Getting started with XSPEC

D. R. Wilkins

April 2, 2024

XSPEC is an X-ray spectral fitting package, used for maximum-likelihood fitting of model functions to spectroscopic data. XSPEC is used to analyze spectral data from all of the major X-ray missions, including *Chandra*, *XMM-Newton*, *NuSTAR*, *Swift*, *NICER*, *IXPE* and more, and is able to load spectra stored in the standard 'OGIP' file formats that are provided by the data reduction pipelines for all of the major missions.

This guide is intended to provide an introduction to X-ray spectral fitting using XSPEC, based upon a practical example of fitting a spectrum from a nearby AGN. It provides a walk through of the most commonly used XSPEC features and techniques that are required to model typical spectra and to perform statistical inference of physical properties of an astronomical source from spectral model parameters, as well as 'just enough' background on some of the statistical concepts involved (though this is by no means a comprehensive guide to statistics and inference, and further reading is *strongly encouraged*).

1 Installing and running XSPEC

XSPEC is distributed as part of the HEASOFT software package (maintained by the HEASARC at the NASA Goddard Spaceflight Center)[1]. In general, it is best to compile HEASOFT yourself from the source code distribution, rather than using the pre-compiled binaries (in particular, you will not be able to install additional XSPEC models if you are using the binary distribution).

To use XSPEC, you will need to initialise HEASOFT in your shell environment, usually by typing at your shell prompt `$` (see the HEASOFT installation instructions for more details).

```
$ heainit
```

There are two interfaces to XSPEC: the comand line interface, and a PYTHON interface, PYXSPEC. While PYXSPEC can be useful for automating analyses and integrating with other routines, it does not implement all of the features of XSPEC and has some limitations. We will therefore use the XSPEC command line interface here. Once you have learned techniques in XSPEC, they can often easily be converted into PYXSPEC. You can start XSPEC by typing

```
$ xspec
```

You will see the welcome screen showing you the version number and patch level you are running and the `XSPEC12>` prompt means that XSPEC is ready to accept a command.

```
                XSPEC version: 12.14.0a
        Build Date/Time: Wed Feb 28 16:05:49 2024

XSPEC12>_
```

[1]HEASOFT downloads and installation guides are available at https://heasarc.gsfc.nasa.gov/lheasoft/download.html

You can use the up arrow key to scroll back through commands you have previously issued. Press CTRL+R then start typing to search backwards through your command history to find the last command you ran that contains the text you type (press enter to run it). XSPEC commands can be abbreviated, typing only the beginning of the command, so long as it is not ambiguous, for example `model`, `plot` and `setplot` can be abbreviated `mo`, `pl` and `setpl`.

You can abort a fit or other operation XSPEC is currently performing by pressing CTRL+C. If at any time you are stuck in the input prompts for a command you wish to get out of, type `/*` and press ENTER to return to the main XSPEC prompt.

2 X-ray spectral data

X-ray spectra are typically stored in a FITS table that lists the number counts per instrument channel, however we usually want to model a spectrum expressed in physical units of flux per photon energy bin (or per wavelength). The channel represents the photon energy, but the mapping between channel and energy is non-trivial, and will depend upon the specific instrument in use and its calibration (which may vary as a function of the time at which the observations were taken). XSPEC therefore requires additional input files to perform this mapping and account for the energy-dependent sensitivity and calibration of the instrument, as well as to account for background signals present in the spectrum:

- **Source spectrum** (also known as the source PI spectrum or PHA spectrum, often with the file extension `.pha` or `.pi`): a FITS file with a table containing the number of counts recorded per channel over the course of the observation. Header keywords contain information about the observation, most notably the exposure time, to convert total counts into count rate.
- **RMF**, **redistribution matrix** or **response matrix** (with the file extension `.rmf`): used to map between photon energies and instrument channels. The RMF is a two-dimensional matrix containing the spectrum that would be observed (in counts per channel) if the detector were illuminated by a monochromatic source of X-ray photons at different energies. In the ideal case, a single photon energy would map onto a single channel, however in real detectors, a single photon energy will produce signals in multiple channels due to how photons interact with the detector. A FITS file containing two tables: one defining the energy channels, and one containing the matrix.
- **ARF**, **ancillary response file** or **effective area curve** (with the file extension `.arf`: contains the effective area of the telescope + detector system as a function of photon energy, which accounts for a number of effects including the quantum efficiency of the detector and the varying ability of the mirrors to collect and focus X-ray photons as a function of energy. The observed spectrum will be the true source spectrum multiplied by the effective area. Note that the ARF depends not just on which telescope and detector are used, but also the geometry of the source being observed (*i.e.* a point source *vs.* an extended source) and also the location of the source in the field of view. Another FITS file containing a table of effective area as a function of energy.
- Sometimes the RMF and ARF are combined into a **single response matrix** or **RSP** file (with the extension `.rsp`. This is essentially the RMF with each response for each energy multiplied by the ARF.
- **Background spectrum** (also with the extension `.pha` or `.pi`): the spectrum obtained from a region of the detector with no source to obtain an estimate of the background signal. XSPEC subtracts the background from the source spectrum before fitting the model (scaling the background spectrum according to the ratio of the areas of the detector from which the two spectra were extracted). The

count rate in the background spectrum is scaled according to the See note below about handling the background.

- **Grouped spectrum** (with the extension `.grp`, `.pha` or `.pi`): another version of the source spectrum, but with the channels grouped into bins. The energy channels themselves are not changed, and all of the original instrument channels will still be in the file. There is an additional column in the table that tells XSPEC which channels to combine into a single bin. Spectra are often grouped or rebinned to have a minimum number of counts or a minimum signal-to-noise ratio per bin. There is also 'optimal' binning to optimally sample the instrument resolution (most useful for high-resolution grating spectra). Often analysis will be performed on these grouped spectra (though note if you are using the Cash statistic or C-statistic, you will want the ungrouped spectra). Grouping or binning is performed outside of XSPEC using the FTOOLS GRPPHA or FTGROUPPHA.

Each of these files should be generated by the data reduction pipeline for the mission you are usuing, and we will assume here that you already have the spectrum and response files for the observations you wish to analyse.

See Chapters 2-4 of "Handbook of X-ray Astronomy" by Arnaud, Smith and Siemiginowska, Cambridge University Press 2011 for more information about response functions and how they are derived from the physical characteristics of an X-ray detector.

3 Loading the data

The `data` command in XSPEC is used to load spectra, and to manage the data sets in use. If we have a spectrum in a file called `src.grp`, we can load it with the command:

```
    XSPEC12> data 1:1 src.grp
```

XSPEC can load multiple spectra and fit a model to them all simultaneously. Each spectrum is assigned a number so that we can refer to it later. Spectra are organised into groups, where spectra in the same group are fit with exactly the same model (with exactly the same parameter values). In general we load spectra taken of the same object at the same time (perhaps with different instruments) into the same group, and either spectra of different objects or spectra of the same object taken at different times (so some of the parameter values may have varied) into different groups. Note, however, that often we will need to load spectra taken at the same time with different instruments into different groups if we need to handle differences or uncertainties in the calibration using model parameters (see later).

The numbers in the `data` command specify the `[group_number]:[spectrum_number]`. `data 1:1` refers to spectrum 1 in group 1. `data 1:2` refers to spectrum 2 in group 1. We would use `data 2:2` if we wanted to put spectrum 2 into group 2 instead. Note that spectrum numbers must be unique and run sequentially, so we can't have a spectrum 1 in group 2.

To load multiple spectra, we might do:

```
    XSPEC12> data 1:1 src1.grp
    XSPEC12> data 1:2 src1.grp
    XSPEC12> data 2:3 src3.grp
    XSPEC12> data 2:4 src4.grp
```

We can remove a spectrum by setting it to none

```
    XSPEC12> data 1:1 none
```

Or change a spectrum by specifying a new file in its place. Note, however, that if we change or remove any one spectrum, all spectra appearing after it in the sequence will be removed. So, in this case, removing or changing spectrum 1 will remove 2, 3 and 4. If we don't wish to remove these spectra, we may put a / at the end of the `data` command to leave all the other spectra in place:

```
    XSPEC12> data 1:1 none /
```

Usually, the headers within a spectrum file will point to the response files (RMF and ARF) and the corresponding background file, so that XSPEC can load these automatically. This, however, is not always the case and you may get a message saying that "one or more spectra are missing responses". If this happens, you can load the RMF, ARF and background corresponding to each spectrum yourself, using the commands:

```
    XSPEC12> resp 1 src1.rmf
    XSPEC12> arf 1 src1.arf
    XSPEC12> back 1 bkg1.pha
```

(where the number corresponds to the spectrum number you are loading the response for). Note that while XSPEC warns you about a missing RMF, you will not get a warning about a missing ARF.

You can view the loaded data files, check all the responses are present and view information like the total net count rate and exposure time with the command:

```
    XSPEC12> show data

    1 file 1 spectrum
    Spectrum 1  Spectral Data File: src1.grp
    Net count rate (cts/s) for Spectrum:1  4.860e+00 +/- 8.956e-03 (99.6 % total)
     Assigned to Data Group 1 and Plot Group 1
      Noticed Channels:  1-2255
      Telescope: XMM Instrument: EPN  Channel Type: PI
      Exposure Time: 6.113e+04 sec
    Using fit statistic: chi
    Using Background File                bkg1.pha
     Background Exposure Time: 6.113e+04 sec
    Using Response (RMF) File            src1.rmf for Source 1
    Using Auxiliary Response (ARF) File  src1.arf
```

4 Selecting the energy range to fit

By default, we will not want to include all of the energy channels in a spectrum in our analysis. There will be energy channels (particularly at the bottom and top of an instrument's sensitivity range) for which the calibration is problematic. In addition, there may be channels at the high energy range where our source is so faint that it drops below the level of the background. We can remove these channels from our analysis using the `ignore` command.

To ignore channels below 0.3 and above 10 keV in spectrum number 1, use the command:

```
    XSPEC12> ignore 1:**-.3,10.-**
```

Here, `**` represents all energy channels, so `**-.3` means all energy channels up to 0.3 keV. `ignore 1:**` will ignore all energy channels. We can ignore multiple energy ranges at once, separating them with a comma. Note that the energy values are entered as floating point numbers, *i.e.* `10.` and not just `10` without the decimal point. Floating point numbers refer to energy values in keV. Integers refer to channel numbers, so `ignore 3-10` will ignore channel numbers 3 to 10, not energies betwen 3 and 10 keV.

If we have multiple spectra loaded, we can apply this command to all loaded spectra by replacing the spectrum number with a *:

```
    XSPEC12> ignore *:**-.3,10.-**
```

The opposite of `ignore` is `notice`. If we have removed some channels from our analysis and would like to include them again, we can use the `notice` command in exactly the same way as the `ignore` command:

```
    XSPEC12> notice 1:**-.3,10.-**
```

The normal energy ranges we use for different instruments are:

- *XMM-Newton* EPIC pn: 0.3-10 keV
- *XMM-Newton* EPIC MOS: 0.2-10 keV
- *NuSTAR* FPMA and FPMB: 3-78 keV
- *Chandra* ACIS-S and ACIS-I: 0.5-7 keV
- *Suzaku* XIS: 0.5-12 keV, ignoring 1.7-2.3 keV
- *Swift* XRT: 0.3-5.5 keV

5 Creating a first plot of the spectrum

XSPEC has built-in plotting capability, and can create plots of the spectrum, the model, the residuals between the spectrum and the model, and more.

The first thing we will need to create a plot is a plotting device. Usually, we want to plot to the screen using the Xwindows device, `/xw` or `/xs`, so change the plot device with the command:

```
    XSPEC12> cpd /xw
```

By default, plots are created with instrument channel on the X-axis. We can change this to plot the spectrum *vs.* energy in keV with the command:

```
    XSPEC12> setplot energy
```

(or `setpl en` for short). Then we can plot the spectrum:

```
    XSPEC12> plot data
```

Or (more usefully) to plot the spectrum on logarithmic axes (Fig. 1):

```
    XSPEC12> plot ldata
```

5.1 Rebinning the plot

It can often be difficult to see the details of a spectrum or the residuals between the data and the model when there are a lot of finely-spaced data points with large error bars. XSPEC can rebin the plot to make things easier to see using the `setplot rebin` command. This takes two parameters. Firstly, the minimum signal-to-noise in each bin, then the number of sequential bins that can be added together to achieve this.

For example:

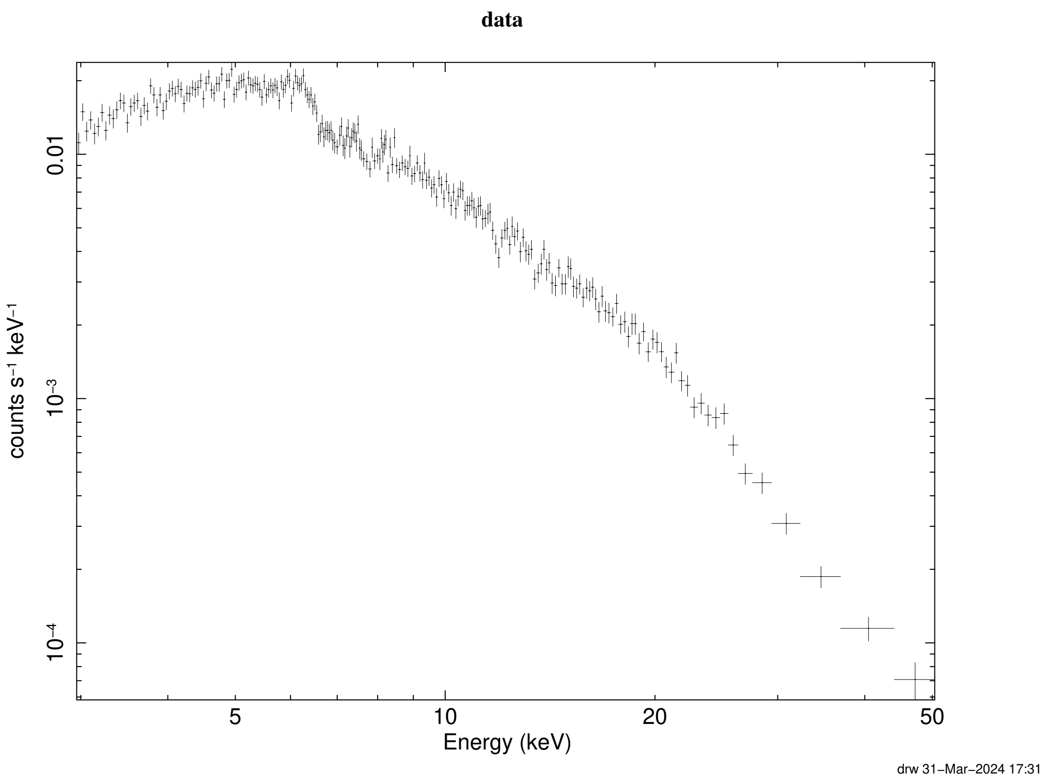

Figure 1: The spectrum (on logarithmic axes) of the AGN Mrk 335 obtained using *NuSTAR* over the 3-50 keV energy range, plotted using `plot ldata`. The plotted spectrum has been rebinned using `setplot rebin 10 20`.

```
XSPEC12> setplot rebin 10 20
```

Will try to achieve a minimum signal-to-noise ratio of 10 in each spectral bin that is plotted, but only by combining a maximum of 20 bins to achieve this. If, after 20 mins, $S/N = 10$ has not been reached, then that bin will be plotted with lower signal-to-noise.

Note that `setplot rebin` only rebins the spectrum for the purposes of plotting. The fit statistic is still evaluated using the original binning or grouping of the input spectrum (see 'grouped spectrum' above).

5.2 Inspecting the background

Before setting the upper energy bound for the analysis, it is useful to inspect the (background-subtracted) spectrum alongside the background spectrum. To do this, we can turn on the background in the plot (Fig. 2):

```
XSPEC12> setplot back
XSPEC12> plot ldata
```

When performing a simple analysis with the background subtracted, we should generally ignore the energy range above which the source spectrum drops below the background.

Turn the background off again in the plot using:

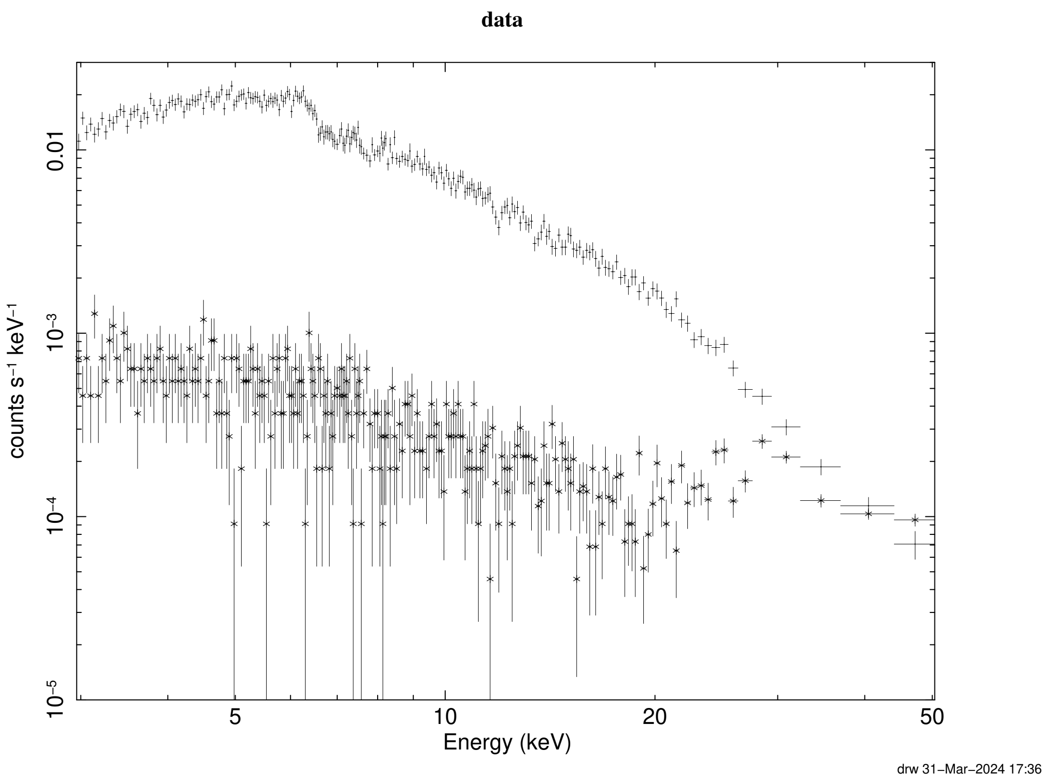

Figure 2: The *NuSTAR* spectrum of Mrk 335 with the instrumental background plotted below. The background is added to the plot using the command `setplot back` before issuing the `plot` command.

```
    XSPEC12> setplot noback
```

6 Fitting a model to a spectrum

A model is a mathematical function that describes the observed spectrum and predicts the number of counts or the flux observed as a function of photon energy or wavelength. Models describe different physical processes that contribute to the emission from a source, and each model will have a number of parameters that can be used to infer physical properties of the source. The model is compared to the data via a fit statistic, and fitting the model to the data involves varying the model parameters to find the minimum possible value of that fit statistic for the data in question. Once the model has been fit, the minimum value of fit statistic achieved indicates whether a given model provides an acceptable fit to the data. Comparing fit statistics between models allows for selection between models to find the one that best describes the data. The parameter values that correspond to the minimum fit statistic enable inferences to be made about the physical properties of the source that the parameters represent (or, put another way, provide a means to measure these properties of the source, once it has been established that the model provides an acceptable fit).

Fitting is performed in XSPEC by considering the fit statistic to be a function of the model parameters, and its minimum value in the parameter space is found using the Levenberg-Marquardt algorithm.

XSPEC follows a *forward modelling* approach. That is to say that we do not 'correct' the data to account for factors including the source redshift, the effective area of the telescope and the response of the instrument, rather we *fold* our model through the response function of the instrument (as described by the RMF and ARF) so as to predict what our telescope would have measured if the source really were described by our model. This is an important philosophical point, treating our measured data points as the 'truth' since when we correct our measured data points we are implicitly applying a model. The instrument effective area, calibration and other correction factors will carry some degree of uncertainty, which we neglect when we merely correct our data points.

6.1 The fit statistic

In maximum likelihood fitting, the fit statistic represents the likelihood function, that is the probability of obtaining the data we have if the model represented the truth, *i.e.* the probability of the data given the model, $L = P(\mathrm{data} \mid \mathrm{model})$. The best-fitting model (with its best fitting parameter values) is that which provides the greatest probability of generating the data that were obtained. The fit statistic is usually expressed as $S = -2\log L$, such that the maximum likelihood and the best-fitting model corresponds to the minimum value of S.

The default fit statistic in XSPEC is χ^2, which corresponds to $-2\log L$ for data points that follow a Gaussian distribution with mean corresponding to the true value and standard deviation σ corresponding to the error bar. X-ray spectral data expressed the number of photon counts per spectral bin, however, follow a Poisson distribution, thus the χ^2 statistic is formally valid only when the Poisson distribution approximates a Gaussian, *i.e.* when the spectrum has been grouped to have at least $20 \sim 25$ counts per bin.

Alternatively, the Cash statistic corresponds directly to the likelihood function for Poisson-distributed data and is valid no matter how many (or few) counts are present per spectral bin. XSPEC defines a modified version of the Cash statistic, referred to as the C-statistic, which asymptotes to the chi-squared statistic in the limit of large count rates. There is no need to use a grouped or binned spectrum when using the C-statistic (and using a grouped spectrum will unnecessarily limit the resolution of your spectrum). Switch to the C-statistic using:

```
    XSPEC12> statistic cstat
```

There is, however, one caveat to using the C-statistic. The default behaviour of XSPEC is to subtract an estimate of the background from the spectrum. It is the total number of counts per bin (including background counts) that follow a Poisson distribution, therefore, formally the C-statistic should be used to fit a model to the *total* spectrum including both source and background components, and should not be used to fit the background-subtracted spectrum that is loaded by default in XSPEC. However, an approximation to the C-statistic can be made for the case of a background-subtracted spectrum that is sometimes referred to as the W-statistic. When a background spectrum is loaded and `cstat` is selected, XSPEC will instead evaluate this modified statistic. Care should be taken in this instance, since this approximation will not be correct when the source and background count rates are comparable. This approximation also diverges if any of the spectral bins contain zero counts (in this case, a very large number will be seen for the C-statistic output. If this occurs, group the spectrum to have a minimum of 1 count per bin.[2]

[2]For more information about statistics in XSPEC, see https://heasarc.gsfc.nasa.gov/xanadu/xspec/manual/XSappendixStatistics.html

6.2 Building a model

Models are built in XSPEC by combining one or more components using a simple syntax that resembles a mathematical expression. Components each describe a different physical process or phenomenological function and fall into four categories:

- **Additive model components:** describe different primary emission processes that can be added together. *e.g.*
  - `powerlaw`: a phenomenological power law to describe continuum emission, for example the primary continuum emission seen from the corona around an accreting black hole.
  - `cutoffpl`: a power law with exponential cut-off at high energy.
  - `nthcomp`: Comptonisation from a thermal plasma of hot electrons, producing an appximately power law spectrum with high-energy cut-off corresponding to the temperature of the plasma.
  - `compTT`: alternative model for Comptonisation.
  - `bbody`: black body emission.
  - `diskbb`: multi-temperature black body emission describing the thermal emission from an accretion disc (where material at different radii emits as a black body with varying temperature).
  - `kerrbb`: thermal emission from a Novikov-Thorne accretion disc, including relativistic effects.
  - `gauss`: Gaussian profile, usually used to model an emission line.
  - `zgauss`: Gaussian line profile with redshift parameter such that the line energy can be expressed in the rest frame of the source.
  - `relline`: relativistically broadened emission line from an accretion disc around a black hole or neutron star
  - `xillver` (and variants): reflection (*i.e.* reprocessed emission) from cold, optically-thick plasma illuminated by an X-ray continuum.
  - `relxill` (and variants): reflection (*i.e.* reprocessed emission) from an accretion disc around a black hole or neutron star, combining *xillver* reflection spectra with relativistic blurring or broadening by *relline*.[3]
  - `brems`: Brensstrahlung emission.
  - `apec`: emission from collisionally ionised diffuse gas.
- **Multiplicative model components:** multiply a spectrum (or a combination of spectral components) by a transmission curve, usually to model absorption. *e.g.*
  - `edge`: photoelectric absorption edge at a specified energy
  - `gabs`: Gaussian absorbtion line
  - `tbabs`: photoelectric absorption from neutral gas in the ISM (see below about modelling absorption in the Milky Way ISM).
  - `ztbabs`: absorption from neutral gas with redshift parameter, to model absorption intrinsic to the source (or at other redshift along the line-of-sight).
  - `TBnew`: enhanced version of TBABS with higher resolution representation of absorption lines and more tunable parameters (useful for high-resolution grating spectra).

[3]The RELXILL and XILLVER model packages need to be installed separately: https://www.sternwarte.uni-erlangen.de/~dauser/research/relxill/

  - `warmabs`: absorption in a photoionised gas (*e.g.* for warm absorbers in AGN). Note that this model is relatively slow to calculate and it is usually better to use pre-calculated absorption table models generated using XSTAR.
  - `zxipcf`: absorption from a partially-covering ionised warm absorber (note that this model assumes a fixed ionising SED and while useful for getting a first guess at the effects of a warm absorber, an absorption model should be calculated specifically for the source in question).

- **Convolution models:** despite the name, these models are more general than simple convolutions. A convolution model is essentially a function that takes in a spectrum and transforms it in some arbitrary way (convolving with some kernel is just one example). In the XSPEC model syntax, convolution models are written as multiplications with the components upon which they act. *e.g.*
  - `gsmooth`: Gaussian smoothing.
  - `relconv`: convolve the input spectrum with the profile of a relativistically broadened emission line from an accretion disc around a black hole (convolving with the reflection spectrum produced in the rest frame of the plasma would predict the observed spectrum from the disc).
  - `cflux`: renormalise a component so that it has specified flux (in $\mathrm{erg\,s^{-1}}$) over a specified energy range (see section on flux measurements later).
  - `cpflux`: renormalise a component so that it has specified photon flux (in $\mathrm{ct\,s^{-1}}$) over a specified energy range (see section on flux measurements later).
  - `zashift`: redshift a component (usually not needed since most components have their own redshift parameter).

- **Mixing models:** see XSPEC manual.

These are just some examples of models available in XSPEC. Most of these models are functions that calculate the model on-the-fly from the set of input parameters. To see all available models type:

```
XSPEC12> model
```

In addition, additive and multiplicative models can be provided via table models using the `atable{model.fits}` and `mtable{model.fits}` commands. Table models are FITS files that contain pre-computed model spectra at specific parameter values. Finally, it is possible to write custom model functions for XSPEC in either C/C++ or Fortran.

Models are defined in XSPEC using the `model` command and components are added and multiplied together. For example, to create a simple model consisting of a power law continuum plus a Gaussian emission line, enter:

```
XSPEC12> model powerlaw + gauss
```

6.3 X-ray absorption in the ISM of the Milky Way

As X-rays pass through the interstellar medium of the Milky Way to reach the Earth, it is necessary to account for absorption of soft X-rays (particularly below 1 keV) in the ISM. This can be done by multiplying the entire model expression by a `tbabs` component with the column density parameter set for the appropriate celestial co-ordinates:

```
XSPEC12> model tbabs(powerlaw + gauss)
```

Column densities through the Milky Way to sources at different locations can be computed from surveys of the ISM using the HEASARC nH Calculator (https://heasarc.gsfc.nasa.gov/cgi-bin/Tools/w3nh/w3nh.pl). Note that in TBABS the column density parameter is in units of $10^{22}\,\mathrm{cm}^{-2}$ (*i.e.* enter 0.01 for $10^{20}\,\mathrm{cm}^{-2}$). Start by freezing (see later) the column density at the calculated value, though note that for some locations, these values may vary slightly, so can be thawed later in the fitting process.

6.4 Setting up the model and its parameters

Once you have decided upon the model components you wish to include, the model can be set up in XSPEC using the `model` command.

As an example, we will model the 3-50 keV spectrum of the AGN Mrk 335 observed using *NuSTAR*.

Model 1: power law

We will start with a simple model consisting of just a power law describing the X-ray continuum (of course accounting for absorption in the Milky Way with column density $4 \times 10^{20}\,\mathrm{cm}^{-2}$):

```
XSPEC12> model tbabs*powerlaw
```

After entering the model, XSPEC will prompt for the starting values of each of the model parameters:

```
Input parameter value, delta, min, bot, top, and max values for ...
          1      0.001(      0.01)          0          0     100000      1e+06
1:TBabs:nH>
```

First off, we are being prompted for the starting value of the TBABS column density parameter, `nH`. Above the prompt, XSPEC shows us the default values for this parameter.

Each parameter has six values associated with it that are entered in order:

1. **value:** The starting value of the parameter. Once the fit has been run, this will be replaced with the best-fitting value.
2. **delta:** The step size used when calculating partial derivatives of the fit statistic with respect to this parameter. If the parameter is variable, it's usually best to leave this at the default value. If you wish to freeze the parameter (*i.e.* not allow it to vary during the fit), set the delta value to 0 or a negative value.
3. **min:** hard lower limit of the parameter
4. **bot(tom):** soft lower limit of the parameter
5. **top:** soft upper limit
6. **max:** hard upper limit

Parameters have both hard and soft limits. The parameter value is not allowed outside the hard limits during the fit. If the best fit is found to be at the hard minimum, the parameter will 'peg' at this value and no lower values will be explored (if you find a best fit at the edge of the hard limits, you may want to try relaxing those limits). Soft limits represent the expected range of the parameter. While the parameter is made 'stiffer' to adjustment outside these limits, the parameter is allowed to go outside the soft limits if this is where the best fit is found to be. If unsure, it's usually best to set the hard and soft limits to the same values.

While a parameter is specified by these six values, you need not provide all of them. If you provide less than six values, they will be assigned in the above order, and the default values will be taken for the rest. Here, if we want to just freeze `nH` at 0.04, we can input:

```
1:TBabs:nH> 0.04 -1
```

(where the negative delta causes the parameter to be frozen).

If we want ot provide only some of the values, but not in order, we can use commas as separators and leave out the values we wish to leave at the default. For example, to set the value to 0.04 and the limits between 0.01 and 1, but leave the delta unchanged we can input `0.04,,0.01,0.01,1,1`.

We will then be prompted for the other model parameters, in this case the photon index and normalisation of the power law:

```
              1       0.01(      0.01)         -3         -2          9         10
2:powerlaw:PhoIndex>
          1       0.01(      0.01)          0          0      1e+20      1e+24
3:powerlaw:norm>
```

If we wish to accept the default values for both of these, we can just press ENTER for each. To accept the default values for all remaining model parameters, enter `/*` then press ENTER.

The final parameter of every additive model component is its normalisation or 'norm'. This is a multiplicative scaling factor applied to the component. For some models, the normalisation has a specific definition (for example the count rate or flux at some specific energy). For some models, however, the flux of the component will depend not just upon the normalisation but upon other parameters. It is therefore wise not to over-interpret normalisation values to make physical measurements without understanding exactly how it is defined for each model (see section later on flux measurements).

We can view the current model definition by typing:

```
XSPEC12> show model

Current model list:

Model TBabs<1>*powerlaw<2> Source No.: 1   Active/Off
    For Data Group(s): 1

Using energies from responses.
```

Every component in our model is assigned a number so we can refer to it later. Here, TBABS is component 1 and POWERLAW is component 2.

We can view the model parameters and their current values using:

```
XSPEC12> show par

Parameters defined:
========================================================================
Model TBabs<1>*powerlaw<2> Source No.: 1   Active/Off
Model Model Component  Parameter  Unit     Value
 par  comp
   1    1   TBabs      nH         10^22    4.00000E-02  frozen
   2    2   powerlaw   PhoIndex            1.00000      +/-  0.0
   3    2   powerlaw   norm                1.00000      +/-  0.0
________________________________________________________________________
```

In addition to the components, each parameter can be referred to by its number (running sequentially from the first parameter of the first component to the last parameter of the last component).

We can use `show free` to just show the parameters that will be allowed to vary during the fit:

```
XSPEC12> show free

Free parameters defined:
========================================================================
Model TBabs<1>*powerlaw<2>  Source No.: 1   Active/On
Model Model Component  Parameter  Unit     Value
 par  comp
   2    2   powerlaw   PhoIndex            1.00000      +/-  0.0
   3    2   powerlaw   norm                1.00000      +/-  0.0
________________________________________________________________________
```

Running `show free` just before `fit` is a useful way to check the model is set up the way you expect it to be and that the expected parameters will be allowed to vary.

6.5 Modifying parameters

We can change the value of a parameter using the `newpar` command (or `new` for short). If we enter `newpar` followed by the parameter number, we will be prompted to enter its new set of values:

```
XSPEC12> newpar 1
Current value, delta, min, bot, top, and max values
       0.04         -1(    0.0004)          0          0     100000      1e+06
1:TBabs[1]:nH:1>
```

We can also enter the new values for the parameter on the same line to change a parameter value with a single command. We can change the value, delta and range of the parameter using the same syntax as for the `newpar` command. For example, to change the value of `nH` (parameter number 1) to 0.033, we can directly type:

```
XSPEC12> newpar 1 0.033
```

We can freeze a parameter if we do not want it to vary during the fit:

```
XSPEC12> freeze 1
```

And then make it variable again with the `thaw` command:

```
XSPEC12> thaw 1
```

6.6 Tying parameter values

It is possible to tie the values of parameters together so that they are free during the fit, but they always take the same value (for example if we have a power law continuum spectrum and its corresponding reflection spectrum and we want to tie the photon index between the primary continuum and its reflection).

We can tie the value of parameter number 6 to the value of parameter number 2:

```
XSPEC12> new 6 = 2
```

We can untie them again using:

```
XSPEC12> untie 6
```

Take care setting the limits of tied parameters to make sure the values are limited to the more restricted of the two models, otherwise errors can occur during the fit if one of the models is pushed outside its allowed parameter range.

6.7 Performing the fit

Now that we have loaded the spectrum, selected the fit statistic and defined the model, we can perform the fit by issuing the `fit` command:

```
    XSPEC12> fit
```

The fit continues until the minimisation of the fit statistic is deemed to have converged. This is based upon a critical delta (*i.e.* the fit has converged when the fit statistic drops by less than this between steps). However, by default, the fit will pause after a fixed number of steps (10 by default) and ask us if we want to continue. Before we run `fit`, we can preempt this question and automatically answer yes to effectively remove the step limit with the command

```
    XSPEC12> query yes
    XSPEC12> fit
```

We can also specify the number of steps before XSPEC pauses and asks us if we want to continue. To run for 1000 steps before prompting to continue:

```
    XSPEC12> fit 1000
```

And we can specify the critical delta. To run for up to 1000 steps until the fit statistic drops by less than 10^{-3} between steps:

```
    XSPEC12> fit 1000 1e-3
```

At each step of the fit, XSPEC will print the current value of the fit statistic, a measure of the convergence (`|beta|/N`, which is the magbitude of the vector of the derivative of the fit statistic with respect to each parameter, and goes to zero at the minimum), and the value of each parameter, so you can see how the fit is proceeding. Then when the fit has finished, a summary will be shown including the covariance matrix, the best-fitting value of each parameter and the minimum value of the fit statistic that was found:

```
    XSPEC12> fit
                                       Parameters
    C-Statistic  |beta|/N    Lvl    2:PhoIndex        3:norm
    1.88119e+06  1.55301e+08  -3       6.69200       116.584
...
    1159.6       47.4721      -5       1.61747    0.00121556
    ==============================
     Variances and Principal Axes
                     2        3
     4.2696E-11| -0.0025   1.0000
     1.0192E-04|  1.0000   0.0025
    ------------------------------

    ========================
      Covariance Matrix
            1           2
       1.019e-04   2.540e-07
       2.540e-07   6.756e-10
    ------------------------

    ========================================================================
    Model TBabs<1>*powerlaw<2> Source No.: 1   Active/On
    Model Model Component  Parameter  Unit     Value
     par  comp
       1    1   TBabs      nH         10^22    3.30000E-02  frozen
       2    2   powerlaw   PhoIndex            1.61747      +/-  1.00957E-02
       3    2   powerlaw   norm                1.21556E-03  +/-  2.59918E-05
```

```
________________________________________________________________________


Fit statistic  : C-Statistic                1159.60     using 575 bins.

Test statistic : Chi-Squared                1162.05     using 575 bins.
 Null hypothesis probability of 2.64e-42 with 573 degrees of freedom
```

We are shown the values of two statistics. The fit statistic is the statistic that was actually minimised during the fit (here, this is the C-statistic). We are also shown a test statistic, chi-squared here, which is not itself minimised, but is evaluated using the best-fitting parameters at the end of the fit (so as to evaluate the goodness of fit using a statistic we are not explicitly minimising. We can recall the fit statistics at any time with the command `show fit`.

6.8 Evaluating the goodness of fit

We can use the statistics shown at the end of the fit to evaluate how well the model describes the data. When the data are perfectly described by the model, the only deviation between the observed values and the true values in each spectral bin will be due to the statistical errors. For Gaussian-distributed data, this means that the contribution to the χ^2 statistic will be ~ 1 for each degree of freedom, hence for ν degrees of freedom, $\chi^2/\nu = 1.0$ (similarly, due to the asymptotic behaviour of the modified C-statistic, $\chi^2/\nu \sim 1.0$). As a rule of thumb, the fit is poor if $\chi^2/\nu > 1.2$, and is reasonable if $\chi^2/\nu < 1.05$. In this case, $\chi^2/\nu > 1162/573 = 2.02$ and the model is a poor description of the data.

More formally, to evaluate the goodness of fit, we wish to evaluate the null hypothesis probability that the deviation between the model and the observed data points is entirely due to statistical fluctuations in the count rate in each bin. If the model accurately describes the data, the measured χ^2 statistic will be a random variable drawn from a χ^2 distribution for the appropriate number of degrees of freedom. We may therefore look up the probability of obtaining a χ^2 as large as the one we have calculated if the model were the correct description of the data. XSPEC gives us this p value of obtaining our value of the statistic if the model were correct, and we can see that in this case we may reject the null hypothesis that the model describes the data at very high probability $(1 - p)$. This is referred to as *Pearson's goodness of fit test*.

For an in-depth discussion of the statistics associated with spectral fitting, evaluating goodness of fit and model comparison, I recommend Chapter 7 of "Handbook of X-ray Astronomy" by Arnaud, Smith and Siemiginowska, Cambridge University Press 2011.

Tabulated p-values to assess the goodness of fit represented by different values of the fit statistic are available for χ^2. The modified C-statistic is defined such that in the limit of a large number of counts, its asymptotes to χ^2, so we can use the equivalent χ^2 values as a *rough guide*. Formal estimation of the goodness of fit using the C-statistic requires running a simulation to compute the distribution of the statistic for the specific data set being analysed (see "On the use of C-stat in testing models for X-ray spectra", J. S. Kaastra, 2017, AA 605, A51).

6.9 Plotting the results of this first fit

Once we have a model, XSPEC will plot this on top of the data points

```
XSPEC12> plot ldata
```

It is also helpful to add a second panel to the plot, showing the ratio between the data and the model as a function of energy (Fig. 3.

```
XSPEC12> plot ldata ratio
```

This helps us visualise where the model fails to describe the data and to see the shape of any additional components we may be missing from our model.

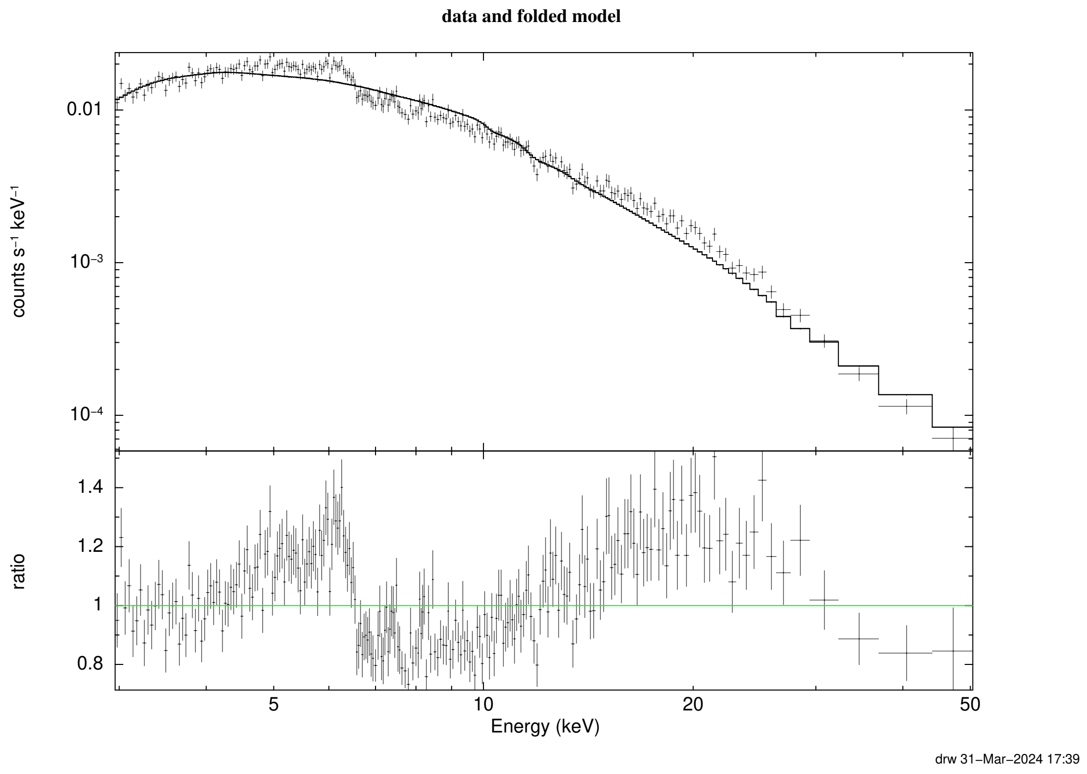

Figure 3: The *NuSTAR* spectrum of Mrk 335 fit by a power law model. When a model is defined, the model will automatically be shown on plots alongside the data. The lower panel shows the ratio of the data/model in each spectral bin. Such a plot is created using `plot ldata ratio`. In this case we can see that the model does not provide a good fit to the data (data/model should be a flat line at approximately 1.0, within its error bar, in each bin). In this case, the ratio plot looks like a reflection spectrum, with a broad emission line around 6.4 keV (the energy of the iron K line) and a Compton hump peaking at 20 keV.

In this case, the ratio plot looks like a reflection spectrum, with a broad emission line around 6.4 keV (the energy of the iron K line) and a Compton hump peaking at 20 keV. This indicates that we have missed the reflection from the inner accretion disc from our model of the spectrum of this AGN.

As an alternative to `ratio`, we can also plot `delchi`, which is the difference between the data and the model divided by the error (*i.e.* the contribution to the χ^2 statistic) at each energy. We can also plot `resid` which is simply the residual (the differenc between data and model).

6.10 Editing the model

After running this first fit and realising the component(s) we are missing, we can modify our model in a number of ways.

We can, of course, issue the `model` command again with a new model expression, however this will delete the current model (including the best-fitting values that have been found for the parameters) and start again with a new set of values for every parameter.

The most comprehensive way to edit a model is the `editmod` command. `editmod` is passed a new model expression and will compare it to the current model expression to work out which component is to changed. The other components will be left intact with the same parameter values. Note, however, that `editmod` can only change one component at a time.

Model 2: X-ray continuum and reflection from the accretion disc

In this case, the residuals to the power law fit suggest that we may want to replace the POWERLAW with the RELXILL model for reflection from the inner accretion disc (ordinarily, we would add the reflection component to our power law model, though here we replace the power law with RELXILL since RELXILL contains both the reflection and primary continuum emission):

```
XSPEC12> editmod tbabs*relxill
```

Since TBABS is already in our model, this component will remain and will keep its parameter values from the previous fit. The power law is replaced by RELXILL and we are prompted for values of each of the RELXILL parameters.

The summary of our new model:

```
========================================================================
Model TBabs<1>*relxill<2> Source No.: 1   Active/On
Model Model Component  Parameter  Unit     Value
 par  comp
   1    1   TBabs      nH         10^22    3.30000E-02  +/-  0.0
   2    2   relxill    Index1              3.00000      frozen
   3    2   relxill    Index2              3.00000      frozen
   4    2   relxill    Rbr                 15.0000      frozen
   5    2   relxill    a                   0.998000     +/-  0.0
   6    2   relxill    Incl       deg      30.0000      +/-  0.0
   7    2   relxill    Rin                 -1.00000     frozen
   8    2   relxill    Rout                400.000      frozen
   9    2   relxill    z                   2.50000E-02  frozen
  10    2   relxill    gamma               2.00000      +/-  0.0
  11    2   relxill    logxi               3.10000      +/-  0.0
  12    2   relxill    Afe                 1.00000      +/-  0.0
  13    2   relxill    Ecut       keV      300.000      frozen
  14    2   relxill    refl_frac           3.00000      +/-  0.0
  15    2   relxill    norm                1.00000      +/-  0.0
________________________________________________________________________
```

Note that some of the parameters that we will, in general, wish to fit are frozen by default. Notably here, `Incl`, the inclination of the accretion disc to the line of sight, and `Index1`, `Index2` and `Rbr`, describing the emissivity profile of the accretion disc (these parameters define a broken power law for the variation in reflected flux as a function of radius on the disc). It is alwyas a good idea to check the output of `show free` before running `fit` to ensure that the parameters you wish to fit (and only those parameters) are free. In this case, hwowever, since these parameters have a more subtle effect on the shape of the spectrum, it can be a good idea to run an initial fit with these parameters frozen at their defautl values, to estimate the values of the other parameters, then `thaw` them and run the fit again.

6.10.1 Adding and deleting model components

We can also add a component to a model. For example, we may want to add the *relxill* reflection model to our existing model. To do this, we can use the `addcomp` command (or `add` for short), specifying the numbered position we wish to place the new component.

We currently have two components, `tbabs<1>*powerlaw<2>`, so to add a component to the end, we will place it in position 3:

```
XSPEC12> addcomp 3 relxill
```

However, we need to be careful with brackets in our model expression. This command will produce `tbabs<1>*powerlaw<2> + relxill<3>`, which is incorrect, since the TBABS absorption is applied only to the POWERLAW, not to RELXILL. Instead, if we place the new component at position 2, XSPEC will place it in brackets with the absorption:

```
XSPEC12> addcomp 2 relxill
```

Now we have `tbabs<1>*(relxill<2> + powerlaw<3>).`

As mentioned above, we don't need the POWERLAW here, since RELXILL also includes the primary continuum emission. We can delete a component using the `delcomp` command (`del` for short), so to remove the POWERLAW:

```
XSPEC12> delcomp 3
```

6.11 Does our new model provide a better fit to the data?

To determine whether our new model provides a better description of the data, we look at the minimised value of the fit statistic at the end of the fit (or alternatively the test statistic), as well as the plot of the residuals between the data and the model as a function of energy.

The new model, using RELXILL in place of the simple POWERLAW reduces the χ^2 statistic from 1162 for 573 degrees of freedom, to 595 for 567 degrees of freedom, *i.e.* $\Delta\chi^2$ = -567 for 6 fewer degrees of freedom. This is a very significant improvement! We also see that the *reduced* χ^2, that is χ^2 divided by the number of degrees of freedom, χ^2/ν, has dropped to 1.05, indicating a reasonable fit, and the null hypothesis probability (that we have obtained a χ^2 value this high purely from statistical fluctuations causing the data points to deviate from the model values) is 0.19.

When comparing models, however, we should employ *Occam's razor*. That is to say that given two models that seem to describe the data equally well, we should always prefer the simpler one. This is where the degrees of freedom come in. The degrees of freedom is equal to the number of data points we are fitting minus the number of free parameters in our model. We therefore always quote the reduction in the fit statistic, be that χ^2 or C for N fewer degrees of freedom, or N additional free parameters. A model with as many free parameters as we have data points will almost always be able to reproduce the data — this does not mean that it is the correct or even a useful description! The preferred model will be the one that provides the greatest reduction in the fit statistic for the least number of additional free parameters. In this case, our new model has given us $\Delta\chi^2$ = -567 for only 6 fewer degrees of freedom or 6 additional free parameters, so is a significant improvement. If the reduction in χ^2 were similar to the number of additional free parameters, the improvement would not be significant and we would not prefer that new model.

There is also such a thing as too low a χ^2 value! Given that the χ^2 statistic is derived from the likelihood of data points x_i distributed around their model values m_i according to a Gaussian distribution with standard deviation according to their statistical errors σ_i, on average for each data point, $(x_i - m_i)^2/\sigma^2 \sim 1$, therefore by definition, if the model describes the true intrinsic values of the data points, $\chi^2/\nu \sim 1$. If the reduced χ^2 value, χ^2/ν is significantly less than one, there is a problem: either the model is over-fitting the data, or the statistical error bars are overestimated.

Inspecting the plot of the residuals between the data and the model, shown in Fig. 4 (either the `ratio` plot for the data/model ratio for each bin, or the `delchi` plot for the contribution to χ^2 for each bin), we can further evaluate the goodness of fit. If the model provides a good description of the data, the data/model ratio points will be scattered around 1.0 according to their error bars (or the $\Delta\chi^2$ points will be scattered around 0). If the fit is good, the majority of points should be around 1.0 within approximately the length of their error bars, though remember that at least for Gaussian distributed random errors, around 32% of points will be more than 1σ (*i.e.* 1x the length of the error bar) away from the mean and 5%, or 5 points out of every 100, will be more than 2σ from the mean. If the deviations are due to random errors, the scatter should be random, however look for structure in the residuals. If multiple, consecutive points in the ratio plot are on one side of the 1.0 line, this could indicate an additional emission feature, such as a narrow emission line, or if there is a trend between consecutive points, there could be a component missing from the model.

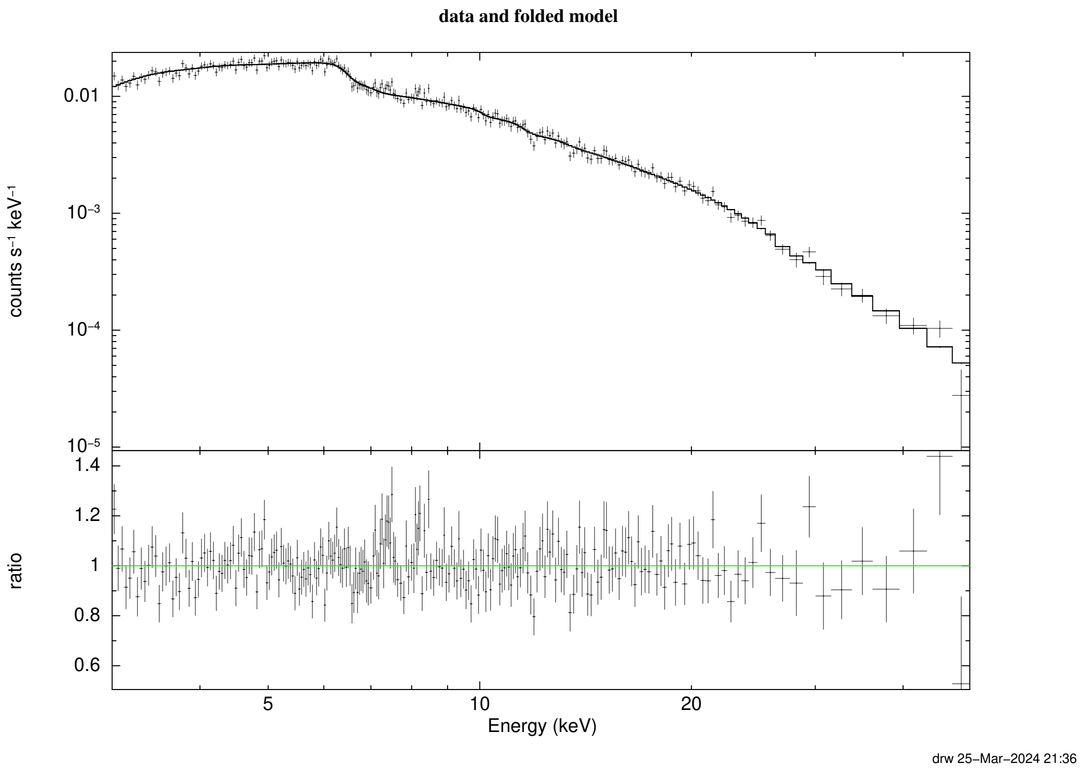

Figure 4: The RELXILL model, describing continuum emission from the corona and its reflection from the inner accretion disc, provides a much better description of the spectrum ($C/\nu = 595/567$. The ratio plot shows almost a flat line, with data/model ~ 1 across most of the spectrum, however residuals around 7 keV resembling a narrow emission line indicate the presence of a narrow component of the iron K line. When we have a good fit to the data, there will be no clear structures within the residuals.

Model 3: X-ray continuum, reflection from the accretion disc and distant material

In the case of our spectrum from Mrk 335 we see residuals around 6.4 keV in the spectrum, suggesting the presence of a narrow component of the iron K emission line produced by reflection from more distant material (in addition to the broad component that is included in our model for relativistically blurred reflection from the inner disc). We therefore may wish to add an additional component.

We could choose just to add a single, narrow emission line with a Gaussian profile using the `zgauss` model:

```
    XSPEC12> editmod tbabs*(relxill + zgauss)
```

In this case, we set the hard limits of the (rest-frame) line energy between 6.4 and 6.97 keV (to ensure that this is a Kα line from neutral, helium-like or hydrogen-like iron), and we freeze the line width (parameterised by σ, the standard deviation of the Gaussian profile) to 0.1, since the line is narrow and unresolved by our detector, so we just freeze the line width to something below our instrumental resolution.

```
    Input parameter value, delta, min, bot, top, and max values for ...
                6.5       0.05(     0.065)          0          0      1e+06      1e+06
    5:zgauss:LineE> 6.4 0.05 6.4 6.4 6.97 6.97
                0.1       0.05(     0.001)          0          0         10         20
    6:zgauss:Sigma> 0.1 -1
                  0      -0.01(      0.01)     -0.999     -0.999         10         10
    7:zgauss:Redshift> 0.025
                  1       0.01(      0.01)          0          0      1e+20      1e+24
    8:zgauss:norm> 1e-6
```

It's best to start the normalisation of the new line at a small value like 10^{-6} to let the fit add the line to our current best-fitting spectrum, rather than add a very large line component that will distort the spectrum and try to fit that.

Including this narrow emission line reduces the C statistic by 11 for just two additional free parameters, thus the new model including the line provides a significantly better description of the data.

Alternatively, if this narrow iron K line is coming from reflection from cold, dense material at a relatively large distance from the black hole, we could instead choose to model it with a self-consistent reflection spectrum using XILLVER:

```
    XSPEC12> editmod tbabs*(relxill + xillver)
```

6.12 Model selection

A common question we may wish to ask of our data is which of a selection of models provides the best description. Formally, we would like to select the model that has the maximum probability of producing the data we have obtained (*i.e.* the model that gives the minimum fit statistic, for the minimum number of free parameters). If model B produces a lower fit statistic than model A, remembering that the fit statistic we obtain is a random variable, drawn from a probability distribution (namely the χ^2 distribution if we are using the χ^2 statistic), we will want to quantify the probability that model A could have given us an equally low value just by random fluctuations. We therefore test the null hypothesis that models A and B provide equally good fits to the data.

One way to do this is the *likelihood ratio test*, also known as the *F-test*, which computes such a p-value strictly for the case of *nested models*. Two models are nested if the entire parameter space of model A lies within the parameter space of model B, for example if model A is the same as model B, just with some of the parameters frozen. Models would also be considered nested if model B is model A with an additional component added (since we could recover model A if we set the normalisation of the new component to zero). In our example, we could consider all three of our models, `tbabs*powerlaw`, `tbabs*relxill` and `tbabs*(relxill + zgauss)` to be nested, but only because RELXILL contains the disc reflection spectrum in addition to the continuum, and the continuum in RELXILL is the same as POWERLAW.

XSPEC can calculate the F-statistic from the chi^2 values and number of degrees of freedom for the two models. For example, comparing our `tbabs*powerlaw` model ($chi^2 = 1162$ for 573 degrees of freedom) to our `tbabs*relxill` model ($chi^2 = 595$ for 567 degrees of freedom):

```
XSPEC12> ftest 595 567 1162 573
 F statistic value = 90.0529 and probability 3.7865e-79
```

As expected, the F statistic is very large, and we can reject the null hypothesis that the RELXILL model provides a comparable fit to the POWERLAW model, and the fit statistics only varied by random chance, to a very high degree of certainty!

Note that we must be using the χ^2 statistic for an F-test. Note also that the F-test cannot be used to test for the presence of narrow line components (see Prossatov et al. 2002, ApJ 571, 545).

One drawback of the F-test is that it only compares the two models via the single value of the fit statistic at the point of best fit in the parameter space. In reality, there will be some uncertainty in the best-fitting values of each of the parameters (see *error estimation* below), so we may wish to compare the models via their goodness of fit over the whole volume of the parameter space admitted by the data. Alternatives include computing the ratio of the *Bayesian evidence* for each model (which is essentially the probability of obtaining the data integrated over the entire parameter space of each model). We can also compute an *information criterion*, which characterises how much information present in the data is lost by the model. Convenient information criteria for model selection include the *Akaike information criterion (AIC)* and the *deviance information criterion (DIC)*. Note that these statistics do not quantify the overall goodness of fit, only the relative improvement between models.

7 Error estimation

When we use the best-fitting values of our model parameters to infer the physical parameters of the source we are observing, there will naturally be some uncertainty in this inference. In the context of spectral fitting, that is to say that there will be a range of values that provide an equally good fit to the data. There will also be a range of values over which the fit statistic, χ^2 or C, increases by only a small amount. Remembering that our fit statistic is itself a random variable drawn from a distribution according to the statistical errors on our data points, the increase in the fit statistic from slightly changing the value of a parameter may be within the bounds of the random fluctuations we naturally expect of that fit statistic, so again would say that this range of values for the parameter provide an equally good description of the data. We therefore wish to define the *confidence limits* or *error bars* on the values of the parameters: the range of parameter values that are consistent with the data.

In general, the uncertainty on a parameter will be greater if either we have lower signal-to-noise data (larger statistical error bars on each spectral data point), or the effect of that parameter on the shape of the spectrum in the band we are observing is more subtle. It is also possible that varying two different parameters will have the same effect on the shape of the spectrum. In this case, the parameters are said to be degenerate with one another, there will be *covariances* between them (as you vary one, you will need to vary the other to obtain an equally good fit) and their uncertainties will be intertwined.

We often discuss parameter uncertainties or confidence limits in the context of the *posterior probability*, that is the probability of the model parameter values given the data we have obtained, $P(\mathrm{model} \mid \mathrm{data})$. This is related to the likelihood function, $P(\mathrm{data} \mid \mathrm{model})$ by *Bayes' Theorem*. If we have no prior knowledge about these parameter values (*i.e.* $P(\mathrm{model})$ is uniform and all values of the parameter are considered to be equally likely), the posterior probability and likelihood function are interchangeable. Given that the fit statistics we use in XSPEC are derived from the likelihood function, we shall make this assumption here.

We often refer to the 1σ (and more generally $n\sigma$) confidence limits, as well as, historically the 90% confidence limit. Assuming the posterior probability distribution of the parameter values to be Gaussian, the 1σ confidence interval is the range of values within one standard deviation of the mean (within which 68.2% of the distribution lies). That is to say that the true value of the parameter will lie within that range 68.2% of the time. 95.5% of the distribution lies within 2σ of the mean, 99.7% within 3σ and 99.9999% within 5σ. The parameter space of an n parameter model will be N-dimensional, and there is a posterior probability associated with every combination of parameter values. When we refer to the posterior of a single parameter, we are *marginalising* over all other parameters. That is to say we are integrating the probability of one parameter having one particular value over all possible values of the other parameters.

The peak of the posterior distribution lies at the best-fitting parameter value, for which the fit statistic is minimised. From the properties of the χ^2 distribution, the 1σ limits correspond to variation in parameter value (in each direction) which leads to an increase of the χ^2 statistic (that is the *total* χ^2, not the reduced χ^2) by 1.0. The 2σ, 3σ and 4σ limits correspond to increases in χ^2 of 4.0, 9.0 and 16.0, respectively. The 90% confidence limit corresponds to an increase in χ^2 of 2.706.

Note that these values are formally correct when using the χ^2 statistic. The asymptotic behaviour of the modified C-statistic used in XSPEC *means that approximately the same values can be used, however formal estimation of confidence limits requires simulations to compute the distribution of the C-statistic for the given data set (or, alternatively, MCMC calculations to directly compute the probability distributions of the parameters, see below). We could instead use the χ^2 statistic, however this would require us to be confident that our uncertainty in every spectral bin is Gaussian, in the limit of a large numebr of counts.*

The parameter table shown at the end of a fit shows an *estimate* of the errors or uncertainties associated with each parameter in the right-most column. These error bars are the 1σ confidence limit derived from the diagonal terms of the Fisher information matrix. This is calculated from the second derivative of the fit statistic with respect to the parameters, and essentially assumes that the likelihood is perfectly Gaussian, thus the shape of the fit statistic as a function of the parameter value is quadratic. This estimate of the errors neglects covariances and degeneracies between parameters, thus will often *underestimate* the true uncertainties. While these error values should never be used in the final inference of parameter values, they can be a useful indication of which parameters are well-constrained by the data. Deriving an accurate estimate of the confidence limits and uncertainties for each parameter requires exploration of the fit statistic across the parameter space using the methods discussed below.

7.1 Stepping through parameter values

A conceptually simple and relatively reliable method of estimating error bars and confidence limits for a parameter is to step through values of that parameter, find the best possible fit at each value (*i.e.* find the minimum possible fit statistic with that value, fitting for the value of all of the other parameters), then find the value of that parameter for which the χ^2 statistic has increased by Δ from the minimum value at the best fit (where Δ is the desired threshold for the 1σ, 90% or other confidence limit). Strictly speaking, the thresholds are defined for the χ^2 statistic, however the C-statistic behaves in approximately the same way due to its asymptotic behaviour. By fitting for all of the other parameters at each step, this procedure is implicitly marginalising over all other parameters.

The `steppar` command in XSPEC is used to step through the values of a parameter in this way.

For example, in our RELXILL model, we may wish to step through values of the photon index parameter (parameter number 10) and find the minimum possible χ^2 or C-statistic at each value. We specify the beginning and end values as well as the number of steps. To step parameter number 10 from 1.8 to 2.5 in 20 linearly spaced steps, the command would be:

```
    XSPEC12> steppar 10 1.8 2.5 20
```

XSPEC will then output a table containing the minimum fit statistic at each step, the delta fit statistic (*i.e.* the difference between the current value and the starting value, which should be the best fit), the step number, and the value of the parameter at this step.

```
        C-Statistic    Delta            gamma
                     C-Statistic            10

              602.79       8.3877    0         1.8
              600.05       5.6511    1       1.835
...
```

We can then plot the (one-dimensional) 'contour' of the fit statistic as a function of the parameter value (Fig. 5):

```
    XSPEC12> plot contour
```

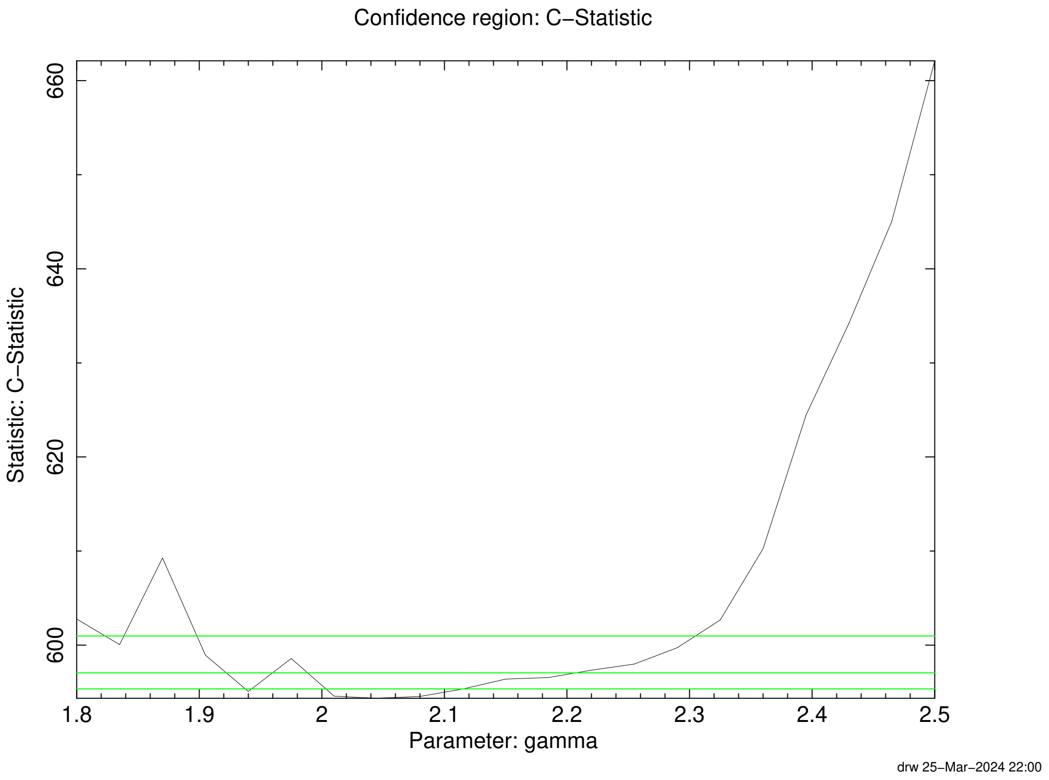

Figure 5: Plot of the minimum C-statistic found stepping through values of `gamma` parameter obtained using the `steppar` command followed by `plot contour`. The green lines show the 1, 2 and 3σ confidence limits with respect to the best fit (the position of minimum fit statistic).

Horizontal lines are drawn on the one-dimensional contour plot indicating the $\Delta\chi^2 = 1$ and 2.706 thresholds, which intersect the curve at the 1σ and 90% confidence intervals.

For some parameters with a large dynamic range (for example a normalisation or temperature), we may wish to step logarithmically through the parameter space instead of linearly. We can do this by specifying `steppar log`:

```
    XSPEC12> steppar log 15 1e-8 1e-4 20
```

Note, however, that running one `steppar` with the `log` option will cause all subsequent `steppar`s to default to logarithmic spacing. To revert to linear spacing, use `steppar nolog`.

`steppar` can also step through two parameters to explore the two-dimensional space between them. For example, we can step over both the photon index (parameter 10, from 1.8 to 2.5 in 20 steps) and normalisation (parameter 15 from 1e-6 to 1e-4 also in 20 steps):

```
    XSPEC12> steppar 10 1.8 2.5 20 15 1e-6 1e-4 20
```

The `plot contour` command will then plot the two-dimensional contours between these two parameters (Fig. 6):

```
    XSPEC12> plot contour
```

(Use `setplot nocontimage` to turn off the shading of probability densities in the background).

In addition to visualising the uncertainty in each of these parameters, we can assess if the parameters are at all degenerate and their values depend on one another. If the parameters are independent, the contours will be circular or elliptical, elongated in either the horizontal or vertical directions. If they are degenerate, the contours will be tilted. If the contours appear jagged, you may need to run `steppar` with more steps. Alternatively, MCMC can be an efficient way to generate contour plots between parameters (see below).

Note that the delta statistics in the `steppar` output and corresponding lines on the plots are calculated with respect to the current fit value, therefore `fit` always needs to be run before `steppar`. If you have saved and reloaded the parameters, `fit` will need to be run again, even if the best fit was saved.

7.2 Automated error searches

The `error` command in XSPEC automates the above procedure to calculate the error bar or confidence limit for a parameter. This command will step through the parameter in each direction until the change in fit statistic reaches the desired threshold, then will output the corresponding parameter values as the upper and lower limits.

To calculate the confidence limit, specify the desired delta statistic (as a floating point number, not as an integer, *i.e.* 1.0 for 1σ, not 1, 2.706 for 90%) then the parameter numbers you wish to calculate the error bars for. For example, to calculate the 1σ error bar on parameters 10, 14 and 15:

```
    XSPEC12> error 1.0 10 14 15
```

The confidence limits for the three parameters will be printed in sequence:

```
     Parameter   Confidence Range (1)
        10      1.98452        2.116    (-0.0627728,0.0687134)
...
```

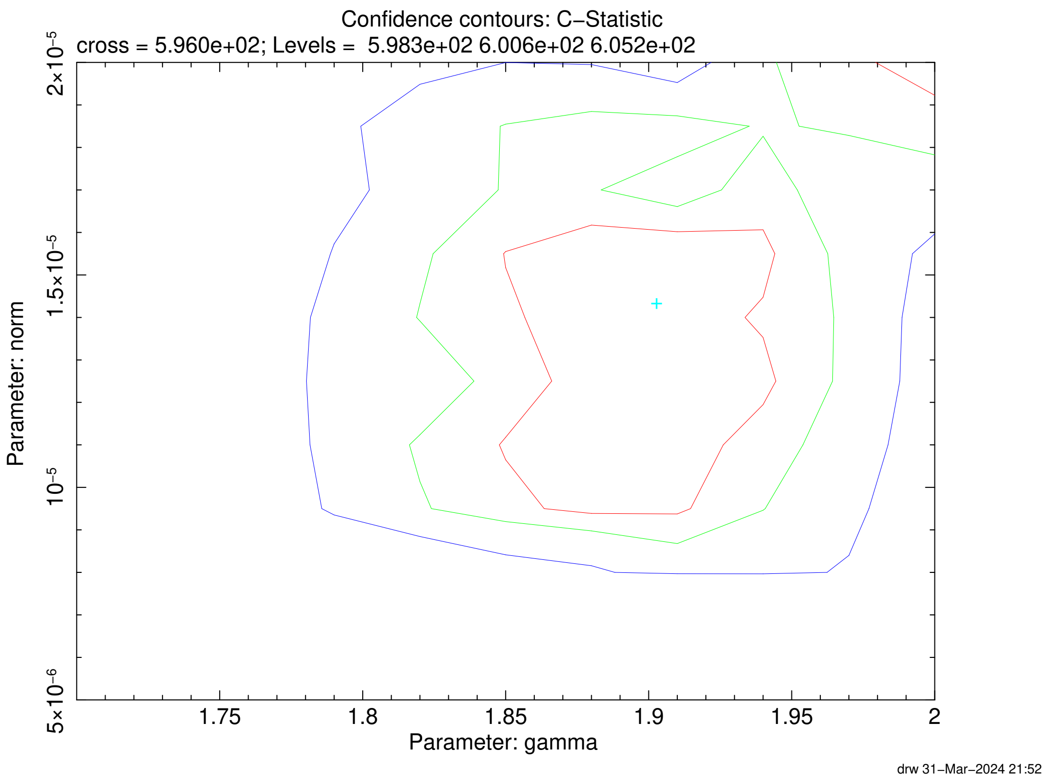

Figure 6: Contour plot created using `steppar` to step through the values of two parameters, followed by running `plot contour`.

7.2.1 Local minima

One common pitfall of maximum likelihood fitting is that the minimisation algorithm will stop at a local minimum of the fit statistic (around which the statistic locally increases), but not at the global minimum, and there is a better fit at another location in the multi-dimensional parameter space. `steppar` and `error` searches can often bring these local minima to light. If a `steppar` is started at a local minimum, you may find a range of parameters for which the delta fit statistic is negative (*i.e.* the statistic is lower than it was at the starting point, which was previously assumed to be the best fit). If this happens, XSPEC will ask if you want to re-run the fit around this local minimum. You should, and you should use the new parameter values found at this new minimum. Error searches should also be re-run, using delta statistics relative to the new minimum.

If XSPEC finds a new minimum during the `error` command, it will ask if you would like to restart the error search around this new best fit. Again, you should. XSPEC may sometimes find new minima many times during an error search and will ask this question each time. Here, it can be useful to have set `query yes` so that XSPEC automatically restarts the error search until the global minimum is found.

7.3 Markov chain Monte Carlo (MCMC) error searches

Markov chain Monte Carlo (MCMC) algorithms provide a means to explore the likelihood function across a multi-dimensional parameter space, and to simultaneously estimate the confidence limits of all of the free parameters. In essence, an MCMC algorithm consists of a number of 'walkers' that take random steps around the parameter space, evaluating the likelihood at each point they land, with the steps forming a *chain*. The probability of a walker stepping from one location to another depends on the relative value of the likelihood between those two points. At the beginning, the walkers will step towards the position of maximum likelihood (*i.e. the minimum fit statistic*), but then once the chain has converged to its steady state, the distribution of the parameter values associated with each of the random steps the walkers take will reflect the underlying probability distribution associated with those parameters (*i.e.* the chain distribution will reflect the likelihood, or the posterior probabilities modulo the priors).

The probability distributions associated with each parameter can be computed and visualised by constructing a histogram of the value of that parameter at each chain step, once the chain has converged its steady state. Because the MCMC algorithm is sampling the probability distribution across the multi-dimensional probability space, it is straightforward to marginalise uncertainties over other parameters, to account for degeneracies between parameters and to plot contours between parameters. Note, however, that it is important that the chain has *converged*. Therefore, it is common practice to discard or 'burn' the first several steps of the chain.

XSPEC implements MCMC calculations using the Goodman-Weare algorithm (also known as the affine sampler, or EMCEE) by default, or alternatively the classic Metropolis-Hastings algorithm. For most purposes, the default Goodman-Weare algorithm will be sufficient, and is typically more robust to complex or degenerate parameter spaces than Metropolis-Hastings (though is not perfect).

The MCMC calculation is set up using the `chain` command (after we have run the `fit`). First we specify that we wish to use the Goodman-Weare algorithm, `gw` (which would be the default if we didn't specify this) and how many walkers we wish to have in our chain. We should have significantly more walkers than we have free parameters in our model, so here we will use 50:

```
XSPEC12> chain type gw
XSPEC12> chain walkers 50
```

Then we specify how long a chain we wish to run, for example 10,000 steps (per walker):

```
XSPEC12> chain length 10000
```

Then we specify how many steps we wish to discard or burn at the beginning to make sure the chain we store and work with has converged. For example to discard 1000 steps:

```
XSPEC12> chain burn 1000
```

We can now run the chain, specifying the name of the FITS file we wish to save it to:

```
XSPEC12> chain run mychain.fits
```

Once the chain has finished running, it will automatically be loaded. The chain will also be saved into a FITS file containing a table of the parameter values, as well as the fit statistic, at each step. We can also reload a previously saved chain, which can be useful if we want to come back to an analysis, or if we want to run the long chain calculation on another computer (*e.g.* a HPC facility):

```
XSPEC12> chain load mychain.fits
```

Once a chain is loaded, it will automatically be used for error calculations, so to calculate the 1σ confidence limits on parameter number 10, we once again run:

```
    XSPEC12> error 1.0 10
```

However now this command will run much more quickly, since XSPEC does not need to perform the error search and can just compute the confidence limits from the percentiles of the points within the chain.

To plot marginalised probability distributions for individual parameters as well as contours between pairs of parameters, we use the `margin` command in place of `steppar`. `margin` takes the same inputs as `steppar`, except it performs the calculations from the chain instead of by fitting the spectrum at each point.

As before, we can obtain the one-dimensional marginalised distribution for parameter number 10 from 1.8 to 2.5 in 20 linearly spaced steps (we could also specify `log`, as for `steppar`) using:

```
    XSPEC12> margin 10 1.8 2.5 20
```

And we can obtain a two-dimensional distribution to create a contour plot between parameter 10, from 1.8 to 2.5 in 20 steps, and parameter 15 from 1e-6 to 1e-4 also in 20 steps):

```
    XSPEC12> margin 10 1.8 2.5 20 15 1e-6 1e-4 20
```

Once the `margin` calculation has completed, we can plot the distribution or contours using:

```
    XSPEC12> plot margin
```

For more detailed analyses, you can read the FITS table containing the chain into another piece of software, for example PYTHON to create corner plots using the CORNER package.

7.3.1 Has the chain converged?

As noted above, before performing any analysis on distributions obtained from an MCMC calculation, it is important to check that the chain has converged to its steady state, such that the chain steps reflect the underlying probability distribution. There are a number of tests we can perform to determine whether the chain has converged.

We can plot the value of a single parameter at each step along the chain, for example to plot parameter 10:

```
    XSPEC12> plot chain 10
```

If the chain is long, it might be somewhat intensive to plot all the chain points, so we can thin the chain, only plotting every nth point (for example every 5th point):

```
    XSPEC12> plot chain thin 5 10
```

If the chain has converged, we should not see a systematic trend in the value of a parameter. The value should more randomly around the best-fit value. If the chain has not converged, we will see the value of a parameter moves towards the peak of the distribution.

We can also calculate convergence statistics for the chain. XSPEC will show these statistics for a chosen parameter using the `chain stat` command, *e.g.* for parameter 10:

```
    XSPEC12> chain stat 10
```

One such convergence statistic is the Geweke statistic, which measures the difference in mean between the first 10% of the chain and the last 50%, divided by the standard error. If the chain has converged, the difference in the mean calculated from these two parts of the chain should be comparable to the standard error, hence the statistic should be around 1.

In searching the parameter space, it is possible that the MCMC calculation will find a better fit to the data than the fit we started at. We can find the parameter values at the best-fitting point of the chain (the point with the mininum fit statistic) using:

```
    XSPEC12> chain best
```

The best fit (*i.e.* the point of maximum likelihood) should coincide with the peaks of the distributions derived from the chain. If the best fit values are offset from the peaks of each parameter's distribution, the chain may not have converged.

It is also useful to check the error estimates derived from the chain to those derived from a `steppar` or classic `error` search (without the chain loaded). If the chain-derived confidence limits on a parameter are much narrower than those calculated using `steppar` or `error`, it is likely the chain has not converged.

If your chain has not converged, you should run it for longer, perhaps with a longer burn-in period. If a chain is loaded, `chain run` will continue the chain from the place it ended. If you want to add the new points to the end of the existing chain file, add > before the filename:

```
    XSPEC12> chain run >mychain.fits
```

7.3.2 Chain proposals

The starting point of the walkers in an MCMC calculation is referred to as the *chain proposal*. By default, XSPEC will start the walkers at random locations scattered around the best fit. The walkers will be scattered according to normal distributions in the parameter values with 1σ widths derived from the errors estimated for each parameter from the covariance matrix.

We may sometimes with to change the chain proposal. In particular, sometimes if the fit is found to be insensitivity to a parameter, XSPEC will freeze its value during the fit and, hence, not derive an uncertainty for it from the covariance matrix. Attempting to run the chain will then lead to an error along the lines of the covariance matrix having fewer elements than there are free parameters in the model. Instead, we can randomly scatter the walkers around the best fit taking $100\times$ the `delta` values for the parameters for the widths of the distributions, by specifying:

```
    XSPEC12> chain proposal gaussian deltas 100
```

The exact distributions chosen for the proposal is not important so long as they are wide enough that the walkers adequately explore the parameter space around the best fit (the chain just may take longer to converge if the distributions are very wide).

8 Simultaneously fitting multiple spectra

8.1 Spectra from different instruments

There are often times we wish to analyse the spectra of a source obtained by two different instruments, for example the separate FPMA and FPMB instruments on board *NuSTAR*, or a coordinated observing campaign between *NuSTAR* and *XMM-Newton*. If the spectra were obtained from the same source at the same time (or at least over a time period during which we would not have expected the source to change), we should be able to describe the spectra using the same model, with the same model parameters. By fitting the spectra simultaneously or jointly, we can combine the information they hold to provide the tightest possible constraints on each of the model parameters. Different instruments, however, will have different response functions and calibrations, therefore we cannot simply add their spectra together, rather we should simultaneously fit the spectra from each instrument using the same model, but folding the model through the specific response function for each instrument.

In XSPEC, we can load multiple spectra to fit them simultaneously, as described in §3. We could load the spectra into the same group to apply exactly the same model and parameter values to each. Unfortunately, the calibration between different instruments carries some degree of uncertainty, and there is often a systematic offset between them. We therefore often need to include an addition model component, a multiplicative constant, that we can fit to the data, to allow for such a calibration offset. We therefore need to load the data into different groups.

For example, we might want to load separately the spectra obtained from *NuSTAR*'s FPMA and FPMB detectors, each into its own group:

```
XSPEC12> data 1:1 src_FPMA.grp
XSPEC12> data 2:2 src_FPMB.grp
```

With multiple spectra loaded, we need to make sure we select the desired energy range for each. To ignore channels outside the 3 to 50 keV in all loaded spectra:

```
XSPEC12> ignore *:**-3.,50.-**
```

We could also explicitly set the energy range for each spectrum:

```
XSPEC12> ignore 1:**-3.,50.-** 2:**-3.,50.-**
```

We then set up our model. We will go back to modelling the spectrum with a simple power law (with Galactic absorption applied), though now we multiply the entire model by a constant (if you have more than one additive component, you will need to put those components in brackets and apply the constant to all of them):

```
XSPEC12> model constant*tbabs*powerlaw
```

When we have spectra loaded into multiple groups, XSPEC will apply the same model to each of them, but the spectra in each group will have different parameter values. As usual, after defining the model we will be prompted for the initial values of the parameters.

First up is the value of the constant for the first data group. Since we wish to define the constant as the relative offset between the two spectra, we will freeze it's value to 1.0 for the first spectrum (*i.e.* we set its value to 1 and its delta to -1):

```
Input parameter value, delta, min, bot, top, and max values for ...
              1       0.01(      0.01)          0          0      1e+10      1e+10
1:data group 1::constant:factor> 1 -1
```

We then get prompted for each of the parameters in turn. We freeze nH for the absorption model at 0.033 (with delta -1) and we just press ENTER to accept the default values for the POWERLAW parameters:

```
              1      0.001(      0.01)          0          0     100000      1e+06
2:data group 1::TBabs:nH> 0.033 -1
              1       0.01(      0.01)         -3         -2          9         10
3:data group 1::powerlaw:PhoIndex>
              1       0.01(      0.01)          0          0      1e+20      1e+24
4:data group 1::powerlaw:norm>
```

Once we have entered all of the parameters for data group 1, we will be asked for each parameter again for data group 2. For this second group, if we accept the default value by just pressing ENTER, each parameter value will be tied to the corresponding parameter in group 1. Importantly, we do not wish to tie the value of the value of the constant, since this represents the offset between data group 2 and 1, so we enter `1 0.01` to start the constant at 1.0 but keep it free (with delta 0.01):

```
    Input parameter value, delta, min, bot, top, and max values for ...
                  1         -1(      0.01)          0          0      1e+10      1e+10
    5:data group 2::constant:factor> 1 0.01
```

Now we are prompted for the remaining model parameters, and we will just press ENTER on each to have them tied to the values in group 1 (or we could just enter `/*` to accept the default value for all remaining inputs, but don't do this if you have further data groups, since you will need to set the constant for each of these too):

```
                  1      0.001(      0.01)          0          0     100000      1e+06
    6:data group 2::TBabs:nH> /*
```

When we're done, XSPEC will show us the parameter table (which we could also get using `show par`):

```
    Parameters defined:
    ========================================================================
    Model constant<1>*TBabs<2>*powerlaw<3> Source No.: 1   Active/On
    Model Model Component  Parameter  Unit     Value
     par  comp
                               Data group: 1
       1    1   constant   factor              1.00000      frozen
       2    2   TBabs      nH         10^22    3.30000E-02  frozen
       3    3   powerlaw   PhoIndex            1.00000      +/-  0.0
       4    3   powerlaw   norm                1.00000      +/-  0.0
                               Data group: 2
       5    1   constant   factor              1.00000      +/-  0.0
       6    2   TBabs      nH         10^22    3.30000E-02  = p2
       7    3   powerlaw   PhoIndex            1.00000      = p3
       8    3   powerlaw   norm                1.00000      = p4
    ________________________________________________________________________
```

We can now see how the model is set up for a simultaneous fit. Each data group has its own set of parameters, numbered sequentially from the first parameter of the first component for the first data group to the last parameter of the last component for the last data group. The CONSTANT is frozen to 1.0 for data group 1, and is a free parameter for each of the subsequent groups. Our Galactic absorption is frozen. Our POWERLAW parameters are free in data group 1, and their values in group 2 are tied to the corresponding parameters in group 1. If we needed to, we could tie parameters using `newpar x = y` and untie them with `untie x`, as described in §6.6.

Now we can run the fit as before:

```
    XSPEC12> fit
```

And when it finishes, we will see the results:

```
    ========================================================================
    Model constant<1>*TBabs<2>*powerlaw<3> Source No.: 1   Active/On
    Model Model Component  Parameter  Unit     Value
     par  comp
                               Data group: 1
       1    1   constant   factor              1.00000      frozen
       2    2   TBabs      nH         10^22    3.30000E-02  frozen
       3    3   powerlaw   PhoIndex            1.67457      +/-  1.18131E-02
       4    3   powerlaw   norm                1.50584E-03  +/-  3.83462E-05
                               Data group: 2
       5    1   constant   factor              1.02213      +/-  1.25340E-02
```

```
   6    2   TBabs      nH         10^22    3.30000E-02  = p2
   7    3   powerlaw   PhoIndex            1.67457      = p3
   8    3   powerlaw   norm                1.50584E-03  = p4
________________________________________________________________________


Fit statistic  : C-Statistic                1451.45     using 1174 bins, spectrum 1, group
1.
                 C-Statistic                1505.93     using 1174 bins, spectrum 2, group
2.
Total fit statistic                         2957.38     with 2345 d.o.f.

Test statistic : Chi-Squared                2627.50     using 2348 bins.

 Null hypothesis probability of 3.45e-05 with 2345 degrees of freedom
```

The best fitting value of the constant in data group 2 is 1.02, indicating approximately a 2% calibration offset between the overall normalisation of the FPMA and FPMB instruments here (this is normal). Note that including a constant in this way only accounts for a calibration offset in the overall normalisation of the spectrum, not in the shape of the spectrum (*e.g.* due to differences in the effective area or sensitivity as a function of energy). We could check for differences in the slope of the spectrum by untying the photon indices of the power law between the two data groups (`untie 7`), then fitting again to look for a significant difference in the value for the two spectra. It is usually sufficient, however, just to include the normalisation offset, and allowing other spectral parameters to vary can make the results somewhat harder to interpret.

XSPEC shows us the fit statistic for each of the individual spectra so that we can see how well the model describes each, in addition to the total fit statistic (*i.e.* the sum of the fit statistics for all the spectra). It is this summed fit statistic that is minimised during a simultaneous or joint fit.

Once we have our model set up to simultaneously fit multiple spectra, we can work with it in the same way we worked with a single spectrum, including running error searches and MCMC analyses.

8.2 Spectra from different times

When we have observed a variable source on multiple occasions, we may wish to simultaneously or jointly fit the spectrum obtained from each time period (whether obtained using the same instrument or different instruments). We can set up such a fit in exactly the same way as described above, loading the spectrum from each observation into its own data group. When we define the model, we will tie together the values of the parameters that should not have changed between the observations (for example the spin of a black hole, or the inclination at which we are viewing an AGN accretion disc), but will keep parameters untied if they can vary on the timescales between the observations (*e.g.* normalisations, representing source luminosity, photon indices of power law spectra). By fitting the model simultenously in this way and calculating the confidence limits of these parameters, we can determine which parameters of an astrophysical system vary significantly over time, and obtain the tightest possible constraints on the parameters that are tied by combining the information available from all of the individual spectra (contrary to fitting each of the spectra separately).

Note that when the normalisations are free to vary between observations, we do not need to include the multiplicative constant, even if the observations are from different instruments, since varying the constant and the normalisation will have the same effect.

9 Saving and loading models and fit results

We will want to save the results of the analyses we perform using XSPEC so that we can return to them later. We can save various aspects of an XSPEC session using the `save` command:

To save everything (including the spectra we have loaded, the channels we have ignored/noticed, the model, the fit statistic selection and the parameter values), we can type:

```
    XSPEC12> save all mysession.xcm
```

This produces a text file (conventionally with the file extension `.xcm`) that is simply a listing of the commands that would be entered into XSPEC one-by-one to restore the session to the current state.

We can reload the session from this file using the `@` command:

```
    XSPEC12> @mysession.xcm
```

Sometimes, we might just want to save the model, fit statistic selection and parameter values so that we can load it into another session to analyse a different data set:

```
    XSPEC12> save all mymodel.xcm
```

We can also just save the data sets we are currently using, as well as the ignored/noticed channels:

```
    XSPEC12> save data mydata.xcm
```

These text files can also be a convenient way to automate XSPEC. We can put any sequence of XSPEC commands into a text file (for example adding the `fit` or the `chain` commands to the end), and them execute them using `@`. We can also feed a text file of commands into `xspec` from the command line (*e.g.* for automating jobs on HPC facilities):

```
    $ xspec < mycommands.xcm
```

It should be noted that XSPEC does not save the current state of the fit, including the covariance matrix or any error estimates, therefore after loading a file, we will usually need to run `fit` again before continuing our analysis.

9.1 Logging console output

Sometimes it is useful to save the console output that XSPEC produces, for example to save the results of an `error` or `steppar` command. We can do this using the `log` command which will save all console output to a text file. Get everything set up, then at the point you wish to start logging (for example just before running `error`), type:

```
    XSPEC12> log mylog.txt
```

Then when you want to stop logging, type:

```
    XSPEC12> log none
```

Be careful with these log files though, since they can often become quite large if there is a lot of output (particularly if you have adjusted the `chatter` level)!

10 Plotting

XSPEC can produce a variety of plots of the spectrum and the model using the `plot` command. The plotting interface uses a back-end known as PGPLOT.

Before creating a plot, you will need to specify a plot device. Usually, we will use the XWINDOWS device, to show the plot on the screen:

```
XSPEC12> cpd /xw
```

Some commonly used plots:

- `plot data`: plot the spectrum on linear axes. If a model is defined, it will be shown as a line on the plot, using the current values of the parameters.

  The spectrum is plotted in units of $\mathrm{ct\,s^{-1}\,keV^{-1}}$, that is to say the count rate in each spectral bin, per unit energy (*i.e. divided by the width of that bin*). The spectrum is shown as it is recorded by the instrument, and the model is *folded* through the instrument response (*i.e.* the model is shown as it would appear if a source described by that model were observed with the telescope).
- `plot data`: as above, but on logarithmic axes.
- `plot ratio`: the ratio (data/model) between the data and the model in each spectral bin.
- `plot resid`: the residual (data $-$ model) between the data and the model in each spectral bin.
- `plot delchi`: the contribution to χ^2 from each data point, (data-model)/error.
- `plot model`: plot the model in units of $\mathrm{ct\,s^{-1}\,keV^{-1}}$, not folded through the instrument response.
- `plot em`: plot the model in units of flux (SED), *i.e.* $\mathrm{keV\,ct\,s^{-1}\,keV^{-1}}$.
- `plot eem`: plot the model in units of the spectral energy distribution (SED), *i.e.* $\mathrm{keV^2\,ct\,s^{-1}\,keV^{-1}}$.
- `plot eff`: plot the effective area curve of the intrument (AKA the efficiency as a function of photon energy)

You can show more than one of these in different panels within the plot by listing them one after the other. A common combination is the data (with model) and ratio:

```
XSPEC12> plot ldata ratio
```

If you just type the `plot` command without specifying an option, it will recreate the last type of plot that was created. This can be convenient if you want to change anything about the data, the fit or the binning then quickly refresh the plot.

If multiple spectra are loaded, all will be shown on the plot. If you wish to only plot only one (or only some) of the spectra, you can specify the spectrum number(s) between after `plot`. For example to plot the data and the ratio for spectrum 1:

```
XSPEC12> plot 1 ldata ratio
```

There are a number of options that can be set using the `setplot` command, including:

- `setplot energy`: plot the spectrum against photon energy in keV on the x-axis (the default is to plot the spectrum against instrument channel number).

- `setplot wave [units]`: plot the spectrum wavelength. The default is to plot the wavelength in units of Angstrom, but you can specify `angstrom`, `cm`, `micron` or `nm`.
- `setplot rebin [min_sn] [max_bins]`: rebin the plotted spectrum to achieve a minimum signal-to-noise ratio of `[min_sn]` in each bin by adding together a maximum of `[max_bins]` neighbouring channels, *e.g.* `setplot rebin 10 20`. This does not affect the binning of the spectrum used in the fit, only what is shown on the plot.
- `setplot add`: show each additive model component separately in addition to the total model (turn off using `setplot noadd`).
- `setplot back`: show the background spectrum on plots of the data. When multiple spectra are loaded, the background is shown for each in the same colour (turn off using `setplot noback`).

10.1 Saving plots to files

We can also specify files as output devices, specifying the plot device as `filename/driver`. The most commonly-used driver is `/ps`, to save the plot to a POSTSCRIPT file, in black and white, or `/cps`, for a colour POSTSCRIPT file. `/png` is also available if you configured it when compiling HEASOFT. For example, to create a colour postscript file called `myplot.ps`, use:

```
XSPEC12> cpd myplot.ps/cps
```

After specifying the output file as the device, create the plot as normal using the `plot` command, and the plot will be created in the file. Once you have created the plot, close the file with:

```
XSPEC12> cpd none
```

(or by switching back to `/xw`).

If you have a plot open on screen, you can also save it to a file using the `hard` command in `iplot` (see below).

POSTSCRIPT is the recommended format for saving XSPEC plots, since the plot will be saved as a vector graphic, meaning it can easily be resized when including it in a paper or slideshow. You can convert the `postscript` file to a PDF using the `ps2pdf` command that comes with most LaTeX distributions.

10.2 Interacting with the plotter

More options to customise the plot are available via the `iplot` interface. Use `iplot` in place of `plot`:

```
XSPEC12> iplot ldata ratio
```

or, if you already have open the plot you wish to work with, just type `iplot` with no further options:

```
XSPEC12> plot ldata ratio
XSPEC12> iplot
```

The prompt will change to `PLT>` and you can issue any `pgplot` commands to customise the prompt.

Since, by default, plots have very small fonts and contain a lot of extraneous information, like time stamps and not-very-informative titles, there is a particularly useful set of customisations, to make plots more suitable for use in papers or slides.

To remove the title (or to change the title, specify the new title at the end of this command:

```
    PLT> lab t
```

Remove the username and timestamp from the bottom corner:

```
    PLT> time off
```

Change the font to Roman:

```
    PLT> fo ro
```

Increase the font size:

```
    PLT> cs 1.2
```

We can also rescale the x and y axes:

```
    PLT> res x 3 50
    PLT> res y 1e-4 1e-2
```

We can save our current plot to a file using the `hard` command (short for `hardcopy`. The filename and type is specified in the same way described above for the plot devices. To save the plot in colour to a POSTSCRIPT file:

```
    PLT> hard myplot.ps/cps
```

When we're finished in `iplot`, we can type `exit` to return to the XSPEC prompt.

10.3 Saving the data points to plot outside of XSPEC

For more control, we may wish to export the data points to a text file so we can recreate the plot outside of XSPEC. We can do this using the `wdata` command within `iplot`:

```
    XSPEC12> iplot data
    PLT> wdata mydata.qdp
```

The text file is saved in the QDP format, which is essentially plain text columns separated by whitespace, and with each data point on a new line. There are some comment lines at the top beginning with `!`. When importing the file into, we will usually skip over these lines. If importing into NUMPY using `genfromtxt`, specify `skip_header=3`.

If we have a single spectrum loaded with a model, the columns of the file will correspond to the central energy of each bin, its error, the count rate, its error, and the model value:

```
    READ SERR 1 2
    @test.pco
    !
    2.98000002 1.99999809E-2 1.11482888E-2 1.07726094E-3 1.18596526E-2
    3.01999998 1.99999809E-2 1.49213038E-2 1.20735099E-3 1.2209136E-2
```

The line `READ SERR 1 2` tells us that data series 1 and 2 each have symmetric error bars, *i.e.* that column 2 is the error for column 1 and column 4 is the error for columns 3.

If multiple spectra are loaded, or we write the data from a plot with multiple series (*e.g.* data plus ratio), the different spectra or data series will be included in the file one after the other. A line reading `NO NO NO NO NO` (with one `NO` for each column) will separate the different series.

I recommend VEUSZ[4] for making plots, which has a plug-in for reading QDP files.

[4]https://veusz.github.io/download/

11 Further reading

- XSPEC manual: https://heasarc.gsfc.nasa.gov/xanadu/xspec/manual/XspecManual.html
- PYXSPEC tutorial (see quick and extended tutorials): https://heasarc.gsfc.nasa.gov/xanadu/xspec/python/html/quick.html
- XSPEC wiki, with notes on various topics: https://asd.gsfc.nasa.gov/XSPECwiki/XSPECPage
- "Handbook of X-ray Astronomy" by Arnaud, Smith and Siemiginowska, Cambridge University Press 2011