XSPEC - Implementation of analytical models

Michal Dovčiak

Astronomical Institute
Academy of Sciences of the Czech Republic
Prague

X-ray data modelling of accreting black holes
Hotel Belaria, Hradec nad Moravicí, Czech Republic
17^{th} – 25^{th} July 2021

Introduction

Information available in XSPEC manual Appendix C: Adding Models to XSPEC (`https://heasarc.gsfc.nasa.gov/xanadu/xspec/manual/node313.html`)

- Create subroutine with the new XSPEC model
- Create description file for the model and its parameters
- Compile the model inside the XSPEC (once)
- Load the model library
- Use the model

Example I – power-law with sharp energy cut-offs – the subroutine

```
#include <math.h>

void pwlco(const double *ear, int ne, const double *param, int ifl,
           double *photar, double *photer, const char* init) {

int    ie;
double e1, e2;

for (ie = 0; ie < ne; ie++){
  if ( ear[ie+1] <= param[1] ){
    photar[ie] = 0.;
    continue;
  }else if ( ear[ie] <= param[1] ) e1 = param[1];
  else e1 = ear[ie];
  if ( ear[ie] >= param[2] ){
    photar[ie] = 0.;
    continue;
  }else if ( ear[ie+1] >= param[2] ) e2 = param[2];
  else e2 = ear[ie+1];

  if(param[0] != 1.)
    photar[ie] = ( pow( e2, -param[0] + 1. ) - pow( e1, -param[0] + 1.) )
                 / ( -param[0] + 1. );
  else photar[ie] = log( e2 / e1 );
}

return;
}
```

- `ear` – energy array with `ne+1` values for energy bin boundaries
- `ne` – size of flux array
- `param` – parameter values
- `ifl` – the spectrum number being calculated
- `photar` – photon flux array with `ne` values defined per energy bin, i.e. not per keV!
- `photer` – photon flux error array with `ne` values
- `init` – this string is read on initialization and available to the model during execution (optional)

Example I – power-law with sharp energy cut-offs – the description

Entry for a model description in `lmodel.dat` file:

```
pwlco   3  0.          1.0e20           c_pwlco  add  0  0
PhoIndex " "      2.0      1.2      1.2      3.       3.        0.1
E0        keV     0.1      0.001    0.001    1.       1.        0.1
Ecut      keV     100.     10.      10.      1000.    1000.     10.
```

The first line for each model:

- the model name
- the number of parameters
- the low and high energies for which the model is valid
- the name of the subroutine to be called
- the type of model (add, mul, mix, or con, or acn)
- the flag to be set to 1 if model variances are calculated by modelfunc
- the flag to be set to 1 if the model should be forced to perform a calculation for each spectrum

Example I – power-law with sharp energy cut-offs – the description

Entry for a model description in `lmodel.dat` file:

```
pwlco   3  0.          1.0e20           c_pwlco  add  0  0
PhoIndex " "      2.0      1.2      1.2      3.       3.       0.1
E0       keV      0.1      0.001    0.001    1.       1.       0.1
Ecut     keV      100.     10.      10.      1000.    1000.    10.
```

The remaining lines specify each parameter in the model:

- the parameter name
- optional units label
- the default parameter value
- hard min and soft min
- soft max and hard max
- fit delta

Example I – power-law with sharp energy cut-offs – the description

Entry for a model description in `lmodel.dat` file:

```
pwlco   3  0.          1.0e20           c_pwlco  add  0  0
PhoIndex " "     2.0      1.2      1.2      3.       3.       0.1
E0       keV     0.1      0.001    0.001    1.       1.       0.1
Ecut     keV     100.     10.      10.      1000.    1000.    10.
```

Three special types of parameters can be used:

- scale parameter – its name is prefixed with a *, it cannot be made variable and may be linked only to another scale parameter, only the initial value is given
- switch parameter – its name is prefixed with a **$**, it is not used directly but switches the model's mode of operation, it only has 2 fields: its name and an integer value
- periodic parameter – **P** is added at the end of the line, if it tries to exceed its hard limits during the fit, it will be assigned a value within its limits: f(max + delta) = f(min + delta), f(min-delta) = f(max-delta)

Example II – model subroutine using FITS tables

```
#define REFSPECTRA "myrefl.fits\0"
#define NPAR 3

extern void tabint(float* ear, int ne, float* param, int npar,
                   const char* filenm, int ifl, const char* tabtyp,
                   float* photar, float* photer);

void myrefl(const double *ear, int ne, const double *param, int ifl,
            double *photar, double *photer, const char* init) {

char   refspectra[12] = REFSPECTRA;
int    ie;
float  fl_ear[ne+1], fl_photar[ne], fl_photer[ne];
float  fl_param[NPAR]={(float) param[0], (float) param[1], (float) param[2]};
const char* tabtyp="add";
double flux;

//Let's read and interpolate the FITS tables using internal XSPEC routine tabint
//Note that we do not use errors here
for(ie = 0; ie <= ne; ie++) fl_ear[ie] = (float) ear[ie];
tabint(fl_ear, ne, fl_param, NPAR, refspectra, ifl, tabtyp, fl_photar, fl_photer);

//Let's e.g. renormalize to the total counts
flux=0.;
for (ie = 0; ie < ne; ie++) flux += fl_photar[ie];
for (ie = 0; ie < ne; ie++) photar[ie] = fl_photar[ie] / flux;

return;
}
```

How to install and use local models

Installation in XSPEC:

- the code is compiled inside XSPEC with the following command:
  initpackage mypackage lmodel.dat /path/to/mypackage

To use local models inside XSPEC:

- the package needs to be loaded:
  lmod mypackage /path/to/mypackage
- then any model from the package may be used, e.g.:
  mo pwlco