin.

Syntax

The following syntax rules apply:

`vbin #i1: #i2 #r` - Simplest command allowed. #i1: is the range in data channels over which the binning needs to take place. #i2 in the minimum amount of data channels that need to be binned, and #r is the minimal S/N ratio one wants the complete data set to have.
`vbin #r1: #i #r2 unit #a` - The same command as above, except that now the ranges over which the data is to be binned (#r1:) are specified in units (#a) different from data channels. These units can be eV, keV, Å, as well as in units of Rydberg (ryd), Joules (j), Hertz (hz) and nanometers (nm).
`vbin [instrument #i1:] [region #i2:] #i3: #i4 #r` - Here #i3: and #i4 are the same as #i1: and #i2 in the first command, also #r is the minimal S/N ratio required. However, here one can specify the instrument range #i1: and the region range #i2: as well, so that the binning only is performed for a certain data set.
`vbin [instrument #i1:] [region #i2:] #r1: #i2 #r2 [unit #a]` - This command is the same as the above, except that here one can specify the range over which the binning should occur in the units specified by #a. These units can be eV, Å, keV, as well as in units of Rydberg (ryd), Joules (j), Hertz (hz) and nanometers (nm).

Examples

`vbin 1:10000 3 10` - Bins the data channels 1:10000 with a minimal bin width of 3 data channels, and a minimum S/N ratio of 10.
`vbin 1:4000 3 10 unit ev` - Does the same as the above, but now the data range to be binned is given in eV, from 1−4000 eV, instead of in data channels.
`vbin instrument 1 region 1 1:19 3 10 unit a` - Bins the data from instrument 1 and region 1 between 1 and 19 Å with the same minimum bin width and S/N as above.

3.34 Watch

Overview

If you have any problems with the execution of SPEX, you may set the watch-options. For example, if the computation time needed for your model is very large, it could be wise to check which parts of the program consume most of the cpu-time. You might then hope to improve the performance by assuming other parameters or by re-structuring a part of the code. In this case, you set the "time" flag to true (see below), and at the end of the program you will see how much time was spent in the most important subroutines.

Another case occurs in the unlikely case that SPEX crashes. In that case it is recommended to re-execute the program, saving all commands onto a log-file and use the "sub" flag to report the entering and exiting of all major subroutines. This makes it more easy to find the source of the error.

Timing is done by the use of the stopwatch package by William F. Mitchell of the NIST, which is free available at the web. If the time flag is set to true, on exit SPEX will report for each subroutine the following execution times (in s):

- The user time, i.e. the cpu-time spent in this subroutine
- The system time, i.e. the overhead system time for this subroutine

- The wall time, i.e. the total time spent while executing this subroutine.

Also the totals for SPEX as a whole are given (this may be more then the sum of the subroutine components, since not all subroutines are listed separately; the wall tiem for SPEX as a whole also includes the time that the program was idle, waiting for input from the user).

Syntax

The following syntax rules apply:
`watch time #l` - set the "time" flag to true or false.
`watch sub #l` - set the flag that SPEX causes to report each major subroutine it enters or exits

Examples

`watch time t` - set the "time" flag to true
`watch sub f` - set the subroutine report flag to false

Chapter 4

Spectral Models

4.1 Overview of spectral components

For more information on the definition of spectral components and the different types, see Sect. 2.3. Currently the following models are implemented in SPEX, see Table 4.1.

4.2 Absm: Morrison & McCammon absorption model

This model calculates the transmission of neutral gas with cosmic abundances as published first by Morrison & McCammon (1983). It is a widely used model. The following is useful to know when this model is applied:

1. The location of the absorption edges is not always fully correct; this can be seen with high resdolution grating spectra
2. The model fails in the optical/UV band (i.e., it does not become transparent in the optical)
3. No absorption lines are taken into account
4. The abundances cannot be adjusted

If all the above is of no concern (as is the case in many situations), then the Morrison & McCammon model is very useful. In case higher precision or more detail is needed, the user is advised to use the "hot" model with low temperature in SPEX, which gives the transmission of a slab in collisional ionisation equilibrium.

The parameters of the model are:

`nh` - Hydrogen column density in 10^{28} m^{-2}. Default value: 10^{-4} (corresponding to 10^{24} m^{-2}, a typical value at low Galactic latitudes).
`f` - The covering factor of the absorber. Default value: 1 (full covering)

Recommended citation: Morrison & McCammon (1983).

4.3 Amol: oxygen edge molecules absorption model

This model calculates the transmission of various molecules.

The following compounds are presently taken into account (see Table 4.2).

Table 4.2: Molecules present in the amol model.

nr	Name	Chemical formula	Atom & shell	Reference
108	molecular oxygen	O_2	O 1s	a
114	silicon crystal	Si	Si 1s	h
126	metallic iron	Fe	Fe 1s,2p	i,j
2001	water	H_2O	O 1s	c
2002	crystalline ice	H_2O	O 1s	d
2003	amorphous ice	H_2O	O 1s	d
2010	carbon monoxide	CO	O 1s	a
2011	carbon dioxide	CO_2	O 1s	a
2020	laughing gas	N_2O	O 1s	a,b
2100	silicon carbide	SiC	Si 1s	h
2101	silicon nitrite	Si_3N_4	Si 1s	h
2102	silicon monoxide	SiO	Si 1s	h
2103	quartz	SiO_2	Si 1s	h
2104	silica	SiO_2	Si 1s	h
2200	eskolaite	Cr_2O_3	O 1s	e
2300	iron monoxide	FeO	Fe 1s	i
2301	iron oxide	$Fe_{1-x}O$	O 1s	e

Continued on next page

nr	Name	Chemical formula	Atom & shell	Reference
2302	magnetite	Fe_3O_4	O, Fe 1s	e,i
2303	hematite	Fe_2O_3	O, Fe 1s; Fe 2p	e,i,j
2304	iron sulfite	FeS_2	Fe 1s	i
2400	nickel monoxide	NiO	O 1s	e
2500	cupric oxide	CuO	O 1s	e
3001	adenine	$C_5H_5N_5$	O 1s	f
3100	enstatite	$MgSiO_3$	Si 1s	h
3101	enstatite_alfa	$MgSiO_3$	Si 1s	h
3102	enstatite_chondrite	$Mg_2Si_2O_6$	Si 1s	h
3103	pyroxene	$MgSiO_3$	O 1s	k
3200	calcite	$CaCO_3$	Ca 1s	g
3201	aragonite	$CaCO_3$	Ca 1s	g
3202	vaterite	$CaCO_3$	Ca 1s	g
3203	perovskite	$CaTiO_3$	O 1s	e
3300	hercynite	$FeAl_2O_4$	O 1s	e
3301	lepidocrocite	FeO(OH)	Fe 2p	j
3302	fayalite	Fe_2SiO_4	Fe 1s, 2p	i,j
3303	iron sulfate	$FeSO_4$	Fe 2p	j
3304	ilmenite	$FeTiO_3$	O 1s	e
3305	chromite	$FeCr_2O_4$	O 1s	e
4001	guanine	$C_5H_5N_5O$	O,N 1s	f
4002	cytosine	$C_4H_5N_3O$	O,N 1s	f
4003	thymine	$C_5H_6N_2O_2$	O,N 1s	f
4004	uracil	$C_4H_4N_2O_2$	O,N 1s	f
4100	andradite	$Ca_3Fe_2Si_3O_{12}$	O 1s	e
4101	acmite	$NaFeSi_2O_6$	O 1s	e
4102	franklinite	$Zn_{0.6}Mn_{0.8}Fe_{1.6}O_4$	O 1s	e
4103	olivine	$Mg_{1.6}Fe_{0.4}SiO_4$	O 1s	e
4104	almandine	$Fe_3Al_2(SiO_4)_3$	O 1s	e
4105	hedenbergite	$CaFeSi_2O_6$	O 1s	e
5001	dna (herring sperm)	$C_{39}H_{61}N_{15}O_{36}P_4$	O,N 1s	f
6001	montmorillonite	$Na_{0.2}Ca_{0.1}Al_2Si_4O_{10}(OH)_2(H_2O)_{10}$	Si 1s	h
6002	nontronite	$Na_{0.3}Fe_2^{3+}Si_3AlO_{10}(OH)_2 \bullet (H_2O)$	Si 1s	h
7001	enstatite_paulite	$Ca_2Mg_4Al_{0.75}Fe_{0.25}(Si_7AlO_{22})(OH)_2$	Si 1s	h

References:
[a] Barrus et al. (1979), 0.5-0.75 eV resolution
[b] Wight & Brion (1974), 0.5 eV resolution
[c] Hiraya et al. (2001), 0.055 eV resolution
[d] Parent et al. (2002), 0.1 eV resolution
[e] van Aken et al. (1998), 0.8 eV resolution
[f] Fujii et al. (2003), unknown resolution
[g] Hayakawa et al. (2008), unknown resolution
[h] Lee (2010), unknown resolution
[i] Lee & Ravel (2005), unknown resolution
[j] Lee et al. (2009), 0.23 eV resolution
[k] Lee et al. (2008), 0.7 eV resolution

The chemical composition of these minerals was mainly taken from the Mineralogy Database of David Barthelmy[1]. For DNA we assume equal contributions of adenine, cytosine, guanine and thymine, plus

[1] http://webmineral.com/

for each of these on average one phosphate and one 2-deoxyribose molecule. We take the cross-sections from the references as listed in Table 4.2 in the energy interval where these are given, and use the cross section for free atoms (Verner & Yakovlev 1995) outside this range.

van Aken et al. (1998) do not list the precise composition of iron oxide. We assume here that $x = 0.5$.

Some remarks about the data from Barrus et al. (1979): not all lines are given in their tables, because they suffered from instrumental effects (finite thickness absorber combined with finite spectral resolution). However, Barrus et al. (1979) have estimated the peak intensities of the lines based on measurements with different column densities, and they also list the FWHM of these transitions. We have included these lines in the table of cross sections and joined smoothly with the tabulated values.

For N_2O, the fine structure lines are not well resolved by Barrus et al. (1979) Instead we take here the relative peaks from Wight & Brion (1974), that have a relative ratio of 1.00 : 0.23 : 0.38 : 0.15 for peaks 1, 2, 3, and 4, respectively. We adopted equal FWHMs of 1.2 eV for these lines, as measured typically for line 1 from the plot of Wight. We scale the intensities to the peak listed by Barrus et al. (1979).

Further, we subtract the C and N parts of the cross section as well as the oxygen 2s/2p part, using the cross sections of Verner & Yakovlev (1995). At low energy, a very small residual remains, that we corrected for by subtracting a constant fitted to the 510–520 eV range of the residuals. The remaining cross section at 600 eV is about 10 % above the Verner cross section; it rapidly decreases; we approximate the high-E behaviour by extrapolating linearly the average slope of the ratio between 580 and 600 eV to the point where it becomes 1. The remaining cross section at 600 eV is about 10% above the Verner & Yakovlev (1995) cross section; it rapidly decreases; we approximate the high-E behaviour therefore by extrapolating linearly the average slope of the ratio between 580 and 600 eV to the point where it becomes 1.

WARNING: *The normalisation is the total molecular column density. Thus, a value of* 10^{-7} *for* CO_2 *means* 10^{21} CO_2 *molecules* m^{-2}*, but of course* 2×10^{21} *O atoms* m^{-2}*, because each* CO_2 *molecule contains 2 oxygen atoms.*

WARNING: *The Table above shows for which edges and atoms the XAFS are taken into account. For all other edges and atoms not listed there, we simply use the pure atomic cross-section (without absorption lines). Note that for almost all constituents this may give completely wrong cross sections in the optical/UV band, as at these low energies the effects of chemical binding, crystal structure etc. are very important for the optical transmission constants. This is contrary to the SPEX models for pure atomic or ionised gas, where our models can be used in the optical band.*

WARNING: *It is possible to change the values of the output atomic column densities of H–Zn, that are shown when you issue the "show par" command of SPEX. However, SPEX completely ignores this and when you issue the "calc" or "fit" commands, they will be reset to the proper values. Morale: just read of those parameters, don't touch them!*

The parameters of the model are:

`n1--n4` - Molecular column density in 10^{28} m^{-2} for molecules 1–4. Default value: 10^{-6} for molecule 1, and zero for the others.
`i1--i4` - the molecule numbers for molecules 1–4 in the list (Table 4.2). Default value: 108 (O_2) for molecule 1, zero for the others. A value of zero indicates that for that number no molecule will be taken into account. Thus, for only 1 molecule, keep *i2–i4* = 0.

The following parameters are common to all our absorption models:
`f` - The covering factor of the absorber. Default value: 1 (full covering)
`zv` - Average systematic velocity v of the absorber

The following parameters are *only* output parameters:

`h--zn` - The column densities in 10^{28} m^{-2} for all *atoms* added together for the all molecules that are present in this component.

Recommended citation: Pinto et al. (2010).

4.4 Bb: blackbody model

The surface *energy* flux of a blackbody emitter is given by

$$F_\nu = \pi B_\nu = \frac{2\pi h\nu^3/c^2}{e^{h\nu/kT}-1} \tag{4.1}$$

(cf. Rybicki & Lightman 1986, chapter 1). We transform this into a spectrum with energy units (conversion from Hz to keV) and obtain for the total *photon* flux:

$$S(E)\mathrm{d}E = 2\pi c[10^3 e/hc]^3 \frac{E^2}{e^{E/T}-1}\mathrm{d}E \tag{4.2}$$

where now E is the photon energy in keV, T the temperature in keV and e is the elementary charge in Coulomb. Inserting numerical values and multiplying by the emitting area A, we get

$$N(E) = 9.883280\times 10^7\, E^2 A/(e^{E/T}-1) \tag{4.3}$$

where N(E) is the photon spectrum in units of 10^{44} photons/s/keV and A the emitting area in 10^{16} m^2.

The parameters of the model are:

`norm` - Normalisation A (the emitting area, in units of 10^{16} m^2. Default value: 1.
`t` - The temperature T in keV. Default value: 1 keV.

Recommended citation: Kirchhoff (1860).

4.5 Cf: isobaric cooling flow differential emission measure model

This model calculates the spectrum of a standard isobaric cooling flow. The differential emission measure distribution $\mathrm{d}Y(T)/\mathrm{d}T$ for the isobaric cooling flow model can be written as

$$D(T) \equiv \mathrm{d}Y(T)/\mathrm{d}T = \frac{5\dot{M}k}{2\mu m_\mathrm{H}\Lambda(T),} \tag{4.4}$$

where $\dot{M}$ is the mass deposition rate, k is Boltzmann's constant, μ the mean molecular weight (0.618 for a plasma with 0.5 times solar abundances), m_H is the mass of a hydrogen atom, and $\Lambda(T)$ is the cooling function. We have calculated the cooling function Λ using our own SPEX code for a grid of temperatures and for 0.5 times solar abundances. The spectrum is evaluated by integrating the above differential emission measure distribution between a lower temperature boundary T_1 and a high temperature boundary T_n. We do this by creating a temperature grid with n bins and calculating the spectrum for each temperature.

WARNING: *Take care that n is not too small in case the relevant temperature is large; on the other hand if n is large, the computational time increases. Usually a spacing with temperature differences between adjacent bins of 0.1 (in* log*) is sufficient and optimal.*

WARNING: *The physical relevance of this model is a matter of debate*

The parameters of the model are:

`norm` - The mass deposition rate $\dot{M}$ in $\mathrm{M}_\odot\,\mathrm{yr}^{-1}$
`t1` - Lower temperature cut-off temperature T_1. Default: 0.1 keV.
`tn` - Upper temperature cut-off temperature T_n. Default: 1 keV.
`nr` - Number of temperature bins n used in the integration. Default value: 16

`p` - Slope $p = 1/\alpha$. Default: 0.25 ($\alpha = 4$).
`cut` - Lower temperature cut-off, in units of $T_{\rm max}$. Default value: 0.1.

The following parameters are the same as for the cie-model:
`ed` - Electron density in 10^{20} m^{-3}
`it` - Ion temperature in keV
`vmic` - Micro turbulent velocity in km/s
`ref` - Reference element
`01...30` - Abundances of H to Zn
`file` - Filename for the nonthermal electron distribution

Recommended citation: Kaastra et al. (2004).

4.6 Cie: collisional ionisation equilibrium model

This model calculates the spectrum of a plasma in collisional ionisation equilibrium (CIE). It consists essentially of two steps, first a calculation of the ionisation balance and then the calculation of the X-ray spectrum. The basis for this model is formed by the mekal model, but several updates have been included.

4.6.1 Temperatures

Some remarks should be made about the temperatures. SPEX knows three temperatures that are used for this model.

First there is the electron temperature $T_{\rm e}$. This is usually referrred to as "the" temperature of the plasma. It determines the continuum shape and line emissivities and the resulting spectrum is most sensitive to this.

Secondly, there is the ion temperature $T_{\rm i}$. This is only important for determining the line broadening, since this depends upon the thermal velocity of the ions (which is determined both by $T_{\rm i}$ and the atomic mass of the ion). Only in high resolution spectra the effects of the ion temperature can be seen.

Finally, we have introduced here the ionization balance temperature $T_{\rm b}$ that is used in the determination of the ionization equilibrium. It is the temperature that goes into the calculation of ionization and recombination coefficients. In equilibrium plasmas, the ratio $R_{\rm b} \equiv T_{\rm b}/T_{\rm e}$ is 1 by definition. It is unphysical to have $R_{\rm b}$ not equal to 1. Nevertheless we allow for the possibility of different values of $R_{\rm b}$, in order to mimick out of equilibrium plasmas. For $R_{\rm b} < 1$, we have an ionizing plasma, for $R_{\rm b} > 1$ a recombining plasma. Note that for ionizing plasmas SPEX has also the nei model, which takes into account explicitly the effects of transient (time dependent) ionization.

It is also possible to mimick the effects of non-isothermality in a simple way. SPEX allows for a Gaussian emission measure distribution

$$Y(x) = \frac{Y_0}{\sqrt{2\pi}\sigma_T} e^{-(x\,-\,x0)^2/2\sigma_T^2} \tag{4.5}$$

where Y_0 is the total, integrated emission measure. By default $x \equiv \log T$ and $x_0 \equiv \log T_0$ with T_0 the average temperature of the plasma (this is entered at the "T" value in SPEX). However, this can be changed to $x \equiv T$ and $x_0 \equiv T_0$ by setting logt to 0. If the parameter sup is set $> 10^{-5}$, then the Gaussian emission measure distribution model becomes asymmetric. The sig parameter determines the slope of the low-temperature part of the Gaussian and sup determines the high-temperature side. Usually (default) $\sigma_T = 0$ and in that case the normal isothermal spectrum is chosen. Note that for larger values of σ_T the cpu time may become larger due to the fact that the code has to evaluate many isothermal spectra.

4.6.2 Line broadening

Apart from line broadening due to the thermal velocity of the ions (caused by $T_\mathrm{i} > 0$) it is also possible to get line broadening due to (micro)turbulence. In this case the line broadening is determined by v_micro, which is given by $v_\mathrm{micro} \equiv \sqrt{2}\sigma_v$ with σ_v the Gaussian velocity dispersion in the line of sight. Thus, without thermal broadening the FWHM of the line would be 1.6651 times v_micro.

4.6.3 Density effects

It is also possible to vary the electron density n_e of the plasma. This does not affect the overall shape of the spectrum (i.e., by changing n_e only SPEX still uses the previous value of the emission measure $Y \equiv n_\mathrm{H} n_\mathrm{e} V$), but spectral lines that are sensitive to the electron density will get different intensities. Usually this occurs for higher densities, for example under typical ISM conditions ($n_\mathrm{e} = 1$ cm^{-3}) this is not an important effect.

4.6.4 Non-thermal electron distributions

The effects of non-thermal electron distribution on the spectrum can be included. See Sect. 5.5 for more details.

4.6.5 Abundances

The abundances are given in Solar units. Which set of solar units is being used can be set using the "abun" command (see Sect. 3.2). For spectral fitting purposes it is important to distinguish two situations.

In the first case there is a strong thermal continuum. Since in most circumstances the continuum is determined mainly by hydrogen and helium, and the X-ray lines are due to the other elements, the line to continuum ratio is a direct measurement of the metal abundances compared to H/He. In this situation, it is most natural to have the hydrogen abundance fixed and allow only for fitting of the other abundances (as well as the emission measure).

In the other case the thermal continuum is weak, but there are strong spectral lines. Measuring for example the Fe abundance will give large formal error bars, not because the iron lines are weak but because the continuum is weak. Therefore, all abundances will get rather large error bars, and despite the fact of strong O and Fe lines, the formal error bars on the O/Fe ratio becomes large. This can be avoided by choosing another reference element, preferentially the one with the strongest lines (for example Fe). Then the O/Fe ratio will be well constrained, and it is now only the H abundance relative to Fe that is poorly constrained. In this case it is important to keep the nominal abundance of the reference element to unity. Also keep in mind that in general we write for the normalisation $n_\mathrm{e} n_\mathrm{X} V$ in this case; when the reference element is H, you mays substitute X=H; however when it is another element, like O, the normalisation is still the product of $n_\mathrm{e} n_\mathrm{X} V$ where X stands for the fictitious hydrogen density derived from the solar O/H ratio.

Example: suppose the Solar O abundance is 1E-3 (i.e. there is 0.001 oxygen atom per hydrogen atom in the Sun). Take the reference atom oxygen ($Z = 8$). Fix the oxygen abundance to 1. Fit your spectrum with a free hydrogen abundance. Suppose the outcome of this fit is a hydrogen abundance of 0.2 and an emission measure 3 (in SPEX units). This means $n_\mathrm{e} n_\mathrm{X} V = 3 \times 10^{64}$ m^{-3}. Then the "true" emission measure $n_\mathrm{e} n_\mathrm{H} V = 0.2 \times 3 \times 10^{64}$ m^{-3}. The nominal oxygen emission measure is $n_\mathrm{e} n_\mathrm{O} V = 0.001 \times 3 \times 10^{64}$ m^{-3}, and nominally oxygen would have 5 times overabundance as compared to hydrogen, in terms of solar ratios.

4.6.6 Parameter description

WARNING: *When you make the temperature too low such that the plasma becomes completely neutral, the model will crash. This is because in that case the electron density becomes zero, and the emission measure*

is undefined. The nominal temperature limits that are implemented in SPEX usually avoid that condition, but the results may depend somewhat upon the metallicity because of the charge exchange processes in the ionisation balance.

WARNING: *In high resolution spectra, do not forget to couple the ion temperature to the electron temperature, as otherwise the ion temperature might keep its default value of 1 keV during spectral fitting and the line widths may be wrong.*

WARNING: *Some people use instead of the emission measure $Y \equiv n_{\rm H} n_{\rm e} V$, the quantity $Y' = n_{\rm e}^2 V$ as normalisation. This use should be avoided as the emission is proportional to the product of electron and ion densities, and therefore use of Y' makes the spectrum to depend nonlinear on the elemental abundances (since an increase in abundances also affects the $n_{\rm e}/n_{\rm H}$ ratio).*

WARNING: *The default line broadening is just Doppler broadening. This is fine and self-consistent for the 'old' line calculation. To incorporate the natural line broadeing for the 'new' calculations, the user must use the* var dopp 4 *option to get Voigt profiles. This is physically better but takes more computation time.*

The parameters of the model are:

`norm` - the normalisation, which is the emission measure $Y \equiv n_{\rm H} n_{\rm e} V$ in units of 10^{64} m^{-3}, where $n_{\rm e}$ and $n_{\rm H}$ are the electron and Hydrogen densities and V the volume of the source. Default value: 1.
`t` - the electron temperature $T_{\rm e}$ in keV. Default value: 1.
`sig` - the width σ_T of the gaussian emission measure profile. Default value: 0. (no temperature distribution i.e. isothermal)
`sup` - the width σ_T of the high-temperature part of the gaussian emission measure profile. If larger than 10^{-5} keV, the sig parameter becomes the sigma value for the low-temperature end. Default value: 0
`logt` - Switch between linear and logarithmic temperature scale for the gaussian emission measure profile. Default value: 1 (logarithmic)
`ed` - the electron density $n_{\rm e}$ in units of 10^{20} m^{-3} (or 10^{14} cm^{-3}). Default value: 10^{-14}, i.e. typical ISM conditions, or the low density limit.
`it` - the ion temperature $T_{\rm i}$ in keV. Default value: 1
`rt` - the ratio of ionization balance to electron temperature, $R_{\rm b} = T_{\rm b}/T_{\rm e}$ in keV. Default value: 1.
`vmic` - the (micro)turbulent velocity $v_{\rm micro}$, in km/s. Default value 0.
`ref` - reference element. Default value 1 (hydrogen). See above for more details. The value corresponds to the atomic number of the reference element.
`01` - Abundance of hydrogen (H, Z=1) in Solar units. Default 1.
`02` - Abundance of helium (He, Z=2) in Solar units. Default 1.

...
`30` - Abundance of zinc (Zn, Z=30) in Solar units. Default 1.
`file` - Filename for the nonthermal electron distribution. If not present, nonthermal effects are not taken into account (default).

Recommended citation: Kaastra et al. (1996a).

4.7 Comt: comptonisation model

This is the model for Comptonisation of soft photons in a hot plasma, developed by Titarchuk (1994). See the XSPEC manual for more details. The code was substantially improved and brought to the SPEX standards.

4.7.1 Some modifications

Titarchuk (1994) gives an analytical approximation for $\beta(\tau)$ to the exact calculations as provided by Sunyaev & Titarchuk (1985) for $0.1 < \tau < 10$. Their approximation has the proper asymptotic behaviour

for small and large values of τ, but unfortunately has deviations up to 7 % over the $0.1 - 10$ range for τ. The approximation by Titarchuk is given in their equations (27) and (28) by

$$\text{disk:}\beta = \frac{\pi^2}{12(\tau + 2/3)^2}(1 - \mathrm{e}^{-1.35\tau}) + 0.45\mathrm{e}^{-3.7\tau} \ln \frac{10}{3\tau}, \tag{4.6}$$

$$\text{sphere:}\beta = \frac{\pi^2}{3(\tau + 2/3)^2}(1 - \mathrm{e}^{-0.7\tau}) + \mathrm{e}^{-1.4\tau} \ln \frac{4}{3\tau}. \tag{4.7}$$

We found an approximation that is better than 1 % using a slight modification of the above equations. We use these new approximations in our calculations:

$$\text{disk:}\beta = \frac{0.8351}{(\tau + 0.867)^2}(1 - \mathrm{e}^{-3.876\tau}) + 0.572\mathrm{e}^{-3.75\tau} \ln \frac{1.06}{\tau}, \tag{4.8}$$

$$\text{sphere:}\beta = \frac{3.504}{(\tau + 1.2)^2}(1 - \mathrm{e}^{-2.09\tau}) + 1.013\mathrm{e}^{-2.2\tau} \ln \frac{1.6}{\tau}. \tag{4.9}$$

WARNING: *For the physically valid range of parameters, consult the XSPEC manual; see also Hua & Titarchuk (1995), their Fig. 7.*

The parameters of the model are:

`norm` - Normalisation A of the power law, in units of 10^{44} ph s^{-1} keV^{-1}. Due to the way the model is calculated, this is not the flux at 1 keV! Default value: 1.
`t0` - The temperature T of the seed photons in keV. Default value: 1 keV.
`t1` - The plasma temperature T in keV. Default value: 100 keV.
`tau` - The optical depth. Default value: 3.
`type` - The type of geometry. Allowed values are 0 for a disk (default) or 1 for a sphere. This is a frozen parameter (cannot be fitted).

Recommended citation: Titarchuk (1994)

4.8 CX: model for charge exchange plasmas

This model calculates the spectrum emitted from a hot plasma when it recombines with cold neutral materials. This model is based on three key assumptions: (1) it considers only single electron capture in ion-neutral collision; (2) all cross section data are obtained only with atomic hydrogen target, (3) electronic collisional excitation and recombination are ignored in the spectral calculation. More informations can be found in Gu et al. (2016).

4.8.1 Charge exchange cross sections

The CX cross section data used in the model are partly taken from literatures, including quantum molecular-orbital close-coupling calculations for C^{5+} and O^{6+} by Wu et al. (2012) and Nolte et al. (2012), multi-channel Laudau-Zener results for Fe^{25+} and Fe^{26+} by Mullen (2016), other data compilations for C^{6+} and O^{8+} by Janev et al. (1993), and NIFS Charge Transfer Database (CHART) [2] for Be^{4+}, B^{5+}, N^{7+}, and Ne^{10+}. For CHART database, we extracted all the data, from both theoretical calculations and experiments (see a full list in Table 1 of Gu et al. (2016)), and fitted them with Eq.2 of Gu et al. (2016) in the energy range of interests. In typical astrophysical velocity range ($\sim 100 - 5000$ km s^{-1}), the useful CHART data are usually from molecular-orbital and atomic-orbital close-coupling methods, and a few classical trajectory Monte Carlo calculations. All the above data are dependent on collision energy, and resolved to levels described by quantum number n and l.

For ions not available in public sources, we developed a new method to interpolate by analyzing the known ions. First we used a scaling law to determine total cross section for each ion, and applied another scaling

[2] http://dbshino.nifs.ac.jp/

law to represent the $n-$ selectivity. The $l-$ dependence is approximated by one of the five empirical weighting functions presented in Eqs.4 – 8 of Gu et al. (2016).

WARNING: *CX model only works with the updated atomic database set through the command " var calc new".*

WARNING: *All Beryllium-like sequence ions are not included in the current version; will be available later.*

WARNING: *We will keep updating the CX model when new data (especially for molecular targets) from theoretical calculations and experiments become available.*

4.8.2 Parameter description

The parameters of the CX model are:

`norm` - the normalisation, which is the emission measure $Y \equiv n_{\rm H} n_{\rm nh} V$ in units of 10^{64} m^{-3}, where $n_{\rm H}$ and $n_{\rm nh}$ are the Hydrogen densities of the ionized and neutral materials, respectively, and V is the effective interaction volume. Default value: 1.
`hden` - Hydrogen density of the neutral materials in units of 10^{20} m^{-3} (or 10^{14} cm^{-3}). Default value: 10^{-14}.
`mode` - Switch between a hot-cold interaction driven by thermal motion of hot plasma, and the one dominated by flow velocity. Default value: 2 (kinematic).
`t` - the ionization temperature of hot matter in keV. It is also used to approximate the thermal motion when mode is set to 1. Default value: 1.
`sig` - the width σ_T of the gaussian emission measure profile. Default value: 0. (no temperature distribution i.e. isothermal)
`sup` - the width σ_T of the high-temperature part of the gaussian emission measure profile. If larger than 10^{-5} keV, the sig parameter becomes the sigma value for the low-temperature end. Default value: 0
`logt` - Switch between linear and logarithmic temperature scale for the gaussian emission measure profile. Default value: 1 (logarithmic)
`zv` - Collision velocity in unit of km s^{-1}, used when mode is set to 2. Default value: 100
`op` - Switch between single and multiple collisions for each ion. In multiple collision case, one ion would continuously undergo CX and produce various emission lines, until it becomes neutral. Default: 1 (single)
`wt` - Weighting functions for subshell $l-$ population. When wt is set to 1, the $l-$ population is approximated by a series of empirical functions that switchs from one to another as a function of collision velocity. See Gu et al. (2016) for details. These empirical functions are defined in Eqs.4 – 8 of Gu et al. (2016), and will be selected when wt is set to 2 – 6, respectively. Default: 1
`vmic` - the (micro)turbulent velocity $v_{\rm micro}$, in km/s. Default value 0.
`ref` - reference element. Default value 1 (hydrogen). See above for more details. The value corresponds to the atomic number of the reference element.
`01` - Abundance of hydrogen (H, Z=1) in Solar units. Default 1.
`02` - Abundance of helium (He, Z=2) in Solar units. Default 1.

...
`30` - Abundance of zinc (Zn, Z=30) in Solar units. Default 1.
`file` - Filename for the nonthermal distribution. If not present, nonthermal effects are not taken into account (default).

Recommended citation: Gu et al. (2016).

4.9 Dabs: dust absorption model

This model allows the user to maake a zeroth order approximation to the effects of dust on an X-ray absorption spectrum. basically, compared to absorption by atomic gas, the effects of dust are two-fold:

- Modifications to the fine structure near the edge
- Reduced opacity due to self-shielding in grains

The *dabs* model only accounts for the second effect, reduced opacity due to self-shielding. The formalism of Wilms et al. (2000) is followed, and we refer to that paper for a full explanation.

In the set of abundances, we use the depletion factors of Wilms et al. (2000).

H	He	Li	Be	B	C	N	O	F	Ne	Na	Mg	Al	Si	P
0	0	1	0	0	0.5	0	0.4	0	0	0.75	0.8	0.98	0.9	0.4
S	Cl	Ar	K	Ca	Sc	Ti	V	Cr	Mn	Fe	Co	Ni	Cu	Zn
0.4	0.5	0	0	0.997	0	0.998	0	0.97	0.93	0.7	0.95	0.96	0	0

The numbers represent the fraction of the atoms that are bound in dust grains. Noble gases like Ne and Ar are chemically inert hence are generally not bound in dust grains, but other elements like Ca exist predominantly in dust grains.

WARNING: *For a realistic model, it is advised to combine the dabs model with a hot model, where (for Solar abundances), the hot model should contain the elemnts not bound in the dust.*

Finally, we note that the *amol* model can deal with the modified structure near the oxygen edge, but that model does not take elf-shielding into account.

The parameters of the model are:

`nh` - Nominal hydrogen column density in 10^{28} m^{-2}. Default value: 1.
`amin` - Minimum grain size in μm. Default value: 0.025.
`amax` - Minimum grain size in μm. Default value: 0.25.
`p` - Power index grain size distribution. Default value: 3.5.
`rho` - Mean density of the grains in units of 1000 $\mathrm{kg\,m}^{-3}$. Default value: 3.
`ref` - Reference atom. Default value: 3 (Lithium). because dust, according to the depletion factors used, does not contain hydrogen, we substitute here formally Lithium. The user is advisednot to change this.
`01...30` - Abundances of H to Zn. See the remark on depletion factors above.

The following parameters are common to all our absorption models:
`fcov` - The covering factor of the absorber. Default value: 1 (full covering)
`zv` - Average systematic velocity v of the absorber

Recommended citation: Wilms et al. (2000) and Kaastra et al. (2009).

4.10 Dbb: disk blackbody model

We take the model for a standard Shakura-Sunyaev accretion disk. The radiative losses from such a disk are given by

$$Q = \frac{3GM\dot{M}(1-\sqrt{r_i/r})}{8\pi r^3}, \tag{4.10}$$

where Q is the loss term in $\mathrm{W\,m}^{-2}$ at radius r, M the mass of the central object, $\dot{M}$ the accretion rate through the disk and r_i the inner radius. If this energy loss is radiated as a blackbody, we have

$$Q = \sigma T^4 \tag{4.11}$$

with σ the constant of Stefan-Boltzmann and $T(r)$ the local temperature of the blackbody. The total spectrum of such a disk is then obtained by integration over all radii. We do this integration numerically. Note that for large r, $T \sim r^{-3/4}$.

WARNING: *A popular disk model (diskbb in XSPEC) assumes this temperature dependence over the full disk. However, we correct it using the factor* $(1-\sqrt{r_i/r})$ *in* Q *which corresponds to the torque-free condition at the inner boundary of the disk.*

The photon spectrum of the disk is now given by

$$N(E) = \frac{8\pi^2 E^2 r_i^2 \cos i}{h^3 c^2} f_d(E/kT_i, r_o/r_i), \tag{4.12}$$

where i is the inclination of the disk (0 degrees for face-on disk, 90 degrees for an edge-on disk), E the photon energy, r_o the outer edge of the disk, and T_i is defined by

$$T_i^4 = 3GM\dot{M}/8\pi r_i^3\sigma \tag{4.13}$$

and further the function $f_d(y, r)$ is defined by

$$f_d(y, r) = \int_1^r \frac{x\mathrm{d}x}{e^{y/\tau} - 1} \tag{4.14}$$

where $\tau(x)$ is defined by $\tau^4(x) = (1 - 1/\sqrt{x})/x^3$.

In addition to calculating the spectrum, the model also allows to calculate the average radius of emission R_e at a specified energy E_r. This is sometimes useful for time variability studies (softer photons emerging from the outer parts of the disk).

Given the fit parameters T_i and r_i, using (4.13) it is straightforward to calculate the product $M\dot{M}$. Further note that if $r_i = 6GM/c^2$, it is possible to find both M and $\dot{M}$, provided the inclination angle i is known.

The parameters of the model are:

`norm` - Normalisation A $(= r_i^2 \cos i)$, in units of 10^{16} m^2. Default value: 1.
`t` - The nominal temperature T_i in keV. Default value: 1 keV.
`ro` - The ratio of outer to inner disk radius, r_o/r_i
`ener` - Energy E_r at which the average radius of emission will be calculated
`rav` - Average radius R_e of all emission at energy E_r specified by the parameter above. Note that this is not a free parameter, it is calculated each time the model is evaluated.

Recommended citation: Shakura & Sunyaev (1973).

4.11 Delt: delta line model

The delta line model is a simple model for an infinitely narrow emission line.

The spectrum is given by:

$$F(E) = A\delta(E - E_0), \tag{4.15}$$

where E is the photon energy in keV, F the photon flux in units of 10^{44} ph s^{-1} keV^{-1}, E_0 is the line energy of the spectral line (in keV) and A is the line normalisation (in units of 10^{44} ph s^{-1}). The total line flux is simply given by A.

To ease the use for dispersive spectrometers (gratings) there is an option to define the wavelength instead of the energy as the basic parameter. The parameter *type* determines which case applies: type=0 (default) corresponding to energy, type=1 corresponding to wavelength units.

WARNING: *When changing from energy to wavelength units, take care about the frozen/thawn status of the line centroid and FWHM*

WARNING: *You need to do a "calc" or "fit" command to get an update of the wavelength (for type=0) or energy (type=1).*

The parameters of the model are:

`norm` - Normalisation A, in units of 10^{44} ph s^{-1}. Default value: 1.
`e` - The line energy E_0 in keV. Default value: 6.4 keV.
`type` - The type: 0 for energy units, 1 for wavelength units.
`w` - The line wavelength λ in Å. Default value: 20 Å.

4.12 Dem: differential emission measure model

This model calculates a set of isothermal CIE spectra, that can be used later in one of the DEM analysis tools (see the "dem" commands of the syntax). The CIE spectra are evaluated for a logarithmic grid of temperatures between T_1 and T_2, with n bins.

WARNING: *For the DEM methods to work, the dem model is the only allowed additive component that can be present. No other additive components are allowed. But of course the spectrum of the dem model may be modified by applying any combination of multiplicative models (redshifts, absorptions, line broadening, etc.).*

WARNING: *Because of the above, do not use the "fit" commands when you have a dem model. If you really need to fit, use the pdem model instead.*

WARNING: *In general, the spacing between the different temperature components should not be smaller than 0.10 in* $\log T$*. Smaller step sizes will produce unstable solutions.*

The parameters of the model are:

`t1` - Lower temperature T_1 in keV. Default value: 0.001 keV.
`t2` - Upper temperature T_2 in keV. Default value: 100 keV.
`nr` - Number of temperature bins. Default value: 64.

The following parameters are the same as for the cie-model:
`ed` - Electron density in 10^{20} m^{-3}
`it` - Ion temperature in keV
`vmic` - Micro turbulent velocity in km/s
`ref` - Reference element
`01...30` - Abundances of H to Zn
`file` - Filename for the nonthermal electron distribution

Recommended citation: Mewe et al. (1995).

4.13 Dust: dust scattering model

This model calculates the effective transmission of dust that scatters photons out of the line of sight. No re-emission is taken into account, i.e. it is assumed that all scattered photons are lost from the line of sight. This makes the model useful for instruments with a high spatial resolution (i.e, the spatial resolution is much better than the typical size of the dust scattering halo).

Dust scattering is described for example by Draine (2003). In that paper, a link is given to the website of Draine (http://www.astro.princeton.edu/ draine/dust/dustmix.html) This website contains slightly updated cross sections for three cases as listed below. The scattering is computed for a Carbonaceous - Silicate Model for Interstellar Dust. The cases are:

1. set=1: for $R_V = 3.1$;
2. set=2: for $R_V = 4.0$;
3. set=3: for $R_V = 5.5$.

WARNING: *For any instrument where the extraction region has a size comparable to the size of the dust scattering halo, this model should not be used, as the scattered X-rays will fall within the exctraction region. Take care when fitting data from different instruments simultaneously.*

WARNING: *This model only calculates the dust scattering, not the dust absorption*

The parameters of the model are:

`nh` - Hydrogen column density in 10^{28} m^{-2}. Default value: 10^{-4} (corresponding to 10^{24} m^{-2}, a typical value at low Galactic latitudes).
`f` - The covering factor of the absorber. Default value: 1 (full covering)
`set` - The set of cross sections being used. See table above.

Recommended citation: Draine (2003).

4.14 Ebv: Galactic interstellar extinction model

This model can be used to correct fluxes of optical/UV data for interstellar reddening in our Galaxy. The reddening is caused by dust absorption and scattering (extinction) in the interstellar medium. The model uses the extinction curve of Cardelli et al. (1989), including the update for near-UV given by O'Donnell (1994).
The extinction law can be expressed as:

$$(A_\lambda/A_V) = a_\lambda + (b_\lambda/R_V) \tag{4.16}$$

where, A_λ and A_V are the total extinctions (measured in magnitude) at wavelength λ and the V-band, respectively. The wavelength- dependent parameters a_λ and b_λ are provided by the aforementioned papers. The scalar R_V is defined as the ratio of total extinction A_V to selective extinction $A_B - A_V$, where A_B is the total extinction at the B-band. So the selective extinction $A_B - A_V$ represents a colour excess, which is commonly denoted as $E(B-V)$. For Milky Way, the typical value for R_V is reported to be 3.1. Thus, by specifying $E(B-V)$ and R_V, the extinction A_λ can be derived, which in turn gives us the unreddened flux (F_λ) from the observed reddened flux (F^*_λ) using $F_\lambda = (10^{\,0.4A_\lambda})\,F^*_\lambda$.

Note that at energies above the Lyman limit, the transmission of the model is set to 1 in SPEX, thus it can be used alongside the Galactic interstellar X-ray absorption models in SPEX.

The parameters of the model are:
`ebv` - The colour excess $E(B-V)$. The value is set by the user.
`rv` - The scalar R_V. Default (recommended) value: 3.1
`fcov` - The covering factor of the absorber. Default value: 1 (full covering)

Recommended citation: Cardelli et al. (1989) and O'Donnell (1994).

4.15 Etau: simple transmission model

This model calculates the transmission $T(E)$ between energies $E1$ and E_2 for a simple (often unphysical!) case:

$$T(E) = e^{-\tau(E)}, \tag{4.17}$$

with the optical depth $\tau(E)$ given by:

$$\tau(E) = \tau_0 E^a. \tag{4.18}$$

In addition, we put here $T \equiv 1$ for $E < E_1$ and $E > E_2$, where E_1 and E_2 are adjustable parameters. This allows the user for example to mimick an edge. Note however, that in most circumstances there are more physical models present in SPEX that contain realistic transmissions of edges! If you do not want or need edges, simply keep E_1 and E_2 at their default values.

Note that τ_0 should be non-negative. For $a > 0$ the spectrum has a high-energy cut-off, for $a < 0$ it has a low-energy cut-off, and for $a = 0$ the transmission is flat. The larger the value of a, the sharper the cut-off is.

The model can be used as a basis for more complicated continuum absorption models. For example, if the optical depth is given as a polynomial in the photon energy E, say for example $\tau = 2 + 3E + 4E^2$, one may define three *etau* components, with τ_0 values of 2, 3, and 4, and indices a of 0, 1 and 2. This is because of the mathematical identity $e^{-(\tau_1+\tau_2)} = e^{-\tau 1} \times e^{-\tau 2}$.

The parameters of the model are:

`tau0` - Optical depth τ_0 at $E = 1$ keV. Default value: 1.
`a` - The index a defined above. Default value: 1.
`e1` - Lower energy E_1 (keV). Default value: 10^{-20}.
`e2` - Upper energy E_2 (keV). Default value: 10^{20}.
`f` - The covering factor of the absorber. Default value: 1 (full covering)

4.16 Euve: EUVE absorption model

This model calculates the transmission of gas with cosmic abundances as published first by Rumph et al. (1994). It is a widely used model for the transimission at wavelenghts covered by the EUVE satellite ($\lambda > 70$ Å). As it these wavelengths metals are relatively unimportant, it only takes into account hydrogen and helium, but for these elements it takes into account resonances. However it is not recommended to use the model for harder X-rays, due to the neglect of metals such as oxygen etc.

Otherwise the user is advised to use the "absm" model or the "hot" model with low temperature in SPEX, which gives the transmission of a slab in collisional ionisation equilibrium.

The parameters of the model are:

`nh` - Hydrogen column density in 10^{28} m^{-2}. Default value: 10^{-4} (corresponding to 10^{24} m^{-2}, a typical value at low Galactic latitudes).
`he1` - The He I / H I ratio. Default value: 0.1.
`he2` - The He II / H I ratio. Default value: 0.01.
`f` - The covering factor of the absorber. Default value: 1 (full covering)

Recommended citation: Rumph et al. (1994).

4.17 File: model read from a file

This model reads a spectrum from a file. The user inputs a spectrum at n energy bins ($n > 1$). The file containing the spectrum should have the follwing format:

- first line: n
- second line: E_1, S_1
- third line: E_2, S_2
- ...
- last line: E_n, S_n

Here E_i is the energy of a photon in keV, and S_i is the spectrum in units of 10^{44} photons $\mathrm{s}^{-1}\,\mathrm{keV}^{-1}$. All energies E_i must be positive, and the grid must be monotonically increasing (two subsequent energies may not have the same value). Also, the spectrum S_i must be strictly positive (i.e. $S_i = 0$ is not allowed).

SPEX then calculates the spectrum by linear interpolation in the $\log E - \log S$ plane (this is why fluxes must be positive). For $E < E_1$ and $E > E_n$ however, the spectrum is put to zero. Finally, the spectrum is multiplied by the scale factor N prescribed by the user.

The parameters of the model are:

`norm` - Normalisation factor N.Default value: 1.
`file` - The file name of the file containing the spectrum

4.18 Gaus: gaussian line model

The Gaussian line model is the simplest model for a broadened emission line.

The spectrum is given by:

$$F(E) = Ae^{(E-E_0)^2/2\sigma^2}, \tag{4.19}$$

where E is the photon energy in keV, F the photon flux in units of 10^{44} ph $\mathrm{s}^{-1}\,\mathrm{keV}^{-1}$, E_0 is the average line energy of the spectral line (in keV) and A is the line normalisation (in units of 10^{44} ph s^{-1}). The total line flux is simply given by A. Further σ is the Gaussian width, which is related to the full width at half maximum $FWHM$ by $FWHM = \sigma\sqrt{\ln 256}$ or approximately $FWHM = 2.3548\sigma$.

To ease the use for dispersive spectrometers (gratings) there is an option to define the wavelength instead of the energy as the basic parameter. The parameter *type* determines which case applies: type=0 (default) corresponding to energy, type=1 corresponding to wavelength units.

WARNING: *Do not confuse σ with $FWHM$ when interpreting your fit results with other data.*

WARNING: *When changing from energy to wavelength units, take care about the frozen/thawn status of the line centroid and FWHM*

WARNING: *You need to do a "calc" or "fit" command to get an update of the wavelength (for type=0) or energy (type=1).*

The parameters of the model are:

`norm` - Normalisation A, in units of 10^{44} ph s^{-1}. Default value: 1.
`e` - The line energy E_0 in keV. Default value: 6.4 keV.
`fwhm` - The line $FWHM$, in keV.
`type` - The type: 0 for energy units, 1 for wavelength units.
`w` - The line wavelength λ in Å. Default value: 20 Å.
`awid` - The line $FWHM$, in Å.

Recommended citation: Gauss (1809).

4.19 Hot: collisional ionisation equilibrium absorption model

This model calculates the transmission of a plasma in collisional ionisation equilibrium with cosmic abundances.

For a given temperature and set of abundances, the model calculates the ionisation balance and then determines all ionic column densities by scaling to the prescribed total hydrogen column density. Using this set of column densities, the transmission of the plasma is calculated by multiplying the transmission of the individual ions.

The transmission includes both continuum and line opacity. For a description of what is currently in the absorption line database, we refer to Sect. 5.2. You can mimick the transmission of a neutral plasma very easy by putting the temperature to 0.5 eV ($5\ 10^{-4}$ keV).

WARNING: *For solar abundances, do not take the temperature much lower than 0.0005 keV, because if the plasma is completely neutral, the code will crash; a tiny fraction of ions such as Fe* II *or Na* II *will help to keep a few free electrons in the gas without affecting the transmission too much. You can check the ion concentrations by giving an "asc ter ... icon" command. Fill in the sector and component number of the hot component for the ... in the "asc ter ... icon" command to get the values for the right component.*

The parameters of the model are:

`nh` - Hydrogen column density in 10^{28} m^{-2}. Default value: 10^{-4} (corresponding to 10^{24} m^{-2}, a typical value at low Galactic latitudes).
`t` - the electron temperature T_e in keV. Default value: 1.
`rt` - the ratio of ionization balance to electron temperature, $R_\mathrm{b} = T_\mathrm{b}/T_\mathrm{e}$ in keV. Default value: 1.

The following parameters are common to all our absorption models:
`f` - The covering factor of the absorber. Default value: 1 (full covering)
`v` - Root mean square velocity σ_v
`rms` - Rms velocity σ_b of line blend components
`dv` - Velocity distance Δv between different blend components
`zv` - Average systematic velocity v of the absorber

The following parameters are the same as for the cie-model (see there for a description):
`ref` - Reference element
`01...30` - Abundances of H to Zn
`file` - Filename for the nonthermal electron distribution

Recommended citation: de Plaa et al. (2004) and Steenbrugge et al. (2005a).

4.20 Hyd: model with user-own hydrodynamical simulation

This model calculates the spectrum of a plasma with a given set of ion concentrations. It allows users to calculate the X-ray spectrum corresponding to his own hydrodynamical simulation results. The basic usage is that the user first runs own calculations to obtain the ion concentrations, and stores them in an ascii file. The name should be either of "spexicon_abs.dat" or "spexicon_rel.dat". Then the user runs SPEX and loads the Hyd model to calculate the spectrum.

For users who are familiar with Fortran, we offer the supporting fortran90 subroutine, *hydro_driver*, to make the ion concentration file conveniently. The usage of the *hydro_driver* is described in Section 7.5.

For more general cases users can directly load the Hyd model and just calculate the spectrum. The model has two modes to specify the format for the ion concentrations: absolute (mode 1) or relative (mode 2). For mode 1, the input file must have the name "spexicon_abs.dat", and the values are treated as absolute ion concentrations (relative to hydrogen). In this mode the parameters of the elemental abundances do not affect the ion concentration nor the spectrum at all (they are ignored). For the case that users wish to treat elemental abundances as fitting parameters, mode=2 can be used. In this mode the input file must have the name "spexicon_rel.dat", which contains the ion concentrations relative to the concentration of the relevant chemical element.

The parameters of the model are:
`hyd` - Hydrogen density in 10^{20} m^{-3}
`mode` - Mode of the model. Mode=1: read absolute ion concentration from "spexicon_abs.dat". mode=2: read relative ion concentration from "spexicon_rel.dat"

The following parameters are the same as for the cie-model:
`norm` - the normalisation, which is the emission measure $Y \equiv n_e n_H V$ in units of 10^{64} m^{-3}, where n_e and n_H are the electron and Hydrogen densities and V the volume of the source. Default value: 1.
`t` - the electron temperature T_e in keV. Default value: 1.
`it` - Ion temperature in keV
`vmic` - Micro turbulent velocity in km/s
`ref` - Reference element
`01...30` - Abundances of H to Zn
`file` - Filename for the nonthermal electron distribution

WARNING: *By default, SPEX starts with "var calc old" (see the var menu for explanation). If you want to use this model with the latest atomic database, you should set the ionisation balance to U16. Note that this is only used to calculate inner-shell ionisation rates for the spectral evaluation, it will not affect the ion concentrations that the user provides.*

Recommended citation (first use): Kosenko et al. (2015).

4.21 Knak: segmented power law transmission model

The knak model is used to model the transmission of any spectrum using a set of contiguous segments with a power-law transmission at each segment.

This component can be useful for new instruments, in order to test large scale calibration errors (effective area errors for example), but other applications can also be made, of course. For example, if the spectrum of the source has an unknown continuum shape with a superimposed absorption model, it is a good trick to model the continuum by a power law, modify that by a knak model with adjustable or fixed transmission at some relevant energies, and then apply the absorption. An example of this last application can be found in Porquet et al. (2004).

The Transmission is given by:

$$T(E) = c_i E^{a_i} \qquad \text{if } E_i < E < E_{i+1} \tag{4.20}$$

for each value of i between 0 and n, the number of grid points. The transmission is 1 for $E < E_1$ and $E > E_2$. Further, instead of using the constants c_i and a_i, we use instead the values of the transmission at E_i, i.e. $T_i \equiv T(E_i) = c_i E_i^{a_i}$. This allows for a continuous connection between neighbouring segments.

Finally, for historical reasons we use here a wavelength grid instead of an energy grid; the corresponding nodes λ_i should therefore be in strictly increasing order.

WARNING: *When applying this model, take care that at least one of the n transmission values is kept fixed (otherwise you may run the risk that your model is unconstrained, for example if the normalisation of the continuum is also a free parameter).*

The parameters of the model are:

`n` - The number of grid points. Maximum value is 9.
`w1` - Wavelength λ_1 (Å) of the first grid point
`f1` - Transmission $T(\lambda_1)$ at λ_1.
`w2` - Wavelength λ_2 (Å) of the second grid point
`f1` - Transmission $T(\lambda_2)$ at λ_2.
- ...
`w9` - Wavelength λ_9 (Å) of the last grid point
`f9` - Transmission $T(\lambda_9)$ at λ_9.

Note that if $n < 9$, the values of T_i and λ_i will be ignored for $i > n$.

Recommended citation: Porquet et al. (2004).

4.22 Laor: relativistic line broadening model

This multiplicative model broadens an arbitrary additive component with a relativistic line profile. The relativistic line profile is interpolated from tables produced by Laor (1991). In his model, the line transfer function is determined for emission from the equatorial plane around a Kerr black hole, assuming an emissivity law $I(\cos\theta) \sim 1. + 2.06\cos\theta$. The transfer function is calculated at a grid of 35 radii ($r_n = 1600/(n+1)^2$ for $n = 1, \ldots, 35$, in units of GM/c^2), 31 inclinations uniformly spaced in $\cos i$, and 300 frequencies, logarithmically spaced between 0.1 and 5 times the emission frequency, respectively.

Using these numbers, a radial integration is done using an emissivity law

$$I(r) \sim 1/(r^2 + h^2)^{q/2}, \tag{4.21}$$

where h is a characteristic scale height and q an asymptotic power law index (for large r, $I(r) \sim r^{-q}$). The integration is done between an inner radius r_1 and an outer radius r_2. Given the radial grid that is provided in Laor's data, the outer radius extends out to at most $400GM/c^2$.

WARNING: *Of course, any line or continuum emission component can be convolved with the this broadening model; for continuum components the computational time may be very large, however, due to the convolution involved.*

WARNING: *The outer radius cannot be larger than* $400GM/c^2$

The parameters of the model are:

`r1` - Inner radius of the disk, in units of GM/c^2. The minimum allowed value is 1.234 (for a maximally rotating Kerr black hole). For a Schwarzschild black hole, one should take $r_i = 6$. Default value: 1.234.
`r2` - Outer radius of the disk, in units of GM/c^2. Keep this radius less than 400 (default value)
`q` - Emissivity slope q as described above. Default value: 2.
`h` - Emissivity scale height. Default value: 0.
`i` - Inclination angle (in degrees) of the disk (angle between line of sight and the rotation axis of the disk). Default value: 45 degrees.

Recommended citation: Laor (1991).

4.23 Line: transmission model for a single spectral line

This model calculates the transmission $T(E)$ for a single spectral emission or absorption model. It is used as a multiplicative component, and can be used both for absorption and emission lines. The line profile is a Voigt profile (the convolution between a Gaussian and a Lorentzian profile):

$$T(E) = e^{-\tau(E)}, \tag{4.22}$$

with the optical depth $\tau(E)$ given by:

$$\tau(E) = \tau_0 K(x, y), \tag{4.23}$$

where $K(x, y)$ is the Voigt profile given by

$$K(x, y) = \frac{y}{\pi} \int_{-\infty}^{\infty} \frac{e^{-t^2}}{y^2 + (x - t)^2} \mathrm{d}t, \tag{4.24}$$

with

$$x = \frac{2\sqrt{\ln 2}}{F_\mathrm{g}}(E - E_0) \tag{4.25}$$

and

$$y = \sqrt{\ln 2}\frac{F_\mathrm{l}}{F_\mathrm{d}} \tag{4.26}$$

where F_g and F_l are the Full Width at Half Maximum of the Gaussian and Lorentzian components, respectively. Further, E_0 is the line centroid.

The program also calculates the equivalent width of the line.

The model can be used both in energy (keV) mode or wavelength (Å) mode. In each case, the quantities for the other scaling (centroid, equivalent width, and FWHMs of the Gaussian and Lorentzian components) are calculated. In addition, like all SPEX absorption models, also the covering factor can be varied.

WARNING: *The equivalent width is calculated at the energy grid in use for SPEX. Line flux outside this range is not taken into account. That may be important in the case of strong Lorentzian wings.*

WARNING: *The parameter τ_0 only represents the optical depth at the line centre for a pure Gaussian case. When the Lorentzian component becomes important, the "true" depth at line center is smaller.*

WARNING: *When changing from energy to wavelength units, take care about the frozen/thawn status of the line centroid and FWHM.*

WARNING: *You need to do a "calc" or "fit" command to get an update of the wavelength (for type=0) or energy (type=1).*

The parameters of the model are:

`tau0` - Optical depth τ_0 at line center. Default value: 1. Positive values correspond to absorption lines, negative values to emission lines.
`el` - Line center energy E_0 (keV). Default value: 6.4 keV.
`fwhg` - FWHM of the Gaussian component F_g in keV. Default value: 0.1 keV.
`fwhl` - FWHM of the Lorentzian component F_l in keV. Default value: 0.1 keV.
`type` - Type of calculation. Type=0: energy units; type=1: wavelength units.
`wl` - Line center wavelength E_0 (Å). Default value: 20 Å.
`awhg` - FWHM of the Gaussian component F_g in Å. Default value: 0.1 Å.
`awhl` - FWHM of the Lorentzian component F_l in Å. Default value: 0.1 Å.
`ewk` - Equivalent width of the line in keV. Calculated by the program, not an input variable.
`ewa` - Equivalent width of the line in Å. Calculated by the program, not an input variable.
`fcov` - The covering factor of the absorption/emission line. Default value: 1 (full covering).

4.24 Lpro: spatial broadening model

This multiplicative model broadens an arbitrary additive component with an arbitrarily shaped spatial profile, in the case of dispersive spectrometers such as the RGS of XMM-Newton. In many instances, the effects of a spatially extended source can be approximated by making use of the fact that for small off-axis angles θ the expression $\mathrm{d}\lambda/\mathrm{d}\theta$ is almost independent of wavelength λ. This holds for example for the RGS of XMM-Newton (for which $\mathrm{d}\lambda/\mathrm{d}\theta = 0.138/m$ Å/arcmin, with m the spectral order).

We can utilize this for a grating spectrum as follows. Make an image $I(\Delta\theta)$ of your source projected onto the dispersion axis, as a function of the off-axis angle $\Delta\theta$. From the properties of your instrument, this can be transformed into an intensity $I(\Delta\lambda)$ as function of wavelength using $\Delta\lambda = \Delta\theta \mathrm{d}\lambda/\mathrm{d}\theta$. Assume that the spatial profile $I(\theta)$ is only non-zero within a given angular range (i.e. the source has a finite extent). Then we can transform $I(\Delta\lambda)$ into a probability distribution $f(x)$ with $f = 0$ for very small or large values of x (here and further we put $x = \Delta\lambda$). The auxilliary task rgsvprof (see Section 7.6) is able to create an input file for the lpro component from a MOS1 image.

The resulting spatially convolved spectrum $S_c(\lambda)$ is calculated from the original spectrum $S(\lambda)$ as

$$S_c(\lambda) = \int f(\lambda - \lambda_0) S(\lambda_0) \mathrm{d}\lambda_0. \tag{4.27}$$

The function $f(x)$ must correspond to a probability function, i.e. for all values of x we have

$$f(x) \geq 0 \tag{4.28}$$

and furthermore

$$\int_{-\infty}^{\infty} f(x)\mathrm{d}x = 1. \tag{4.29}$$

In our implementation, we do not use $f(x)$ but instead the cumulative probability density function $F(x)$, which is related to $f(x)$ by

$$F(x) \equiv \int_{-\infty}^{x} f(y)\mathrm{d}y, \tag{4.30}$$

where obviously $F(-\infty) = 0$ and $F(\infty) = 1$. The reason for using the cumulative distribution is that this allows easier interpolation and conservation of photons in the numerical integrations.

If this component is used, you must have a file available which we call here "vprof.dat" (but any name is allowed). This is a simple ascii file, with n lines, and at each line two numbers: a value for x and the corresponding $F(x)$. The lines must be sorted in ascending order in x, and for $F(x)$ to be a proper probability distribution, it must be a non-decreasing function i.e. if $F(x_i) \leq F(x_{i+1})$ for all values of i between 1 and $n-1$. Furthermore, we demand that $F(x_1) \equiv 0$ and $F(x_n) \equiv 1$.

Note that $F(x)$ is dimensionless. In addition, we allow for two other parameters: a scale factor s and an offset λ_o. Usually, $s = 1$, but if s is varied the resulting broadening scales proportional to s. This is useful if for example one has an idea of the shape of the spatial profile, but wants to measure its width directly from the observed grating spectrum. In addition, the parameter λ_o can be varied if the absolute position of the source is unknown and a small linear shift in wavelength is necessary.

WARNING: *This model can be applied to grating spectra (like RGS), but if you include in your fit also other data (for example EPIC), the same broadening will also be applied to that other data SET. This can be avoided by using a separate sector for each detector type.*

WARNING: *The above approximation of spatially extended sources assumes that there are no intrinsic spectral variations over the surface area of the X-ray source. Only total intensity variations over the surface area are taken into account. Whenever there are spatial variations in spectral shape (not in intensity) our method is strictly speaking not valid, but still gives more accurate results than a point-source approximation. In principle in those cases a more complicated analysis is needed.*

The parameters of the model are:

`s` - Scale parameter s, dimensionless. Default value: 1.
`dlam` - Offset parameter λ_o, in Å. Default value: 0 Å.
`file` - Ascii character string, containing the actual name of the vprof.dat file

Recommended citation: Tamura et al. (2004).

4.25 Mbb: modified blackbody model

This model describes the spectrum of a blackbody mdified by coherent Compton scattering. This is in several instances a much better description than a simple blackbody (for example accretion disk spectra of AGN). The physical background is described for example by Rybicki & Lightman (1986), pages 218–219. The formulae that we use here with a derivation are given by Kaastra & Barr (1989). From that work we derive the spectrum (10^{44} photons/s/keV):

$$N(E) = 1358.\frac{AE^{0.25}}{e^{E/T}(e^{E/T}-1)} \tag{4.31}$$

where E is the photon energy in keV, T the temperature in keV and A the normalisation in units of 10^{26} m$^{0.5}$, defined by

$$A = n_{\mathrm{e}}^{0.5} O \tag{4.32}$$

with n_e the electron density (units: 10^{20} m^{-3}) and O the emitting source area (units: 10^{16} m).

The parameters of the model are:

`norm` - Normalisation A (in units of 10^{26} m$^{0.5}$). Default value: 1.
`t` - The temperature T in keV. Default value: 1 keV.

Recommended citation: Kaastra & Barr (1989) and Rybicki & Lightman (1986).

4.26 Musr: User defined multiplicative model

The user model provides an interface to create an multiplicative model component calculated by a user program. The interface is rather simple and therefore very flexible. The interface uses a set of two ASCII files: input-?-?.prm and output-?-?.spc, where the '?' stands for the sector and component number of the musr model component.

The input-?-?.prm is generated by the musr model and contains the parameter values and the energy grid that SPEX uses to calculate the model. The file has a fixed structure:

- The first line contains one integer number containing the number of parameters that the model needs to calculate (current maximum is 40).
- On the second line the array of parameter values is listed (with the length indicated on the previous line.
- On the third line, an integer value is shown which indicates the number of model bins SPEX has in its model grid.
- On the following lines, the model grid points are shown in three columns. The first column contains the bin upper boundary (egb), the second column is the bin center (eg) and the third column is the full bin width (deg). The energy values are in keV.

The user program is supposed to read the input file and calculate the multiplicative factors for each energy bin on the grid. The spectrum array is called 'sener' and has the same length as the energy grid. In addition, it is also possible to calculate the model for the emission weighted center of the bin. In that case the spectrum calculation is a bit more complicated, but it leads to better results. The user has the option of also calculating the weighted spectrum, which is the average photon energy $E_{\rm aver}$ minus the bin centroid $E_{\rm centroid}$ times the flux (F):

$$W = (E_{\rm aver} - E_{\rm centroid}) * F \tag{4.33}$$

The W values end up in the 'wener' array. If you do not use weighing by average photon energy, 'wener' values can be 0.

The user program should create an output ASCII file named output-?-?.spc with the following structure:

- On the first row, put the number of model bins. This should be the same number as written in the input-?-?.prm file.
- Write the 'sener' and 'wener' values in two columns. The first column is the 'sener' value that represents the multiplication factor and the second column is the 'wener' value.

In the directory 'user' in the main SPEXdirectory some example user model programs can be found. The source code can be adapted to accommodate the user's wishes.

The parameters of the model are:

`exec` - User executable, for example './model.py' for a python script in the same directory where you run SPEX
`npar` - Number of free parameters in the model. Current maximum is 40.
`p01...p40` - Model parameters of the user model.

4.27 Neij: non-equilibrium ionisation jump model

This model calculates the spectrum of a plasma in non-equilibrium ionisation (NEI). For more details about NEI calculations, see Sect. 5.4.

The present model calculates the spectrum of a collisional ionisation equilibrium (CIE) plasma with uniform electron density $n_{\rm e}$ and temperature T_1, that is instantaneously heated or cooled to a temperature T_2. It then calculates the ionisation state and spectrum after a time t. Obviously, if t becomes large the plasma tends to an equilibrium plasma again at temperature T_2.

The ionisation history can be traced by defining an ionisation parameter,

$$u \equiv \int n_{\rm e} {\rm d}t \tag{4.34}$$

with $u = 0$ defined at the start of the shock.

By default the model describes a classical NEI condition with a flat temperature profile:

$$u < 0 \quad : \quad T = T_1, \tag{4.35}$$
$$u > 0 \quad : \quad T = T_2. \tag{4.36}$$

For the case the user wands to calculate more complex situations, SPEX offers two modes to treat a temperature profile $T(u)$: analytic expression (mode 1) or ascii-file input (mode 2).

The temperature profile in mode=1 (analytic case) is given by

$$u < 0 \quad : \quad T = T_1, \tag{4.37}$$
$$0 < u < U \quad : \quad T = T_2, \tag{4.38}$$
$$U < u < U + dU \quad : \quad T = T(u). \tag{4.39}$$

By setting a non-zero value for dU, this model offers the opportunity to calculate more complex evolution in the last epoch ($U < u < U+dU$); e.g. with secondary heating/cooling process and/or change in density. We introduce parameters α and β, which describe a power-law like evolution respectively for temperature and density of the plasma after the "break" of constancy at time $t_{\rm br}$:

$$T(t) = T_2 \ (t/t_{\rm br})^{\alpha} \tag{4.40}$$
$$n_{\rm e}(t) = n_{\rm e} \ (t/t_{\rm br})^{\beta}, \tag{4.41}$$

An immediate application of this break feature would be a recombining plasma due to rarefaction (adiabatic expansion). Such a condition can be realised with $\alpha = -2$ and $\beta = -3$. Note that we include the effect of the density change here in the NEI calculation for the ion concentration, but of course the line emission is calculated at the density prescribed by the parameter *ed* of the model, which represents the true density at the epoch of emission of the spectrum.

To get the expression for $T(u)$, we first calculate the increase of the ionisation parameter after $t = t_{\rm br}$ as follows:

$$u - U = \int_{t_{\rm br}} n(t) dt = \int_{t_{\rm br}} n_{\rm e} (t/t_{\rm br})^{\beta} dt \tag{4.42}$$
$$= n_{\rm e} t_{\rm br} \int_1 (t/t_{\rm br})^{\beta} \ d(t/t_{\rm br}) \tag{4.43}$$
$$= U/(\beta+1) \cdot [(t/t_{\rm br})^{\beta+1} - 1], \tag{4.44}$$

Then, by combining equations 4.40 and 4.44, we obtain:

$$T(u) = T_2 \cdot [1 + (\beta+1) \cdot (u-U)/U]^{\alpha/(\beta+1)}, \tag{4.45}$$

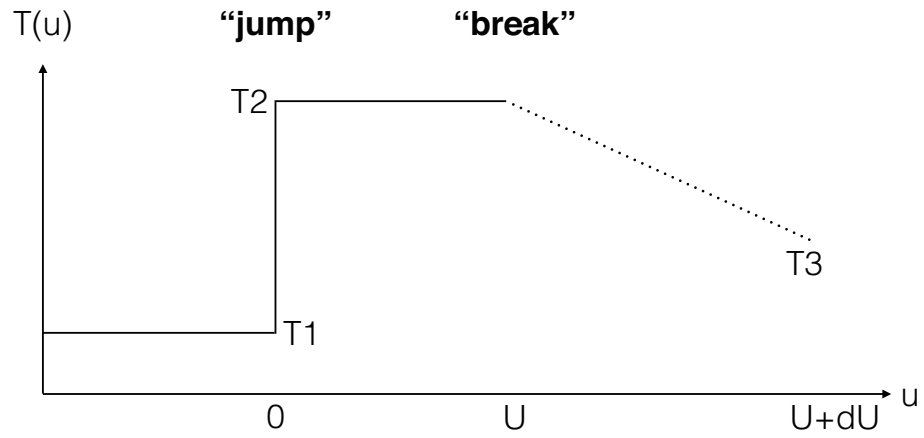

Figure 4.1: The temperature profile with mode=1.

and we get the final temperature at $u = U + dU$ to be

$$T_3 = T_2 \cdot [1 + (\beta + 1) \cdot dU/U]^{\alpha/(\beta+1)}. \tag{4.46}$$

It should be noted that, for fixed values of α and β, the temperature change after the break is determined by the ratio dU/U rather than dU itself. The user can check T_3 with the `ascdump plas` command (see § 3.3) and also the histories of u and $T(u)$ with the `ascdump nei` command (see § 3.3).

In some rare cases with a large negative β, T_3 can get an unphysical value ($T_3 < 0$). In such a case the calculation will automatically be stopped at a lower-limit of $T(u) = 10^{-4}$ keV.

For mode 2, the user may enter an ascii-file with u- and T-values. The format of this file is as follows: the first line contains the number of data pairs (u, T). The next lines contain the values of u (in the SPEX units of 10^{20} s m^{-3}) and T (in keV). Note that $u_1 = 0$ is a requirement, all T_i should be positive, and the array of u-values should be in ascending order. The pairs (u, T) determine the ionisation history, starting from $T = T_1$ (the pre-shock temperature), and the final (radiation) temperature is the temperature of the last bin.

The parameters of the model are:
`t1` - Temperature T_1 before the sudden change in temperature, in keV. Default: 0.002 keV.
`t2` - Temperature T_2 after the sudden change in temperature, in keV. Default: 1 keV.
`u` - Ionization parameter $U = n_e t$ before the “break”, in 10^{20} m^{-3} s. Default: 10^{-4}.
`du` - Ionization parameter dU after the “break” in 10^{20} m^{-3} s. Default value is 0 (no break).
`alfa` - Slope α of the $T(t)$ curve after the “break”. Default value is 0 (constant T).
`beta` - Slope β of the $n(t)$ curve after the “break”. Default value is 0 (constant n).
`mode` - Mode of the model. Mode=1: analytical case; mode=2: $T(u)$ read from a file. In the latter case, also the parameter *hisu* needs to be specified.
`hisu` - Filename with the $T(u)$ values. Only used when mode=2.

The following parameters are the same as for the cie-model:
`ed` - Electron density in 10^{20} m^{-3}
`it` - Ion temperature in keV
`vmic` - Micro turbulent velocity in km/s
`ref` - Reference element
`01...30` - Abundances of H to Zn
`file` - Filename for the nonthermal electron distribution

Recommended citation: Kaastra & Jansen (1993).

4.28 Pdem: DEM models

The pdem model is intended to be used for differential emission measure analysis, simultaneous with fitting of abundances etc. of an optically thin plasma.

It works as follows. The user gives a a number of temperature grid points n, a minimum temperature T_1, a maximum temperature T_n, a total emission measure Y and relative contributions $y_1 \dots y_n$. SPEX assumes that the grid points between T_1 and T_n are distributed logarithmically.

The relative contributions $y_1 \dots y_n$ represents the values of $\mathrm{d}Y/\mathrm{d}\ln T$ (note the logarithm!) at the grid points. SPEX then interpolates in the $\log T_i - \log y_i$ space on a finer grid using splines. That temperature grid on which the data are being evaluated has a fine mesh: step size is about 0.10 in $\log T$ (finer is not usefull because uniqueness is no longer guaranteed), with the additional constraint that the number of mesh points is at least n and not larger than 64 (practical limit in order to avoid excessive cpu-time). The emission measure distribution is then simply scaled in such a way that its sum over the fine mesh equals the total emission measure Y that went into the model.

WARNING: *At least one of the y_i values should be kept frozen during fitting, when Y is a free parameter, otherwise no well defined solution can be obtained! If Y is fixed, then all y_i can be free.*

The parameters of the model are:

`norm` - Normalisation, i.e. total integrated emission measure Y in units of 10^{64} m^{-3}
`t1` - Lower temperature T_1 in keV
`tn` - Upper temperature T_n in keV
`npol` - Number of temperature grid points n; minimum value 2, maximum value 8
`y1` - Relative contribution y_1 at T_1
`y2` - Relative contribution y_2 at T_2

...
`y8` - Relative contribution y_8 at T_8; note that the higher contributions y_i are neglected if $i > n$
The following parameters are the same as for the cie-model:
`ed` - Electron density in 10^{20} m^{-3}
`it` - Ion temperature in keV
`vmic` - Micro turbulent velocity in km/s
`ref` - Reference element
`01...30` - Abundances of H to Zn
`file` - Filename for the nonthermal electron distribution

Note that the true emission measure on the finer mesh can be displayed by using the "ascdump term # # dem" command; you will get a list of temperature (keV) versus emission measure.

4.29 Pion: SPEX photoionised plasma model

The *pion* model calculates the transmission and emission of a slab of photo-ionized plasma, where all ionic column densities are linked through a photoionisation model. The relevant parameter is the ionization parameter $\xi = L/nr^2$, with L the source luminosity, n the hydrogen density and r the distance from the ionizing source. The major difference is that while for the *xabs* model the photoionisation equilibrium is pre-calculated for a grid of ξ-values by an external code, for instances Cloudy or XSTAR (or even with the present *pion* model ...), in the present *pion* model the photoionisation equilibrium is calculated self-consistently using the available plasma routines of SPEX.

WARNING: *The default energy grid in SPEX has 8192 bins between 0.001 and 100 keV. This may not be sufficient for the pion model when you use it without data. Recommended is a logarithmic grid between $10^{-6} - -10^{6}$ keV with a step size of 0.005. You can get this by issuing the following command: "Egrid log 1E-6 : 1E6 step 0.005 keV". Note that if you have read in data, SPEX automatically expands the energy grid to this range and resolution, plus including all energy boundaries from the response matrix.*

WARNING: *This model is still under development and not all atomic data is fully updated. For instance, no cooling by collisional excitation for ions of the K- to Zn-isolelectronic sequences is taken into account yet. So use with care!*

WARNING: *When setting up the model, be aware that the pion model is both additive and multiplicative (even if you put the emission to zero). Therefore, the pion model needs the same com rel sequence as you use for your absorption component. Example: 'com pow — com reds — com pion — com rel 1 3,2' (a powerlaw that powers a pion model and is then redshifted) needs as next command 'com rel 3 2', telling SPEX that the pion emission model is also redshifted.*

WARNING: *Please note that all PION components must be multiplied by at least one continuum component (using the usual comp rel command). Otherwise, photoionisation cannot be calculated and SPEX may crash. A typical AGN SED spanning optical to X-ray energies can be set up with three continuum components in SPEX: pow (primary hard X-ray power-law emission), refl (reflected power-law emission), comt (thermal optical/UV disk continuum + the soft X-ray excess). See the SED of NGC 5548 derived in Mehdipour et al. (2015).*

The main advantage, however, is that the user can define his own ionising continuum by combining any additive components of SPEX, and that ionising spectrum is fed through the *pion* component to get the transmitted spectrum. In addition, multiple *pion* components can be used, with the light leaking from one layer to the next. And it allows for spectral fitting, where the parameters of the ionising continuum are determined simultaneously with the parameters of the absorbing layer. The number of parameters is therefore unlimited (well, the memory and cpu speed of the computer will be the true limiting factors).

Prior to fitting a PION component, it is best to first calculate the model spectrum with your initial parameter values (see `calculate` command). This helps to see if the initial values are reasonable numbers and the model is not too far off the data.

4.29.1 Emission from the *pion* model

Latest news: we have now incorporated a first version of emission from the same layer. We cannot give you any guarantee at the moment that it is bug-free. We know at the moment that the source has probelms when the density gets too high; this is different for each ion; so unless you limit the density in fitting, SPEX may encounter a situation where you surpass the critical density and you may get a warning message. Here is an example. For a photoionised case, with $\log \xi = 3$ (resulting $kT = 0.64$ keV) the nominal occupation of the ground state of H I becomes negative for a density $> 0.15 \times 10^{20}$ m^{-3}. This can be traced down to incomplete atomic data. For H I, we include collisional excitation and de-excitation up to principal quantum number $n = 5$ but not above. As a result, in this example the 1s–5s levels are mainly populated/depopulated by collisions, while 6s–16s are mainly populated by radiative recombination and depopulated by radiative cascades downward or photon absorption. The nominal occupation of 1s–5s then decreases from 0.035 (1s) to 0.0024 (5s), while for 6s–16s they increase slowly from 0.020 to 0.027. This is of course physically unacceptable. It causes the lower levels to leak to the higher levels, with eventually the catastrophe of negative occupation for the ground state. We mitigate this by replacing the level populations by the LTE populations and issueing a warning. Without this mitigation, SPEX would crash.

WARNING: *You can get the emission by putting the covering factor (omeg) to a non-zero value; it will slow down the calculations compared to absorption-only calculations, so be aware*

Normally, to calculate the emission from a full thin shell surrounding an ionising source, you should set the parameter *omeg* to unity (a full shell of 4π steradians). Smaller factors could be associated e.g. to ionisation cones; values larger than unity make physically no sense but you can formally play around with it (for very large values, the emitted spectrum would start dominating the absorbed primary continuum, but if you want to suppress the primary continuum in the observed spectrum, it is better to define your model like the example below as:

com pow
com pion

```
com etau
com rel 1 2,3
com rel 2 0
par 3 tau v 1e10
par 3 a v 0
```

In this example, the powerlaw goes through the *pion* component and is killed afterwards by the *etau* component, while the emission from the *pion* component is not attenuated by *etau*.

You can vary the new parameter *mix* to get a different ratio of forwards to backwards emission. Putting it to 1 (default) means you get the forward emission, putting it to 0 the backwards emission, and intermediate values give you a mix.

WARNING: *The emission model uses currently only one layer. When the continuum optical depth of the absorbed continuum, weighted with the incoming flux, becomes of order unity, the layer will become inhomogeneous in terms of temperature structure, and our single-layer approximation will break down.*

In order to make a PION component produce emission only, fix the covering fraction (cf) parameter to zero so that no absorption is produced. Then fit the omega parameter. Note that any PION component with a non-zero omega acts as an additive component in SPEX. Therefore, multiply these components with your multiplicative components (like the Galactic absorption) using the `comp rel` command.

For more information on this model, the atomic data and parameters we refer to Sect. 5.2.3.

4.29.2 More options

No energy balance solution needed

The default option (*tmod=0*) for the *pion* model is to solve simultaneously for the ionisation balance and the energy balance equations. This option is useful for e.g. photoionised winds of AGN or X-ray binaries.

However, there are situations where photo-ionisation or photo-excitation play a role but do not determine the thermal structure. Examples are winds of hot stars, where shocks heat the wind but UV radiation from the star can affect He-like triplet line emission ratios. Another example are the most teneous parts of the WHIM, where photoionisation by the cosmic background can be important compared to collisional ionisations.

For such cases, the user can set the parameter *tmod=1*; in that case, the user should also provide the temperature *tinp* of the plasma. In this case, only the ionisation balance equation is solved, and there is in general no energy balance (this can be checked by using the ascii-output option *heat*). Do not forget to set the parameter *omeg* to a finite value (the default is zero), otherwise the emitted spectrum is zero.

External heat sources

In some cases there may be an other external heat or cooling source, like shock heating, magnetic reconnection, adiabatic expansion etc. If one wishes to solve for the photoionisation equilibrium, then this additional heat source can be used by putting the parameter *exth* to the proper value (units: $\mathrm{W\,m^{-3}}$. A negative value would mean a cooling contribution.

Multiple solutions

There are situations where there is not a unique solution to the energy balance equations. A simple example can be obtained as follows: take a logarithmic energy grid between $10^{-6} - 10^{6}$ keV, use a powerlaw with photon index 1.5, apply the pion model to it and put *exth* to 5×10^{-25} $\mathrm{W\,m^{-3}}$. In this case there are 3 solutions. SPEX chooses by default the hottest solution. You can see all solutions by putting the parameter *fmod=1* and using the *heat* ascii output option. Or check the behaviour of the hating balance by issuing the *ebal* ascii output option. You can select which solution you want to use in SPEX by setting the *soln* parameter. Default is 0 (hottest solution), and for the above case of 3 solutions

values of 1, 2 and 3 renders you the coldest, second ant hottest solution. Test this with the *heat* or *plas* output options.

WARNING: *When you set soln to a non-zero value, use fmod=1, otherwise SPEX may crash.*

No equilibrium solution

There are also situations where there is no equilibrium solution to the energy balance equations. This may happen for instance if you put so much heat in the plasma that it cannot be balanced anymore by cooling. Another example is a too hard powerlaw without high energy cut-off, where Compton-heating might be very strong. In this case SPEX renders an error message, and you cannot trust the result of the calculation anymore. The only remedie is to adjust your model parameters or the allowed range for them in case of spectral fitting or error searches.

4.29.3 Model parameters

The parameters of the model are:

`nh` - Hydrogen column density in 10^{28} m^{-2}. Default value: 10^{-4} (corresponding to 10^{24} m^{-2}, a typical value at low Galactic latitudes).
`xi` - the ^{10}log of the ionisation parameter $\log \xi$ in units of 10^{-9} W m. Default value: 1.
`u` - the Davidson (Cloudy) ionisation parameter U (dimensionless). This is calculated from the SED and the value of ξ. Not fittable, just output.

The following parameters are common to all our absorption models:
`fcov` - The covering factor of the absorber. Default value: 1 (full covering)
`v` - Root mean square velocity $\sigma_{\rm v}$
`rms` - Rms velocity $\sigma_{\rm b}$ of line blend components
`dv` - Velocity distance Δv between different blend components
`zv` - Average systematic velocity v of the absorber

The following parameters are the same as for the cie-model (see there for a description):
`ref` - Reference element
`01...28` - Abundances of H to Ni; only here we take H, He, C, N, O, Ne, Na, Mg, Al, Si, S, Ar, Ca, Fe, Ni.
`file` - File name for the electron distribution (in case of a sum of Maxwellians distribution)

The following parameters are unique for the *pion* model:
`type` - If type equals 0 (default value), it uses ξ as its main parameter; if type equals 1, it uses lixi (see next parameter) as its main parameter
`lixi` - Optional alternative ionisation parameter, defines as $L_{\rm ion}/\xi$ in units of 10^{39} m^{-1}. This is useful for time-variable spectra where ξ has been determined from one spectrum and where one wants to calculated the transmitted spectrum for fixed nr^2 for a different ionising spectrum; in that case lixi can be kept constant.
`omeg` - Covering factor $\Omega/4\pi$, needed for emission. At this stage, keep it to zero, please.
`mix` - Fraction of emitted spectrum to the forward direction relative to the total. default value: 1 (all emission forward). A value of 0 means SPEX gives all backwards emission.
`exth` - External heating in W m^{-3}. default value: 0.
`fmod` - Show all solutions in ascii output of the heating (fmod=1). Default is fmod=0. Set fmod=1 also when you set soln> 0.
`soln` - The temperature solution to be used, from low to high values. Default value is 0 (hottest solution). If this parameter is larger than the hootest solution, it adopts the hottest solution instead. Should be used with fmod=1 in case soln> 0.
`tmod` - Temperature mode. Default value: 0 (solve for the temperature that provides energy balance). If

tmod=1, use *tinp* instead as temperature and do not solve for energy balance.
`tinp` - Temperature of the plasma in keV. Default: 1 keV. Only relevant if *tmod=1*.

Recommended citation: Miller et al. (2015) and Mehdipour et al. (2016).

4.30 Pow: power law model

The power law spectrum as given here is a generalization of a simple power law with the possibility of a break, such that the resultant spectrum in the $\log F - \log E$ plane is a hyperbola.

The spectrum is given by:

$$F(E) = AE^{-\Gamma} e^{\eta(E)}, \tag{4.47}$$

where E is the photon energy in keV, F the photon flux in units of 10^{44} ph s^{-1} keV^{-1}, and the function $\eta(E)$ is given by

$$\eta(E) = \frac{r\xi + \sqrt{r^2\xi^2 + b^2(1-r^2)}}{1-r^2}, \tag{4.48}$$

with $\xi \equiv \ln(E/E_0)$, and E_0, r and b adjustable parameters. For high energies, ξ becomes large and then η approaches $2r\xi/(1-r^2)$, while for low energies ξ approaches $-\infty$ and as a consequence η goes to zero. Therefore the break $\Delta\Gamma$ in the spectrum is $\Delta\Gamma = 2r\xi/(1-r^2)$. Inverting this we have

$$r = \frac{\sqrt{1+(\Delta\Gamma)^2} - 1}{|\Delta\Gamma|}. \tag{4.49}$$

The parameter b gives the distance (in logarithmic units) from the interception point of the asymptotes of the hyperbola to the hyperbola. A value of $b = 0$ therefore means a sharp break, while for larger values of b the break gets smoother.

The simple power law model is obtained by having $\Delta\Gamma = 0$, or the break energy E_0 put to a very large value.

WARNING: *By default, the allowed range for the photon index Γ is (-10,10). If you manually increase the limits, you may run the risk that SPEX crashes due to overflow for very large photon indices.*

WARNING: *Note the sign of Γ: positive values correspond to spectra decreasing with energy. A spectrum with $\Delta\Gamma > 0$ therefore steepens/softens at high energies, for $\Delta\Gamma < 0$ it hardens.*

As an extension, we allow for a different normalisation, namely the integrated luminosity L in a given energy band E_1–E_2. If you choose this option, the parameter "type" should be set to 1. The reason for introducing this option is that in several cases you may have a spectrum that does not include energies around 1 keV. In that case the energy at which the normalisation A is determined is outside your fit range, and the nominal error bars on A can be much larger than the actual flux uncertainty over the fitted range. Note that the parameters E_1 and E_2 act independently from whatever range you specify using the "elim" command. Also, the luminosity is purely the luminosity of the power law, not corrected for any transmission effects that you may have specified in other spectral components.

WARNING: *When you do spectral fitting, you* **must** *keep either A or L a fixed parameter! The other parameter will then be calculated automatically whenever you give the calculate or fit command. SPEX does not check this for you! If you do not do this, you may get unexpected results / crashes.*

WARNING: *The conversion factor between L and A is calculated numerically and not analytically (because of the possible break). In the power law model, photon fluxes above the nominal limit (currently $e^{34} = 5.8 \times 10^{14}$ in unscaled units) are put to the maximum value in order to prevent numerical overflow. This implies that you get inaccurate results for low energies, for example for a simple power law with $\Gamma = 2$ the results (including conversion factors) for $E < 10^{-7}$ keV become inaccurate.*

WARNING: *Note that when you include a break, the value of Γ is the photon index at energies below the break. Also, the normalisation A is the nominal normalisation of this low-energy part. In such a case of*

a break, the true flux at 1 keV may be different from the value of A! Of course, you can always calculate the flux in a given band separately.

The parameters of the model are:

`norm` - Normalisation A of the power law, in units of 10^{44} ph s^{-1} keV^{-1} at 1 keV. Default value: 1. When $\Delta\Gamma$ is not equal to 0, it is the asymptotic value at 1 keV of the low-energy branch.
`gamm` - The photon index Γ of the spectrum. Default value: 2. When $\Delta\Gamma$ is not equal to 0, it is the slope of the low-energy branch.
`dgam` - The photon index break $\Delta\Gamma$ of the spectrum. Default value: 0. and frozen. If no break is desired, keep this parameter 0 (and frozen!).
`e0` - The break energy E_0 (keV) of the spectrum. Default value: 10^{10} and frozen.
`b` - Smoothness of the break b. Default: 0.
`type` - Type of normalisation. Type= 0 (default): use A; type= 1: use L.
`elow` - E_1 in keV, the lower limit for the luminosity calculation. Default value: 2 keV.
`eupp` - E_2 in keV, the upper limit for the luminosity calculation. Default value: 10 keV. Take care that $E_2 > E_1$.
`lum` - Luminosity L between E_1 and E_2, in units of 10^{30} W.

4.31 Reds: redshift model

This multiplicative model applies a redshift z to an arbitrary additive component. If a photon is emitted at energy E, the redshift model will put it at energy $(1+z)E$. In addition, a time dilatation correction is applied such that the spectrum $S(E)$ (expressed as photons/s per bin) is divided by $1+z$.

However, it is necessary to distinguish two different cases:

case 1 (flag=0): the redshift component can be used to calculate the effects of the cosmological redshift. The cosmological redshift has the effect of the energy shift and the time dilatation as outlined above. Therefore, the above procedure is essentially correct and this is what was used in older versions of SPEX (before version 2.00.10). However, what was overlooked there is that in the determination of the finally observed spectrum/flux as seen on Earth, a division by $4\pi d^2$ is necessary. For this distance d, we took the luminosity distance. However, the factors $(1+z)^2$ are absorbed into the definition of the luminosity distance. Therefore, in the older versions all fluxes were too small by a factor of $(1+z)^2$. Since we want to retain the luminosity distance as the natural distance measure for spectra, it appears necessary to multiply the spectrum as calculated by the original subroutine by a factor of $(1+z)^2$. But in the other case (redshifts caused by the motion of a source at any distance), this re-correction should not be done, reason why we introduce the other option. *in summary, for redshift components corresponding to cosmological redshifts, the option flag=0 (default) must be used.*

case 2 (flag=1): this is the old case for a source with a redshift caused by motion away from us. It should be used for any velocity fields other than the Hubble flow.

WARNING: *Note that this component should be used in tandem with the distance command (Section 3.9) to take into account the cosmological redshift and its influence on the flux completely.*

The parameters of the model are:

`z` - Redshift z. Default value: 0.
`flag` - Flag: 0 (cosmological redshift) or 1 (velocity redshift). Default value: 0.

4.32 Refl: reflection model

This model was kindly provided by Piotr Zycki. The related programs in XSPEC are FELIPL ans FERFSCHW. The first one gives the reflected continuum (i.e. PEXRAV/PEXRIV) plus the line with correct energy

and intensity, the second one is the first one with the relativistic smearing added. Both are combined into the single refl model in SPEX.

It is a model of reflection from a constant density X-ray illuminated atmosphere. It computes the Compton-reflected continuum (cf. Magdziarz & Zdziarski 1995) and the iron K alpha line (cf. Zycki & Czerny 1994), as described in Zycki et al. (1999). In addition it can be convolved with a relativistic diskline model (for Schwarzschild geometry).

Chemical elements taken into account in this model are H, He, C, N, O, Ne, Mg, Si, S and Fe. The standard abundances are taken from Morrison & McCammon (1983).

The incoming spectrum is characterized by:

$$N_i(E) = AE^{-\Gamma} \exp\left[-E/E_c\right], \tag{4.50}$$

where E is the photon energy in keV, $N_i(E)$ the number of photons per per second per keV, Γ is the photon index and E_c a cut-off energy. The normalisation A is in units of 10^{44} photons $\mathrm{s}^{-1}\,\mathrm{keV}^{-1}$ at 1 keV, just the same as in the standard power law model. The total spectrum $N(E)$ is then given by

$$N(E) = N_i(E) + sR(E), \tag{4.51}$$

where $R(E)$ is the reflected spectrum and s is a scaling factor.

The parameters of the model are:

`norm` - Normalisation A of the power law
`gamm` - The photon index Γ of the ionising spectrum
`ecut` - The cut-off energy (keV) of the ionising spectrum. If no cut-off is desired, take this parameter zero (and keep it frozen!).
`pow` - If pow=1, the incoming power law is added to the spectrum (default); if pow=0, only the reflected spectrum is given
`disk` - If disk=1, the spectrum will be convolved with an accretion disk profile (default); if disk=0, this is not done
`fgr` - Full general relativity used (default, for fgr=1)
`t` - Temperature of the reflector (disk) in keV
`xi` - Ionisation parameter $\xi = L/nr^2$ in the usual (c.g.s. based) units of 10^{-9} W m
`abun` - The abundance of all metals excluding H and He, in solar units
`feab` - The iron abundance with respect to the other metals
`cosi` - The cosine of the inclination angle of the disk. $\cos i = 0$ ($i = \pi/2$) corresponds to edge-on
`scal` - Scale s for reflection. For an isotropic source above the disk $s = 1$. This value corresponds to seeing equal contributions from the reflected and direct spectra.
`q` - Emissivity index for the accretion disk; default value -3 (the emissivity scales with r^{+q} at large radii, so $q = -3$ means r^-3. Note the sign difference with the Laor model.
`r1` - Inner radius of the disk in units of GM/c^2. Default: 10.
`r2` - Outer radius of the disk in units of GM/c^2. Default: 10^4.

Recommended citation: Magdziarz & Zdziarski (1995) and Zycki et al. (1999).

4.33 Rrc: radiative recombination continuum model

This is a simplified model, aimed to calculate radiative recombination continua for photoionized plasmas. It is a simple, limited shortcut to a more fully complete model for the emission from a recombining plasma.

The user essentially prescribes the emission measure for each ion as well as the radiation temperature, and then this model calculates the *continuum* emission corresponding to this temperature and set of ionic emission measures. Line radiation is *not* taken into account. However, for physical self-consistency, we take account of all three continuum emission components: Bremsstrahlung, two photon emission and free-bound radiation (= the RRC).

The reason for having no physical model to couple the ionic emission measures (contrary to for example the CIE model), is that this allows the user to fit these emission measures without making a priori assumptions about the ionization equilibrium. The user might then combine later the set of derived emission measures with any of his relevant models.

WARNING: *Take care that for too high temperatures, two photon emission might be stronger than the free-bound (RRC) emission!*

WARNING: *Take care that the fit parameters are emission measures of a given ion, while the radiation occurs in the next ion. For example radiative recombination of O* IX *to O* VIII *is proportional to the emission measure of O* IX *($n_e n_{\mathrm{O\,IX}} V$), but produces an emission edge in O* VIII *at 14.22 Å.*

WARNING: *No recombination is possible to neutrals, so therefore there is no H* I*, O* I *or Fe* I *in this model.*

The parameters of the model are:

`t` - The temperature T in keV. Default value: 1 keV.
`h2` - The H II emission measure $n_e n_{\mathrm{H\,II}} V$ in units of 10^{64} m^{-3}. Default value: 0.
`he2` - The He II emission measure $n_e n_{\mathrm{He\,II}} V$ in units of 10^{64} m^{-3}. Default value: 0.
`he3` - The He III emission measure $n_e n_{\mathrm{He\,III}} V$ in units of 10^{64} m^{-3}. Default value: 0.

...
`ni29` - The Ni XXIX emission measure $n_e n_{\mathrm{Ni\,XXIX}} V$ in units of 10^{64} m^{-3}. Default value: 0.

4.34 Slab: thin slab absorption model

The *slab* model calculates the transmission of a slab of material, where all ionic column densities can be chosen independently. This has the advantage that a spectrum can be fit without any knowledge of the ionisation balance of the slab. After a spectral fit has been made, one may try to explain the observed column densities by comparing the with predictions from any model (as calculated by SPEX, Cloudy, XSTAR, ION or any other existing (photo)ionisation code).

For more information on this model, the atomic data and parameters we refer to Sect. 5.2.3. Below we do not list all the parameters of the model, but for each ion of H, He, C, N, O, Ne, Na, Mg, Al, Si, S, Ar, Ca, Fe and Ni there is a parameter, namely the logarithm (base 10) of the column density in SI units (m^{-2}). So, a H 1 column of 10^{28} m^{-2} (10^{24} cm^{-2} should be entered as 28.0. The default values of 1 therefore correspond to an almost negligible column.

WARNING: *We do include here fully stripped ions, as their free electrons do produce Thomson scattering. However, the user is advised not to take more than one column density of a bare nucleus as a free parameter, as the shape of the Thomson scattering contribution is the same for all electrons, regardless from which ion they came from. In the spectral fitting, there would be 100% correlations between the column densities of these bare ions, which is undesirable.*

The parameters of the model are:

`h1` - log column density (m^{-2}) of H I. Default value: 1.
`h2` - log column density (m^{-2}) of H II. Default value: 1.
`he1` - log column density (m^{-2}) of He I. Default value: 1.
`he2` - log column density (m^{-2}) of He II. Default value: 1.
`he2` - log column density (m^{-2}) of He III. Default value: 1.
`c1` - log column density (m^{-2}) of H I. Default value: 1.
... -
`ni27` - log column density (m^{-2}) of Ni XXVII. Default value: 1.
`ni28` - log column density (m^{-2}) of Ni XXVIII. Default value: 1.

The following parameters are common to all our absorption models:

`fcov` - The covering factor of the absorber. Default value: 1 (full covering)
`v` - Root mean square velocity $\sigma_{\rm v}$
`rms` - Rms velocity $\sigma_{\rm b}$ of line blend components
`dv` - Velocity distance Δv between different blend components
`zv` - Average systematic velocity v of the absorber

Recommended citation: Kaastra et al. (2002).

4.35 Spln: spline continuum model

Sometimes the continuum of an X-ray source may be too complex to model with known physical components. A situation like that may be found in AGN continua, which are a complex superposition of hard power law, soft continuum excess, relativistically broadened and "normal" broad lines with a priori unknown line shape, etc., while in addition a superimposed warm absorber may have well defined narrow absorption lines. In that case it might be useful to fit the continuum with an arbitrary profile in order to get first an accurate description of the absorber, and then after having "removed" the absorber try to understand the underlying continuum spectrum.

For these situations the *spln* model introduced here is useful. It allows the user to model the continuum within two boundaries b_1 and b_2 with a cubic spline.

The algorithm works as follows. The user selects the limits b_1 and b_2 as well as the number of grid points n. SPEX then creates a grid x_1, x_2, ..., x_n with uniform spacing in b (see below for details). The spectrum at these grid points is contained in the corresponding array y_1, y_2, ..., y_n. These have the usual units of 10^{44} photons s^{-1} keV^{1} used throughout SPEX, and is the spectrum emitted at the source. The parameters y_i can be adjusted during the spectral fitting, but b_1, b_2 and n and thereby x_i remain fixed. At intermediate points between the x_i, the photon spectrum is determined by cubic spline interpolation on the (x_i, y_i) data pairs. We take a natural spline, i.e. at x_1 and x_n the second derivative of the spline is zero.

Outside of the range b_1–b_2 however, the photon spectrum is put to zero, i.e. no extrapolation is made!

Finally note that we have chosen to define the spline in logarithmic space of y, i.e. the log of the photon spectrum is fit by a spline. This is done in order to guarantee that the spectrum remains non-negative everywhere. However, the y-values listed is the (linear) photon spectrum itself.

There are four different options for the energy grid x_i, indicated by the parameter *type*:

- *type*=1: b_1 is the lower energy in keV, b_2 is the upper energy in keV, and the grid is linear in energy in between.
- *type*=2: b_1 is the lower energy in keV, b_2 is the upper energy in keV, and the grid is logarithmic in energy in between.
- *type*=3: b_1 is the lower wavelength in Å, b_2 is the upper wavelength in Å, and the grid is linear in wavelength in between.
- *type*=4: b_1 is the lower wavelength in Å, b_2 is the upper wavelength in Å, and the grid is logarithmic in wavelength in between.

Note that the logarithmic grids can also be used if one wants to keep a fixed velocity resolution (for broadened line features for example). Further, each time that the model is being evaluated, the relevant values of the x_i grid points are evaluated.

WARNING: *Be aware that if you just set b_1, b_2 and n but do not issue the "calc" command or the "fit" command, the x_i values have not yet been calculated and any listed values that you get with the "par show" command will be wrong. After the first calculation, they are right.*

WARNING: *If at any time you change one of the parameters type, b_1, b_2 or n, the y_i values will not be appropriate anymore as they correspond to the previous set of x_i values.*

The maximum number n of grid points that is allowed is 999, for practical reasons. Should you wish to have a larger number, then you must define multiple *spln* components, each spanning its own (disjunct) b_1–b_2 range.

It should be noted, however, that if you take n very large, spectral fitting may become slow, in particular if you take your initial guesses of the y_i parameters not too close to the true values. The reason for the slowness is two-fold; first, the computational time for each fitting step is proportional to the number of free parameters (if the number of free parameters is large). The second reason is unavoidable due to our spectral fitting algorithm: our splines are defined in log photon spectrum space; if you start for example with the same value for each y_i, the fitting algorithm will start to vary each parameter in turn; if it changes for example parameter x_j by 1, this means a factor of 10; since the neighbouring points (like x_{j-1} and x_{j+1} however are not adjusted in thid step, the photon spectrum has to be drawn as a cubic spline through this very sharp function, and it will show the well-known over-and undershooting at the intermediate x-values between x_{j-1} and x_j and between x_j and x_{j+1}; as the data do not show this strong oscillation, χ^2 will be poor and the fitting algorithm will decide to adjust the parameter y_j only with a tiny amount; the big improvement in χ^2 would only come if *all* values of y_i were adjusted simultaneously.

The parameters of the model are:

`type` - The parameter *type* defined above; allowed values are 1–4. Default value: 1.
`n` - The number of grid points n. Should be at least 2.
`low` - Lower x-value b_1.
`upp` - Upper x-value b_2. Take care to take $b_2 > b_1$.
`x001` - First x-value, by definition equal to b_1. x-values are not allowed to vary (i.e. you may not fit them).
`x002` - Second x-value
`x003` - Third x-value
. . . - Other x-values
`x999` - last x-value, by definition equal to b_n. If $n < 999$, replace the 999 by the relevant value (for example, if $n = 237$, then the last x-value is x237).
`y001` - First y-value. This is a fittable parameter.
`y002` - Second y-value
`y003` - Third y-value
. . . - Other y-values
`y999` - last y-value. If $n < 999$, replace the 999 by the relevant value (for example, if $n = 237$, then the last y-value is y237).

4.36 User: User defined model

The user model provides an interface to create an additive model component calculated by a user program. The interface is rather simple and therefore very flexible. The interface uses a set of two ASCII files: input-?-?.prm and output-?-?.spc, where the '?' stands for the sector and component number of the user model component.

The input-?-?.prm is generated by the user model and contains the parameter values and the energy grid that SPEX uses to calculate the model. The file has a fixed structure:

- The first line contains one integer number containing the number of parameters that the model needs to calculate (current maximum is 40).
- On the second line the array of parameter values is listed (with the length indicated on the previous line.
- On the third line, an integer value is shown which indicates the number of model bins SPEX has in its model grid.

- On the following lines, the model grid points are shown in three columns. The first column contains the bin upper boundary (egb), the second column is the bin center (eg) and the third column is the full bin width (deg). The energy values are in keV.

The user program is supposed to read the input file and calculate the model spectrum for each energy bin on the grid. The spectrum array is called 'sener' and has the same length as the energy grid and unit photons/s/bin. In addition, it is also possible to calculate the model for the emission weighted center of the bin. In that case the spectrum calculation is a bit more complicated, but it leads to better results. The user has the option of also calculating the weighted spectrum, which is the average photon energy $E_{\rm aver}$ minus the bin centroid $E_{\rm centroid}$ times the flux (F):

$$W = (E_{\rm aver} - E_{\rm centroid}) * F \tag{4.52}$$

The W values end up in the 'wener' array. If you do not use weighing by average photon energy, 'wener' values can be 0.

The user program should create an output ASCII file named output-?-?.spc with the following structure:

- On the first row, put the number of model bins. This should be the same number as written in the input-?-?.prm file.
- Write the 'sener' and 'wener' values in two columns. The first column is the 'sener' value in photons/s/bin and the second column is the 'wener' value.

In the directory 'user' in the main SPEXdirectory some example user model programs can be found. The source code can be adapted to accommodate the user's wishes.

The parameters of the model are:

`exec` - User executable, for example './model.py' for a python script in the same directory where you run SPEX
`npar` - Number of free parameters in the model. Current maximum is 40.
`p01...p40` - Model parameters of the user model.

4.37 Vblo: rectangular velocity broadening model

This multiplicative model broadens an arbitrary additive component with a rectangular Doppler profile, characterized by the half-width v. Therefore if a delta-line at energy E is convolved with this component, its full energy width will be $2Ev/c$, and line photons get a rectangular distribution between $E - Ev/c$ and $E + Ev/c$. Of course, any line or continuum emission component can be convolved with the this broadening model.

The parameters of the model are:

`vblo` - Velocity broadening half-width v, in km/s. Default value: 3000 km/s.

4.38 Vgau: gaussian velocity broadening model

This multiplicative model broadens an arbitrary additive component with a Gaussian Doppler profile, characterized by the Gaussian σ. The broadening kernel is therefore proportional to $e^{-(c^2/2\sigma^2)(E-E_0)^2/E_0^2}$.

The parameters of the model are:

`sig` - Gaussian velocity broadening σ, in km/s. Default value: 1 km/s

4.39 Vpro: velocity profile broadening model

This multiplicative model broadens an arbitrary additive component with an arbitrarily shaped Doppler profile, characterized by the half-width v and a profile shape $f(x)$. The resulting spectrum $S_c(E)$ is calculated from the original spectrum $S(E)$ as

$$S_c(E) = \int f\left(\frac{E - E_0}{E_0}\,\frac{v}{c}\right) S(E_0)\mathrm{d}E_0. \tag{4.53}$$

The function $f(x)$ must correspond to a probability function, i.e. for all values of x we have

$$f(x) \geq 0 \tag{4.54}$$

and furthermore

$$\int_{-\infty}^{\infty} f(x)\mathrm{d}x = 1. \tag{4.55}$$

In our implementation, we do not use $f(x)$ but instead the cumulative probability density function $F(x)$, which is related to $f(x)$ by

$$F(x) \equiv \int_{-\infty}^{x} f(y)\mathrm{d}y, \tag{4.56}$$

where obviously $F(-\infty) = 0$ and $F(\infty) = 1$. The reason for using the cumulative distribution is that this allows easier interpolation and conservation of photons in the numerical integrations.

If this component is used, you must have a file available which we call here "vprof.dat" (but any name is allowed). This is a simple ascii file, with n lines, and at each line two numbers: a value for x and the corresponding $F(x)$. The lines must be sorted in ascending order in x, and for $F(x)$ to be a proper probability distribution, it must be a non-decreasing function i.e. if $F(x_i) \leq F(x_{i+1})$ for all values of i between 1 and $n-1$. Furthermore, we demand that $F(x_1) \equiv 0$ and $F(x_n) \equiv 1$.

Note that both x and $F(x)$ are dimensionless. The parameter v serves as a scaling parameter for the total amount of broadening. Of course for a given profile there is freedom for the choice of both the x-scale as well as the value of v, as long as e.g. $x_n v$ remains constant. In practice it is better to make a logical choice. For example, for a rectangular velocity broadening (equivalent to the *vblo* broadening model) one would choose $n = 2$ with $x_1 = -1$, $x_2 = 1$, $F(x_1) = 0$ and $F(x_2) = 1$, and then let v do the scaling (this also allows you to have v as a free parameter in spectral fits). If one would instead want to describe a Gaussian profile (for which of course also the *vgau* model exists), one could for example approximate the profile by taking the x-scale in units of the standard deviation; an example with a resolution of 0.1 standard deviation and a cut-off approximation at 5 standard deviations would be x =-5, -4.9, -4.8, ..., 4.8, 4.9, 5.0 with corresponding values for F given by F = 0, 0.00000048, 0.00000079, ..., 0.99999921, 0.99999952, 1.

The parameters of the model are:

`v` - Velocity broadening parameter v, in km/s. Default value: 1 km/s.
`file` - Ascii character string, containing the actual name of the vprof.dat file

4.40 Warm: continuous photoionised absorption model

The *warm* model is a multi-component version of the *xabs* model. In the *warm* model, we construct a model for a continuous distribution of column density N_{H} as a function of ξ. It is in some sense comparable to the differential emission measure models used to model the emission from multi-temperature gas. Here we have absorption from multi-ionization gas. Depending upon the physics of the source, this may be a better approximation than just the sum of a few *xabs* components. A disadvantage of the model may be

(but this also depends upon the physics of the source), that all dynamical parameters for each value of ξ are the same, like the outflow velocity and turbulent broadening. If this appears to be the case in a given source, one may of course avoid this problem by taking multiple, non-overlapping *warm* components.

The model assumes a logarithmic grid of n equidistant values of $\log \xi$, between a lower limit ξ_1 and an upper limit ξ_2. For each of the grid points ξ_i, a value of f_i can be adjusted; here f_i is given as $f_i = \mathrm{d}N_\mathrm{H}/\mathrm{d}\ln\xi$ evaluated at ξ_i, where the differential column density is assumed to be a continuous function of ξ. At each intermediate point, the value of $\mathrm{d}N_\mathrm{H}/\mathrm{d}\ln\xi$ is then determined by doing cubic spline interpolation in the $\ln f$ – $\ln \xi$ space. In order not to consume too much computer time, the step size for numerical integration $\Delta \log \xi$ can be set. A typical, recommended value for this (the default) is 0.2.

For more information on this model, the atomic data and parameters we refer to Sect. 5.2.3.

The parameters of the model are:

`xil1` - $\log \xi_1$ of the lower limit of the ionisation parameter range in units of 10^{-9} W m. Default value: -4.
`xil2` - $\log \xi_2$ of the upper limit of the ionisation parameter range in units of 10^{-9} W m. Default value: 5.
`npol` - The number of grid points for the $\log \xi$ grid, including the end points for ξ_1. Default value: 19; lower values are generally recommended; minimum value is 2.
`dxi` - step size for numerical integration $\Delta \log \xi$. Default value: 0.2.
`f01`...`f19` - Values of $f_i = \mathrm{d}N_\mathrm{H}/\mathrm{d}\ln\xi$ at the grid points. Default values 10^{-6}. When npol< 19, the remaining values of f_i are simply ignored.

The following parameters are common to all our absorption models:
`fcov` - The covering factor of the absorber. Default value: 1 (full covering)
`v` - Root mean square velocity σ_v
`rms` - Rms velocity σ_b of line blend components
`dv` - Velocity distance Δv between different blend components
`zv` - Average systematic velocity v of the absorber

The following parameter is the same as for the xabs-model (see there for a description):
`col` - File name for the photoionisation balance parameters

Recommended citation: Steenbrugge et al. (2005b).

4.41 Wdem: power law differential emission measure model

This model calculates the spectrum of a power law distribution of the differential emission measure distribution. It appears to be a good empirical approximation for the spectrum in cooling cores of clusters of galaxies. It was first introduced by Kaastra et al. (2004) and is defined as follows:

$$\frac{\mathrm{d}Y}{\mathrm{d}T} = \begin{cases} 0 & \text{if } T \le \beta T_\mathrm{max}; \\ cT^\alpha & \text{if } \beta T_\mathrm{max} < T < T_\mathrm{max}; \\ 0 & \text{if } T \ge T_\mathrm{max}. \end{cases} \tag{4.57}$$

Here Y is the emission measure $Y \equiv n_\mathrm{H} n_\mathrm{e} V$ in units of 10^{64} m^{-3}, where n_e and n_H are the electron and Hydrogen densities and V the volume of the source.

For $\alpha \to \infty$, we obtain the isothermal model, for large α a steep temperature decline is recovered while for $\alpha = 0$ the emission measure distribution is flat. Note that Peterson et al. (2003) use a similar parameterisation but then for the differential luminosity distribution). In practice, we have implemented the model (4.57) by using the integrated emission measure Y_tot instead of c for the normalisation, and instead of α its inverse $p = 1/\alpha$, so that we can test isothermality by taking $p = 0$. The emission measure distribution for the model is binned to bins with logarithmic steps of 0.10 in $\log T$, and for each bin the spectrum is evaluated at the emission measure averaged temperature and with the integrated emission

measure for the relevant bin (this is needed since for large α the emission measure weighted temperature is very close to the upper temperature limit of the bin, and not to the bin centroid). Instead of using $T_{\rm min}$ directly as the lower temperature cut-off, we use a scaled cut-off β such that $T_{\rm min} = \beta T_{\rm max}$.

From the parameters of the wdem model, the emission weighted mean temperature $kT_{\rm mean}$ can be calculated (de Plaa et al. 2006):

$$T_{\rm mean} = \frac{(1+\alpha)}{(2+\alpha)} \frac{(1-\beta^{2+\alpha})}{(1-\beta^{1+\alpha})} T_{\rm max} \tag{4.58}$$

WARNING: *Take care that $\beta < 1$. For $\beta = 1$, the model becomes isothermal, regardless the value of α. The model also becomes isothermal for p=0, regardless of the value of β.*

WARNING: *For low resolution spectra, the α and β parameters are not entirely independent, which could lead to degeneracies in the fit.*

The parameters of the model are:

`norm` - Integrated emission measure between $T_{\rm min}$ and $T_{\rm max}$
`t0` - Maximum temperature $T_{\rm max}$, in keV. Default: 1 keV.
`p` - Slope $p = 1/\alpha$. Default: 0.25 ($\alpha = 4$).
`cut` - Lower temperature cut-off β , in units of $T_{\rm max}$. Default value: 0.1.

The following parameters are the same as for the cie-model:
`ed` - Electron density in 10^{20} m^{-3}
`it` - Ion temperature in keV
`vmic` - Micro turbulent velocity in km/s
`ref` - Reference element
`01...30` - Abundances of H to Zn
`file` - Filename for the nonthermal electron distribution

Recommended citation: Kaastra et al. (2004).

4.42 Xabs: photoionised absorption model

The *xabs* model calculates the transmission of a slab of material, where all ionic column densities are linked through a photoionisation model. The relevant parameter is the ionization parameter $\xi = L/nr^2$, with L the source luminosity, n the hydrogen density and r the distance from the ionizing source. The advantage of the *xabs* model over the *slab* model is that all relevant ions are taken into account, also those which would be detected only with marginal significance using the *slab* model. In some circumstances, the combined effect of many weak absorption features still can be significant. A disadvantage of the *xabs* model happens of course when the ionization balance of the source is different from the ionization balance that was used to produce the set of runs with the photo ionization code. In that case the *xabs* model may fail to give an acceptable fit, while the *slab* model may perform better.

The *xabs* model needs an ascii-file as input. The user can provide such a file (see parameter "col" in the parameter list), but there is also a default file in SPEX that is read if the user does not provide a separate file. The default is based on a run with Cloudy, using the spectral energy distribution of NGC 5548 as used in Steenbrugge et al. (2005a). Such an ascii-files contains a pre-calculated list of ionic column densities versus ionisation parameter, as well as electron temperatures versus ionisation parameter (needed for the thermal line broadening). If you want to produce such a file, you can use the auxiliary program *xabsinput*, that will run Cloudy for you and make the file to be used in SPEX. See Sect. 7.2 for more details how to use that program.

For more information on this model, the atomic data and parameters we refer to Sect. 5.2.3.

The parameters of the model are:

`nh` - Hydrogen column density in 10^{28} m^{-2}. Default value: 10^{-4} (corresponding to 10^{24} m^{-2}, a typical value at low Galactic latitudes).

`xi` - the ^{10}log of the ionisation parameter $\log \xi$ in units of 10^{-9} W m. Default value: 1.

The following parameters are common to all our absorption models:
`fcov` - The covering factor of the absorber. Default value: 1 (full covering)
`v` - Root mean square velocity $\sigma_{\rm v}$
`rms` - Rms velocity $\sigma_{\rm b}$ of line blend components
`dv` - Velocity distance Δv between different blend components
`zv` - Average systematic velocity v of the absorber

The following parameters are the same as for the cie-model (see there for a description):
`ref` - Reference element
`01...28` - Abundances of H to Ni; only here we take H, He, C, N, O, Ne, Na, Mg, Al, Si, S, Ar, Ca, Fe, Ni.

The following parameter is unique for the xabs-model and the warm model:
`col` - File name for the photoionisation balance parameters

Recommended citation: Steenbrugge et al. (2003).

Table 4.1: Available spectral components

acronym	description
	additive components:
pow	Power law
delt	Delta line
gaus	Gaussian line
bb	Blackbody
mbb	Modified blackbody
dbb	Accretion disk blackbody
cie	Collisional ionisation equilibrium spectrum
comt	Comptonisation model Titarchuk
cx	Charge exchange model
neij	Non-equilibrium ionisation spectrum
sed	Sedov adiabatic SNR model
chev	Chevalier adiabatic SNR model with reverse shock
soli	Solinger isothermal SNR model
band	Band isothermal SNR model with reverse shock
pdem	Differential emission measure model, polynomials
cf	Isobaric cooling flow model
wdem	Power law differential emission measure with high T cut-off
dem	Differential emission measure model, for DEM analysis
refl	Reflection model of Zycki
file	Table model from file
	multiplicative components, shifts:
reds	Redshift model
	multiplicative components, convolutions:
vgau	Gaussian velocity profile
vblo	Square velocity profile
vpro	Arbitrary velocity profile (needs input file)
lpro	Spatial profile modeling for RGS (needs input file)
laor	Laor relativistic line profile
	multiplicative components, absorption/transmission:
absm	Morrison & McCammon ISM absorption
euve	EUVE absorption model Rumph et al. (1994) (H+He)
ebv	Galactic interstellar extinction model
hot	SPEX absorption by plasma in CIE
slab	Absorption by a slab with adjustable ionic columns
xabs	Absorption by a slab in photoionization equilibrium
pion	Photoionised absorption model
warm	Absorption by a slab with continuous distribution in ionization
knak	Transmission piecewise power law
etau	Transmission for optical depth $\tau(E)$ a power law
dust	Dust scattering model
amol	Absorption by molecular oxygen

Chapter 5

More about spectral models

5.1 Plasma model in SPEX 3.0

5.1.1 The core of the plasma model

The old plasma code used by SPEX in version 2.0 and below is essentially the same plasma code as developed originally by Rolf Mewe and colleagues, with relatively minor updates to the atomic data (like wavelength improvements, corrections of a few typo's, improvements for Fe XVII).

Its basis were precalculated and parametrized line emissivities for each spectral line, as a function of temperature, with for relevant lines empirical density corrections. For some transitiuons, like the He-like triplets, the calculations were rather complex and required several correction factors to account for the full density dependence.

In the new approach presented here, the basic plasma processes are evaluated for each individual level, and then the occupation numbers of the excited states are calculated for the full ion, solving a matrix equation. This has the great advantage that with the same effort a multitude of processes can be taken into account, including effects of photo-excitation and photo-ionisation. From the occupation numbers and the radiative transition probabilities it is then straightforward to calculate the emitted spectrum.

In order to keep the code fast and flexible, we have chosen for a procedure to parametrise all relevant processes, and using simple analytical formulae with a limited number of parameters for each process. This is beneficial both in terms of computation time and storage demands and formed the basis for the succes of Mewe's original work.

The production of the relevant files is not yet complete, but in the first release of version 3.0 we incorporate the data for the H, He, and Li isoelectronic sequences, and some data for the other sequences, including the Fe-L ions. For an overview of what is in the code see Section 5.1.2 below.

By default, the plasma code is the old version 2.0 code, but by giving the command "var calc new", for the ions for which new data are available, the new code is used. This leads in principle to higher accuracy and many more spectral lines. A disadvantage is of course that the computations become somewhat slower. For spectral fitting, one could envision a procedure where in a first run the old code is used to get close to the best parameters, and then to refine using the new plasma core.

If you want to use the new plasma model, it is important to change the ionisation balance from the default (Bryans et al. 2009) to the new *u16* balance. This new balance has improved collisional ionisation rates and allows to calculate properly inner-shell transitions that are needed for the new calculations. The paper describing this (Urdampilleta et al.) will be published soon.

Finally, it is advised that the users have a look to the various ascii-output options that are available for the plasma models, allowing to get deeper insight into the physical process and parameters that are being used.

5.1.2 Ions for which updated calculations are available

Below we list for each iso-electronic sequence what data is available for the new plasma calculations. Each line corresponds to one ion, with *ii, iz and jz* corresponding to the iso-electronic sequence, nuclear charge and ionisation stage, respectively. Then for a number of process we list two quantities: N, the number of entries we have (e.g., number of energy levels or transition rates), and $n_{\max}$ the maximum principal quantum number for wich we use data for that ion and process. Note that $n_{\max}$ refers to the highest level included for a transition between two levels.

The processes incorporated and shown in the table are:

1. *levels* – Energy levels
2. *Arad* – Radiative transition probabilities (Einstein coefficients)
3. *Atwo* – Two-photon processes
4. *El col* – Electron collisional excitation
5. *Pr col* – Proton collisional excitation
6. *Aug* – Auger rates (auto-ionisation, needed for dielectronic recombination)
7. *CX* – Charge exchange recombination
8. *RR* – Radiative recombination
9. *II* – Inner-shell processes (this is merely the number of quantum states after ionisation; the number of fluorescent lines will be larger)

Table 5.1: Hydrogen like.

ii	iz	jz	levels		Arad		Atwo		El Col		Pr col		Aug		CX		RR		II	
			N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}
1	1	1	256	16	3245	16	1	2	289	5	3	2	0	0	16	4	256	16	0	0
1	2	2	256	16	3244	16	1	2	354	10	3	2	0	0	0	0	256	16	0	0
1	3	3	256	16	3242	16	1	2	358	10	3	2	0	0	0	0	256	16	0	0
1	4	4	256	16	3253	16	1	2	362	10	3	2	0	0	18	3	256	16	0	0
1	5	5	256	16	3267	16	1	2	360	10	3	2	0	0	32	4	256	16	0	0
1	6	6	284	20	4305	20	1	2	430	20	3	2	0	0	456	8	256	16	0	0
1	7	7	284	20	4359	20	1	2	430	20	3	2	0	0	542	7	256	16	0	0
1	8	8	284	20	4378	20	1	2	430	20	3	2	0	0	744	9	256	16	0	0
1	9	9	256	16	3345	16	1	2	360	10	3	2	0	0	780	8	256	16	0	0
1	10	10	284	20	4431	20	1	2	430	20	3	2	0	0	1034	9	256	16	0	0
1	11	11	256	16	3364	16	1	2	360	10	3	2	0	0	1306	10	256	16	0	0
1	12	12	284	20	4456	20	1	2	428	20	3	2	0	0	1526	10	256	16	0	0
1	13	13	256	16	3388	16	1	2	360	10	3	2	0	0	1736	11	256	16	0	0
1	14	14	284	20	4506	20	1	2	430	20	3	2	0	0	2084	12	256	16	0	0
1	15	15	256	16	3448	16	1	2	360	10	3	2	0	0	2078	13	256	16	0	0
1	16	16	284	20	4557	20	1	2	430	20	3	2	0	0	2282	14	256	16	0	0
1	17	17	256	16	3502	16	1	2	360	10	3	2	0	0	2544	15	256	16	0	0
1	18	18	284	20	4608	20	1	2	430	20	3	2	0	0	2826	15	256	16	0	0
1	19	19	256	16	3544	16	1	2	360	10	3	2	0	0	2988	15	256	16	0	0
1	20	20	284	20	4664	20	1	2	430	20	3	2	0	0	3038	15	256	16	0	0
1	21	21	256	16	3587	16	1	2	360	10	3	2	0	0	3038	15	256	16	0	0
1	22	22	284	20	4691	20	1	2	430	20	3	2	0	0	3092	15	256	16	0	0
1	23	23	256	16	3613	16	1	2	360	10	3	2	0	0	3092	15	256	16	0	0
1	24	24	284	20	4725	20	1	2	430	20	3	2	0	0	3150	15	256	16	0	0
1	25	25	256	16	3640	16	1	2	360	10	3	2	0	0	2700	15	256	16	0	0
1	26	26	284	20	4757	20	1	2	434	20	3	2	0	0	3150	15	256	16	0	0
1	27	27	256	16	3666	16	1	2	360	10	3	2	0	0	3150	15	256	16	0	0
1	28	28	284	20	4790	20	1	2	430	20	3	2	0	0	3150	15	256	16	0	0
1	29	29	256	16	3702	16	1	2	360	10	3	2	0	0	3150	15	256	16	0	0
1	30	30	256	16	3783	16	1	2	360	10	3	2	0	0	3150	15	256	16	0	0

Table 5.2: Helium like.

ii	iz	jz	levels		Arad		Atwo		El Col		Pr col		Aug		CX		RR		II	
			N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}
2	2	1	511	16	7934	16	1	2	84	7	3	2	0	0	0	0	127	8	2	2
2	3	2	558	16	8777	16	1	2	132	7	3	2	46	3	0	0	127	8	2	2
2	4	3	558	16	8912	16	1	2	754	7	3	2	46	3	0	0	127	8	2	2
2	5	4	558	16	9030	16	1	2	760	7	3	2	46	3	60	4	127	8	2	2
2	6	5	678	16	12345	16	1	2	1678	10	3	2	166	5	560	7	127	8	2	2
2	7	6	678	16	12499	16	1	2	1690	10	3	2	166	5	872	9	127	8	2	2
2	8	7	678	16	12647	16	1	2	1675	10	3	2	166	5	980	6	127	8	2	2
2	9	8	678	16	12289	16	1	2	1591	7	3	2	166	5	108	6	127	8	2	2
2	10	9	678	16	12798	16	1	2	1692	10	3	2	166	5	1702	8	127	8	2	2
2	11	10	678	16	12606	16	1	2	1588	7	3	2	166	5	2006	9	127	8	2	2
2	12	11	678	16	13070	16	1	2	1679	10	3	2	166	5	2530	10	127	8	2	2
2	13	12	678	16	12997	16	1	2	1588	7	3	2	166	5	2994	11	127	8	2	2
2	14	13	678	16	13343	16	1	2	1688	10	3	2	166	5	3416	11	127	8	2	2
2	15	14	678	16	13288	16	1	2	1595	7	3	2	166	5	0	0	127	8	2	2
2	16	15	678	16	13564	16	1	2	1672	10	3	2	166	5	204	10	127	8	2	2
2	17	16	678	16	13437	16	1	2	1589	7	3	2	166	5	204	10	127	8	2	2
2	18	17	678	16	13683	16	1	2	1669	10	3	2	166	5	228	11	127	8	2	2
2	19	18	678	16	13542	16	1	2	1574	7	3	2	166	5	228	11	127	8	2	2
2	20	19	678	16	13744	16	1	2	1672	10	3	2	166	5	252	12	127	8	2	2
2	21	20	678	16	13586	16	1	2	1569	7	3	2	166	5	252	12	127	8	2	2
2	22	21	678	16	13783	16	1	2	1625	10	3	2	166	5	302	13	127	8	2	2
2	23	22	678	16	13615	16	1	2	1537	7	3	2	166	5	276	13	127	8	2	2
2	24	23	678	16	13818	16	1	2	1668	10	3	2	166	5	330	14	127	8	2	2
2	25	24	678	16	13627	16	1	2	1520	7	3	2	166	5	330	14	127	8	2	2
2	26	25	678	16	13834	16	1	2	1645	10	3	2	166	5	4018	15	127	8	2	2
2	27	26	678	16	13618	16	1	2	1576	7	3	2	166	5	358	15	127	8	2	2
2	28	27	678	16	13850	16	1	2	1652	10	3	2	166	5	324	15	127	8	2	2
2	29	28	678	16	13662	16	1	2	1571	7	3	2	166	5	386	16	127	8	2	2
2	30	29	678	16	13679	16	1	2	1603	7	3	2	166	5	348	16	127	8	2	2

Table 5.3: Lithium like.

ii	iz	jz	levels		Arad		Atwo		El Col		Pr col		Aug		CX		RR		II	
			N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}
3	4	2	815	16	19775	16	0	0	262	5	0	0	247	3	0	0	63	8	1	2
3	5	3	815	16	18944	16	0	0	242	5	0	0	263	3	0	0	63	8	1	2
3	6	4	885	16	13679	16	0	0	996	6	0	0	1041	5	30	4	63	8	1	2
3	7	5	885	16	13565	16	0	0	994	6	0	0	1040	5	42	5	63	8	1	2
3	8	6	885	16	13727	16	0	0	978	5	0	0	1041	5	69	7	63	8	1	2
3	9	7	271	16	3483	16	0	0	66	5	0	0	0	0	54	6	63	8	1	2
3	10	8	885	16	13882	16	0	0	981	5	0	0	1041	5	54	6	63	8	1	2
3	11	9	271	16	3520	16	0	0	66	5	0	0	0	0	66	7	63	8	1	2
3	12	10	885	16	14131	16	0	0	981	5	0	0	1041	5	66	7	63	8	1	2
3	13	11	271	16	3564	16	0	0	66	5	0	0	0	0	78	8	63	8	1	2
3	14	12	885	16	14520	16	0	0	989	5	0	0	1041	5	78	8	63	8	1	2
3	15	13	271	16	3591	16	0	0	66	5	0	0	0	0	90	9	63	8	1	2
3	16	14	885	16	14613	16	0	0	975	5	0	0	1051	5	90	9	63	8	1	2
3	17	15	271	16	3646	16	0	0	66	5	0	0	0	0	102	10	63	8	1	2
3	18	16	885	16	14602	16	0	0	976	5	0	0	1051	5	102	10	63	8	1	2
3	19	17	271	16	3710	16	0	0	66	5	0	0	0	0	114	11	63	8	1	2
3	20	18	885	16	14572	16	0	0	975	5	0	0	1051	5	114	11	63	8	1	2
3	21	19	271	16	3748	16	0	0	66	5	0	0	0	0	126	12	63	8	1	2
3	22	20	885	16	14518	16	0	0	973	5	0	0	1051	5	126	12	63	8	1	2
3	23	21	271	16	3781	16	0	0	66	5	0	0	0	0	151	13	63	8	1	2
3	24	22	885	16	14351	16	0	0	969	5	0	0	1051	5	138	13	63	8	1	2
3	25	23	271	16	3812	16	0	0	66	5	0	0	0	0	165	14	63	8	1	2
3	26	24	889	16	14257	16	0	0	1202	7	0	0	2039	5	165	14	63	8	1	2
3	27	25	271	16	3843	16	0	0	66	5	0	0	0	0	150	14	63	8	1	2
3	28	26	885	16	14080	16	0	0	972	5	0	0	1051	5	179	15	63	8	1	2
3	29	27	271	16	3867	16	0	0	66	5	0	0	0	0	162	15	63	8	1	2
3	30	28	885	16	14097	16	0	0	964	5	0	0	1051	5	193	16	63	8	1	2

Table 5.4: Berilium like.

ii	iz	jz	levels		Arad		Atwo		El Col		Pr col		Aug		CX		RR		II	
			N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}
4	6	3	753	16	13857	16	0	0	2121	6	0	0	215	5	0	0	127	8	8	2
4	7	4	778	16	16951	16	0	0	2901	6	0	0	215	5	58	4	127	8	8	2
4	8	5	727	16	11204	16	0	0	2263	6	0	0	215	5	190	7	127	8	8	2
4	9	6	578	16	9524	16	0	0	402	3	0	0	0	0	84	5	127	8	4	2
4	10	7	753	16	13514	16	0	0	2153	5	0	0	215	5	108	6	127	8	8	2
4	11	8	578	16	9774	16	0	0	403	3	0	0	0	0	108	6	127	8	4	2
4	12	9	753	16	14555	16	0	0	2197	5	0	0	215	5	132	7	127	8	8	2
4	13	10	578	16	10112	16	0	0	404	3	0	0	0	0	132	7	127	8	4	2
4	14	11	753	16	14907	16	0	0	2187	5	0	0	215	5	156	8	127	8	8	2
4	15	12	578	16	10435	16	0	0	403	3	0	0	0	0	156	8	127	8	4	2
4	16	13	753	16	15391	16	0	0	2208	6	0	0	215	5	180	9	127	8	8	2
4	17	14	578	16	10727	16	0	0	404	3	0	0	0	0	180	9	127	8	4	2
4	18	15	753	16	15421	16	0	0	2185	5	0	0	215	5	204	10	127	8	8	2
4	19	16	577	16	11014	16	0	0	394	3	0	0	0	0	204	10	127	8	4	2
4	20	17	753	16	15629	16	0	0	2187	5	0	0	215	5	228	11	127	8	8	2
4	21	18	578	16	11299	16	0	0	404	3	0	0	0	0	228	11	127	8	4	2
4	22	19	753	16	15762	16	0	0	2173	5	0	0	215	5	252	12	127	8	8	2
4	23	20	578	16	11509	16	0	0	404	3	0	0	0	0	252	12	127	8	4	2
4	24	21	753	16	15831	16	0	0	2165	5	0	0	215	5	302	13	127	8	8	2
4	25	22	578	16	11632	16	0	0	404	3	0	0	0	0	276	13	127	8	4	2
4	26	23	808	16	23191	16	0	0	3297	7	3	2	493	5	330	14	127	8	8	2
4	27	24	578	16	11745	16	0	0	404	3	0	0	0	0	330	14	127	8	4	2
4	28	25	753	16	15965	16	0	0	2133	5	0	0	204	5	300	14	127	8	8	2
4	29	26	578	16	11849	16	0	0	404	3	0	0	0	0	358	15	127	8	4	2
4	30	27	753	16	15982	16	0	0	2106	5	0	0	215	5	324	15	127	8	8	2

Table 5.5: Boron like.

ii	iz	jz	levels		Arad		Atwo		El Col		Pr col		Aug		CX		RR		II	
			N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}
5	6	2	719	16	18431	16	0	0	549	5	0	0	1117	5	0	0	63	8	16	2
5	7	3	973	16	21616	16	0	0	1372	5	0	0	1117	5	0	0	65	8	16	2
5	8	4	973	16	22272	16	0	0	1298	5	0	0	1117	5	28	4	66	8	16	2
5	9	5	744	16	16610	16	0	0	1012	5	0	0	0	0	42	5	66	8	8	2
5	10	6	973	16	22576	16	0	0	687	5	0	0	1117	5	42	5	67	8	16	2
5	11	7	744	16	16965	16	0	0	1020	5	0	0	0	0	54	6	67	8	8	2
5	12	8	973	16	22211	16	0	0	689	5	0	0	1117	5	54	6	67	8	16	2
5	13	9	744	16	16100	16	0	0	1014	5	0	0	0	0	66	7	67	8	8	2
5	14	10	972	16	22360	16	0	0	650	5	0	0	1117	5	66	7	68	8	16	2
5	15	11	744	16	17123	16	0	0	1013	5	0	0	0	0	78	8	68	8	8	2
5	16	12	973	16	22769	16	0	0	691	5	0	0	1117	5	78	8	68	8	16	2
5	17	13	744	16	17161	16	0	0	1013	5	0	0	0	0	90	9	69	8	8	2
5	18	14	973	16	22615	16	0	0	665	5	0	0	1117	5	90	9	69	8	16	2
5	19	15	744	16	16583	16	0	0	1022	5	0	0	0	0	102	10	69	8	8	2
5	20	16	973	16	22275	16	0	0	701	5	0	0	1117	5	102	10	69	8	16	2
5	21	17	744	16	16705	16	0	0	1022	5	0	0	0	0	114	11	69	8	8	2
5	22	18	973	16	22439	16	0	0	690	5	0	0	1117	5	114	11	69	8	16	2
5	23	19	744	16	16719	16	0	0	1022	5	0	0	0	0	126	12	69	8	8	2
5	24	20	973	16	22101	16	0	0	685	5	0	0	1117	5	126	12	69	8	16	2
5	25	21	972	16	18063	16	0	0	1401	5	0	0	1112	5	151	13	69	8	15	2
5	26	22	1110	16	35949	16	0	0	5028	6	0	0	8512	5	138	13	69	8	16	2
5	27	23	973	16	17956	16	0	0	1404	5	0	0	1117	5	165	14	69	8	16	2
5	28	24	973	16	21700	16	0	0	656	5	0	0	1117	5	165	14	69	8	16	2
5	29	25	744	16	16204	16	0	0	1019	5	0	0	0	0	150	14	69	8	8	2
5	30	26	973	16	21351	16	0	0	1397	5	0	0	1117	5	179	15	69	8	16	2

Table 5.6: Carbon like.

ii	iz	jz	levels		Arad		Atwo		El Col		Pr col		Aug		CX		RR		II	
			N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}
6	6	1	1199	7	23942	5	0	0	4429	5	0	0	0	0	0	0	61	7	20	2
6	7	2	1192	6	24112	5	0	0	5293	5	0	0	0	0	0	0	55	6	20	2
6	8	3	1218	5	24282	5	0	0	5581	5	0	0	5520	5	0	0	55	5	20	2
6	10	5	1218	5	24273	5	0	0	5706	5	0	0	5520	5	204	5	56	5	20	2
6	12	7	1218	5	24899	5	0	0	5788	5	0	0	5520	5	0	0	57	5	20	2
6	14	9	1218	5	24319	5	0	0	5760	5	0	0	5520	5	0	0	57	5	20	2
6	16	11	1218	5	24416	5	0	0	5816	5	0	0	5520	5	0	0	57	5	20	2
6	18	13	1218	5	24151	5	0	0	5929	5	0	0	5520	5	0	0	57	5	20	2
6	20	15	1218	5	24716	5	0	0	5946	5	0	0	5520	5	0	0	57	5	20	2
6	22	17	1218	5	24237	5	0	0	5939	5	0	0	5520	5	0	0	57	5	20	2
6	24	19	1218	5	23259	5	0	0	5982	5	0	0	5520	5	0	0	57	5	20	2
6	26	21	1525	5	25505	5	0	0	17831	5	0	0	68831	5	0	0	57	5	20	2
6	28	23	1218	5	22644	5	0	0	5956	5	0	0	5520	5	0	0	57	5	20	2
6	30	25	1218	5	22378	5	0	0	5945	5	0	0	5520	5	0	0	57	5	20	2

Table 5.7: Nitrogen like.

ii	iz	jz	levels		Arad		Atwo		El Col		Pr col		Aug		CX		RR		II	
			N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}
7	26	20	2073	10	160752	10	0	0	18842	10	0	0	7612	4	0	0	57	8	16	2

Table 5.8: Oxygen like.

ii	iz	jz	levels		Arad		Atwo		El Col		Pr col		Aug		CX		RR		II	
			N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}
8	26	19	2913	10	126083	10	0	0	12244	6	0	0	6838	5	0	0	225	8	0	0
8	28	21	2760	10	115818	10	0	0	4520	5	0	0	7890	5	0	0	225	8	0	0

Table 5.9: Fluor like.

ii	iz	jz	levels		Arad		Atwo		El Col		Pr col		Aug		CX		RR		II	
			N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}
9	26	18	3226	10	124224	10	0	0	2694	9	0	0	3183	6	0	0	252	8	0	0

Table 5.10: Neon like.

ii	iz	jz	levels		Arad		Atwo		El Col		Pr col		Aug		CX		RR		II	
			N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}
10	12	3	894	10	18911	10	0	0	701	10	0	0	846	10	0	0	215	8	0	0
10	13	4	894	10	18940	10	0	0	536	10	0	0	846	10	0	0	215	8	0	0
10	14	5	894	10	19321	10	0	0	1021	10	0	0	842	10	204	5	215	8	0	0
10	16	7	894	10	20629	10	0	0	1311	10	0	0	826	10	276	6	215	8	0	0
10	18	9	894	10	19871	10	0	0	769	10	0	0	814	10	348	7	215	8	0	0
10	20	11	894	10	21169	10	0	0	318	6	0	0	798	10	100	6	215	8	0	0
10	22	13	894	10	20802	10	0	0	864	10	0	0	790	10	492	9	215	8	0	0
10	24	15	894	10	19759	10	0	0	861	10	0	0	762	10	564	10	215	8	0	0
10	26	17	894	10	20434	10	0	0	1302	10	0	0	762	10	400	10	215	8	0	0
10	28	19	894	10	19878	10	0	0	871	10	0	0	740	10	212	10	215	8	0	0
10	30	21	894	10	20140	10	0	0	813	10	0	0	733	10	212	10	215	8	0	0

Table 5.11: Na like.

ii	iz	jz	levels		Arad		Atwo		El Col		Pr col		Aug		CX		RR		II	
			N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}	N	n_{max}
11	20	10	431	10	20248	10	0	0	353	5	0	0	0	0	0	0	60	8	0	0
11	22	12	431	10	20310	10	0	0	352	5	0	0	0	0	0	0	60	8	0	0
11	24	14	431	10	20280	10	0	0	353	5	0	0	0	0	0	0	60	8	0	0
11	26	16	1646	10	98896	10	0	0	2572	5	0	0	0	0	102	10	60	8	0	0
11	28	18	430	10	20240	10	0	0	352	5	0	0	0	0	0	0	60	8	0	0

5.2 Absorption models

5.2.1 Introduction

In most astrophysical sitiuations we have to take into account absorption of photons between the emitting region and the observer. Apart from a few standard models like the ones by Morrison & McCammon (1983, our "absm" model) and Rumph et al. (1994, our "euve" model) we have constructed our own absorption models based upon the atomic database used by SPEX.

Essentially, we adopt a few steps, which will be described below. First, we produce a set of column densities, in different ways for the diferent absorption models (see Sect. 5.2.3). Next, using a dynamical model for the absorber we calculate its transmission (Sect. 5.2.4). For these calculations, we use our atomic database as described in Sect. 5.3.

A basic assumption in all the absorption models is that there is no re-emission, i.e. we look through an absorbing nmedium that has a very small solid angle as seen from the X-ray source. This allows essentially to calculate the transmission simply as $e^{-\tau(E)}$ with $\tau(E)$ the optical depth.

5.2.2 Thomson scattering

The above approach also allows us to include Thomson-scattering in the transmission. Any source photon aimed at the observer but that suffers from Thomson scattering is scattered out of the line of sight and hanec in this approximation is not seen by the observer. We have included not simply the Thomson cross-section σ_T but have taken the Klein-Nishina correction into account (see Rybicki & Lightman 1986), eqn. 7.5 (exact expression) and 7.6 (approximations)). The evaluation of the exact formula for the cross section is non-trivial, as terms up to the third power in $x = E/m_e c^2$ cancel; we have extended the low-energy polynomial approximation (7.6.a) of Rybicki & Lightman by comparing to quadruple precision calculations using the exact formula, and made the following approximation that has a relative accuracy of better than 3×10^{-4} for all energies, when evaluated using single precision arithmetics:

$$\sigma = \begin{cases} \sigma_T(1 - 2x + 5.2x^2 - 13.3x^3 + 32.685x^4) & x < 0.05; \\ 0.75\sigma_T \left[\frac{1+x}{x^3}\left\{\frac{2x(1+x)}{1+2x} - \ln(1+2x)\right\} + \frac{\ln(1+2x)}{2x} - \frac{1+3x}{(1+2x)^2}\right] & 0.05 < x < 5000; \\ 0.375\sigma_T(\ln(2x) + 0.5)/x & x > 5000. \end{cases} \tag{5.1}$$

5.2.3 Different types of absorption models

We have a set of spectral models available with different levels of sophistication and applicability, that we list below.

Slab model

The *slab* model calculates the transmission of a slab of material, where all ionic column densities can be chosen independently. This has the advantage that a spectrum can be fit without any knowledge of the ionisation balance of the slab. After a spectral fit has been made, one may try to explain the observed column densities by comparing the with predictions from any model (as calculated by SPEX, Cloudy, XSTAR, ION or any other existing (photo)ionization code).

Xabs model

In the *xabs* model, the ionic column densities are not independent quantities, but are linked through a set of runs using a photo ionization code. See the description of the *xabs* model for more details about this. The relevant parameter is the ionization parameter $\xi = L/nr^2$, with L the source luminosity, n the hydrogen density and r the distance from the ionizing source. The advantage of the *xabs* model over the *slab* model is that all relevant ions are taken into account, also those which would be detected only with

marginal significance using the *slab* model. In some circumstances, the combined effect of many weak absorption features still can be significant. A disadvantage of the *xabs* model happens of course when the ionization balance of the source is different from the ionization balance that was used to produce the set of runs with the photo ionization code. In that case the *xabs* model may fail to give an acceptable fit, while the *slab* model may perform better.

Warm model

In the *warm* model, we construct a model for a continuous distribution of column density $N_{\rm H}$ as a function of ξ. It is in some sense comparable to the differential emission measure models used to model the emission from multi-temperature gas. Here we have absorption from multi-ionization gas. Depending upon the physics of the source, this may be a better approximation than just the sum of a few *xabs* components. A disadvantage of the model may be (but this also depends upon the physics of the source), that all dynamical parameters for each value of ξ are the same, like the outflow velocity and turbulent broadening. If this appears to be the case in a given source, one may of course avoid this problem by taking multiple, non-overlapping *warm* components.

Hot model

In the *hot* model, we link the different ionic column densities simply by using a collisional ionsation (CIE) plasma. It may be useful in situations where photoionisation is relatively unimportant but the source has a non-negligible optical depth. A special application is of course the case for a low temperature, where it can be used to mimick the absorption of (almost) neutral gas.

Pion model

Finally we have in SPEX the pion model, which does a self-consistent photo ionization calculation of the slab of material.

5.2.4 Dynamical model for the absorbers

For each of the absorption models described in the previous section, we have the freedom to prescribe the dynamics of the source. The way we have implemented this in SPEX is described below.

The transmission $T(\lambda)$ of the slab is simply calculated as

$$T(\lambda) = \exp[-\tau_c(\lambda) - \tau_l(\lambda)] \tag{5.2}$$

with τ_c and τ_l the total continuum and line optical depth, respectively. As long as the thickness of the slab is not too large, this most simple approximation allows a fast computation of the spectrum, which is desirable for spectral fitting.

In particular UV observations of AGN show that the absorption lines can often be decomposed into multiple velocity components. In the X-ray band these are not always fully resolvable, which led us to the following approach. Each absorption line is split into different velocity components, using

$$\tau_l(v) = \sum_i \tau_i \exp\left[-(v - v_i)^2/2\sigma_{\rm v}^2\right] \tag{5.3}$$

(or the equivalent generalisation to the Voigt profile). Further, we take

$$v_i = v_0 + i\,\Delta v, \tag{5.4}$$

$$\tau_i = K \exp\left[-v_i^2/2\sigma_{\rm b}^2\right], \tag{5.5}$$

where v_0 is the average velocity of the blend (a negative value corresponds to a blue-shift or outflow), Δv is the separation between the velocity components, and the r.m.s. width of the blend $\sigma_{\rm b}$ is in general

larger than the intrinsic width σ_v of the components (do never confuse both σ's!). The normalization K is defined in such a way that $\sum \tau_i = \tau_0$. Finally, the total optical depth τ_0 is given by

$$\tau_0 = 0.106 f N_{20} \lambda / \sigma_{v,100}. \tag{5.6}$$

Here f is the oscillator strength, λ the wavelength in Å, $\sigma_{v,100}$ the velocity dispersion in units of 100 km/s and N_{20} the total column density of the ion in units of 10^{20} m^{-2}.

This dynamical structure offers the user a broad range of applicability. However, we advise the user to use the extension with σ_b with caution! Always start with the most simple case. The default values for SPEX are defined in such a way that $\sigma_b = 0$. This will produce the "normal" case of single absorption lines. In that case, the velocity separation Δv is an irrelevant parameter.

Finally, we make a remark on the r.m.s. line width of individual lines, σ_v. In our code, this *only* includes the turbulent broadening of the lines. The thermal broadening due to motion of the ions is included by adding it in quadrature to the tutbulent broadening. The only exception is the *slab* model, where of course due to the lack of underlying physics the thermal broadening is unknown, and therefore in using the slab model one should be aware that σ_v also includes a thermal contribution.

5.3 Atomic database for the absorbers

The continuum opacities are taken from Verner & Yakovlev (1995). Line opacities and wavelengths for most ions are from Verner et al. (1996), with the following additions and exceptions.

5.3.1 K-shell transitions

For some hydrogenic ions (C, N, O, Ne, Mg, Si, S and Ar) we added the transitions from principal quantum numbers 11–20 to the ground using our own SPEX database.

C-K transitions

Three inner-shell K-shell transitions for C I were calculated by Raassen using the Cowan code, but here only the oscillator strengths were calculated, and therefore calculated equivalent widths for large optical depths will be too weak. We added two C IV lines from the work of Nahar et al. (2001).

N-K transitions

For N I the K-shell transitions were calculated by Raassen using the Cowan code. We adjusted the wavelength of the strongest 1s–2p line acording to measurements. However, for the 1s–2p lines towards $1s2s^22p^4\ ^4P$, we used oscilator strengths and Auger widths from García et al. (2009).

Inner-shell K-shell transitions for N II were also calculated by Raassen using the Cowan code, but here only the oscillator strengths were calculated, and therefore calculated equivalent widths for large optical depths will be too weak. The exception to this are the 1s–2p absorption lines to the $1s2s^22p^3\ ^3S_1$, $^3P_1$ and $^3D_1$ levels, for which we use García et al. (2009).

For N III we added the 1s–2p absorption lines towards the $1s2s^22p^2\ ^2S$, ^2P and $^2D_{3/2}$ levels from the paper by García et al. (2009). Wavelengths are theoretical, so may be somewhat off.

For N IV we added the 1s–2p absorption lines towards the $1s2s^22p\ ^1P_1$ and $1s2p^3\ ^1P_1$ levels from the paper by García et al. (2009). Wavelengths are theoretical, so may be somewhat off.

For N V we added the 1s–2p absorption lines towards the 1s(2S)2s2p(3P) ^{2}P doublet and 1s(2S)2s2p(1P) ^{2}P doublet from the paper by García et al. (2009). Wavelengths were corrected according to the measurements of Beiersdorfer et al. (1999).

For N VI we added the 1s–np lines for $n = 5 - 7$ from our SPEX database.

O-K transitions

We included the inner shell K-shell transitions of oxygen ions (O I – O VI) from earlier work of Behar (HULLAC calculations, private commumication).

We adjusted the wavelength of the strongest O V line to 22.370 ± 0.010 Å, taken from Gu et al. (2005).

The two 1s–2p $^2P_{1/2,3/2}$ lines of O IV were adjusted to 22.741 and 22.739 Å, respectively following Gu et al. (2005). The lines to the $^2D_{3/2}$ and $^2S_{1/2}$ terms were not adjusted as no values are given by Gu et al. (2005).

For O III, Gu et al. (2005) give only a single line instead of the three dominant lines to $^3D_1$, $^3S_1$, $^3P_1$. The oscillator-strength weighted average wavelength for these three lines using Behar's HULLAC calculations is 23.058 Å, compared to 23.065 Åas given by Gu et al. (2005). Other sets of calculation differ much from this, up to 0.05–0.10 Å (Olalla et al. 2002; Pradhan et al. 2003; Juett et al. 2004); so we keep the Behar values lacking more reliable calculations.

For O II, Juett et al. (2004) identified a feature in observed Chandra HETGS spectra towards several sources with the 1s–2p line from this ion; their measured wavelength is 23.351±0.003 Å. This identification seems to be justified given the Auger electron measurements of Krause (1994) and Caldwell et al. (1994) as cited by García et al. (2005). The relevant transitions are from the ground $^4S_{3/2}$to the $^4P_{5/2}$, $^4P_{3/2}$, and $^4P_{1/2}$ levels; the oscillator strength weighted wavelength from Behar is 23.3012 Å. Therefore we shift the wavelengths of these transitions by +0.0498 Å in order to match the Juett et al. (2004) value.

Finally, in a similar way we shift the strong 1s–2p doublet from O I to match the weighted average of the value found by Juett et al. (2004) with the Chandra HETGS (23.508±0.003 Å) and in Mrk 421 with RGS (23.5130±0.0022 Å) to (23.5113±0.0018 Å). For all O I lines, we have updated the oscillator strengths and transition rates to values obtained by Ton Raassen using Cowan's code, and benchmarked to an adjusted wavelength scale.

Lines from O VII up to $n = 100$ were calculated by extrapolation of the data for $n \leq 10$.

Ne-K to Ca-K transitions

The strongest K-shell transitions from Li-like to F-like ions of Ne, Mg, Al, Si, S, Ar, and Ca were taken from Behar & Netzer (2002).

Fe-K transitions

K-shell transitions in Fe II–Fe XXIV were included from the series of papers by Palmeri et al. (2003b), Mendoza et al. (2004) and Bautista et al. (2004).

In addition, the strongest 1s–3p transitions in Fe XXIII were added from Behar & Netzer (2002).

5.3.2 L-shell transitions

Fe-L transitions

For the L-shell ions of iron (Fe XVII − Fe XXIV) we used HULLAC calculations. The wavelengths of these L-shell transitions were adjusted according to Phillips et al. (1999). Also the L-shell ions of Si VIII–Si XII, S X–S XIV and Ar XV were calculated using the HULLAC code. In addition, we added the 2p–3d inner shell resonance lines of Fe I−Fe XVI from Behar et al. (2001). These inner shell resonance lines occur mainly in the 16−17 Å band and appear to be extremely important diagnostic tools, since the normal resonance lines of these lowly ionized iron ions have their wavelengths mainly in the inaccessible EUV band.

Ni-L transitions

L-shell transitions of Ni I–Ni XVIII were calculated by Raassen using Cowan's code, with the neglect of line broadening due to auto-ionizations.

5.3.3 M-shell transitions

For M-shell transitions of Fe ions, we have used data calculated by Ehud Behar using HULLAC (Fe I – Fe XVI). The calculations for (Fe VI – Fe XVI) were replaced on November 21, 2008 by updated calculations by Ming Feng Gu using FAC and compared with the NGC 3783 spectrum in Gu et al. (2006). They represent slight (0.03 Angstrom) shifts from Behar's initial calculations. Also, they exhibit typically 2–3 times higher total transition rates (Auger rates).

5.4 Non-equilibrium ionisation (NEI) calculations

Our current implementation of time-dependent, non-equilibrium ionisation spectra is based upon the work of Kaastra & Jansen (1993), KJ hereafter. The method consists essentially upon calculating the transition matrix (containing ionisation and recombination rates) for a grid of temperatures, calculating the eigenvalues and eigenvectors of these matrices and also the inverse eigenvector matrix, and storing those on disk. The ionisation balance is calculated then by dividing the temperature history in several steps, with the steps defined in such a way that the temperature over each time interval is close to one temperature grid point. Then for each step the time evolution of the plasma is calculated using eigenvalue decomposition and using the pre-calculated coefficients for the relevant temperature. For more details see KJ.

Here we make a few changes with respect to KJ. First, KJ adopted a prescribed history of the Hydrogen density, and calculate the corresponding electron density at each time from the ion concentrations at that time. This was motivated by the argument that for a fixed plasma element the number of Hydrogen atoms remains constant over time, while the number of electrons depends upon the degree of ionisation and may be time-dependent, thus not known before the ionisation balance has beeen solved completely. But the situation is slightly more complicated. If the plasma is not (nearly) completely ionised, then for normal abundances there must be a considerable population of neutral Hydrogen or Helium. But in that case charge exchange reactions become important, and should be taken into account in the transition matrix. This however is impossible to incorporate in the KJ method, since in this case there are 3 free parameters (n_e/n_H, n_e/n_{He} and T) instead of 1 (T), and storage of matrices etc. becomes physically impossible. Knowing that the method of KJ becomes inaccurate for low ionisation rates we decided to leave the prescribed n_H concept and return to the prescribed n_e concept. This has the advantage that the equations for the ionisation balance of each chemical element can be calculated independently from the other elements; KJ need to calculate the concentrations of all elements for each time step in order to determine the conversion from Hydrogen to electron density.

Another rationale behind the above-mentioned modification is the following. For self-similar supernova remnants, a class of problems where non-equilibrium ionisation applies, the hydrodynamics is inconsistent with a time- or place varying electron/Hydrogen ratio: that would introduce another scale length and destroy the assumptions behind self-similarity. Thus, for SNR models it is necessary to assume that the plasma is (almost) completely ionised in order to assure self-consistency. We will neglect small changes in the electron/Hydrogen ratio of SNRs therefore.

The second modification concerns the temperature grid. Contrary to KJ, we use a keV grid instead of a K grid. Further, since we now include 30 chemical elements instead of 15, we had to reduce the step size h of the grid from 0.05 in $\log T$ to 0.10, in order not to get disk storage problems. This change however is compensated by a better interpolation technique used here. KJ just rounded all temperatures to the nearest grid point. Here we interpolate the calculated concentrations between two temperature grid points linearly, which is one order of magnitude (h) better. A rough estimate learns that the error in the method of KJ per decade is nh^2 which gives for $n = 1/h$ an error of order 0.05, while the current

implementation has typical errors per decade of nh^3 which gives a typical error of 0.01 per decade.

5.5 Non-thermal electron distributions

In the current version of SPEX it is possible to include the effects of non-thermal (NT) electron distributions. Such NT distributions affect the spectrum in two ways: by changing the ionization balance and the emitted spectrum. Currently we take both effects into account for all our plasma models. The only exception is the NEI model, for non-equilibrium ionisation. The way we calculate the ionisation balance does not allow us to include the NT effects in the ionisation balance, but we do take it into account in the spectrum.

The way we implemented it is as follows. Our atomic data (like collision strengths) are all parameterized in a way that allow for analytical integration over simple electron distributions like delta-functions, Maxwellians or power laws. But there are other processes like radiative recombination where such an analytical approach fails. However, one way or the other, we use expressions for Maxwellian-averaged rates in all our calculations. In principle, it is possible to decompose any electron distribution as a linear combination of Maxwellians. The total rate, for example a recombination rate, is then the sum of the rates caused by the individual Maxwellian components.

Therefore, in all our rate calcualtions we build the total rate based upon such a linear combination of Maxwellians. This is done for all relevant processes (ionization, recombination, excitation, etc.)

In all our thermal models, there is an ascii-parameter called "file"; if this value is defined (i.e. when a file name of an existing file is entered), it will read the parameters of the Maxwellians from an ascii-file with that filename. If there is not such a file, or if the filename is reset by entering the "par file aval none" command, no file will be taken into account (i.e., we have a simple, single Maxwellian again).

The (ascii) file should have the following format. On the first line, the number of Maxwellians is given. Then for each Maxwellian there is a separate line containing two numbers: the first number is the temperature of the Maxwellian, in units of the main temperature of the plasma; the second number is the total number of electrons in this Maxwellian, relative to the main component. It is wise to have the parameters of the main component as the first line of this list.

Let us give an example. Suppose we have a plasma with three components: 2, 20 and 200 keV Maxwellians, with electron densities of 3000, 300 and 30 m^{-3}, respectively. In this case the parameters of the *cie* model should be: temperature 2 keV, electron density 3000 m^{-3}, and the corresponding file should be as follows:

```
3
1 1
10 0.1
100 0.01
```

The first line tells us that there are three Maxwellians. The second line contains the parameters of the first Maxwellian, that we scale here to 1 1 (it is the one with temperature 2 keV and electron density 3000 m^{-3}). The third lines contain the second Maxwellian, which has a 10 times higher temperature but also a 10 times lower electron density as the first component. Finally, the fourth line contains the parameters of the third Maxwellian, which has a 100 times higher temperature and a 100 times lower electron density as the first component.

WARNING: *Note that you can give your ascii-file any name, except of course the name "none", which is reserved for resetting the file name.*

WARNING: *Note that both numbers for each Maxwellian must be positive, i.e. positive temperatures and electron densities.*

5.6 Supernova remnant (SNR) models

For the calculation of SNR models we follow Kaastra & Jansen (1993). But also here we make some modifications in the implementation.

The X-ray spectrum of a SNR is characterised in lowest order by its average temperature T, ionisation parameter U and emission measure Y. If one calculates a specific model however, these average values are not known a priori. A relevant set of input parameters could be the interstellar density, the shock radius and the explosion energy. These are the parameters used by KJ. A disadvantage is that this choice is not suitable for spectral fitting: changing one of the parameters gives correlated changes in T, U and Y, which is an undesirable feature in spectral fitting. KJ solved this by inputting a guess of T, U, and Y and using these guess parameters as the fitting parameters. After fitting, the model had to be calculated another time in order to find the relation between guessed and true parameters. here we take another choice. We will select input parameters which scale with T, U and Y respectively but are connected to the SNR parameters independent of the hydrodynamical model. The first parameter is the shock temperature

$$T_s, \tag{5.7}$$

the second the shock ionisation parameter defined by

$$U_s \equiv n_{\rm e} t \tag{5.8}$$

where $n_{\rm e}$ is the electron density before the shock (for a completely ionised gas), and t the age of the remnant. Further we define the shock reduced emission measure by

$$Y_s \equiv n_{\rm e} n_{\rm H} r_s^3 / d^2 \tag{5.9}$$

where r_s is the main shock radius, and d the distance to the source. We have introduced here also the electron density n_e instead of the more natural Hydrogen density, because this allows an efficient use of the model for spectral fitting: if one fits a spectrum with variable chemical abundances, then using the current scaling we need to calculate (for a given T_s, U_s and Y_s) the relative ion concentrations only once; for other abundances, the relative concentrations may be simply scaled without a need to redo completely the ionisation balance calculations. Finally we introduced the distance in the definition of Y_s in order to allow an easy comparison with spectra observed at earth. These 3 input parameters T_s, U_s and Y_s will serve as input parameters for all hydrodynamical models. They are linked to other parameters as follows:

$$f_{\rm p} k T_s = f_{\rm T} \epsilon_\rho m_{\rm p} r_s^2 / t^2 \tag{5.10}$$

where ϵ_ρ is a dimensionless factor representing the average mass in units of a proton mass per Hydrogen atom ($\rho = \epsilon_\rho m_{\rm p} n_{\rm H}$ with $m_{\rm p}$ the proton mass), and $f_{\rm p}$ is another dimensionless factor, representing the ratio of electron plus ion density over the Hydrogen density. Further, $f_{\rm T}$ is a model-dependent dimensionless scaling factor which is given by

$$f_{\rm T} = \frac{f_v^2}{\eta - 1} \tag{5.11}$$

with f_v the velocity scaling discussed below and η the density jump at the main shock front, which depends upon the Hydrodynamical model. Another scaling law links the gas velocity to the temperature:

$$v_2 = f_v r_s / t \tag{5.12}$$

with v_2 the post-shock gas velocity (not to be confused with the shock velocity!). The velocity scaling factor is given by

$$f_v \equiv \frac{(n-3)(\eta-1)}{(n-s)\eta} \tag{5.13}$$

with the understatement that for the Sedov and Solinger model one should take formally the limit $n = 5$. The explosion energy E_0 is linked to the other parameters by

$$E_0 = \frac{\alpha r_s^5 \rho}{t^2}, \tag{5.14}$$

where α is the dimensionless energy integral introduced by Sedov. The value of α depends both upon the hydrodynamical model and the value for s and n, the density gradient scale heights of the interstellar medium and the stellar ejecta.

If we express T_s in keV, U_s in 10^{20} m^{-3}s, Y_s in 10^{20} m^{-5}, r_s in units of 10^8 m, $n_{\rm H}$ in 10^{20} m^{-3}, d in units of 10^{22} m and t in s, we obtain from (5.10) the value for the proton mass of $m_{\rm p} = 1.043969 \times 10^5$ keVs^2(10^{16}m^2)$^{-1}$. Using this value and defining $f_m = f_T \epsilon_\rho m_{\rm p}/f_{\rm p}$, the scaling laws for SNRs may be written as

$$n_{\rm e} = \frac{T_s^{1.5} U_s^3}{f_m^{1.5} Y_s d^2} \tag{5.15}$$

$$t = \frac{f_m^{1.5} Y_s d^2 n_{\rm e}}{T_s^{1.5} U_s^2 n_{\rm H}} \tag{5.16}$$

$$r = \frac{f_m Y_s d^2 n_{\rm e}}{U_s^2 T_s n_{\rm H}} \tag{5.17}$$

$$v = \frac{f_v T_s^{0.5}}{f_m^{0.5}} \tag{5.18}$$

$$E_0 = [1.6726231 \times 10^{33}\ \text{J}] \frac{\alpha \epsilon_\rho f_m^{0.5} Y_s^2 d^4 n_{\rm e}}{U_s^3 T_s^{0.5} n_{\rm H}} \tag{5.19}$$

$$M = [8.409 \times 10^{-14}\ \text{M}_\odot] \frac{\epsilon_\rho \beta \eta f_m^{1.5} Y_s^2 d^4 n_{\rm e}}{U_s^3 T_s^{1.5} n_{\rm H}} \tag{5.20}$$

where M is the mass integral, and β is the dimensionless mass integral (defined by $\beta = 4\pi \int \frac{\rho}{\rho_s} \frac{r}{r_s}^2 \mathrm{d}\frac{r}{r_s}$).

Chapter 6

More about plotting

6.1 Plot devices

The following devices are available (may depend somewhat on your local operating system, i.e. the allowed devices in your local pgplot installation):

1. null : Null device (no output)
2. xwin : Workstations running X Window System
3. xser : Persistent window on X Window System
4. ps : PostScript printers, monochrome, landscape
5. vps : Postscript printers, monochrome, portrait
6. cps : PostScript printers, color, landscape
7. vcps : PostScript printers, color, portrait
8. gif : GIF-format file, landscape
9. vgif : GIF-format file, portrait

WARNING: *Note that when you enter the device name in SPEX you do not need to enter the leading shlash; this is only required in other applications using the pgplot library.*

6.2 Plot types

There are several plot types available in SPEX. below we list them together with the acronym that should be used when defining this plot type (in a "plot type" command).

- data – a spectrum observed with an instrument plus optionally the folded (predicted) model spectrum and/or the subtracted background spectrum.
- model – the model spectrum, not convolved by any instrumental profile
- area – the effective area of the instrument. If the response matrix of the instrument contains more components, also the contribution of each component is plotted separately.
- fwhm – the peak energy of the response matrix (as the model) as well as the FWHM limits (as the data) are plotted. The FWHM limits are the energies at which the response reaches half of its maximum value. If for the y-axis the option "de/e" is chosen, all calculated energies are divided by the energy of the incoming photon.

Table 6.1: Available plot colours.

Value	Colour
00	Black (background)
01	White (default)
02	Red
03	Green
04	Blue
05	Cyan (Green + Blue)
06	Magenta (Red + Blue)
07	Yellow (Red + Green)
08	Orange (Red + Yellow)
09	Green + Yellow
10	Green + Cyan
11	Blue + Cyan
12	Blue + Magenta
13	Red + Magenta
14	Dark Gray
15	Light Gray

- chi – the fit residuals, either expressed in units such as counts/s or the same units as the spectrum, or expressed in terms of numer of standard deviations, or as a relative error.
- dem – a differential emission measure distribution).

6.3 Plot colours

Table 6.1 lists the plot colours that are available. See also Fig. 6.3.

6.4 Plot line types

When drawing lines, the following line types are possible:

1. `----------`
2. `- - - - - - -`
3. `.-.-.-.-.-.-.`
4. `............`
5. `-..-..-..-..-`

The first is the default value.

Further, the user can specify the thickness of the lines. Default value is 1, larger values give thicker lines. Maximum value is 200. Recommended value for papers: use line weight 3. For an example of different line weights, see also Fig. 6.4.

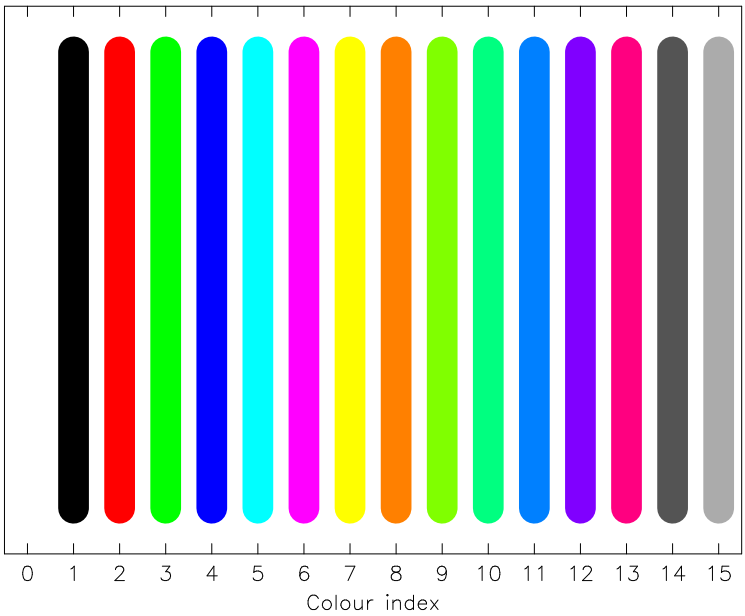

Figure 6.1: Plot colours available with pgplot. Note that on a hardcopy colour 1 is black and the background (colour 0) is white, while on most terminal screens colour 0 is balck and colour 1 is white.

6.5 Plot text

In SPEX it is possible to draw text strings on the plot surface or near the frame of the plot. Several properties of the text can be adjusted, like the fontheight, font type, colour, orientation and location of the text. See Fig. 6.5.1 for an example of possible fonts.

6.5.1 Font types (font)

The following fonts types are allowed (values between 1–4):

1. normal font (default)
2. roman font
3. italic font
4. script font

6.5.2 Font heights (fh)

The font height can be entered as a real number, the default value is 1. A useful value for getting results that are still readable for publications is 1.3. Note that if you make the font height for the captions of the figure too large, you may loose some text off the paper.

6.5.3 Special characters

The texts strings that are plotted may be modified using the pgplot escape sequences. These are character–sequences that are not plotted, but are interpreted as instructions to change the font, draw superscripts or subscripts, draw non-ASCII characters, Greek letters, etc. All escape sequences start with

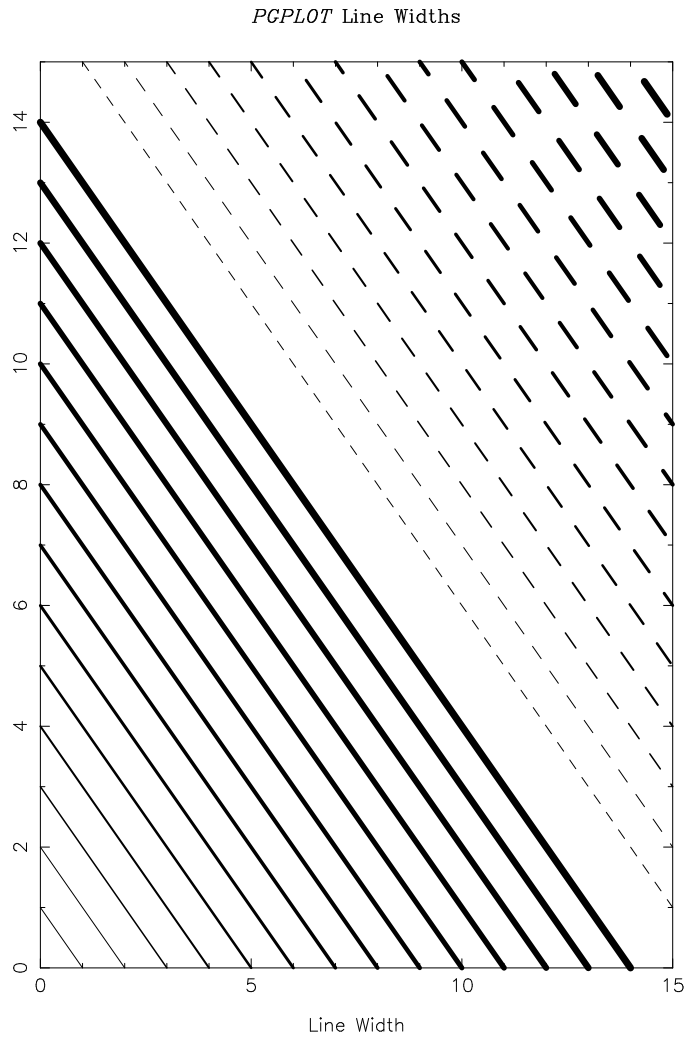

Figure 6.2: Example of different line weights.

a backslash character (\). A list of the defined escape sequences is given in tab. 6.2. A lookup table for Greek letters is presented in tab. 6.3. Some useful non-ASCII characters are listed in tab. 6.4. Fig. 6.4 shows some examples of the use of pgplot escape sequences in character strings.

6.6 Plot captions

Spex has a set of captions defined for each plot. The properties of these captions as well as the text can be modified by the user. For the properties (font types etc.) see Sect. 6.5.

The following captions exist:

- x : the x-axis, plotted below the frame
- y : the y-axis, plotted left from the frame
- z : the z-axis (for contour plots and images), plotted at the upper left edge of the frame
- ut : the upper title text, plotted in the middle above the frame

Table 6.2: A list of available escape sequences.

Seq.	description
`\u`	start a superscript or end a subscript. A `\u` must be ended by a `\d`!
`\d`	start a subscript or end a superscript. A `\d` must be ended by a `\u`!
`\\`	backspace (i.e. do not advance textpointer after plotting the previous character)
`\A`	Ångstrom symbol (Å)
`\g`x	greek letter corresponding to roman letter x
`\fn`	switch to Normal font
`\fr`	switch to Roman font
`\fi`	switch to Italic font
`\fs`	switch to Script font
\(n)	character number n, see *pgplot manual appendix B, tab. 1*

Table 6.3: List of upper- and lower case Greek letters (G) and their corresponding Roman letters (R).

R :	A	B	G	D	E	Z	Y	H	I	K	L	M	N	C	O	P	R	S	T	U	F	X	Q	W
G :	A	B	Γ	Δ	E	Z	H	Θ	I	K	Λ	M	N	Ξ	O	Π	P	Σ	T	Υ	Φ	X	Ψ	Ω
R :	a	b	g	d	e	z	y	h	i	k	l	m	n	c	o	p	r	s	t	u	f	x	q	w
G :	α	β	γ	δ	ϵ	ζ	η	θ	ι	κ	λ	μ	ν	ξ	o	π	ρ	σ	τ	v	ϕ	χ	ψ	ω

Table 6.4: Some useful non-ASCII character sequences.

$\sim$	2248	$\pm$	2233	∇	0583
$\approx$	0248	$\mp$	2234	$\surd$	2267
$\cong$	0250	$\times$	2235	$\int$	2268
$\propto$	2245	$\div$	2237	$\oint$	2269
$\neq$	2239	$\equiv$	2240	∞	2270
$\leqq$	2243	†	2277	∂	2286
$\geqq$	2244	‡	2278	$\odot$	2281

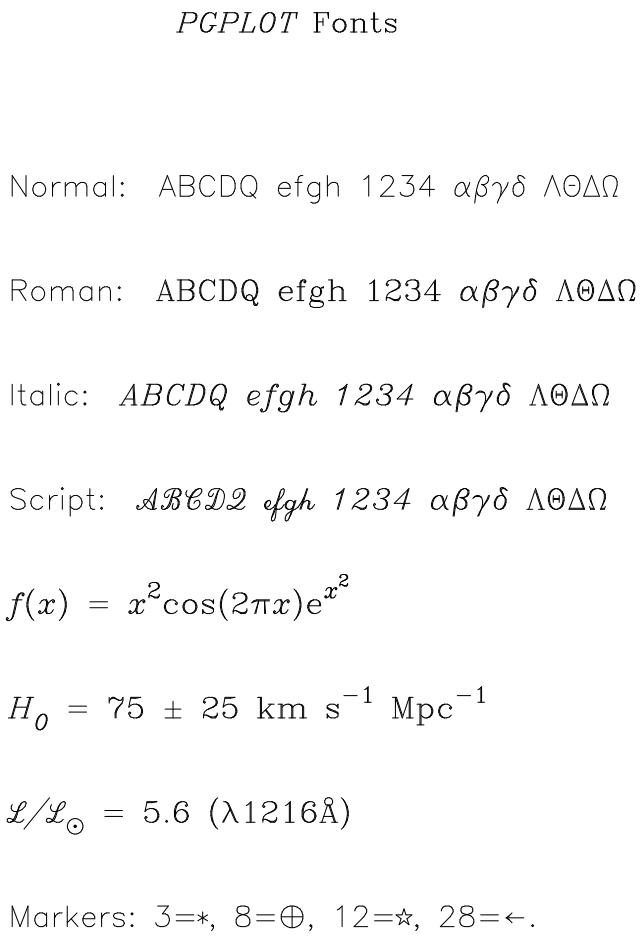

Figure 6.3: Example of different fonts.

- lt : the lower title text, plotted between the upper title text and the frame
- id : the plot identification, plotted to the upper right corner of the frame (usually contains the SPEX version and time and date).

With the "plot cap" series of commands, the text, appearance and position of all these captions can be modified.

6.7 Plot symbols

When plotting data, the data can be labeled with different symbols as indicated in Fig. 6.7.

6.8 Plot axis units and scales

6.8.1 Plot axis units

Different units can be chosen for the quantities that are plotted. Which units are available depends upon the plot type that is used, for example an observed spectrum (data), a Differential Emission Measure

Displayed	pgplot escape sequence
$f(x) = x^2 \cos(2\pi x)$	`\fif(x) = x\u2\d \frcos\fi(\fr2\fi\gpx)`
$H_0 = 75 \pm 25$ km s^{-1} Mpc^{-1}	`\fiH\d0\u\fr = 75 \(2233) 25 km s\u-1\d Mpc\u-1\d`
$\mathcal{L}/\mathcal{L}_\odot = 5.6$ (λ1216Å)	`\fsL/L\fr\d\(2281)\u = 5.6 (\gl1216\A)`

Figure 6.4: Some examples of the use of pgplot escape sequences.

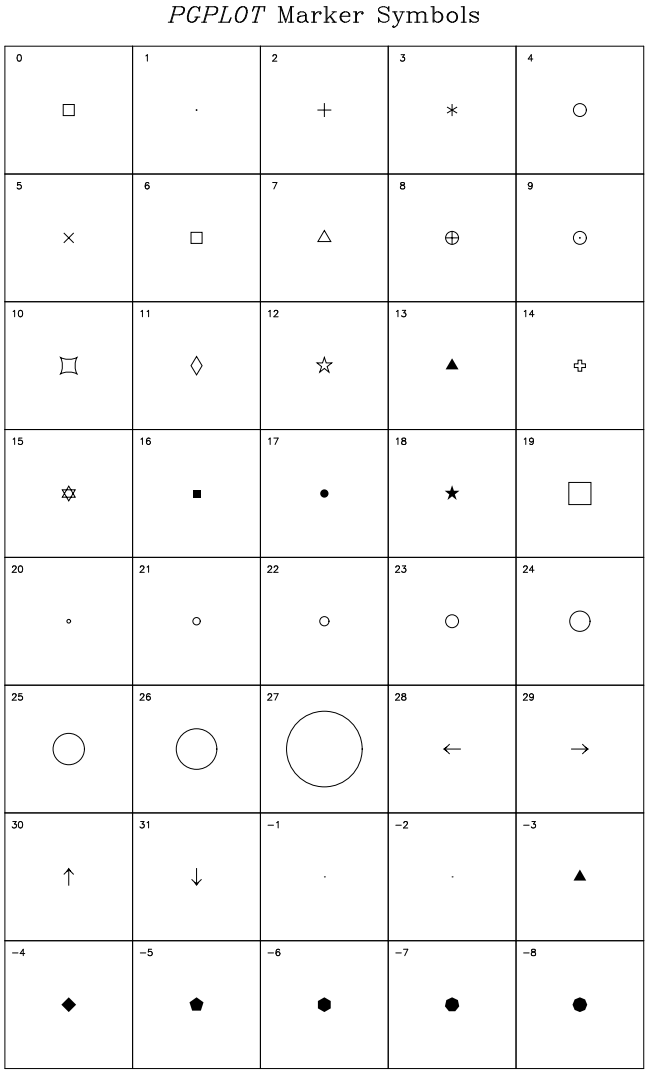

Figure 6.5: Plot symbols that are available.

distribution (DEM) etc. For the availabe plot types see Sect. 6.2.

Note also that depending upon the plot type, there are two or three axes available, the x-axis, y-axis and z-axis. The last axis is only used for contour maps or images, and corresponds to the quantity that is being plotted in the contours or image.

The units that are allowed for the x-axis are listed in Tables 6.5–6.6, and for the y-axis in Tables 6.7–6.12.

Note: when using the velocity scale, negative values correspond to a blueshift, positive values to a redshift. Also note that in this case one needs to provide the reference energy or wavelength.

WARNING: *For the plot type data option, there are options (starting with "f") that are essentially counts* m^{-2}*. In these cases, the observed count rates (counts/s/keV for instance) are divided by the nominal effective area at the relevant energy. This nominal effective area is determined for a flat input spectrum, and is defined as the ratio of predicted count rate and model spectrum flux. Response parts below 0.75 times the original photon energy are ignored in the estimation of the predicted count rate, in an attempt to remove higher order spectral lines. Note however that if your spectrum really contains significant higher order lines, you cannot remove these from your plot. Also beware of end-effects. If you want to see the underlying model, it is often better to select the plot type model!*

6.8.2 Plot axis scales

The axes in a plot can be plot in different ways, either linear or logarithmic.

For the y-axis we also have made an option for a mixed linear and logarithmic scale. The lower part of the plot will be on a linear scale, the upper part on a logarithmic scale. This option is useful for example for plotting a background-subtracted spectrum. Often there are bins with a low count rate which scatter around zero after background suntraction; for those channels a linear plot is most suited as a logarithmic scaling cannot deal with negative numbers. On the other hand, other parts of the spectrum may have a high count rate, on various intemsity scales, and in that case a logarithmic representation might be optimal. The mixed scaling allows the user to choose below which y-value the plot will be linear, and also which fraction of the plotting surface the linear part will occupy. For an example see Fig. 6.8.2.

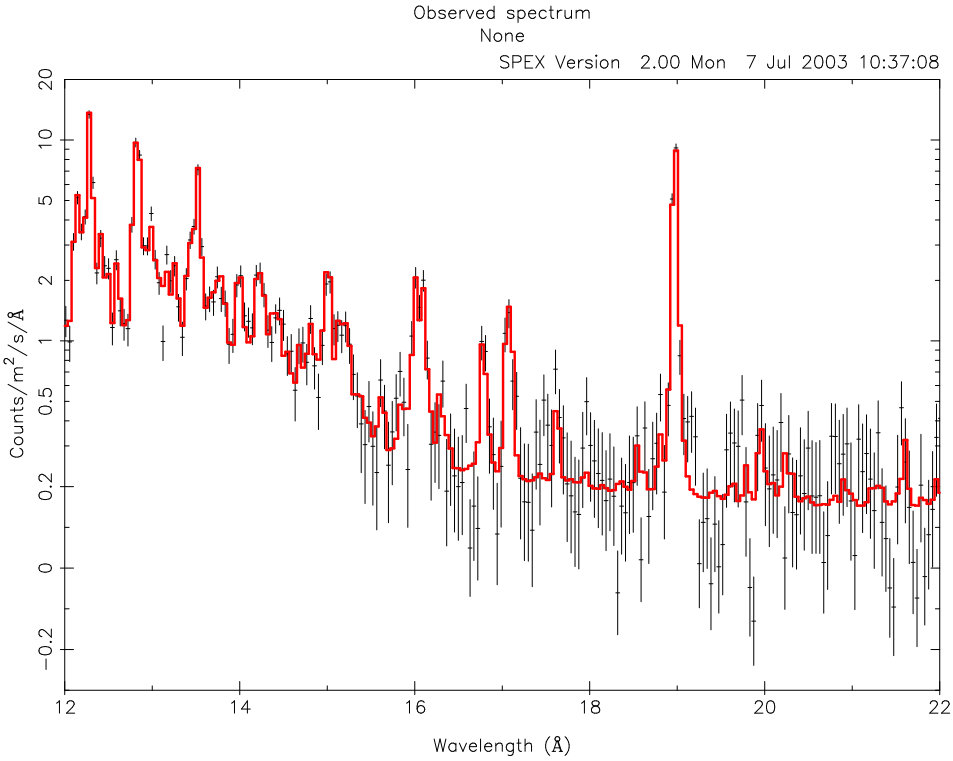

Figure 6.6: Example of the mixed y-axis scale. Simulation of a Chandra LETGS spectrum for a thermal plasma with a temperature of 1 keV.

Table 6.5: x-axis units for plot type model, data, chi, area, fwhm and spec:

bin	Bin number
kev	keV (kilo-electronvolt), default value
ev	eV (electronvolt)
ryd	Rydberg units
hz	Hertz
ang	Å (Ångstrom)
nm	nm (nanometer)
vel	$\mathrm{km\,s^{-1}}$

Table 6.6: x-axis units for plot type dem:

bin	Bin number
kev	keV (kilo-electronvolt), default value
ev	eV (electronvolt)
ryd	Rydberg units
k	K (Kelvin)
mk	MK (Mega-Kelvin)

Table 6.7: y-axis units for plot type model, spec:

	Photon numbers:
cou	$\mathrm{Photons\,m^{-2}\,s^{-1}\,bin^{-1}}$
kev	$\mathrm{Photons\,m^{-2}\,s^{-1}\,keV^{-1}}$ (default value)
ev	$\mathrm{Photons\,m^{-2}\,s^{-1}\,eV^{-1}}$
ryd	$\mathrm{Photons\,m^{-2}\,s^{-1}\,Ryd^{-1}}$
hz	$\mathrm{Photons\,m^{-2}\,s^{-1}\,Hz^{-1}}$
ang	$\mathrm{Photons\,m^{-2}\,s^{-1}\,Å^{-1}}$
nm	$\mathrm{Photons\,m^{-2}\,s^{-1}\,nm^{-1}}$
	Energy units:
wkev	$\mathrm{W\,m^{-2}\,keV^{-1}}$
wev	$\mathrm{W\,m^{-2}\,eV^{-1}}$
wryd	$\mathrm{W\,m^{-2}\,Ryd^{-1}}$
whz	$\mathrm{W\,m^{-2}\,Hz^{-1}}$
wang	$\mathrm{W\,m^{-2}\,Å^{-1}}$
wnm	$\mathrm{W\,m^{-2}\,nm^{-1}}$
jans	Jansky (10^{-26} $\mathrm{W\,m^{-2}\,Hz^{-1}}$
	νF_ν units:
iw	$\mathrm{W\,m^{-2}}$
ij	$\mathrm{Jansky\,Hz}$ (10^{-26} $\mathrm{W\,m^{-2}}$)
	Various:
bin	Bin number

Table 6.8: y-axis units for plot type data:

bin	Bin number
cou	Counts
cs	Counts s^{-1}
kev	Counts s^{-1} keV^{-1} (default value)
ev	Counts s^{-1} eV^{-1}
ryd	Counts s^{-1} Ryd^{-1}
hz	Counts s^{-1} Hz^{-1}
ang	Counts s^{-1} $Å^{-1}$
nm	Counts s^{-1} nm^{-1}
fkev	Counts m^{-2} s^{-1} keV^{-1}
fev	Counts m^{-2} s^{-1} eV^{-1}
fryd	Counts m^{-2} s^{-1} Ryd^{-1}
fhz	Counts m^{-2} s^{-1} Hz^{-1}
fang	Counts m^{-2} s^{-1} $Å^{-1}$
fnm	Counts m^{-2} s^{-1} nm^{-1}

Table 6.9: y-axis units for plot type chi:

bin	Bin number
cou	Counts
cs	Counts s^{-1}
kev	Counts s^{-1} keV^{-1} (default value)
ev	Counts s^{-1} eV^{-1}
ryd	Counts s^{-1} Ryd^{-1}
hz	Counts s^{-1} Hz^{-1}
ang	Counts s^{-1} $Å^{-1}$
nm	Counts s^{-1} nm^{-1}
fkev	Counts m^{-2} s^{-1} keV^{-1}
fev	Counts m^{-2} s^{-1} eV^{-1}
fryd	Counts m^{-2} s^{-1} Ryd^{-1}
fhz	Counts m^{-2} s^{-1} Hz^{-1}
fang	Counts m^{-2} s^{-1} $Å^{-1}$
fnm	Counts m^{-2} s^{-1} nm^{-1}
dchi	(Observed - Model) / Error (default)
rel	(Observed - Model) / Model

Table 6.10: y-axis units for plot type area:

bin	Bin number
m2	m^2 (default)
cm2	cm^2

Table 6.11: y-axis units for plot type fwhm:

bin	Bin number
kev	keV (kilo-electronvolt), default value
ev	eV (electronvolt)
ryd	Rydberg units
hz	Hertz
ang	Å (Ångstrom)
nm	nm (nanometer)
de/e	$\Delta E/E$

Table 6.12: y-axis units for plot type dem: the emission measure Y is defined as $Y \equiv n_{\mathrm{e}} n_{\mathrm{H}} V$ and is expressed in units of $10^{64}\,\mathrm{m}^{-3}$. Here n_{e} is the electron density, n_{e} is the Hydrogen density and V the emitting volume.

bin	Bin number
em	Y (default)
demk	$\mathrm{d}Y/\mathrm{d}T$, per keV
demd	$\mathrm{d}Y/\mathrm{d}T$, per K

Chapter 7

Auxilary programs

7.1 Trafo

Program *trafo* allows you to transform spectra and response matrices from standard OGIP format (or from SPEX version 1 format) to the format used by SPEX. It also gives you some options to combine data sets for simultaneous spectral fitting, and to optimses the spectral data set for speed, We discuss here first the different file formats, and then explain the use of *trafo*.

Some theoretical background and technical details on the SPEX file format are given in Chapter 8 and we refer the interested reader to that chapter for full details. Here we summarise some of the basics.

The basic philosophy behind the SPEX file format is:

keep things as simple as possible unless there is absolutely no alternative.

Therefore, SPEX uses one file containing all the information on the spectrum, including subtracted background, that must have the extension *.spo* and has FITS format with well-defined extensions and columns. Also, there is only a single file containing all the information on the response matrix, including effective area information, that must have the extension *.res* and also has FITS format with well-defined extensions and columns.

Trafo always produces the .spo and the .res files at the same time, as both are linked together tightly.

Examples of the usage of *trafo* can be found in the SPEX Cookbook, which is available on the SPEX website: http://www.sron.nl/spex

7.1.1 File formats spectra

There are a number of differences between OGIP and SPEX spectral file formats. The OGIP format has been documented at http://heasarc.gsfc.nasa.gov /docs /heasarc /ofwg /docs /spectra /ogip_92_007 /ogip_92_007.html.

In Table 7.1 we show the difference in structure between the OGIP spectral file format and the format used by SPEX. In addition to that, we make the following remarks.

For the energy grids, only energy (keV) units are allowed (not wavelength units), as there is a one-to-one relation between both. This number should be in double precision: for high-resolution spectra the difference between upper- and lower energy can be small, and close enough to the machine precision for single precision to cause some annoying numerical problems.

In the error column, it is essential to give real Poissonian errors (in case the spectrum is based on counts), and the use of e.g. Gehrels statistics must be avoided; in those cases, it is better to calculate the formal errors just from the square root of the number of raw counts. Chandra grating spectra are sometimes delivered with Gehrels errors, but this gives problems when the data need to be re-binned, and this is

Table 7.1: Differences between OGIP and SPEX in spectral file structure

Property	OGIP	SPEX
Spectrum:		
Files	Separate files for source & optionally background and correction file	One file for all
Channel	2-byte or 4-byte column in spectral file	Not used (should be by definition the row number of the table, starting to count at 1 without gaps)
Bin boundaries	Real, but not in spectral file but in *ebounds* extension matrix	Double precision (for high-res spectra), and in spectral file
Data	Counts (2-byte or 4-byte integer) or count rate (real, in counts s^{-1})	Only count rate (real, in counts s^{-1})
Errors	Either explicitly given, or calculated using Poissonian statistics if *poiserr* keyword is given	Always explicitly given
Exposure (s)	One value for full spectrum	Can differ for each bin
Background	Unscaled value in separate file; needs *backscal* keyword or column in source & background file to determine scaling for source region	Background subtracted value in one column; subtracted background (and its statistical error) in other columns
Quality flag	2-byte integer with 4 different values (0=good, 1=bad by s/w, 2=dubious by s/w, 5=set bad by user	"no-nonsense" logical, bin is either to be used (true) or not (false)
Grouping	2-byte integer, +1 for start channel, −1 for continuation bin, 0 if no grouping defined	Two logicals: true/false is bin is first/last of a group (i.e., for no grouping, both are true)
Area scaling	*Areascal* keyword or column	Not allowed (can be handled with exposure time or response matrix)
Background scaling	*Backscal* keyword or column	Worked out through explicit background subtraction in the spectrum
Systematic error	Keyword or column *sys_err* in both source & background files	Two columns: one for fraction of source, one for fraction of background

usually the case as the Chandra-data are delivered over-sampled. Also, never use 90 % errors, but always r.m.s. ("1σ") errors.

For the same reasons, the use of systematic errors should be avoided in the spectral file, because after rebinning they would become smaller. It is better to use no systematic errors in the spectral file, but in case you really need them, within SPEX you can set them after you have done the proper binning.

Also the use of quality flags should be avoided. It is better to provide the user either with only "good" data, or to make the user aware of any systematic limitations of the dats. When needed (either for instrumental or astrophysical reasons), within SPEX it is possible to ignore data ranges.

The usefulness of the *areascal* keywords is not very clear; the XMM-Newton RGS software uses the *arescal* for some dead-time corrections, but SPEX solves this in a more elegant way by allowing the exposure time to be different for each data bin. Whenever *trafo* encounters the arescal, it uses it to adjust the exposure time per bin. If you give the "show data" command in SPEX, you see for each data set some statistics, including mean, minimum and maximum exposure time per bin.

Finally, OGIP uses three columns (the background count rate, and the backscal for the source and background region) to determine the background that needs to be subtracted. In SPEX this reduces to two columns containing essentially the same information, namely the scaled background count rate and its statistical error. Actually, what is important in this is only the ratio of the backscal columns, not their individual values.

In summary, whenever possible we recommend to use only the first seven columns (bin boundaries, exposure time, source and background count rates with their errors), and leave the other columns empty / default (first/last channel flags, used, systematic errors).

7.1.2 File formats responses

The OGIP standard is described in https://heasarc.gsfc.nasa.gov /docs /heasarc /caldb /docs /memos /cal_gen_92_002 /cal_gen_92_002.html.

In Table 7.2 we show the difference in structure between the OGIP response file format and the format used by SPEX.

7.1.3 Multiple spectra

With *trafo*, you can combine different datasets into one combined spectrum and response file. There can be various reasons to do so:

1. You may want to combine different, similar instruments from the same observation (e.g. RGS1 and RGS2) or different spectral orders (e.g. RGS1 -1 and RGS1 -2 spectral order), or even combine pn with RGS. However, in these cases you may prefer to have the spectra in separate files, as that allows you easier to use only part of the data.
2. You may have time-resolved spectra of the same source taken with the same instrument
3. You may have multiple spatial regions of the same source, observed with the same instruments.

For more info, we refer to Sect. 2.2. *Trafo* allows you to achieve this.

7.1.4 How to use *trafo*

Trafo is an interactive program. It asks a few questions, which are usually self-explanatory. However, here we give a brief overview.

1. The first question *trafo* asks if you want to transform data from OGIP format (option 1), the now abandoned SPEX version 1 binary format (option 2) or the new SPEX format (option 3).

Table 7.2: Differences between OGIP and SPEX in response matrix structure

Property	OGIP	SPEX
Response:		
Files	Separate files for response (.rmf) & ancillary response file (.arf)	One file for all
Rmf extension:		
Components	1 component only	Matrix may have multiple components
Energy grid	One grid for the matrix, single precision	Each component can have its own grid, double precision
Response groups (≡ contiguous row of non-zero matrix elements for the same energy)	Multiple groups per row; specify number of groups (2-byte), first channel & number of channels (2-byte or 4-byte) and the matrix elements for each group	In extension "spex_resp_group" specify bin lower and upper energy, first channel, last channel and number of channels for the group; in extension "spex_resp_resp give the matrix elements
Optimalisation	No	Matrix may also contain derivatives of the matrix with respect to photon energy
Ebounds extension:		
Channel energies	Single precision	Not here, but in spectral file and double precision
Arf extension:		
Columns	Contains lower, upper energy and area (in cm^2)	N/A (included in matrix; but note units are SI, i.e. m^2)

2. The next question is how many spectra you want to transform. Usually you will enter here the number 1, but if you want to create concantinated spectra (see previous section) you should enter here the number of spectra. In that case, the next questions will be repeated for each spectrum.

3. Now you are asked to enter the maximum number of response groups per energy per spectrum. This is needed in order to allocate scratch memory; for almost any "sane" spectrum that we have seen sofar, a large number like 10000 is always sufficent here. Anyway, *trafo* will warn you and quit whenever you will reach this limit.

4. Optional: for multiple spectra, you are asked how many sectors you want to create. See the description in Sect. 2.2 for more info on sectors and regions.

5. Optional: if you have more than 1 spectrum, *trafo* asks you to enter the sector and region number that you want to assign to the spectrum that you will enter next. If The sector or region number are out of the allowed range as specified before, *trafo* will keep repeating this question until you have entered valid numbers.

6. Next a question is asked about partitioning of the matrix. You have here basically three options. Option 1: keep the structure of the matrix essentially as provided by the software package that created the OGIP files. The SPEX matrix will have 1 component, with no re-arrangements. Option 2: rearrange the matrix into contiguous groups. Your matrix may have been splitted into multiple regions, and for one photon energy you might have multiple regions (for instance, higher spectral orders for grating spectrometers without energy-sensitive detectors like the LETG/HRC-S or EUVE spectrometers); or a main diagonal and a fluorescence component, etc. *Trafo* will attempt to sort your response matrix accrding to these physically distinct components, by checking if for a given energy a response group has overlap in energy with an already existing group. Option 3: split into N roughly equal-sized components. This option is recommended for large matrices of high-resolution instruments such as RGS. It allows for the optimal usage of multiple processors during spectral fitting, provided your machine has of course more than one processor.

   Note that if you combine spectra that are already in SPEX version 2.0 or higher format, you do not have this option because we assume you have already optimised your matrix when creating the original version 2.0 files.

7. Optional: if you have selected option 3 above, *trafo* ask you the number of components. This can be any number, but experience has shown that a power of 2, typically between 8 and 32 works best (even on dual core processors).

8. Now you must enter the file name of the OGIP-type spectrum. Alternatively, if you combine SPEX version 2.0 files, you are prompted for the .spo file and you do not need to answer some of the questions below.

9. If the OGIP spectral file does not provide you the name of a background spectrum, you are prompted if you want to subtract a background spectrum. Be aware that sometimes background subtraction has already been taken into account in OGIP-spectra. Check carefully. If you answer "yes" to this question, then *trafo* will ask you for the filename of the background file.

10. In a few rare cases (e.g. Einstein SSS data), there is a second background file needed, the so-called "correction file". If such a file is to be used, *tarfo* will read it but it *must* be indicated then as the corfil extension in the spectral file.

    for each file

11. *Trafo* now makes a few sanity checks. If there is some problem, *trafo* will report it and stop. It checks for the same number of data channels in source and background (or correction) file. Further, and this is important to know, data bins that are being qualified as "bad" in the background or correction files, but "good" in the source file, will end up as "bad" bins in the final, background subtracted spectrum.

12. *Trafo* reports the total number of bad channels. You now can decide if you want to ignore the bad channels. Default is no (keep data; why would you otherwise have taken the burden to keep them in your datasets), but if you answer "yes", the bad channels will be ignored by *trafo* (well, flaged as not to be used in the .spo file).

13. Optional: if the OGIP-spectrum contains grouping information, *trafo* asks if that grouping should be used or ignored in the final spectrum.

14. Optional: if there is no response matrix file specified in the spectral pha-file, *trafo* asks for the name of the matrix.

15. Optional: while reading the response matrix, *trafo* makes some sanity checks. For instance, if the lower bin boundary of an energy bin is not smaller than the upper bin boundary, the user can correct this manually (some matrices are provided erroneously with zero width bins). But be sure that you understand here whjat you are doing!

16. Optional: also, some instruments assign an energy 0 to the lower energy boundary of the first bin. SPEX does not like this (as it gives trouble if you convert to a wavelength scale, for instance), so you can change the lower boundary manually to a small, non-zero number here.

17. Optional: in a few rare cases, matrices/data are poorly designed, such that the spectrum starts with a channel 0, but the matrix starts with channel 1. It is then not always clear which spectral element corresponds to which response element. *Trafo* tests for occurrences where the "flchan" keyword in the matrix equals 1, but the first channel in the data is 0. In this case it is possible to shift the response array by 1 channel, although this should be done as a last resort, and needs careefull checking if no mistakes are made! *Trafo* also tests for occurrences where the "flchan" keyword in the matrix does not equal 1 (usually 0), but the first channel in the data is 0. In this case it is advisable and possible to shift the response array by 1 channel, but again care should be taken!

18. Optional: if there is no ancillary (effective area) file specified in the spectrum, *trafo* reports this and asks if the user wants to use such an arf-file nevertheless. Default is "no", but if "yes" is entered, the name of the arf-file is prompted for.

19. As a next step, model energy bins with zero effective area are deleted from the file. Such bins usually occur at the low and high-energy side of the matrix. Deleting them saves computing time. Further, any necessary rebinning (if indicated by the grouping falgs) is done.

20. *Trafo* will tell you how many components it has found or created. It now asks you if you want to apply a shift to your spectra. Usually you should enter here 0. Useful cases to enter a non-zero shift s are situations where for instance your energy or wavelength scale is not yet sufficiently well calibrated. *trafo* then in that case puts the data of bin nr $k + s$ into bin s.

21. *Trafo* reports if the spectrum and matrix will be swapped (in case the original OGIP data were in wavelength order). Remember that SPEX *always* uses energy order.

22. The pre-last question is the filename for the spectrum that has been created (without the .spo extension, that *trafo* will add automatically).

23. Finally the filename for the response matrix that has been created is asked (without the .res extension, that *trafo* will add automatically).

7.2 Xabsinput

For the *xabs* and *warm* models, an input file is needed that contains the temperature and ionic column densities as a function of the ionisation parameter ξ. The standard input file provided by SPEX is based on a run with an older version of Cloudy, using the spectral energy distribution of NGC 5548 as used in Steenbrugge et al. (2005a).

The present program *xabsinput* allows the user to create such a file. The user basically needs to provide the following information, discussed in more detail below: spectral energy distribution (SED), abundances, root directory, and how to call Cloudy.

The SED should be provided in an ascii-file with two columns. The first column gives the energy in Rydberg units (note these units!), and the second column gives the spectral energy distribution in terms of $E\,\mathrm{d}N/\mathrm{d}E$ (for instance, in units of $\mathrm{keV\,m^{-2}\,s^{-1}\,keV^{-1}}$ or $\mathrm{Ryd\,m^{-2}\,s^{-1}\,Ryd^{-1}}$, or Jansky, etc. Note that the absolute normalisation of the SED does not matter here, only the shape is used, so therefore all these units can be used as long as it corresponds to energy per area per time per unit energy.

WARNING: *In order to have a sufficiently broad energy band, xabsinput adds a value of* 10^{-10} *at energies of* 10^{-8} *and* 10^{9} *Ryd. Take care that the SED scaling would essentially imply* 10^{-10} *to be a very small value. Also, the energies in the file should be in increasing order, and within the* 10^{-8} *and* 10^{9} *Ryd interval.*

Default abundances are presently the Lodders et al. (2009) proto-Solar values, presently the default of SPEX.

WARNING: *These default abundances are not equal to the default abundances of Cloudy*

The user is prompted for a file that contains the abundances relative to these standard abundances. This ascii files should have one line for each element with non-standard abundances. Each such line has two numbers: the atomic number of the element and the abundance relative to standard (in linear units, i.e. if you have iron two times solar, enter "26 2.0", without the quotes, of course). If all abundances are standard, simply use an empty but existing file.

The user also should provide the name of a directory where the files that are calculated will be stored. Basically, the following files will appear in this directory:

- a file "run.in", which contains the input file for Cloudy; at the end of the program, it contains the input of the last Cloudy run (with the highest ξ).
- a file "run.out", which contains the output of Cloudy; at the end of the program, it contains the output of the last Cloudy run (with the highest ξ).
- a file "xabs_inputfile", which is the input file that you will need for the *xabs* and *warm* models of SPEX; you can rename or move that file after finishing this program. The format of that file is as follows: first part two columns ($\log\xi$ (in cgs, bah) and temperature (in eV); this is followed by blocks for each ion from H I to Zn XXXI, with as first entry the nuclear charge and ionisation stage, and then simply the logs of the column densities, scaled to 28 for hydrogen.
- ascii files named "column01.dat", "column02.dat", ... "column30.dat" containing for each element with nuclear charge $Z = 1 - 30$ the ion concentrations as a function of ξ. SPEX does not need those files, but it can be educative to look at them (or to plot using qdp). The format of those files is $\log\xi$, followed by the logs of the column densities for each ion (1 is neutral, 2 is singly ionised etc). The column densities take account of the abundances and have, for practical reasons, been scaled in such a way that Hydrogen has log column 10; values below 0 are cut-off to 0.

Finally, the user should provide the name of a script that runs Cloudy. At SRON, this is /opt/local/bin/cloudy, and the script is given as:

```
#!/bin/csh -f
#
setenv CLOUDY_DATA_PATH /opt/local/HEA/cloudy/c13.01/data
setenv LD_LIBRARY_PATH
/opt/local/HEA/cloudy/c13.01/source/cloudy.exe < ${1}
```

WARNING: *We presently run Cloudy version 13. When newer versions of Cloudy become available, the Cloudy input or output formats may change, and the present program would need updates.*

Finally, our runs with Cloudy are done using the following default settings (apart from the SED and abundances that are entered by the user):

- Hydrogen density small, i.e. 10^{14} m^{-3} (10^{8} cm^{-3})
- Column density small, i.e. 10^{20} m^{-2} (10^{16} cm^{-2})
- Use the "iterate to convergence" command of Cloudy
- Values of $\log \xi$ between -8.5 and $+6.5$ with steps of 0.1

Note that, depending on the computer used, thi program may run for several hours; during execution it displays the present value of $\log \xi$ and the electron temperature in eV for each step of $\log \xi$ between -8.5 and $+6.5$. If you want to change the $\log \xi$ range, provide the flag `-r` at the command line when xabsinput is started.

WARNING: *We note that up to and including version 13.03 of Cloudy, $L_{\rm ion}$ in the definition of ξ was actually the total bolometric ionising luminosity. However, from the upcoming version 13.04 of Cloudy this is corrected in the Cloudy code to be consistent with the commonly used definition, where $L_{\rm ion}$ ranges between 1 to 1000 Ryd. Thus, we recommend using the xabsinput program with Cloudy version 13.04.*

7.3 RGS_fluxcombine

Program *RGS_fluxcombine* combines fluxed spectra, created by the XMM-Newton SAS task *rgsfluxer*. It has two options:

1. Option 1: combine RGS spectra of the same detector and spectral order
2. Option 2: combine spectra of RGS1, RGS2 and first and second order of both (4 spectra) into one spectrum.

This is not a trivial task, due to the presence of bad pixels, CCD gaps etc, that for each individual spectrum can be slightly different, for instance because of the use of the multi-pointing mode (MPM) of the RGS, so that the same bad pixel falls at a different wavelength for each individual spectrum, or due to the transient nature of some bad pixels/columns.

We discuss both options separately. In general, if you have RGS data for multiple epochs of a source, you may have to run this program first four times with option 1 (RGS1 and 2, orders 1 and 2), and then run it ones with option 2 to get a single stacked RGS spectrum.

WARNING: *You must run the SAS task rgsfluxer with 3600 wavelength bins between 4–40 Å. Thus, do not use the former β-bins!. Also, do not change these wavelength limits or number of bins.*

The program needs an input file, basically telling it how many spectra should be combined, followed by one line for each spectrum (fixed format, in fortran (F9.2,1X,A) for option 1), containing the exposure time in s (unfortunately not contained in the output of *rgsfluxer*, but you can get it from the spectral files) and the name of the relevant output file of *rgsfluxer*. For option 2 there are two additional numbers, namely the RGS (1 or 2 for RGS1 and RGS2), and the spectral order (1 or 2 for first and second spectral order), in such a way that the format is (I1,1X,I1,1X,F11.2,1X,A).

7.3.1 Option 1: combining RGS spectra of the same detector and spectral order

It is important to realise that when data of a time-variable source are combined, artifacts due to missing data (bad pixels) will arise. This can be simply illustrated by an example. Suppose that source X is observed twice with RGS, with exactly the same exposure time, the first time with 100 counts/bin, the second time with 20 counts/bin. Suppose that at wavelength bin j the first observation has a bad pixel, hence no counts, while the second observation has no bad pixel at bin j. The averaged spectrum will have (100+20)/2=60 counts per bin, except for bin j where the average value is (0+20)/2=10. Now since both exposure times are the same, the model will predict 60 counts for all bins (as required), except for bin j where it will predict 30 counts (the pixel is dead for half of the exposure). Thus, the model has

30 counts while the data have only 10 counts. An innocent observeer might therefore think there is an absorption line at bin j taking awat 20 counts, but it is obvious that that is not true.

For this reason, when we combine spectra, we will weigh with the exposure time for all "normal" bins, but when in some of the datasets there is a bad pixel j, we will use a different procedure. In that case, we look to the neighbouring pixels, and *assume that the spectral shape in this local neighbourhoud remains the same*. From the neighbours with complete data we can estimate how much flux the spectra with good data contribute to the total flux, and we use this fraction to estimate with what factor we should scale the flux measured at pixel j in these good data sets, to obtain the best estimate of the flux at pixel j for the total spectrum.

This procedure gives reliable results as long as the spectral shape does not change locally; as there is less exposure at pixel j, only the error bar on the flux will be larger.

WARNING: *W*hen there is reason to suspect that the bad pixel is at the location of a spectral line that changes in equivalent width, this procedure cannot be applied!

To alleviate this problem, the user has to enter a minimum exposure fraction. If this value is zero, for spectra with constant shape the best result is obtained. On the other hand, if this value is 1, only bins will be included that have no bad pixel in any of the observations that are combined. Advantage in that case is that there is no bias in the stacked spectrum, but a disadvantage may be that a significant part of the spectrum may be lost near important diagnostic lines (in particular for the multi-pointing mode).

Due to the binning and randomisation procedures within the SAS task *rgsfluxer*, it is still possible that despite careful screening on bad pixels, a few bad pixels remain in the fluxed spectrum, often adjacent or close to discarded pixels. For this reason, the present program checks for any bins that are more than 3σ below their right or left neighbouring bin (taking the statistical errors in both into account). Typically, the algorithm finds a few additional bad pixels in an individual observation that is discarded for combination.

The program further allows to apply two other corrections (both are done in the same step). The first correction is an effective area correction. In very high signal to noise data, there are broadband flux differences between RGS1 and RGS2 and first and second order spectra, in some parts of the spectrum, at the few percent level. A simpel model has been produced to correct for this, in the sense that when RGS1 and RGS2 are combined, the true spectrum is assumed to be the average of both. The correction is based on the monitoring campaign on Mrk 509, taken in 2009, and is found to be consistent with a large data set of Mrk 421 spectra that are present in the archive.

Local continuum fits to spectral bands that are poor of lines yield in general χ^2 values that are too low. This is an artifact of the SAS procedures related to the rebinning of the data. Data have to be binned from the detector pixel grid to the fixed wavelength grid that we use in our analysis. The bin boundaries of both grids do not match. As a consequence of this process, the natural Poissonian fluctuations on the spectrum as a function of detector pixel are distributed over the wavelength bin coinciding most with the detector pixel and its neighbours. In addition, there is a small smoothing effect caused by pointing fluctuations of the satellite. Due to this tempering of the Poissonian fluctuations, chi^2 values will be lower for a "perfect" spectral fit.

We have quantified the effect by fitting a linear model $F_\lambda = a + b\lambda$ to fluxed spectra in relatively line-poor spectral regions, in 1 Å wide bins in the 7–10, 17–18 and 35–38 Å ranges. The median reduced χ^2 is 0.72, with a 67 % confidence range between 0.65 and 0.79.

The rebinning process conserves the number of counts, hence the nominal error bars (square root of the number of counts) are properly determined. The lower reduced χ^2 is caused by the smoothing effect on the data. For correct inferences about the spectrum, such a bias in χ^2 is not appropriate. As we cannot change the observed flux values, we opt to multiply the nominal errors on the fluxes by $\sqrt{(0.72)} = 0.85$, in order to get get acceptable fits for ideal data and models.

7.3.2 Option 2: combine spectra of RGS1, RGS2 and first and second order of both (4 spectra) into one spectrum

This procedure is similar to option 1, but with some differences.

No corrections for effective area or binning bias are allowed here, as this should be taken into account in the steps that use option 1. In case there is only one observation, the present program can also be run with only one spectrum for option 1, to get the efective area and binning corrections in.

Further, the spectra are not combined according to exposure time (that should be in general almost the same for RGS1 and RGS2), but according to signal to noise ratio at each bin. Reason is that ofteh the second order spectra have poorer statistics than the first order spectra, hence they should be weighted less in the combined spectra.

WARNING: *S*pectra created with rgsfluxcombine can only be fitted using χ^2 statistics. C-statistics will produce wrong values.

7.4 RGS_fmat

Program *RGS_fmat* produces a response matrix for a fluxed spectrum created by the XMM-Newton SAS task *rgsfluxer*.

The user should provide an FITS file that is the output of the *rgsfluxer* program. That program *must* be run with 3600 equidistant wavelength bins between 4 and 40 Å. The program takes this fluxed spectrum, and creates a response matrix with effective area 1 m^2 for each wavelength, and a redistribution function normalised to unity for each wavelength.

This redistribution function is slightly different for RGS1 and RGS2, and also different for the second order spectra. Hence, the user must provide *RGS_fmat* the information about which RGS and spectral order is used.

In addition, it is also possible to make a single response matrix for combined RGS1 and RGS2 data, first and second order. In this case, first the program *RGS_fluxcombine* must have been run with option=2. That program then creates the *rgsfluxer*-compatible file, with the addition of four extra columns, containing for each wavelength the relative weight of the different instrument/order to the total spectrum. These weights are used to weigh the redistribution functions in the combined spectra.

The redistribution function has been parametrised as the sum of three (first order) or 4 (second order) Gaussians, with wavelength-dependent width and relative normalisations. These parametrisations have been obtained by fitting the "true" redistribution function as it is in SAS in 2014 after having re-investigated the RGS line-spread-function by the RGS consortium.

The model energy grid used for the matrix has a uniform spacing of 1/9th of the instrumental FWHM, approxinmated here simply by $0.06/|m|$, where m is the spectral order. The matrix is fully SPEX-compatible, in the sense that it also uses the derivatives of the matrix with respect to energy to get the most accurate results. There is no way back from this to OGIP-typs spectra!

WARNING: *The redistribution function is limited to ±1 Å around the central wavelength. For most continuum and absorption line spectra this is sufficient. In a few rare cases of very bright emission line sources, the user is advised to compare the resulting fit with a direct fit using the original SAS response matrices and spectral files.*

The matrix is much smaller: the combined RGS matrix has a size of 8.2 Mbyte. This should be compared to the total size of the four matrices in OGIP format, which is over 200 Mbyte.

7.5 Hydro_driver

Some users run complex and computationally expensive hydrodynamical simulations, and want to be able to calculate the corresponding X-ray spectrum. Without having to implement these hydro-codes

into SPEX it is now possible to do this.

For that purpose we offer the fortran90 subroutine *hydro_driver*. The source code of that routine is available in the SPEX distribution. Basically, it does the following:

1. It reads the needed parameters as subroutine arguments.
2. It does a few sanity checks (ion concentrations should be non-negative, and the total H I+H II column density must be 1.
3. It then creates a file called "spexicon.dat" containing the ion concentrations.
4. It also creates a file called "spexgrid.egr" containing the energy grid.
5. It then creates a file called "spexdriver.com" that contains the commands to run SPEX in batch mode.
6. This file is executed, and SPEX creates a file "spexoutput.asc" containing the emitted spectrum.
7. SPEX terminates.
8. The subroutine reads the file "spexoutput.asc" and puts the spectrum in the appropriate output array of the subroutine.

The input arguments of the subroutine are as follows:

`hden` - The Hydrogen density in SPEX units of 10^{20} m^{-3}. Real number.
`t` - The electron temperature in keV. Real number.
`it` - The ion temperature in keV, used for line broadening. Real number.
`vmic` - The microturbulence velocity, in km s^{-1}, used for line broadening. Real number.
`volume` - The emitting volume, in SPEX units of 10^{24} m^3. Real number.
`con(31,30)` - Array with ion concentrations relative to hydrogen (number densities). The abundances of the metals should be taken into account in this; thus, for example, for cosmic abundances, oxygen has a number density of about 0.001 per hydrogen atom, hence the sum of all oxygen ion concentrations should then be 0.001. The array con(jz,iz) contains for each element (labeled with atomic number iz, H=1, He=2 etc.) the concentration; jz indicates the ionisation stage (1=I=neutral, 2=II=singly ionized, etc.; do not forget to include the bare nucleus as the last one. Array elements with $jz > iz + 1$ will be ignored. Note that as things are normalised to hydrogen, $con(1,1) + con(2,1) = 1$ is obligatory! Real array.
`neg` - Number of energy bins. Integer number.
`eg(0:neg)` - Energy bin boundaries in keV; the boundaries of the first bin are stored in eg(0) and eg(1); the second bin in eg(1) and eg(2); etc. Real array.

The output arguments of the subroutine are as follows:

`spec(neg)` - The emitted spectrum in units of 10^{44} photons/s/keV. Divide this number by $4\pi d^2$ with d the source distance to get the flux at Earth. Real array
.

7.6 Rgsvprof

In SPEX, the *lpro* component (see Section 4.24) can be used to fold the spectrum with a user defined broadening profile. This is particularly useful for the analysis of extended sources with grating spectrometers, like RGS aboard XMM-Newton. The rgsvprof program creates an input file (usually called vprof.dat) for the *lpro* component from a MOS1 detector image.

The program will ask for the following input:

- MOS1 detector image. In order to obtain a profile along the dispersion direction of RGS within the same field of view, the program asks for a MOS1 image of the source in detector coordinates

(DETX,DETY) created by, for example, the XMM-Newton SAS task *evselect*. Rgsvprof does not require a particular resolution for the image, but a resolution of about 1/3 of the XMM-Newton PSF, i.e. 4″, is recommended. It is recommended to extract an image using events with energies falling in the RGS band: ∼0.3-2.0 keV.

- Cross-dispersion selection region. If the RGS spectrum is extracted from a certain area in the cross-dispersion direction, then provide the lower and upper boundary here in arcminutes with respect to the centre of the field of view. The full RGS strip corresponds to a range between -2.5 and 2.5 arcminutes.
- Source width in the dispersion direction. This parameter determines the boundaries of the resulting profile in wavelength space. Rgsvprof asks for the width of the source in arcmin centred on the peak of the emission. Choose this width to be somewhat larger than the actual width of the source to be sure that the brightest parts are included in the profile. A width of >10 arcmin is typically enough, but a larger width will increase the size of the vprof.dat file and increase processing times.
- Output file name. The name of the output file (usually vprof.dat). Note that rgsvprof does not overwrite files that already exist.

Rgsvprof creates an ASCII file with two columns. The left column consists of the grid of wavelengths that the profile is based upon and the right column gives the normalised cumulative profile over this range starting with 0 and ending at 1. The profile will be centred on the peak of the emission.

Note: The cross-dispersion axis in RGS is parallel to the DETX coordinate axis in MOS1 and has the same direction. This is particularly helpful when one extracts spatial regions in the cross-dispersion direction of RGS.

WARNING: *This program works best for bright extended sources, where the background contribution is negligible. For sources with a low surface brightness, in principle a background subtracted image should be used. This program, however, cannot deal with negative values in the MOS1 image.*

WARNING: *This program does not take into account vignetting and chip gaps. For offset observations or very extended sources, this may start to play a role.*

7.6.1 Theory

The wavelength scale corresponding to the angular width of the source in RGS can be calculated using the dispersion equation defined in e.g. Peterson et al. (2004):

$$m\lambda = d(\cos\beta - \cos\alpha), \tag{7.1}$$

where m is the spectral order, λ the wavelength, d the grating spacing, α the incidence angle of a photon, and β the diffracted angle. The wavelength shift $\Delta\lambda$ due to an offset angle $\Delta\theta$ relative to the telescope boresight can be derived by differentiating the equation above with respect to $\theta = \frac{F}{L}(\alpha - \alpha_0)$, where F is the distance between the gratings (RGA) and the camera (RFC), L the focal length of the telescope, and α_0 the incidence angle corresponding to the telescope boresight. The resulting relation for $\Delta\lambda$ is:

$$\Delta\lambda = \frac{dL}{mF}\sin(\alpha_0)\Delta\theta. \tag{7.2}$$

We then use the following numbers from den Herder et al. (2001): d is 1/645.6 mm, L is 7.5 m, F is 6.7 m, and α_0 is 1.5762° to calculate the typical conversion factor from $\Delta\theta$ in arcmin:

$$\Delta\lambda = 0.1387 \text{ Å } \Delta\theta \tag{7.3}$$

Note that the value in Peterson et al. (2004) is not correct, because the factor $\frac{L}{F}$ was not applied. The value in the above equation is used in rgsvprof to convert the spatial profile from arcminute scales to wavelength scales.

7.7 Stepcontour

The stepcontour task produces 1D and contour plots of step searches produced using the `step` commands. At the end of the step command definitions (see 3.28), add the text `file` and a file name to the step command to write out the step results to a .stp file.

For example:

```
SPEX> step dimension 2
SPEX> step axis 1 parameter 1 1 norm range 0.5 1.5 n 10
SPEX> step axis 2 parameter 1 1 t range 0.75 1.25 n 10
SPEX> step file norm-kt
```

These commands will produce a norm-kt.stp file. Subsequently, the `stepcontour` command can be run from the command line or the SPEX¿ prompt:

```
linux:~> stepcontour
```

or

```
SPEX> sys exe "spexcontour"
```

The program will start and ask the user for the name of the .stp file:

```
 Program stepcontour version 1.0.
 Use this program to plot contours from SPEX steppar (.stp) files.

 Enter filename of SPEX steppar file (.stp): norm-kt.stp
```

Then provide a PGPLOT device to show the plot. For example '/xw' for X output or 'file.ps/ps' for Postscript plots:

```
 Number of axes in your steppar file:            2
 Graphics device/type (? to see list, default /NULL): /xw
 PGPLOT device type: XWINDOW
 Plotting contours for: 68.3%, 90%, 95.4%, 99%, 99.99%.
```

If you would like to write the plot data to QDP format, say yes to the following question:

```
 Would you like to write this plot to a QDP file (y/n)? y
```

The program asks for an output QDP filename and writes the data to that file.

WARNING: *The stepcontour program does not support logarithmic axes on 2D plots. Use another program to plot these or perform the step command on a linear grid.*

Chapter 8

Response matrices

8.1 Optimal definition of respons matrices

The SPEX FITS format allows us to bin spectra and response matrices on arbitrary grids, which enables us to bin spectra and responses in an optimal way. In Kaastra & Bleeker (2016) a theoretical framework is developed in order to estimate the optimum binning of X-ray spectra. Expressions are derived for the optimum bin size for the model spectra as well as for the observed data using different levels of sophistication. It is shown that by not only taking into account the number of photons in a given spectral model bin but also its average energy (not necessarily equal to the bin center) the number of model energy bins and the size of the response matrix can be reduced by a factor of 10–100. The response matrix should then not only contain the response at the bin center, but also its derivative with respect to the incoming photon energy. In the coming sections we describe practical guidelines how to construct the optimum energy grids as well as how to structure the response matrix. Finally a few examples are given to illustrate the present methods.

8.2 Proposed file formats

In Kaastra & Bleeker (2016) it was shown how the optimum data and model binning can be determined, and what the corresponding optimum way to create the instrumental response is. Now I focus upon the possible data formats for these files.

For large data sets, the fastest and most transparant way to save a response matrix to a file would be to use direct fortran or C write statements. Unfortunately not all computer systems use the same binary data representation. Therefore the FITS-format has become the *de facto* standard in many fields of astronomical data analysis. For that reason I propose here to save the response matrix in FITS-format.

A widely used response matrix format is NASA's OGIP format. This is used e.g. as the data format for XSPEC. There are a few reasons that we propose *not* to adhere to the OGIP format here, as listed below:

1. The OGIP response file format as it is currently defined does not account for the possibility of response derivatives. As was shown in the previous sections, these derivatives are needed for the optimum binning. Thus, the OGIP format would need to be updated.
2. In the later versions of OGIP new possibilities of defining the energy grid have been introduced that are prone to errors. In particular the possibility of letting grids start at another index than 1 may (and has led) often to errors. The sofware also becomes unncessarily complicated, having to account for different possible starting indices. Moreover, the splitting into arf and rmf files makes it necessary to check if the indexing in both files is consistent, and also if the indexing in the corresponding spectra is consistent.

3. There are some redundant quantities in the OGIP format, like the "areascal" keyword. When the effective area should be scaled by a given factor, this can be done explicitly in the matrix.
4. As was shown in this work, it is more efficient to do the grouping within the response matrix differently, splitting the matrix into components where each component may have its own energy grid. This is not possible within the present OGIP format.

8.2.1 Proposed response format

I propose to give all response files the extension ".res", in order not to confuse with the ".rmf" or ".arf" files used within the OGIP format.

The file consists of the mandatory primary array that will remain empty, plus a number of extensions. In principle, we could define an extension for each response component. However, this may result into practical problems. For example, the fitsio-package allows a maximum of 1000 extensions. The documentation of fitsio says that this number can be increased, but that the access time to later extensions in the file may become very long.

In principle we want to allow for data sets with an unlimited number of response components. For example, when a cluster spectrum is analysed in 4 quadrants and 10 radial annuli, one might want to extract the spectrum in 40 detector regions and model the spectrum in 40 sky sectors, resulting in principle in at least 1600 response components (this may be more if the response for each sky sector and detector region has more components).

Therefore I propose to use only three additional and mandatory extensions.

The first extension is a binary table with 4 columns and contains for each component the number of data channels, model energy bins, sky sector number and detector region number (see table 8.1).

The second extension is a binary table with 5 columns and contains for each model energy bin for each component the lower model energy bin boundary (keV), the upper model energy bin boundary (keV), the starting data channel, end data channel and total number of data channels for the response group (see table 8.2).

The third extension is a binary table with 2 columns and contains for each data channel, for each model energy bin for each component the value of the response at the bin center and its derivative with respect to energy (see table 8.3). SI units are mandatory (i.e. m^2 for the response, $m^2 keV^{-1}$ for the response derivative).

Any other information that is not needed for the spectral analysis may be put into additional file extensions, but will be ignored by the spectral analysis program.

Finally I note that the proposed response format is used as the standard format by version 2.0 of the SPEX spectral analysis package.

8.2.2 Proposed spectral file format

There exists also a standard OGIP FITS-type format for spectra. As for the OGIP response file format, this format has a few redundant parameters and not absolutely necessary options.

There is some information that is absolutely necessary to have. In the first place the *net source count rate* S_i (counts/s) and *its statistical uncertainty* ΔS_i (counts/s) are needed. These quantities are used e.g. in a classical χ^2-minimization procedure during spectral fitting.

In some cases the count rate may be rather low; for example, in a thermal plasma model at high energies the source flux is low, resulting in only a few counts per data channel. In such cases it is often desirable to use different statistics for the spectral fitting, for example maximum likelihood fitting. Such methods are often based upon the number of counts; in particular the difference between Poissonian and Gaussian statistics might be taken into account. In order to allow for these situations, also the *exposure time* t_i per data channel is needed. This exposure time needs not to be the same for all data channels; for example, a part of the detector might be switched off during a part of the observation, or the observer might want to use only a part of the data for some reason.

Table 8.1: First extension to the response file

keyword	description
EXTNAME (=RESP_INDEX)	Contains the basic indices for the components in the form of a binary table
NAXIS1 = 16	There are 16 bytes in one row
NAXIS2 =	This number corresponds to the total number of components (the number of rows in the table)
NSECTOR =	This 4-byte integer is the number of sky sectors used.
NREGION =	This 4-byte integer is the number of detector regions used.
NCOMP =	This 4-byte integer is the totalnumber of response components used (should be equal to NAXIS2).
TFIELDS = 4	The table has 4 columns; all columns are written as 4-byte integers (TFORM='1J')
TTYPE1 = 'NCHAN'	First column contains the number of data channels for each component. Not necessarily the same for all components, but it must agree with the number of data channels as present in the corresponding spectrum file.
TTYPE2 = 'NEG'	Second column contains the number of model energy grid bins for each component. Not necessarily the same for all components.
TTYPE3 = 'SECTOR'	Third column contains the sky sector number as defined by the user for this component. In case of simple spectra, this number should be 1.
TTYPE4 = 'REGION'	Fourth column contains the detector region number as defined by the user for this component. In case of simple spectra, this number should be 1.

Further, in several situations the spectrum will contain a contribution from background events. The observed spectrum can be corrected for this by subtracting the *background spectrum* B_i scaled at the source position. In cases where one needs to use Poissonian statistics, including the background level, this subtracted background B_i must be available to the spectral fitting program. Also for spectral simulations it is necessary to include the background in the statistics.

The background can be determined in different ways, depending upon the instrument. For the spectrum of a point-source obtained by an imaging detector one could extract the spectrum for example from a circular region, and the background from a surrounding annulus and scale the background to be subtracted using the area fractions. Alternatively, one could take an observation of an "empty" field, and using a similar extraction region as for ther source spectrum, one could use the empty field observation scaled by the exposure times to estimate the background.

The background level B_i may also contain noise, e.g. due to Poissonion noise in the background observation. In some cases (e.g. when the raw background count rate is low) it is sometimes desirable to smooth the background to be subtracted first; in this case the *nominal background uncertainty* ΔB_i no longer has a Poissonian distribution, but its value can nevertheless be determined. In spectral simulations one may account for the background uncertainty by e.g. simply assuming that the square of the background signal-to-noise ratio $(B_i/\Delta B_i)^2$ has a Poissonian distribution.

For spectral simulations, however, one should be able to know the *expected* background. This cannot

Table 8.2: Second extension to the response file

keyword	description
EXTNAME (=RESP_COMP)	Binary table with for each row relevant index information for a single energy of a component; stored sequentially, starting at the lowest component and within each component at the lowest energy.
NAXIS1 = 20	There are 20 bytes in one row
NAXIS2 =	This number must be the sum of the number of model energy bins added for all components (the number of rows in the table).
TFIELDS = 5	The table has 5 columns
TTYPE1 = 'EG1'	The lower energy (keV) as a 4-byte real of the relevant model energy bin
TTYPE2 = 'EG2'	The upper energy (keV) as a 4-byte real of the relevant model energy bin
TTYPE3 = 'IC1'	The lowest data channel number (as a 4-byte integer) for which the response at this model energy bin will be given. The response for all data channels below IC1 is zero. Note that IC1 should be at least 1 (i.e. start counting at channel 1!).
TTYPE4 = 'IC2'	The highest data channel number (as a 4-byte integer) for which the response at this model energy bin will be given. The response for all data channels above IC2 is zero.
TTYPE5 = 'NC'	The total number of non-zero response elements for this model energy bin (as a 4-byte integer). NC is redundant, and should equal IC2-IC1+1, but it is convenient to have directly available in order to allocate memory for the response group.

be derived from the observed (Poissonian) background if by chance a data channel has zero background counts. To alleviate that problem, we have introduced an additional column in the spectral file that represents the exposure ratio of the background region divided by that of the source region, where the exposure is the product of exposure time times extraction area.

Furthermore, there may be systematic calibration uncertainties, for example in the instrumental effective area or in the background subtraction. These systematic errors may be expressed as a fraction of the source count rate (such that the total *systematic source uncertainty* is $\epsilon_{si}S_i$) and/or as a fraction of the subtracted background (such that the total *systematic background uncertainty* is $\epsilon_{bi}B_i$). Again, these systematic errors may vary from data channel to data channel. They should also be treated different than the statistical errors in spectral simulations: they must be applied to the simulated spectra *after* that the statistical errors have been taken into account by drawing random realisations. Also, whenever spectral rebinning is done, the systematic errors should be averaged and applied to the rebinned spectrum: a 10 % systematic uncertainty over a given spectral range may not become 1 % by just rebinning by a factor of 100, but remains 10 %, while a statistical error of 10 % becomes 1 % after rebinning by a factor of 100.

In the previous sections we have shown how to choose the optimal data binning. The observer may want to rebin further in order to increase the significance of some low-statistics regions in the spectrum, or

Table 8.3: Third extension to the response file

keyword	description
EXTNAME (=RESP_RESP)	Contains the response matrix elements and their derivatives with respect to model energy. The data are stored sequentially, starting at the lowest component, within each component at the lowest energy bin, and within each energy bin at the lowest channel number.
NAXIS1 = 8	There are 8 bytes in one row
NAXIS2 =	This number must be the sum of the NC values, added for all model energy bins and all components (the number of rows in the table).
TFIELDS = 2	The table has 2 columns
TTYPE1 = 'Response'	The response values $R_{ij,0}$, stored as a 4-byte real **in SI units** i.e. in units of m^2.
TTYPE1 = 'Response_Der'	The response derivative values $R_{ij,1}$, stored as a 4-byte real **in SI units** i.e. in units of $m^2 keV^{-1}$.

may want to inspect the unbinned spectrum. Also, during spectral analysis or beforehand the observer may wish to discard certain parts of the spectrum, e.g. data with bad quality or parts of the spectrum that he is not interested in. For these reasons it is also usefull to have the proposed rebinning scheme available in the spectral file.

I propose to add to each data channel three logical flags (either true or false): first, if the data channel is the *first channel* of a group (f_i); next, if the data channel is the *last channel* of a group (l_i); and finally, if the data channel is *to be used* (u_i). A channel may be both first and last of a rebinning group (f_i and l_i both true) in the case of no rebinning. The first data channel $i = 1$ always *must* have f_i true, and the last data channel l_i true. Whenever there are data channels that are not used (u_i false), the programmer should check that the first data channel after this bin that is used gets f_i true and the last data channel before this bin taht is used gets l_i true. The spectral analysis package needs also to check for these conditions upon reading a data set, and to return an error condition whenever this is violated.

Finally, I propose to add the nominal energies of the data bin boundaries c_{i1} and c_{i2} to the data file. This is very usefull if for example the observed spectrum is plotted outside the spectral fitting program. In the OGIP format, this information is contained in the response matrix. I know that sometimes objections are made against the use of these data bin boundaries, expressed as energies. Of course, formally speaking the observed data bin boundaries often do not have energy units; it may be for example a detector voltage, or for grating spectra a detector position. However, given a proper response. However given a corresponding response matrix there is a one-to-one mapping of photon energy to data channel with maximum response, and it is this mapping that needs to be given here. In the case of only a single data channel (e.g. the DS detector of EUVE) one might simply put here the energies corresponding to the FWHM of the response. Another reason to put the data bin boundaries in the spectral file and not in the response file is that the response file might contain several components, all of which relate to the same data bins. And finally, it is impossible to analyse a spectrum without knowing simultaneously the response. Therefore, the spectral analysis program should read the spectrum and response together.

As a last step we must deal with multiple spectra, i.e. spectra of different detector regions that are related through the response matrix. In this case the maximum number of FITS-file extensions of 1000 is a much smaller problem then for the response matrix. It is hard to imagine that anybody might wish to fit more than 1000 spectra simultaneously; but maybe future will prove me to be wrong. An example could be the following. The supernova remnant Cas A has a radius of about $3'$. With a spatial resolution

of 10″, this would offer the possibility of analysing XMM-EPIC spectra of 1018 detector regions. At the scale of 10″ the responses of neighbouring regions overlap so fitting all spectra simultaneously could be an option···.

Therefore it is wise to stay here on the conservative side, and to write the all spectra of different detector regions into one extension. As a result we propose the following spectral file format.

After the null primary array the first extension contains the number of regions for the spectra, as well as a binary table with the number of data channels per region (see table 8.4). This helps to allocate memory for the spectra, that are stored as one continuous block in the second extension (see table 8.5).

Table 8.4: First extension to the spectrum file

keyword	description
EXTNAME (=SPEC_REGIONS)	Contains the spectral regions in the form of a binary table
NAXIS1 = 4	There are 4 bytes in one row
NAXIS2 =	This number corresponds to the total number of regions (spectra) contained in the file (the number of rows in the table)
TFIELDS = 1	The table has 1 column, written as 4-byte integer (TFORM='1J').
TTYPE1 = 'NCHAN'	Number of data channels for this spectrum.

Table 8.5: Second extension to the spectrum file

keyword	description
EXTNAME (=SPEC_SPECTRUM)	Contains the basic spectral data in the form of a binary table
NAXIS1 = 28	There are 28 bytes in one row
NAXIS2 =	This number corresponds to the total number of data channels as added over all regions (the number of rows in the table)
TFIELDS = 12	The table has 12 columns.
TTYPE1 = 'Lower_Energy'	Nominal lower energy of the data channel c_{i1} in keV; 4-byte real.
TTYPE2 = 'Upper_Energy'	Nominal upper energy of the data channel c_{i2} in keV; 4-byte real.
TTYPE3 = 'Exposure_Time'	Net exposure time t_i in s; 4-byte real.
TTYPE4 = 'Source_Rate'	Background-subtracted source count rate S_i of the data channel in counts/s; 4-byte real.
TTYPE5 = 'Err_Source_Rate'	Background-subtracted source count rate error ΔS_i of the data channel in counts/s (i.e. the statistical error on 'Source_Rate'); 4-byte real.
TTYPE6 = 'Back_Rate'	The background count rate B_i that was subtracted from the raw source count rate in order to get the net source count rate S_i; in counts/s; 4-byte real.
TTYPE7 = 'Err_Back_Rate'	The statistical uncertainty ΔB_i of 'Back_Rate' in counts/s; 4-byte real.
TTYPE8 = 'Exp_Rate'	ratio of the exposures (time × area) for the background region divided by that of the source region; dimensionless; 4-byte real.
TTYPE9 = 'Sys_Source'	Systematic uncertainty ϵ_{si} as a fraction of the net source count rate S_i; dimensionless; 4-byte real.
TTYPE10 = 'Sys_Back'	Systematic uncertainty ϵ_{bi} as a fraction of the subtracted background count rate B_i; dimensionless; 4-byte real.
TTYPE11 = 'First'	True if it is the first channel of a group; otherwise false. 4-byte logical.
TTYPE12 = 'Last'	True if it is the last channel of a group; otherwise false. 4-byte logical.
TTYPE13 = 'Used'	True if the channel is to be used; otherwise false. 4-byte logical.

Chapter 9

Installation

9.1 Binary installation

We support SPEX on two platforms: Linux 64-bit and Mac OSX 64-bit. Please consult our website for the most recent list of (`http://www.sron.nl/spex`). For instructions, please follow the steps below for your operating system.

9.1.1 Binary installation on Linux

These are the steps to install SPEX on Linux 64-bit (Intel/AMD) systems:

1. Download the tar file from our web site named `SPEX-<version>-Linux.tar.gz` where ¡version¿ is the current version number (for example 3.00.00). Copy this file into the directory where you want to install SPEX, like `/home/user/software/`.

2. Extract the tar file in the desired directory:

   ```
   linux:~/software> tar xvfz SPEX-3.05.00-Linux.tar.gz
   ```

   Tar will create the directory `~/software/SPEX-3.05.00-Linux`.

3. For SPEX to function, the environment needs to be set. The `SPEX90` variable should point to the top SPEX directory (`/path/to/your/SPEX-3.05.00`). Then source the included environment file called `spexdist.sh` or `spexdist.csh`:

   For BASH shells:

   ```
   linux:~/software> export SPEX90=/path/to/your/SPEX-3.05.00-Linux
   linux:~/software> source $SPEX90/spexdist.sh
   ```

   For CSH shells:

   ```
   linux:~/software> setenv SPEX90 /path/to/your/SPEX-3.05.00-Linux
   linux:~/software> source $SPEX90/spexdist.csh
   ```

4. To automate the setting of the environments, you can add the commands for your shell to either `~/.bashrc` (bash) or `~/.cshrc` (csh).

If you encounter problems or if you have questions, please visit the troubleshooting and frequently asked questions sections on our website first, before consulting us directly.

9.1.2 Binary installation on Mac OSX

We always compile SPEX on one of the most recent Mac OSX versions and try to make the executables compatible with earlier OSX versions. However, due to the dependency on X-windows (XQuartz), this compatibility is limited. Please consult our web site for the list of supported Mac OSX versions.

SPEX can be installed following these steps:

1. Download and install on your system first (if you have not done so already) from `http://www.xquartz.org/`
2. Download the SPEX-3.05.00-OSX.dmg file from our web site `http://www.sron.nl/spex`
3. Click the downloaded file and open the installation package. You can walk through the graphical installer. The default settings will be fine.
4. SPEX will be installed in `/opt/spex`, but the still needs to be set. The included spexdist.sh and spexdist.csh files can be found in the directory `/opt/spex`:

   For BASH shells:

   ```
   darwin:~> source /opt/spex/spexdist.sh
   ```

   For CSH shells:

   ```
   darwin:~> source /opt/spex/spexdist.csh
   ```

5. To automate the setting of the environments, you can add the command for your shell to either `~/.bashrc` (bash) or `~/.cshrc` (csh).

If you encounter problems or if you have questions, please visit the troubleshooting and frequently asked questions sections on our website first, before consulting us directly.

9.1.3 Running SPEX using Docker

SPEX is written mostly in Fortran 90 and depends on a few system libraries. This makes it difficult to provide a few binary versions that will continue to run on multiple platforms over many years. Therefore, we have also created a Docker image for SPEX that can be run on the Docker platform (https://www.docker.com), which is available for Linux, Mac OS and Windows.

Step 1: Download and install Docker on your computer

To run a Docker image, please install Docker on your computer. See the Docker website (https://www.docker.com) for details and look for the Docker Engine community edition. Once you have downloaded and installed Docker, you can continue with this tutorial.

Step 2: Download the SPEX docker image from Zenodo

The SPEX Docker image is available on this Zenodo page as a tar.gz file. Please download the file called spex-[version]-docker.tar.gz, for example:

```
spex-3.05.00-docker.tar.gz
```

Step 3: Import the SPEX image into Docker

Before you can run the docker file, it should be imported into the docker system. This can be done on the command line:

```
user@linux:~> docker load -i spex-3.05.00-docker.tar.gz
```

The image will be named spexxray/spex with the tag 3.05.00 and can be found with the command:

```
user@linux:~> docker images
REPOSITORY     TAG      IMAGE ID      CREATED        SIZE
spexxray/spex  3.05.00  0a0a0a0a0a0   1 minute ago   311MB
```

The docker images are also available on Docker hub. You can download and import the SPEX image easily with the following command:

```
docker pull spexxray/spex:latest
```

Step 4: Run SPEX on Docker

Using the `docker run` command, the image can be executed and SPEX can be run in a so-called container. To enjoy all the capabilities of SPEX, two things need to be arranged in the `docker run` command: access to local directories and a graphical X11 connection. To arrange this, additional flags need to be specified on the command line.

Mounting local directories to the container

If you would like to mount your own home directory into the SPEX container such that you can use some spo and res files there or save the output files, then you need the following flags:

```
-e LOCAL_USER_ID=‘id -u $USER‘
-v /home/myusername:/home/user
```

The first flag arranges that the user in the container will have the same user ID as you. This will allow you to read and write to your home directory from within the container.

The second flag arranges that your true local home directory called `/home/myusername` is mounted to `/home/user` inside the SPEX container.

MAC users: Please note that on OSX your home directory is in `/Users/myusername`.

Arranging the X11 connection

To make sure PGPLOT can connect to the X11 server on the host, we need to make a few connections from the container to the host machine. This is done with the following flags:

```
-e DISPLAY=$DISPLAY
-v /tmp/.X11-unix:/tmp/.X11-unix
```

The first flag sets the DISPLAY variable inside the container to the DISPLAY variable of the host machine. The second flag mounts the X11 temporary directory of the host to the same directory inside the container.

MAC users: To use X11 on Mac, you need to install XQuartz (https://www.xquartz.org/) (or a similar X11 server) and set it to 'Allow connections from network clients' in the XQuartz settings. In addition, the X server should be set to allow incoming connections from localhost on the command line:

```
user@macos:~> xhost +127.0.0.1
```

Then the DISPLAY variable on the `docker run` line should be set to `host.docker.internal:0`.

The complete docker run command

The full run commands for docker now look like below, where `-w` means that the container will start in working directory `/home/user`.

For Linux:

```
docker run -it \
           -e DISPLAY=$DISPLAY \
           -e LOCAL_USER_ID=‘id -u $USER‘ \
           -v /tmp/.X11-unix:/tmp/.X11-unix \
           -v /home/myusername:/home/user \
           -w /home/user \
           spexxray/spex:3.05.00
```

For Mac:

```
docker run -it \
           -e DISPLAY=host.docker.internal:0 \
           -e LOCAL_USER_ID=‘id -u $USER‘ \
           -v /tmp/.X11-unix:/tmp/.X11-unix \
           -v /Users/myusername:/home/user \
           -w /home/user \
           spexxray/spex:3.05.00
```

The `docker run` command above will provide you with a prompt that will allow you to run spex:

```
user@linux:~> docker run -it -e DISPLAY=$DISPLAY -e LOCAL_USER_ID=‘id -u $USER‘ \
-v /tmp/.X11-unix:/tmp/.X11-unix -v /home/myusername:/home/user -w /home/user \
spexxray/spex:3.05.00

Welcome to the SPEX Docker Container!
Just type ’spex’ to start the program.
user@0922f2e4ff85:~>
```

In this environment, you can just run `spex` or `trafo`:

```
user@0922f2e4ff85:~> spex
 Welcome user to SPEX version 3.05.00

 NEW in this version of SPEX:
11-06-2018 Added Ext_Rate column to new spo files
18-12-2018 SPEX is now using the GPL license
18-12-2018 Release of version 3.05.00

SPEX>
```

9.2 Source code install

The SPEX source code is available on Zenodo and is distributed under the GNU public license version 3 since version 3.05.00.

The SPEX source code can be compiled using the multi-platform Cmake build system. See http://www.cmake.org/ for more information and downloads, or check the package manager of your Linux distribution. The SPEX install needs CMake version 3.0 or higher.

Since SPEX is programmed mostly in Fortran 90, it is recommended to use a recent Fortran compiler. SPEX has been tested with GFortran (version 4 and above) and the Intel Fortran Compiler.

9.2.1 Library dependencies

SPEX depends on a few external libraries to function. For some of those, the library source code has been included in the SPEX source code package. By default, CMake will look for system libraries to link to. If they are not there, then the version in the source package will be used.

The following libraries and packages are required to compile SPEX:

- CMake
- X11
- Readline
- CFITSIO (*)
- BLAS (*)
- LAPACK (*)
- PGPLOT (*)

(*) The SPEX source tree also contains the library if necessary.

All these libraries are commonly available in Linux distributions, so please read the documentation of your distribution to find out how to install these libraries. Please note that some distributions require you to also install the 'development' package of a library to be able to use them during compilation. In the Debian repository, for example, the development package of readline is called 'libreadline-dev'.

Below, we list some library-specific comments that can be helpful in case of problems.

Readline

Note for Mac OSX users: The OSX readline library is NOT compatible with the GNU readline library. You need to compile your own readline library from source or find a GNU readline library elsewhere on your system to link to. Compilation may work in, for example, a MacPorts environment, although this has not been tested. The official Mac version of SPEX statically links to a compiled version of readline downloaded from:

```
https://tiswww.case.edu/php/chet/readline/rltop.html
```

CFITSIO

The SPEX source tree contains an old fortran version of fitsio, which is sufficient for the vast majority of applications. However, in exceptional cases, the old library cannot handle very large fits files. In those cases, it is better to link to a system CFITSIO library.

BLAS and LAPACK

Some of the SPEX models depend heavily on the BLAS and LAPACK linear algebra packages. The default routines are available in the SPEX source tree, but compiling those will not provide the best performance. The performance improves substantially if an optimized BLAS or LAPACK library is used. There are two tested options:

- Intel Math Kernel Library (MKL)
- OpenBLAS

When compiling with the Intel Fortran compiler, using MKL is quite obvious. To link the MKL library, add the following option to the cmake command:

```
cmake . -DMKL=YES
```

If MKL is not set, cmake will look for other options, like OpenBLAS, if they are installed on your machine. If nothing is found, the non-optimized code in the SPEX source tree is used.

9.2.2 MacOS instructions

The compilation of SPEX on MacOS is slightly more demanding. SPEX can run natively on MacOS (without ports), but then it needs a few pre-installed programs:

- Xcode (Through the App store)
- CMake (https://cmake.org/download/)
- XQuartz (https://www.xquartz.org/)
- GNU readline (https://tiswww.case.edu/php/chet/readline/rltop.html) Compile and install readline with 'clang' and install in /usr/local.
- Fortran compiler (for example): GCC/GFORTRAN (http://hpc.sourceforge.net/)

9.2.3 General Compilation Instructions

When all library dependencies are installed, the compilation process can begin. Execute cmake in the root directory of the SPEX source tree, where CMakeLists.txt is located (mind the dot):

```
unix:~/spex> cmake .
```

If no errors occured and all libraries were found, then type 'make':

```
unix:~/spex> make
```

When the program needs to be installed system wide, then execute:

```
unix:~/spex> sudo make install
```

The program will be installed to /opt/spex by default. Usually, administrator rights are necessary to copy the files to the right location.

9.2.4 Optional features

There are several options that can be passed to CMake to influence the build process through the -D operator. Of course, all options can be combined in a single cmake call. See the cmake documentation and the CMakeLists.txt file for details.

Compiler selection

Select a different fortran compiler:

```
unix:~> cmake . -DCMAKE_Fortran_COMPILER=ifort
```

Install prefix

Install SPEX at a different location in the 'make install' step:

```
unix:~> cmake . -DCMAKE_INSTALL_PREFIX=/home/user/software
```

Set SPEX90 variable in source files

If the build needs to be moved to a different directory structure, then it may be helpful not to include the SPEX90 variable definition in spexdist.sh and spexdist.csh, and define it separately.

```
unix:~> cmake . -DSET_SPEX90=YES
```

Force use of SPEX libraries

The use of the SPEX libraries in the source tree can be forced:

```
unix:~> cmake . -DCFITSIO=YES -DPGPLOT=YES
```

The command above will compile these libraries from the SPEX source tree. See the CMakeLists.txt file for more options.

Create install packages

CMake is shipped with a program to create install packages, called CPack. Install packages can be generated by simply calling 'cpack' after the 'make' has finished. Please see the CPack documentation and CMakeLists.txt for packageing options.

Chapter 10

Acknowledgements

We would like to express our special thanks to numerous people who have substantially contributed to the development of various aspects of this work, in alphabetical order:

Frans Alkemade, Gert-Jan Bartelds, Ehud Behar, Alex Blustin, Elisa Costantini, Max Duysens, Jacobo Ebrero, Irma Eggenkamp, Andrzej Fludra, Yan Grange, Ed Gronenschild, Liyi Gu, Theo Gunsing, Jörgen Hansen, Wouter Hartmann, Kurt van der Heyden. Fred Jansen, Jelle Kaastra, Jim Lemen, Duane Liedahl, Junjie Mao, Pasquale Mazzotta, Missagh Mehdipour, Rolf Mewe, Hans Nieuwenhuijzen, Bert van den Oord, Ken Phillips, Ciro Pinto, Ton Raassen, Remco van der Rijst, Makoto Sawada, Hans Schrijver, Karel Schrijver, Tiemen Schut, Janusz Sylwester, Rob Smeets, Katrien Steenbrugge, Jeroen Stil, Igone Urdampilleta, Dima Verner, Jacco Vink, Frank van der Wolf, Piotr Zycki.

During the years the many people have contributed as follows:

Software:

Most software is written by Kaastra. Nieuwenhuijzen contributed to the software design.

De Plaa maintains the software on different platforms.

The philosophy of the fitting procedure is based on the experiences obtained with the fitting of EXOSAT data, especially on the software package ERRFIT designed by Jansen and Kaastra.

Tiemen Schut has initiated the parallelisation of important parts of the code.

This product includes software developed by the University of Chicago, as Operator of Argonne National Laboratory. In particular code from the MINPACK package.

Plasma model:

The line and continuum emission routines were written by Kaastra, based on previous routines by Mewe, Gronenschild, van den Oord, and Lemen (1981-1986).

The original line and continuum emission data are based on the work by Mewe and Kaastra.

Raassen has calculated and compiled a large amount of atomic data for the models.

Nieuwenhuijzen, Hansen, Duysens and Van der Rijst contributed to the extension of the atomic database.

Phillips contributed to improvements of the wavelengths of strong lines.

Liedahl contributed to the Fe-L calculations.

The photo-ionization cross-sections have been provided by Verner. Behar and Raassen contributed to the atomic data for the absorption models.

Missagh Mehdipour helped with the pion model.

Mazzotta and Urdampilleta worked on the ionization balance.

Mao incorporated new radiative recombination data.

Gu developed the charge exchange models.

The non-equilibrium ionization balance routine was developed by Jansen, based on earlier work by Hans

Schrijver (1974) and Gronenschild (1981). It was extended and updated together with Sawada (2014).

SPEX models:

The SNR models are based on the work by Kaastra and Jansen (1992), with further extensions and testing by Stil, Eggenkamp and Vink.

Costantini and Pinto contributed to the dust models.

Lemen, Smeets, and Alkemade developed an earlier version (1986-1990) of a spectrum synthesis program.

The original DEM modelling was based on the work by Sylwester and Fludra (1980), while the current DEM regularisation algorithm has been designed by Karel Schrijver and Alkemade. Bartelds contributed to the testing of these models.

Piotr Zycki provided us the *refl* model code.

Documentation:

The first version (1.0) of the documentation has been prepared – with starting help by Gunsing – by van der Wolf and Nieuwenhuijzen. The second version was set-up by Kaastra with contributions from Steenbrugge, van der Heyden and Blustin. The later versions were managed by de Plaa.

The cookbook contains contributions from De Plaa, Ebrero, Grange, Pinto, Kaastra, Steenbrugge, Gu, and Mehdipour.

Bibliography

Allen, C. W. 1973, Astrophysical quantities

Anders, E. & Grevesse, N. 1989, Geochim. Cosmochim. Acta, 53, 197

Arnaud, M. & Raymond, J. 1992, ApJ, 398, 394

Arnaud, M. & Rothenflug, R. 1985, A&AS, 60, 425

Baker, S. & Cousins, R. D. 1984, Nuclear Instruments and Methods in Physics Research, 221, 437

Barrus, D. M., Blake, R. L., Burek, A. J., Chambers, K. C., & Pregenzer, A. L. 1979, Phys. Rev. A, 20, 1045

Bautista, M. A., Mendoza, C., Kallman, T. R., & Palmeri, P. 2004, A&A, 418, 1171

Behar, E. & Netzer, H. 2002, ApJ, 570, 165

Behar, E., Sako, M., & Kahn, S. M. 2001, ApJ, 563, 497

Beiersdorfer, P., López-Urrutia, J. R. C., Springer, P., Utter, S. B., & Wong, K. L. 1999, Review of Scientific Instruments, 70, 276

Bryans, P., Landi, E., & Savin, D. W. 2009, ApJ, 691, 1540

Caldwell, C. D., Schaphorst, S. J., Krause, M. O., & Jiménez-Mier, J. 1994, J. Electron. Spectrosc. Relat. Phenom., 67, 243

Cardelli, J. A., Clayton, G. C., & Mathis, J. S. 1989, ApJ, 345, 245

Cash, W. 1979, ApJ, 228, 939

de Plaa, J., Kaastra, J. S., Tamura, T., et al. 2004, A&A, 423, 49

de Plaa, J., Werner, N., Bykov, A. M., et al. 2006, A&A, 452, 397

den Herder, J. W., Brinkman, A. C., Kahn, S. M., et al. 2001, A&A, 365, L7

Doron, R. & Behar, E. 2002, ApJ, 574, 518

Draine, B. T. 2003, ApJ, 598, 1026

Fabian, A. C., Arnaud, K. A., Bautz, M. W., & Tawara, Y. 1994, ApJ, 436, L63

Fujii, K., Akamatsu, K., Muramatsu, Y., & Yokoya, A. 2003, Nuclear Instruments and Methods in Physics Research B, 199, 249

García, J., Kallman, T. R., Witthoeft, M., et al. 2009, ApJS, 185, 477

García, J., Mendoza, C., Bautista, M. A., et al. 2005, ApJS, 158, 68

Gauss, K. F. 1809, Theoria motus corporum coelestium in sectionibus conicis solem ambientium. (Perthes and Besser, Hamburg, 1809.)

Grevesse, N., Noels, A., & Sauval, A. J. 1992, in ESA Special Publication, Vol. 348, Coronal Streamers, Coronal Loops, and Coronal and Solar Wind Composition, ed. C. Mattok, 305–308

Grevesse, N. & Sauval, A. J. 1998, Space Sci. Rev., 85, 161

Gronenschild, E. H. B. M. & Mewe, R. 1982, A&AS, 48, 305

Gu, L., Kaastra, J., & Raassen, A. J. J. 2016, A&A *in press*

Gu, M. F., Holczer, T., Behar, E., & Kahn, S. M. 2006, ApJ, 641, 1227

Gu, M. F., Schmidt, M., Beiersdorfer, P., et al. 2005, ApJ, 627, 1066

Hayakawa, S., Hajima, Y., Qiao, S., Namatame, H., & Hirokawa, T. 2008, Analytical Sciences, 24, 835

Hiraya, A., Nobusada, K., Simon, M., et al. 2001, Phys. Rev. A, 63, 042705

Hua, X.-M. & Titarchuk, L. 1995, ApJ, 449, 188

Janev, R. K., Phaneuf, R. A., Tawara, H., & Shirai, T. 1993, Atomic Data and Nuclear Data Tables, 55, 201

Juett, A. M., Schulz, N. S., & Chakrabarty, D. 2004, ApJ, 612, 308

Kaastra, J. S. 2008, Clusters of galaxies: Beyong the Thermal View (Springer Science+Business media,

BV)
Kaastra, J. S. 2017, ArXiv e-prints, 1707.09552
Kaastra, J. S. & Barr, P. 1989, A&A, 226, 59
Kaastra, J. S. & Bleeker, J. A. M. 2016, A&A, 587, A151
Kaastra, J. S., de Vries, C. P., Costantini, E., & den Herder, J. W. A. 2009, A&A, 497, 291
Kaastra, J. S. & Jansen, F. A. 1993, A&AS, 97, 873
Kaastra, J. S., Mewe, R., Liedahl, D. A., et al. 1996a, A&A, 314, 547
Kaastra, J. S., Mewe, R., & Nieuwenhuijzen, H. 1996b, in UV and X-ray Spectroscopy of Astrophysical and Laboratory Plasmas, ed. K. Yamashita & T. Watanabe, 411–414
Kaastra, J. S. & Paerels, F. 2011, High-Resolution X-ray Spectroscopy (Springer Science+Business media, BV)
Kaastra, J. S., Steenbrugge, K. C., Raassen, A. J. J., et al. 2002, A&A, 386, 427
Kaastra, J. S., Tamura, T., Peterson, J. R., et al. 2004, A&A, 413, 415
Kirchhoff, G. 1860, Annalen der Physik, 185, 275
Kosenko, D., Hillebrandt, W., Kromer, M., et al. 2015, MNRAS, 449, 1441
Krause, M. O. 1994, Nuclear Instruments and Methods in Physics Research B, 87, 178
Laor, A. 1991, ApJ, 376, 90
Lee, J. C. 2010, X-ray Spectroscopy of Astrophysical Dust: A High Spectral Resolution (Re)View & Look to the Future, http://www.sron.nl/files/HEA/XRAY2010/talks/3/lee.pdf, presentation at conference "High-resolution X-ray spectroscopy: past, present and future", Utrecht, March 15-17, 2010
Lee, J. C. & Ravel, B. 2005, ApJ, 622, 970
Lee, J. C., Xiang, J., Ravel, B., Kortright, J., & Flanagan, K. 2009, ApJ, 702, 970
Lee, S. K., Lin, J.-F., Cai, Y. Q., et al. 2008, PNAS, 105, 7925
Lodders, K. 2003, ApJ, 591, 1220
Lodders, K., Palme, H., & Gail, H.-P. 2009, Landolt Börnstein, 44
Magdziarz, P. & Zdziarski, A. A. 1995, MNRAS, 273, 837
Mao, J., Kaastra, J., & Badnell, N. R. 2017, A&A, 599, A10
Mehdipour, M., Kaastra, J. S., Kriss, G. A., et al. 2015, A&A, 575, A22
Mendoza, C., Kallman, T. R., Bautista, M. A., & Palmeri, P. 2004, A&A, 414, 377
Mewe, R. 1972, A&A, 20, 215
Mewe, R., Gronenschild, E. H. B. M., & van den Oord, G. H. J. 1985, A&AS, 62, 197
Mewe, R., Kaastra, J. S., Schrijver, C. J., van den Oord, G. H. J., & Alkemade, F. J. M. 1995, A&A, 296, 477
Mewe, R., Lemen, J. R., & van den Oord, G. H. J. 1986, A&AS, 65, 511
Mewe, R., Schrijver, C. J., Kaastra, J. S., Alkemade, F. J. M., & Haisch, B. M. 1994, in Astronomical Society of the Pacific Conference Series, Vol. 64, Cool Stars, Stellar Systems, and the Sun, ed. J.-P. Caillault, 41
Miller, J. M., Kaastra, J. S., Miller, M. C., et al. 2015, Nature, 526, 542
Morrison, R. & McCammon, D. 1983, ApJ, 270, 119
Mullen, P. D. e. a. 2016, ApJS, TBD
Nahar, S. N., Pradhan, A. K., & Zhang, H. L. 2001, Phys. Rev. A, 63, 060701
Nolte, J. L., Stancil, P. C., Liebermann, H. P., et al. 2012, Journal of Physics B Atomic Molecular Physics, 45, 245202
O'Donnell, J. E. 1994, ApJ, 422, 158
Olalla, E., Wilson, N. J., Bell, K. L., Martin, I., & Hibbert, A. 2002, MNRAS, 332, 1005
Palmeri, P., Mendoza, C., Kallman, T. R., & Bautista, M. A. 2003a, A&A, 403, 1175
Palmeri, P., Mendoza, C., Kallman, T. R., Bautista, M. A., & Meléndez, M. 2003b, A&A, 410, 359
Parent, P., Laffon, C., Mangeney, C., Bournel, F., & Tronc, M. 2002, J. Chem. Phys., 117, 10842
Peterson, J. R., Jernigan, J. G., & Kahn, S. M. 2004, ApJ, 615, 545
Peterson, J. R., Kahn, S. M., Paerels, F. B. S., et al. 2003, ApJ, 590, 207
Phillips, K. J. H., Mewe, R., Harra-Murnion, L. K., et al. 1999, A&AS, 138, 381
Pinto, C., Kaastra, J. S., Costantini, E., & Verbunt, F. 2010, A&A, 521, A79
Porquet, D., Kaastra, J. S., Page, K. L., et al. 2004, A&A, 413, 913
Pradhan, A. K., Chen, G. X., Delahaye, F., Nahar, S. N., & Oelgoetz, J. 2003, MNRAS, 341, 1268

Raymond, J. C. & Smith, B. W. 1977, ApJS, 35, 419
Ross, J. E. & Aller, L. H. 1976, Science, 191, 1223
Rumph, T., Bowyer, S., & Vennes, S. 1994, AJ, 107, 2108
Rybicki, G. B. & Lightman, A. P. 1986, Radiative Processes in Astrophysics
Shakura, N. I. & Sunyaev, R. A. 1973, A&A, 24, 337
Steenbrugge, K. C., Kaastra, J. S., Crenshaw, D. M., et al. 2005a, A&A, 434, 569
Steenbrugge, K. C., Kaastra, J. S., de Vries, C. P., & Edelson, R. 2003, A&A, 402, 477
Steenbrugge, K. C., Kaastra, J. S., Sako, M., et al. 2005b, A&A, 432, 453
Sunyaev, R. A. & Titarchuk, L. G. 1985, A&A, 143, 374
Tamura, T., Kaastra, J. S., den Herder, J. W. A., Bleeker, J. A. M., & Peterson, J. R. 2004, A&A, 420, 135
Titarchuk, L. 1994, ApJ, 434, 570
Urdampilleta, I., Kaastra, J. S., & Mehdipour, M. 2017, A&A, 601, A85
van Aken, P. A., Liebscher, B., & Styrsa, V. J. 1998, Physics and Chemistry of Minerals, 25, 494
Verner, D. A., Verner, E. M., & Ferland, G. J. 1996, Atomic Data and Nuclear Data Tables, 64, 1
Verner, D. A. & Yakovlev, D. G. 1995, A&AS, 109, 125
Wheaton, W. A., Dunklee, A. L., Jacobsen, A. S., et al. 1995, ApJ, 438, 322
Wight, G. R. & Brion, C. E. 1974, Journal of Electron Spectroscopy and Related Phenomena, 3, 191
Wilms, J., Allen, A., & McCray, R. 2000, ApJ, 542, 914
Wu, Y., Stancil, P. C., Schultz, D. R., et al. 2012, Journal of Physics B Atomic Molecular Physics, 45, 235201
Zycki, P. T. & Czerny, B. 1994, MNRAS, 266, 653
Zycki, P. T., Done, C., & Smith, D. A. 1999, MNRAS, 305, 231

Index