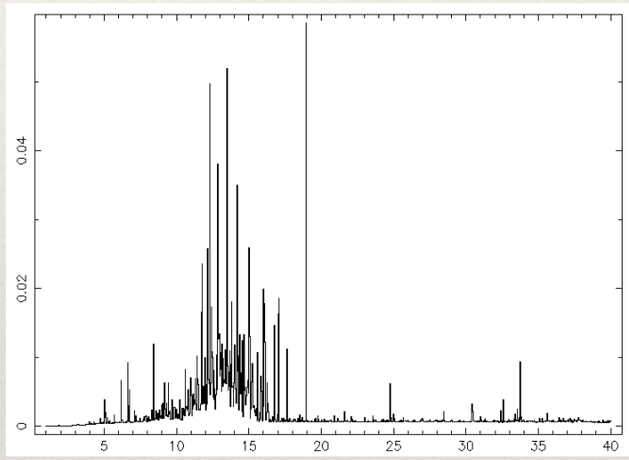
High resolution X-ray spectroscopic software and tools: XSTAR and XSTARDB

Navigating atomic databases with ISIS



L. Corrales, May 13, 2016

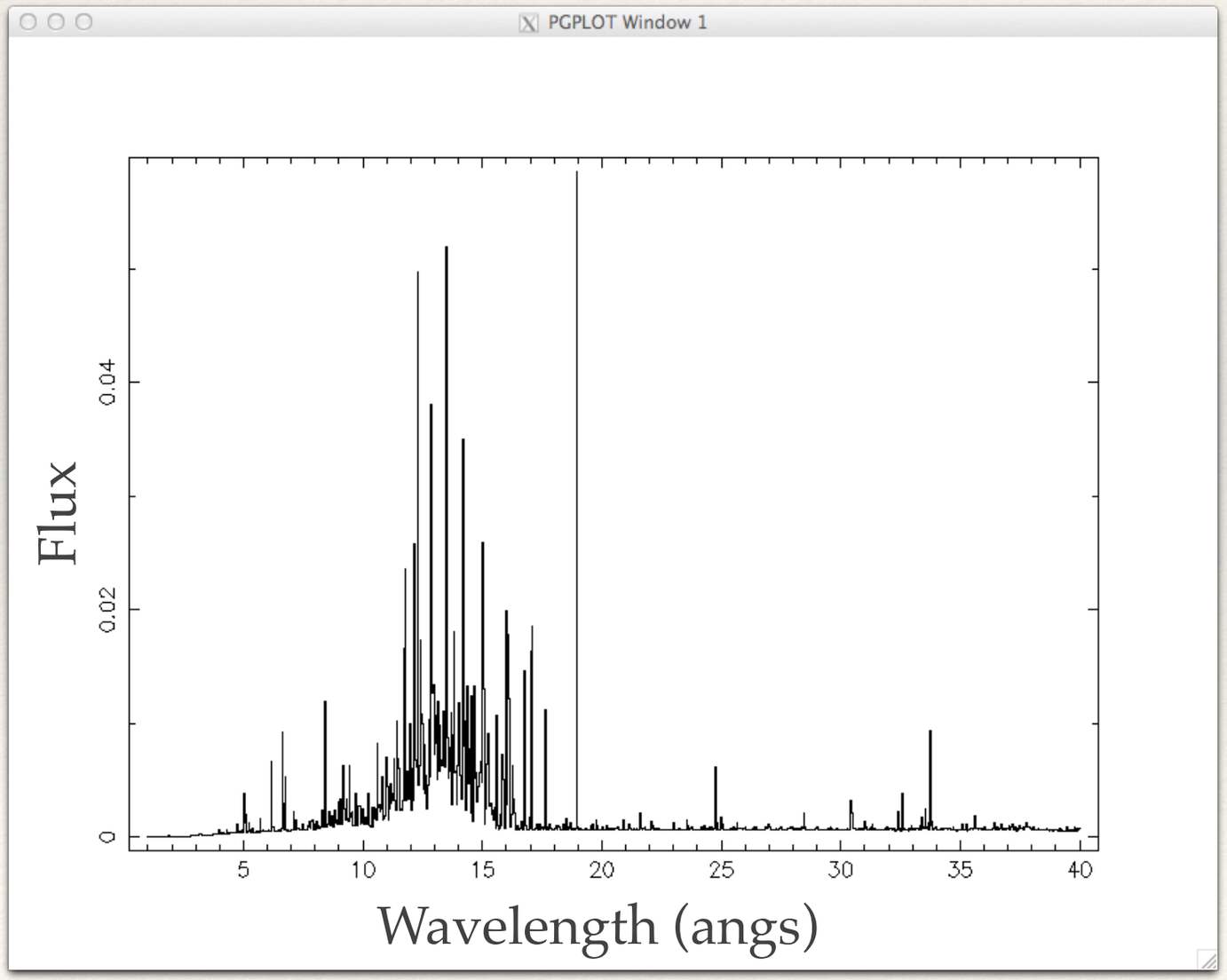
Navigating AtomDB with ISIS

Modeling astrophysical plasmas (apec in XSPEC)

```
isis> (x1, x2) = linear_grid( 1, 40, 1000 );
isis> fit_fun( "apec(1)" );
 Solar Abundance Vector set to angr:  Anders E. & Grevesse N. Geochimica et Cosmochimica Acta 53, 197 (1989)
 Cross Section Table set to bcmc:  Balucinska-Church and McCammon, 1998
isis> y = eval_fun( x1, x2 );
isis> hplot( x1, x2, y );
```

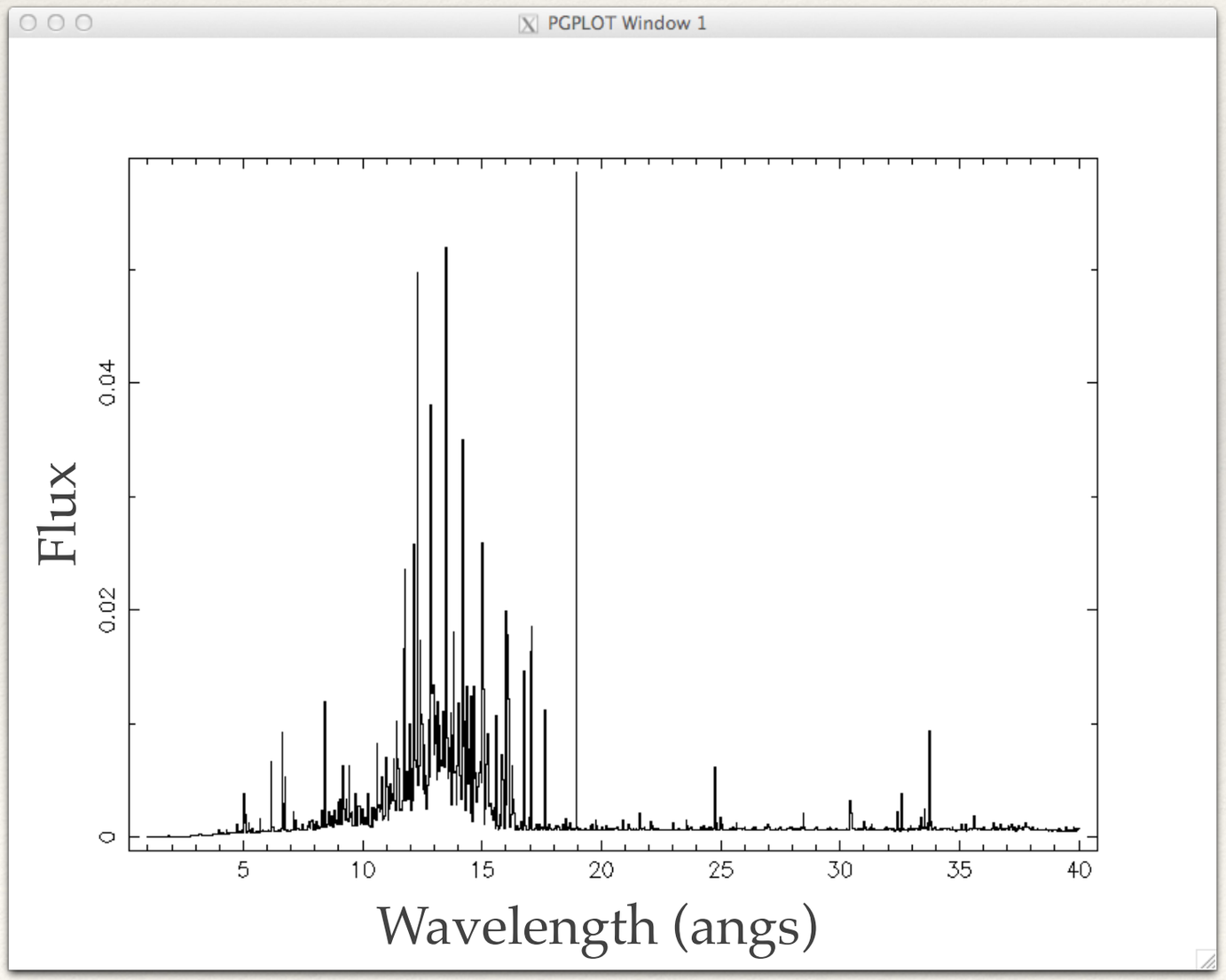
apec, vapec, vvapec: APEC emission spectrum

An emission spectrum from collisionally-ionized diffuse gas calculated using the ATOMDB code v2.0.2. More information can be found at http://atomdb.org/ which should be consulted by anyone running this model. This default version number can be changed by modifiying the ATOMDB_VERSION string in your Xspec.init file.



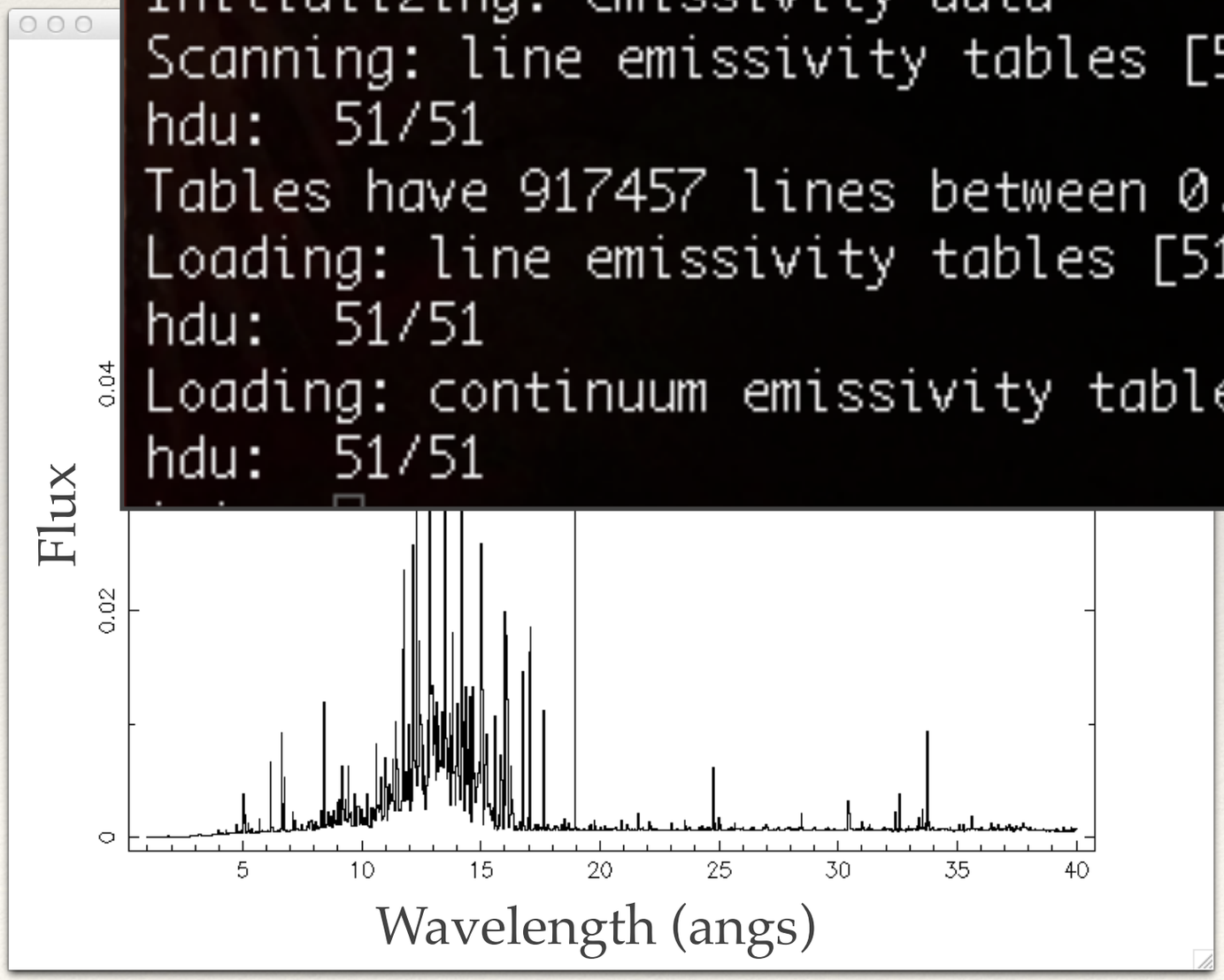
Modeling astrophysical plasmas (apec in XSPEC)

```
isis> list_par;
apec(1)
 idx   param           tie-to  freeze          value          min          max
   1   apec(1).norm       0       0              1            0        1e+10
   2   apec(1).kT         0       0              1        0.008           64  keV
   3   apec(1).Abundanc   0       1              1            0            5
   4   apec(1).Redshift   0       1              0       -0.999           10
```



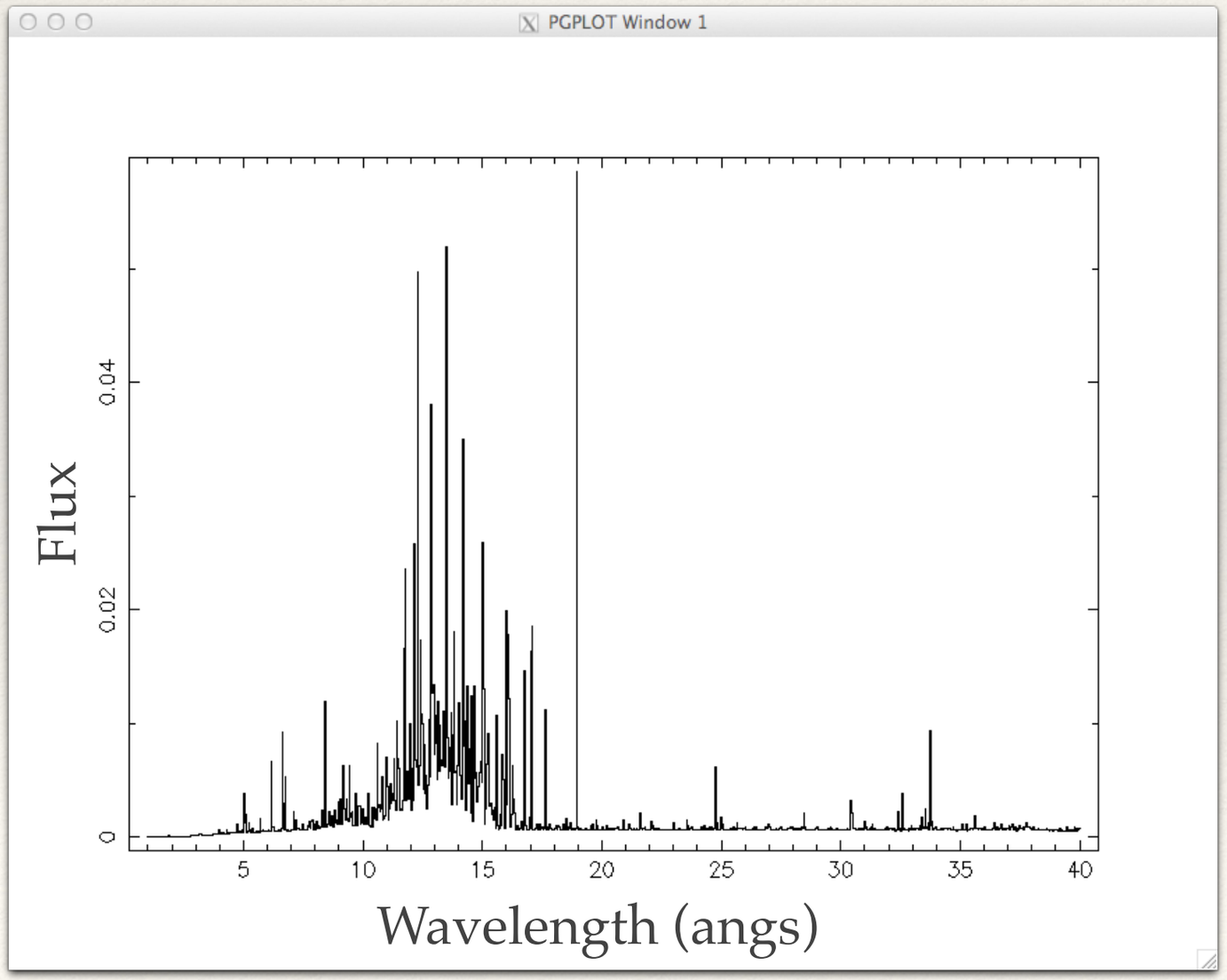
Modeling astrophysical plasmas (ISIS)

```
isis> plasma(aped);
Reading database version 2.0.2 from /usr/local/atomdb_v2.0.2/
Initializing: atomic data
Read 268 energy level files
Read 268 wavelength files
Tables have 914520 lines between 0.6979 and 1e+10 Angstrom
Initializing: emissivity data
Scanning: line emissivity tables [51 hdus]
hdu:  51/51
Tables have 917457 lines between 0.6979 and 1e+10 Angstrom
Loading: line emissivity tables [51 hdus]
hdu:  51/51
Loading: continuum emissivity tables [51 hdus]
hdu:  51/51
```



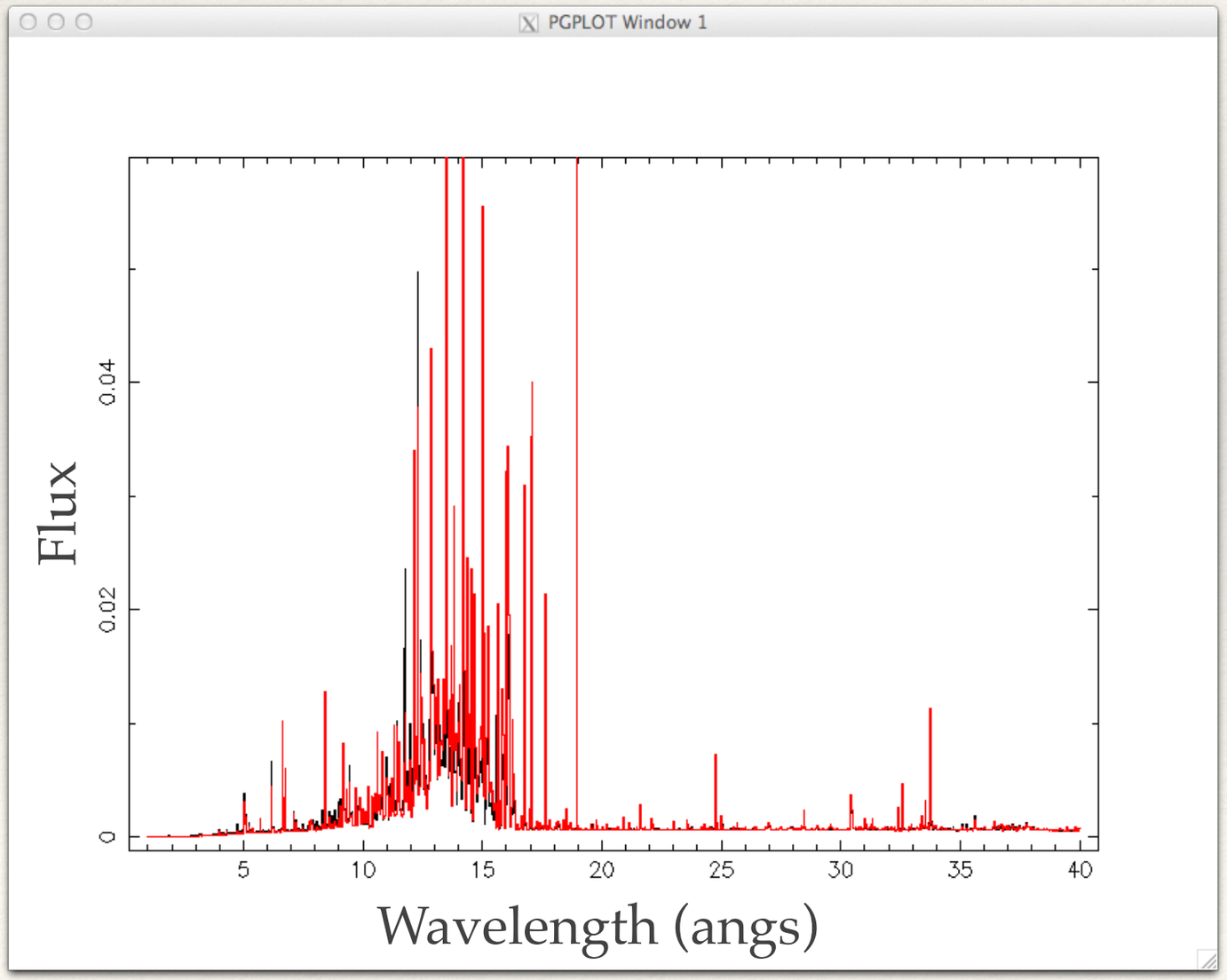
Modeling astrophysical plasmas (ISIS)

```
isis> print( aped );
Reading database version 2.0.1 from /nfs/cxc/a1/share/atomdb
{dir="/nfs/cxc/a1/share/atomdb",
 atomic_data_filemap="filemap",
 abundance="APED/misc/Abundances.fits",
 ion_balance="APED/ionbal/MM98_ionbal.fits",
 line_emissivity="apec_v2.0.1_line.fits",
 continuum_emissivity="apec_v2.0.1_coco.fits"}
```



Modeling astrophysical plasmas (ISIS)

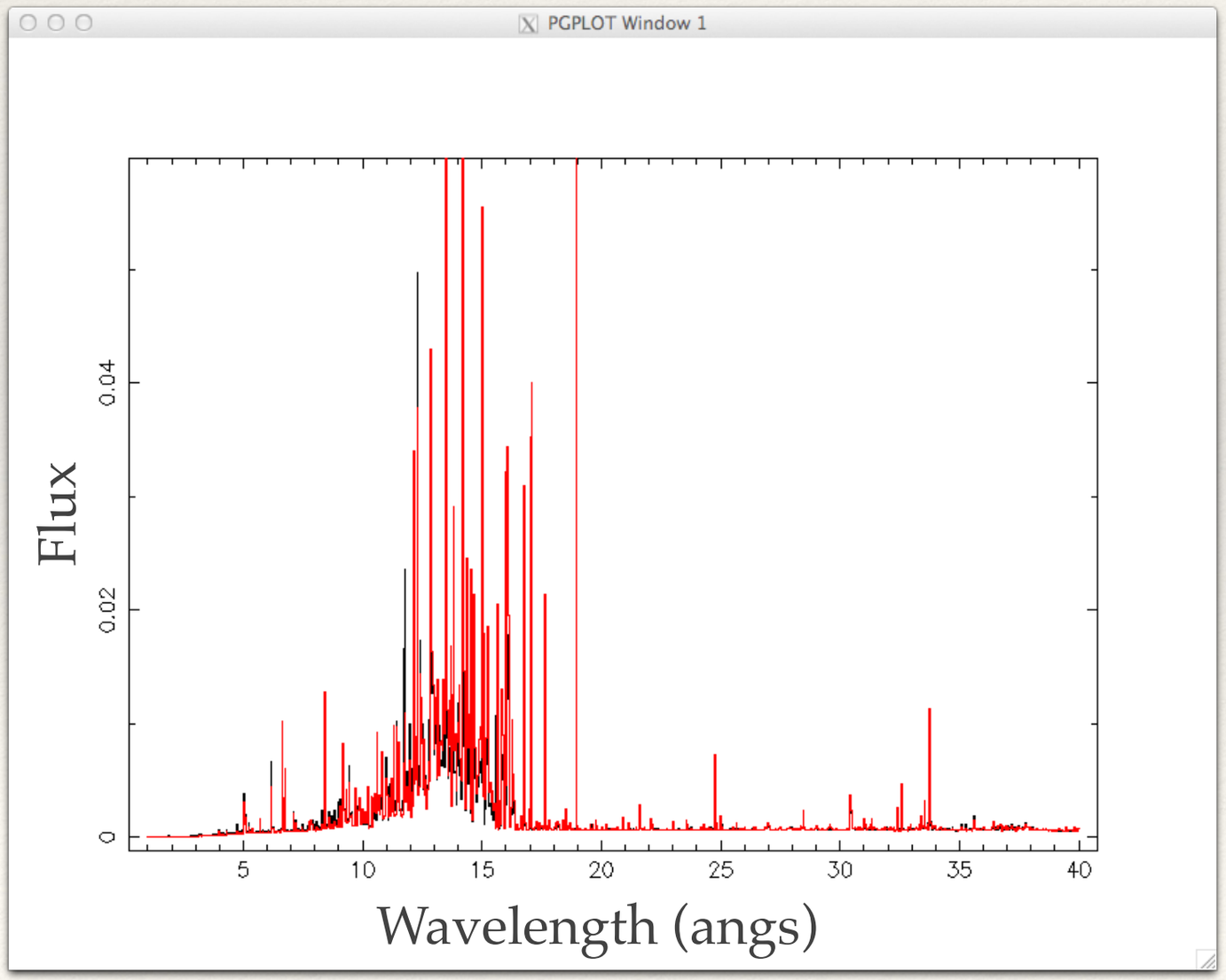
```
isis> (x1, x2) = linear_grid(1, 40, 1000);
isis> create_aped_fun( "myplasma", default_plasma_state );
isis> fit_fun( "myplasma(1)" );
isis> y2 = eval_fun( x1, x2 );
isis> ohplot( x1, x2, y2, 2 );
```




```
isis> print( default_plasma_state );
{norm=1.0,
 temperature=1e+07,
 density=1.0,
 metal_abund=1.0,
 elem_abund=NULL,
 elem=NULL,
 vturb=0.0,
 redshift=0.0}
```

Modeling astrophysical plasmas (ISIS)

```
isis> list_par;
myplasma(1)
 idx  param                    tie-to  freeze          value          min          max
  1  myplasma(1).norm             0       1              1            0            0
  2  myplasma(1).temperature      0       1          1e+07            0            0
  3  myplasma(1).density          0       1              1            0            0
  4  myplasma(1).vturb            0       1              0            0            0
  5  myplasma(1).redshift         0       1              0            0            0
  6  myplasma(1).metal_abund      0       1              1            0            0
```



Navigating AtomDB with ISIS

SEARCH options

- `wl(min, max);`
- `el_ion(Z[, ion]);`
- `trans(Z[, ion[, upper[, lower]]]);`

returns a boolean array, easy to combine
use where to get a line list:

```
ll = where( wl(18, 20) & el_ion(O) );
```

Navigating AtomDB with ISIS

SEARCH options

- `wl(min, max);`
- `el_ion(Z[, ion]);`
- `trans(Z[, ion[, upper[, lower]]]);`

returns a boolean array, easy to combine
use where to get a line list:

```
ll = where( wl(18, 20) & el_ion(O) );
```

SCIENCE

- `bl = brightest(n, ll);`
- `line_em(ll, temps[, dens]);`
- `ratio_em(ll[0], ll[1], temps[, dens]);`

Navigating AtomDB with ISIS

SEARCH options

- `wl(min, max);`
- `el_ion(Z[, ion]);`
- `trans(Z[, ion[, upper[, lower]]]);`

returns a boolean array, easy to combine
use where to get a line list:

```
ll = where( wl(18, 20) & el_ion(O) );
```

SCIENCE

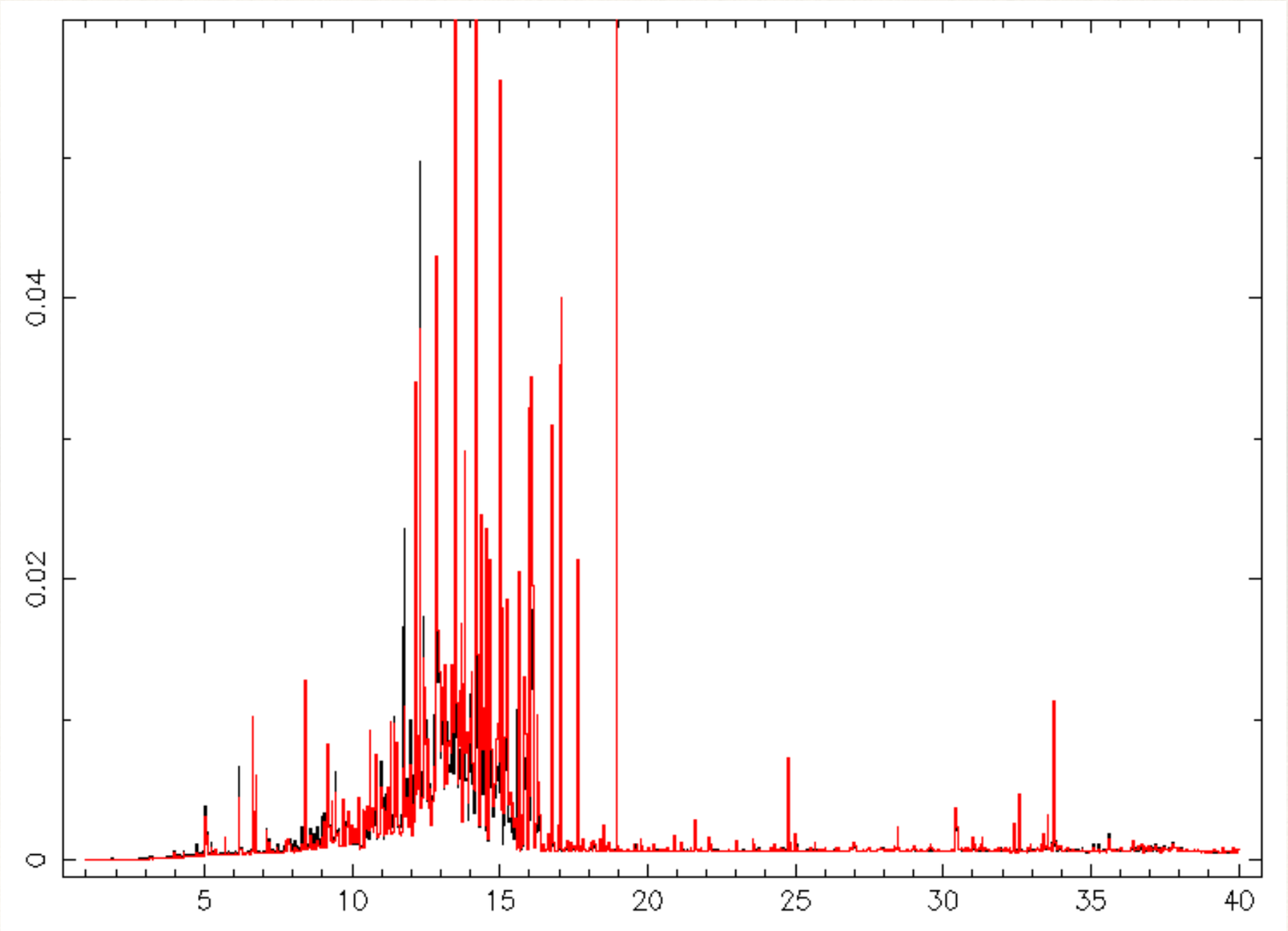
- `bl = brightest(n, ll);`
- `line_em(ll, temps[, dens]);`
- `ratio_em(ll[0], ll[1], temps[, dens]);`

Get database INFO

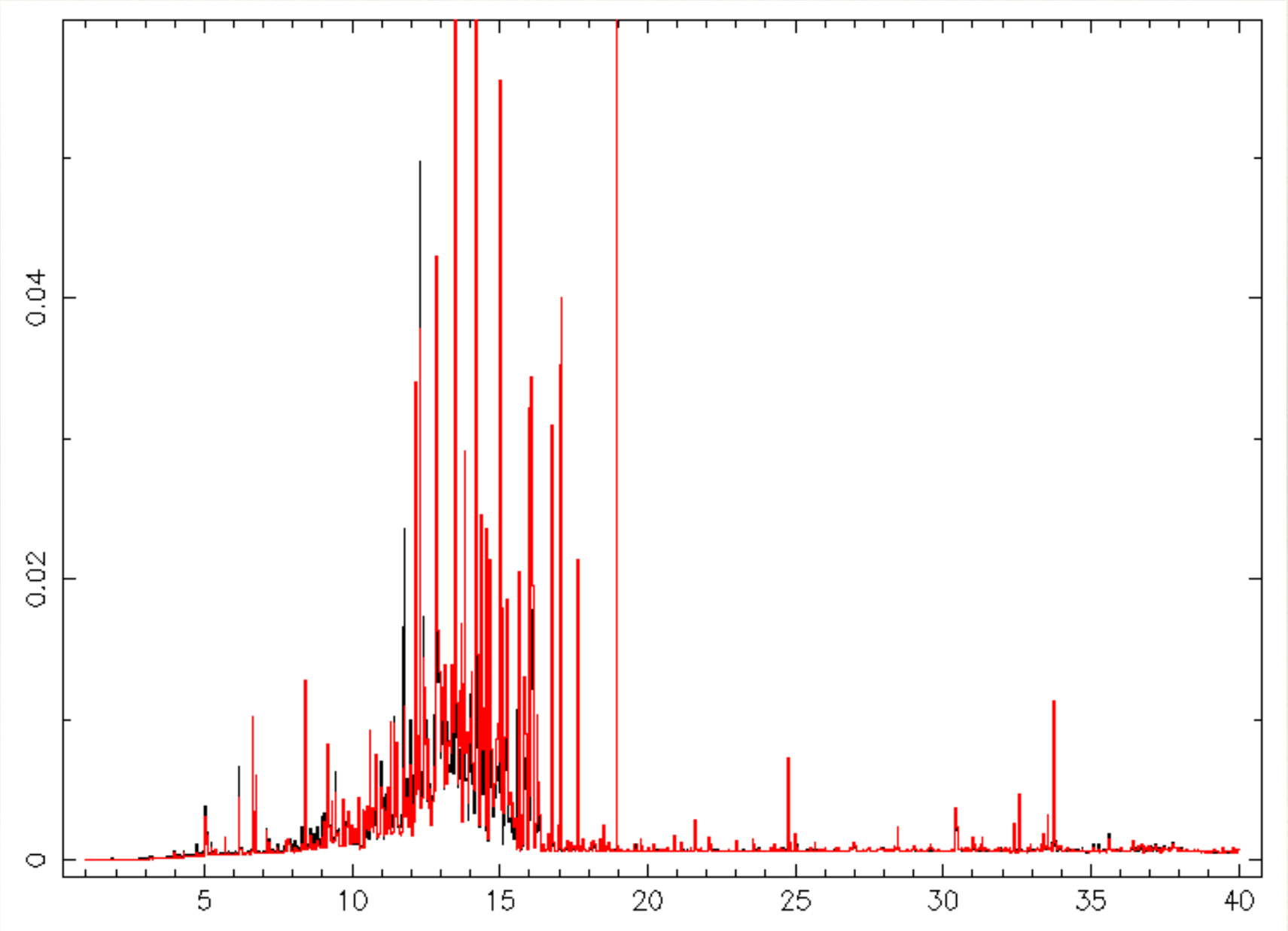
- `page_group(bl);`
- `plot_group(bl);`
- `s = line_info(bl[0]);`

```
isis> print(s);
{flux=0.04733396397989178,
 lambda=18.96711,
 lambda_err=9.0000001e-05,
 A=2.5658209e+12,
 A_err=1.1754944e-38,
 id=50033,
 ion=8,
 Z=8,
 upper=4,
 lower=1,
 upper_g=4.0,
 lower_g=2.0,
 upper_E=653.77435,
 lower_E=0.0,
 upper_L=1,
 upper_S=0,
 lower_L=0,
 lower_S=0,
 up_name="2p1",
 lo_name="1s1"}
```

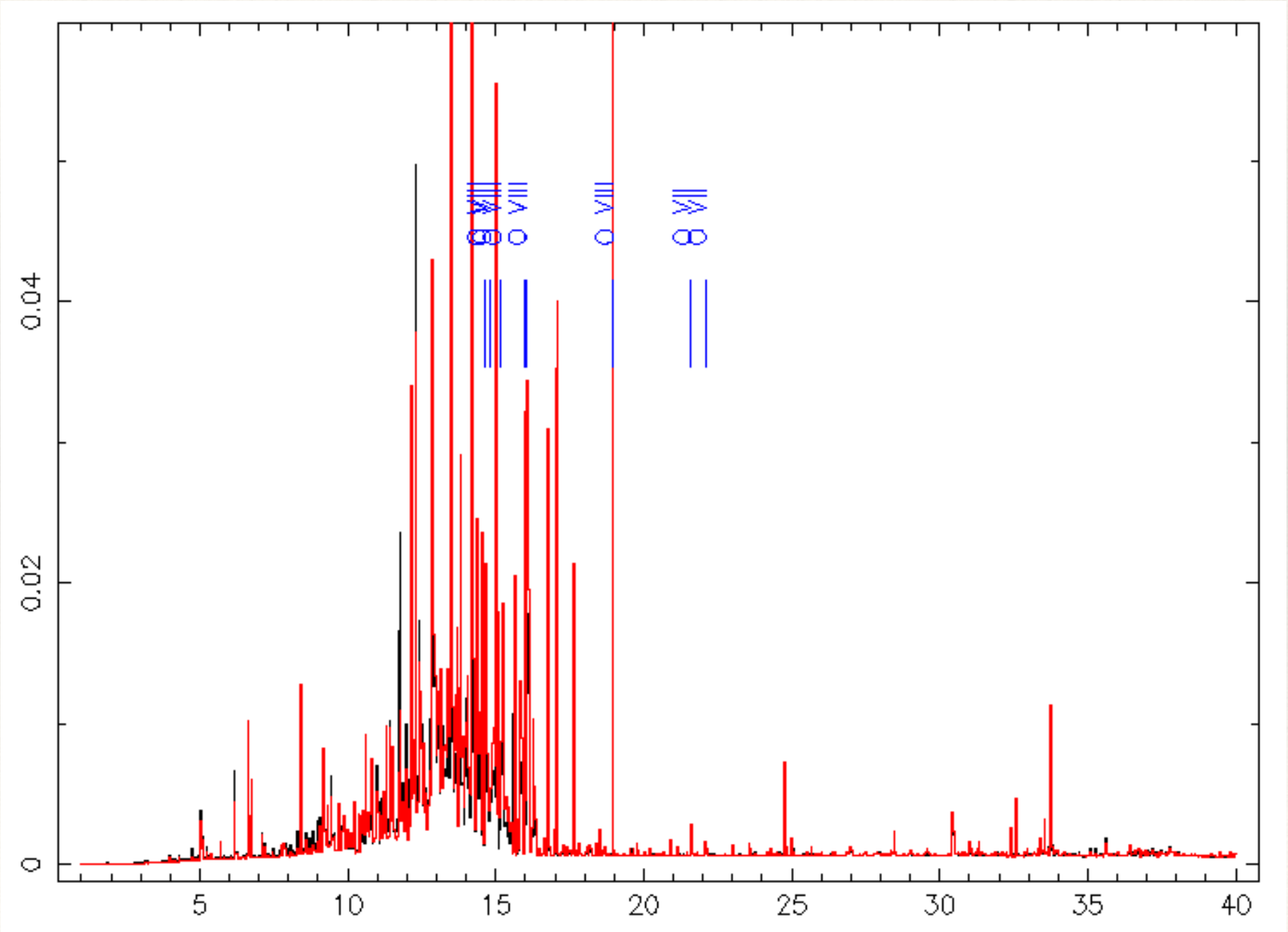
```
isis> olines = brightest(10, where( el_ion(0) ) );
isis> page_group( olines );
#  index         ion   lambda    F (ph/cm^2/s)  A(s^-1)  upper lower   label
  50134 *     0 VIII    14.633    6.564e-04   7.951e+10    28     1 6p1 - 1s1
  50091 *     0 VIII    14.821    1.121e-03   1.393e+11    19     1 5p1 - 1s1
  50058 *     0 VIII    15.176    2.431e-03   2.774e+11    12     1 4p1 - 1s1
  50045 *     0 VIII    15.176    1.330e-03   5.562e+11    11     1 4p1 - 1s1
  50041 *     0 VIII    16.006    6.461e-03   6.829e+11     7     1 3p1 - 1s1
  50034 *     0 VIII    16.007    3.902e-03   1.367e+12     6     1 3p1 - 1s1
  50033 *     0 VIII    18.967    4.733e-02   2.566e+12     4     1 2p1 - 1s1
  50031 *     0 VIII    18.973    2.371e-02   2.564e+12     3     1 2p1 - 1s1
  40783 *      0 VII    21.602    2.153e-03   3.430e+12     7     1 1s1 2p1 - 1s2
  49857 *      0 VII    22.098    1.012e-03   9.120e+02     2     1 1s1 2s1 - 1s2
isis> plot_group( olines );
```



```
isis> olines = brightest(10, where( el_ion(0) ) );
isis> page_group( olines );
#  index          ion   lambda    F (ph/cm^2/s)  A(s^-1)  upper lower   label
  50134 *    0 VIII    14.633    6.564e-04   7.951e+10    28     1 6p1 - 1s1
  50091 *    0 VIII    14.821    1.121e-03   1.393e+11    19     1 5p1 - 1s1
  50058 *    0 VIII    15.176    2.431e-03   2.774e+11    12     1 4p1 - 1s1
  50045 *    0 VIII    15.176    1.330e-03   5.562e+11    11     1 4p1 - 1s1
  50041 *    0 VIII    16.006    6.461e-03   6.829e+11     7     1 3p1 - 1s1
  50034 *    0 VIII    16.007    3.902e-03   1.367e+12     6     1 3p1 - 1s1
  50033 *    0 VIII    18.967    4.733e-02   2.566e+12     4     1 2p1 - 1s1
  50031 *    0 VIII    18.973    2.371e-02   2.564e+12     3     1 2p1 - 1s1
  40783 *     0 VII    21.602    2.153e-03   3.430e+12     7     1 1s1 2p1 - 1s2
  49857 *     0 VII    22.098    1.012e-03   9.120e+02     2     1 1s1 2s1 - 1s2
isis> plot_group( olines );
```



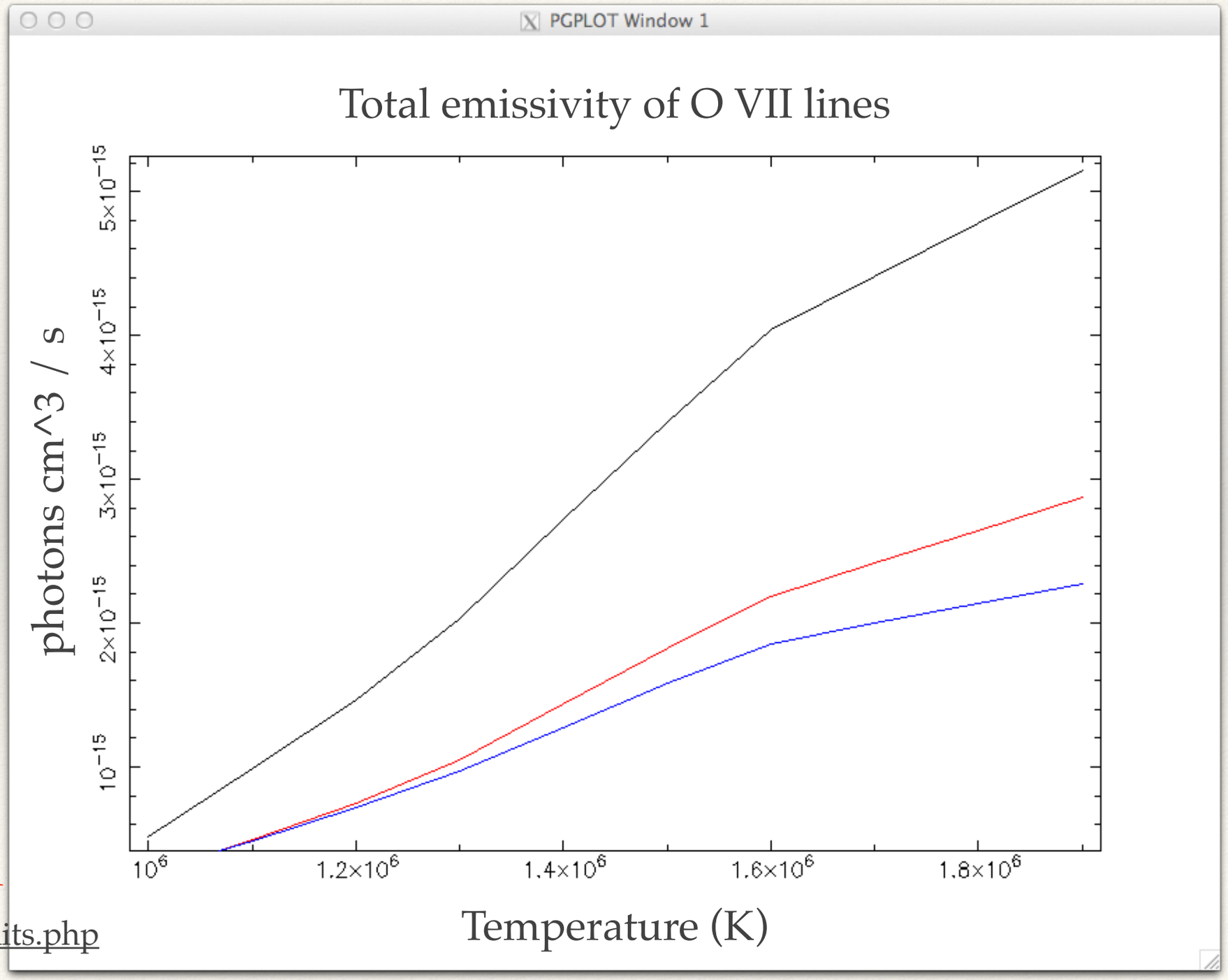
```
isis> olines = brightest(10, where( el_ion(O) ) );
isis> page_group( olines );
#  index         ion   lambda    F (ph/cm^2/s)  A(s^-1)  upper lower   label
  50134 *     O VIII    14.633    6.564e-04   7.951e+10    28     1 6p1 - 1s1
  50091 *     O VIII    14.821    1.121e-03   1.393e+11    19     1 5p1 - 1s1
  50058 *     O VIII    15.176    2.431e-03   2.774e+11    12     1 4p1 - 1s1
  50045 *     O VIII    15.176    1.330e-03   5.562e+11    11     1 4p1 - 1s1
  50041 *     O VIII    16.006    6.461e-03   6.829e+11     7     1 3p1 - 1s1
  50034 *     O VIII    16.007    3.902e-03   1.367e+12     6     1 3p1 - 1s1
  50033 *     O VIII    18.967    4.733e-02   2.566e+12     4     1 2p1 - 1s1
  50031 *     O VIII    18.973    2.371e-02   2.564e+12     3     1 2p1 - 1s1
  40783 *      O VII    21.602    2.153e-03   3.430e+12     7     1 1s1 2p1 - 1s2
  49857 *      O VII    22.098    1.012e-03   9.120e+02     2     1 1s1 2s1 - 1s2
isis> plot_group( olines );
```



```
isis> ovii = brightest(2, where( el_ion(0,7) ) );
isis> temps = [1.e6 : 2.e6 : 1.e5];
isis> emis = line_em( ovii, temps );
```

line_em gives **sum of line-list emissivities** as a function of temperature (or dens)

```
isis> plot( temps, emis );
isis> oplot( temps, line_em(ovii[0], temps), 2 );
isis> oplot( temps, line_em(ovii[1], temps), 4 );
```



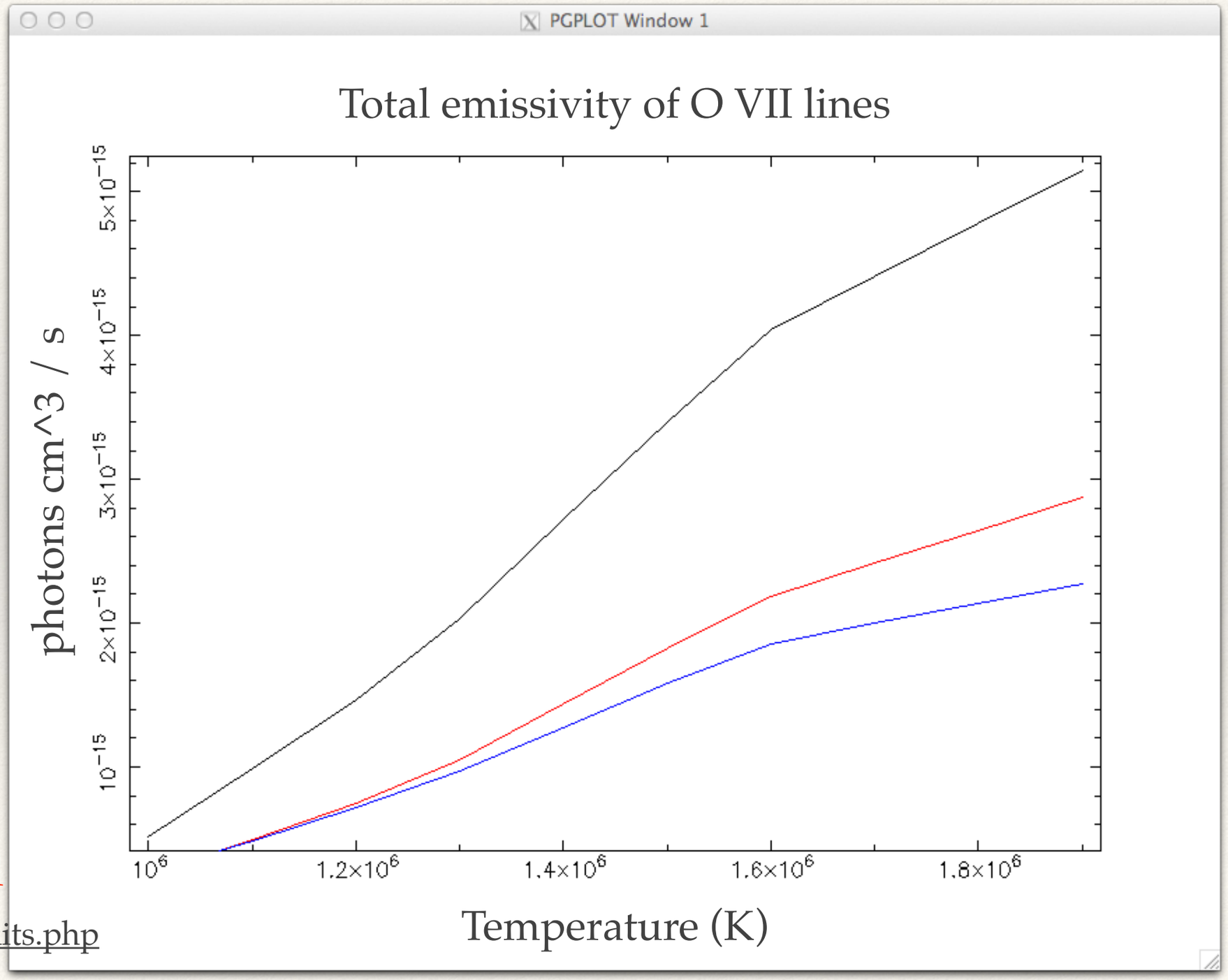
How emissivity is calculated

http://www.atomdb.org/Physics/units.php

```
isis> ovii = brightest(2, where( el_ion(0,7) ) );
isis> temps = [1.e6 : 2.e6 : 1.e5];
isis> emis = line_em( ovii, temps );
```

line_em gives **sum of line-list emissivities** as a function of temperature (or dens)

```
isis> plot( temps, emis );
isis> oplot( temps, line_em(ovii[0], temps), 2 );
isis> oplot( temps, line_em(ovii[1], temps), 4 );
```



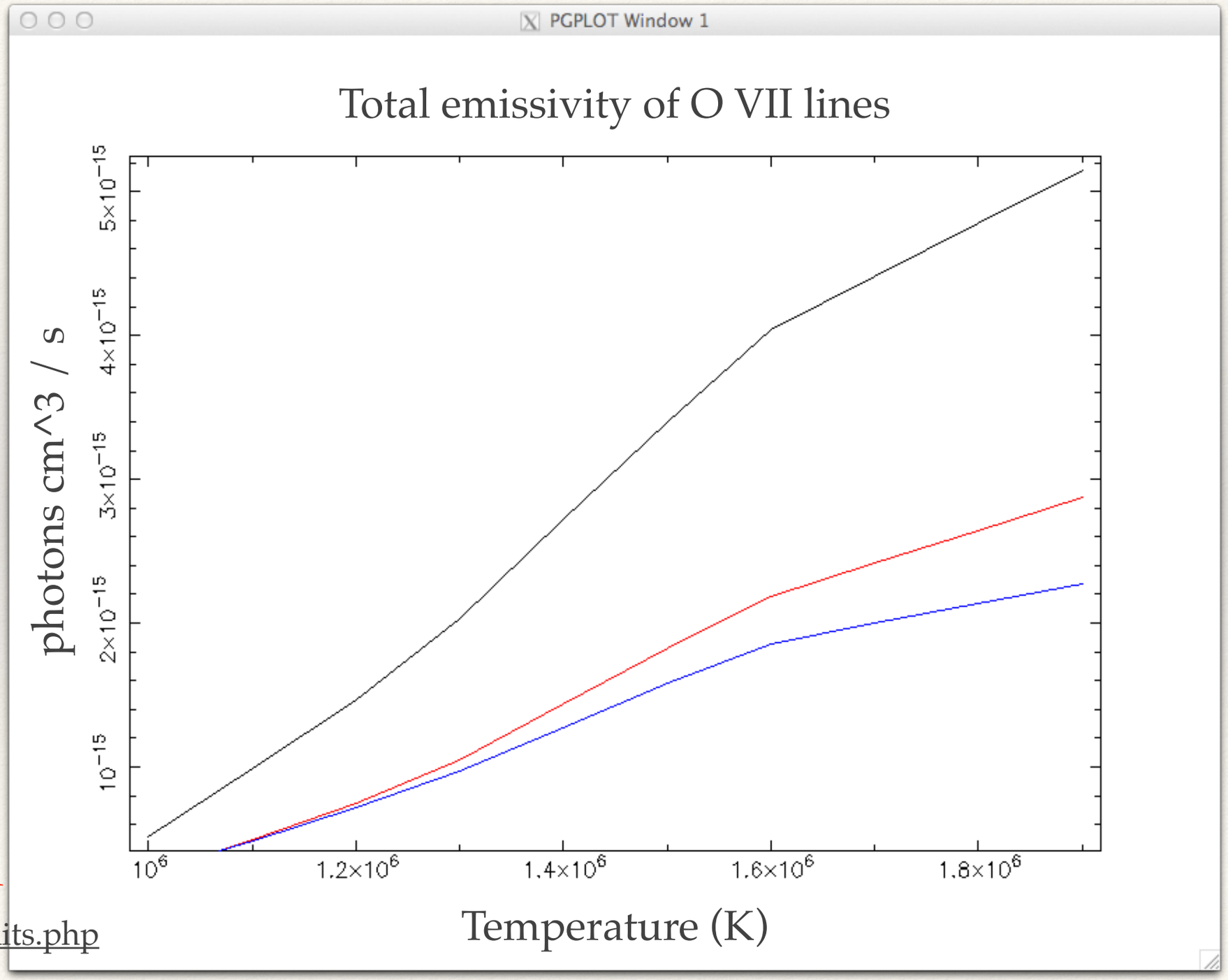
How emissivity is calculated

http://www.atomdb.org/Physics/units.php

```
isis> ovii = brightest(2, where( el_ion(0,7) ) );
isis> temps = [1.e6 : 2.e6 : 1.e5];
isis> emis = line_em( ovii, temps );
```

line_em gives **sum of line-list emissivities** as a function of temperature (or dens)

```
isis> plot( temps, emis );
isis> oplot( temps, line_em(ovii[0], temps), 2 );
isis> oplot( temps, line_em(ovii[1], temps), 4 );
```

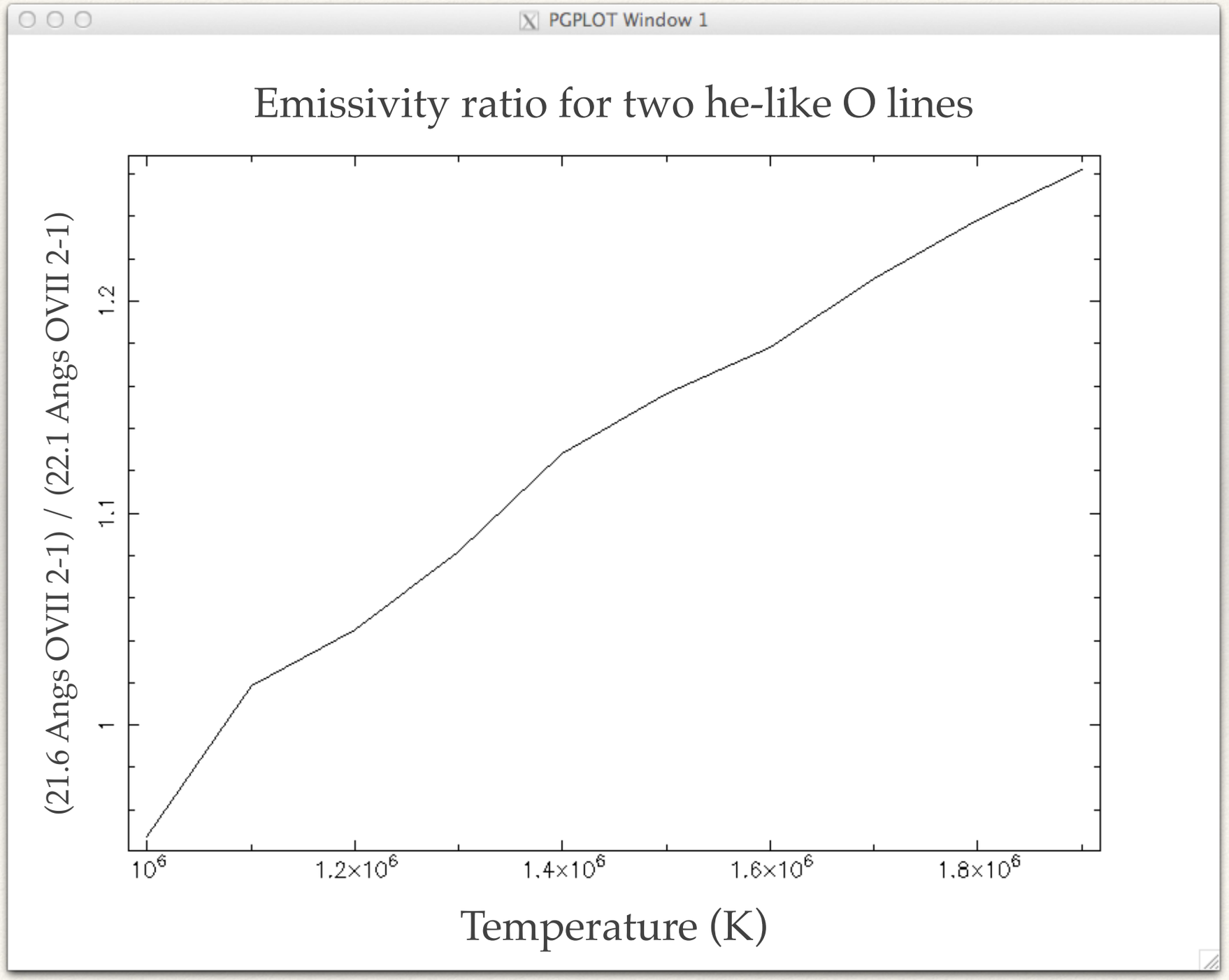


How emissivity is calculated

http://www.atomdb.org/Physics/units.php

ratio_em gives **ratios of emissivities** as a function of temperature (or dens)
and will keep line-list groups intact

```
isis> rat = ratio_em( ovii[0], ovii[1], temps);
isis> plot( temps, rat );
```



Navigating XSTAR with ISIS:

— XSTARDB —

http://space.mit.edu/cxc/analysis/xstardb/

XSTARDB extension to XSTAR

Similar philosophy to AtomDB navigation, except:

1. Each model run produces a **separate fits file database**
2. That database file must then be **loaded into a structure**
3. Each **function call acts on the database structure** (instead of environment)

http://space.mit.edu/cxc/analysis/xstardb/

XSTARDB extension to XSTAR

Similar philosophy to AtomDB navigation, except:

1. Each model run produces a **separate fits file database**
2. That database file must then be **loaded into a structure**
3. Each **function call acts on the database structure** (instead of environment)

LOAD:

- `db = rd_xstar_output("warmabs_1.fits");`

http://space.mit.edu/cxc/analysis/xstardb/

XSTARDB extension to XSTAR

Similar philosophy to AtomDB navigation, except:

1. Each model run produces a **separate fits file database**
2. That database file must then be **loaded into a structure**
3. Each **function call acts on the database structure** (instead of environment)

LOAD:

- `db = rd_xstar_output("warmabs_1.fits");`

SEARCH:

- `xstar_wl(db, min, max);`
- `xstar_el_ion(db, Z[, ion]);`
- `xstar_trans(db, Z[, ion[, upper[, lower]]]);`

http://space.mit.edu/cxc/analysis/xstardb/

XSTARDB extension to XSTAR

Similar philosophy to AtomDB navigation, except:

1. Each model run produces a **separate fits file database**
2. That database file must then be **loaded into a structure**
3. Each **function call acts on the database structure** (instead of environment)

LOAD:

- `db = rd_xstar_output("warmabs_1.fits");`

SEARCH:

- `xstar_wl(db, min, max);`
- `xstar_el_ion(db, Z[, ion]);`
- `xstar_trans(db, Z[, ion[, upper[, lower]]]);`

INFO:

- `xstar_page_group(db, ll[; sort]);`
- `xstar_plot_group(db, ll);`

http://space.mit.edu/cxc/analysis/xstardb/

XSTARDB extension to XSTAR

Similar philosophy to AtomDB navigation, except:

1. Each model run produces a **separate fits file database**
2. That database file must then be **loaded into a structure**
3. Each **function call acts on the database structure** (instead of environment)

LOAD:

- `db = rd_xstar_output("warmabs_1.fits");`

SEARCH:

- `xstar_wl(db, min, max);`
- `xstar_el_ion(db, Z[, ion]);`
- `xstar_trans(db, Z[, ion[, upper[, lower]]]);`

INFO:

- `xstar_page_group(db, ll[; sort]);`
- `xstar_plot_group(db, ll);`

SCIENCE:

- `xstar_strong(n, db[; wmin, wmax]);`

http://space.mit.edu/cxc/analysis/xstardb/

XSTARDB extension to XSTAR

Similar philosophy to AtomDB navigation, except:

1. Each model run produces a **separate fits file database**
2. That database file must then be **loaded into a structure**
3. Each **function call acts on the database structure** (instead of environment)

LOAD:

```
db = rd_xstar_output("warmabs_1.fits");
dbm = xstar_merge(["warmabs_1.fits","warmabs_2.fits"]);
db_grid = xstar_load_grid(filenames);
```

SEARCH:

```
xstar_wl(db, min, max);
xstar_el_ion(db, Z[, ion]);
xstar_trans(db, Z[, ion[, upper[, lower]]]);
```

INFO:

```
xstar_page_group(db, ll[; sort]);
xstar_plot_group(db, ll);
```

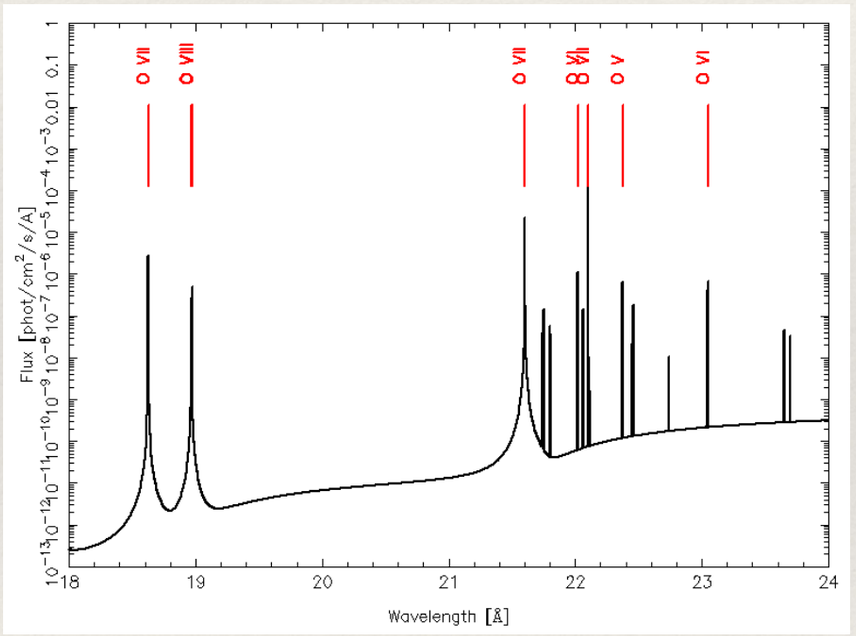
SCIENCE:

```
xstar_strong(n, db[; wmin, wmax]);
xstar_run_model_grid(model_info, "path"[; istart]);
xstar_line_prop(db_grid, ll, "luminosity");
xstar_get_grid_par(db_grid, "rlogxi");
```

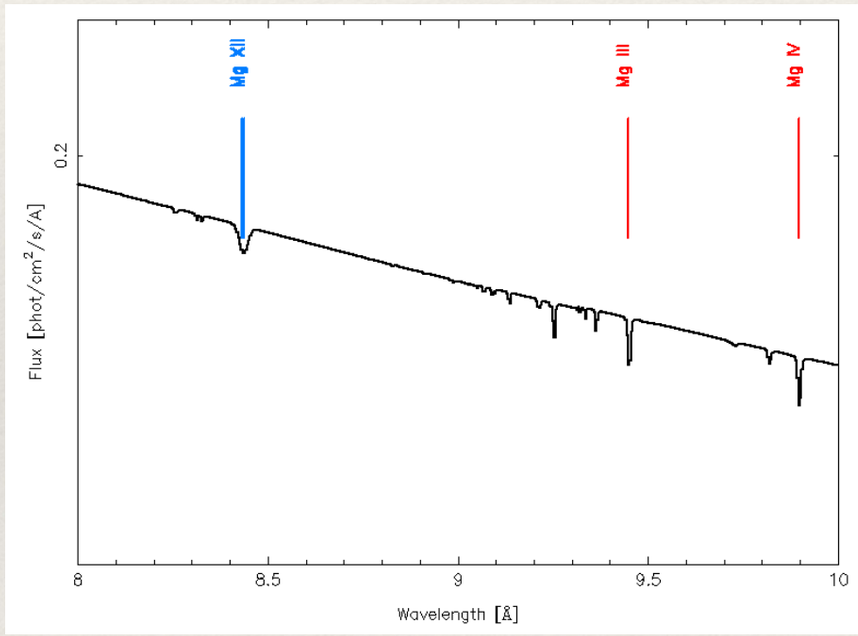
http://space.mit.edu/cxc/analysis/xstardb/

Three essential examples

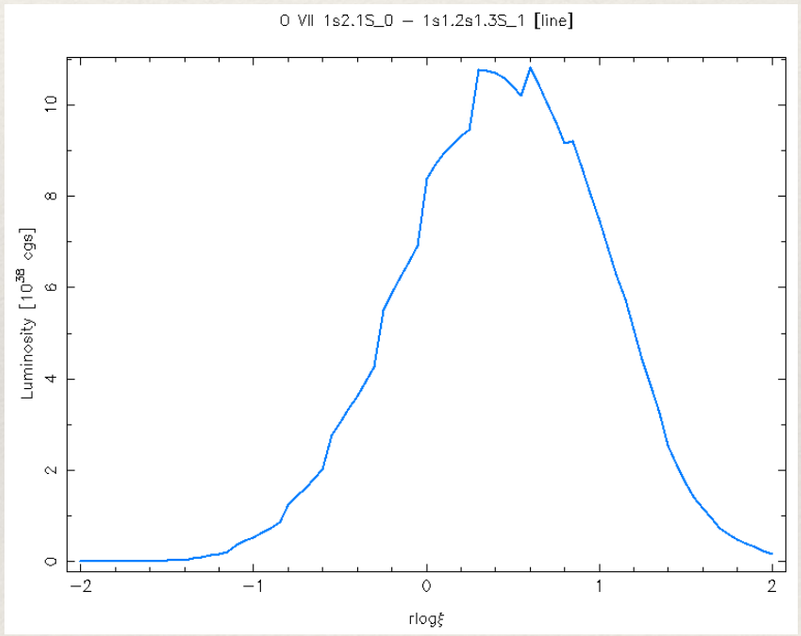
http://space.mit.edu/cxc/analysis/xstardb/examples.html



single photemis model



multiple warmabs models



grid of photemis models
(for line luminosity vs rlogxi)

Starting up XSTARDB

```
isis> require( "xstardb" );

%% LMODDIR = /usr/local/xspec_lmodels/warmabs22
%% WARMABS_DATA = /usr/local/xspec_lmodels/warmabs22
%% WARMABS_POP = pops.fits


xstardb_utils 0.5.0 loaded

isis> fit_fun( "powerlaw(1) * warmabs2(1)" );
isis> set_par( "warmabs2(1).write_outfile", 1 );
isis> set_par( "warmabs2(1).autoname_outfile", 1 );
```

Starting up XSTARDB

```
isis> list_par;
powerlaw(1) * warmabs2(1)
 idx  param                    tie-to  freeze        value         min         max
  1  powerlaw(1).norm             0     0             1           0        1e+10
  2  powerlaw(1).PhoIndex         0     0             1          -2           9
  3  warmabs2(1).column           0     0             0          -3           2  cm^-2
  4  warmabs2(1).rlogxi           0     0             0          -4           5
  5  warmabs2(1).Cabund           0     1             1           0        1000
  6  warmabs2(1).Nabund           0     1             1           0        1000
  7  warmabs2(1).Oabund           0     1             1           0        1000
  8  warmabs2(1).Fabund           0     1             1           0        1000
  9  warmabs2(1).Neabund          0     1             1           0        1000
 10  warmabs2(1).Naabund          0     1             1           0        1000
 11  warmabs2(1).Mgabund          0     1             1           0        1000
 12  warmabs2(1).Alabund          0     1             1           0        1000
 13  warmabs2(1).Siabund          0     1             1           0        1000
 14  warmabs2(1).Pabund           0     1             1           0        1000
 15  warmabs2(1).Sabund           0     1             1           0        1000
 16  warmabs2(1).Clabund          0     1             1           0        1000
 17  warmabs2(1).Arabund          0     1             1           0        1000
 18  warmabs2(1).Kabund           0     1             1           0        1000
 19  warmabs2(1).Caabund          0     1             1           0        1000
 20  warmabs2(1).Scabund          0     1             1           0        1000
 21  warmabs2(1).Tiabund          0     1             1           0        1000
 22  warmabs2(1).Vabund           0     1             1           0        1000
 23  warmabs2(1).Crabund          0     1             1           0        1000
 24  warmabs2(1).Mnabund          0     1             1           0        1000
 25  warmabs2(1).Feabund          0     1             1           0        1000
 26  warmabs2(1).Coabund          0     1             1           0        1000
 27  warmabs2(1).Niabund          0     1             1           0        1000
 28  warmabs2(1).Cuabund          0     1             1           0        1000
 29  warmabs2(1).Znabund          0     1             1           0        1000
 30  warmabs2(1).write_outfile    0     1             1           0           1
 31  warmabs2(1).outfile_idx      0     1             0           0           1
 32  warmabs2(1).vturb            0     1             0           0       10000  km/s
 33  warmabs2(1).Redshift         0     1             0           0          10
 34  warmabs2(1).autoname_outfile 0     1             1           0           1
```

Starting up XSTARDB

```
isis> (x1, x2) = linear_grid( 1, 40, 1000 );
isis> y = eval_fun( x1, x2 );
 doing setup:           0           1           0           0           0           0           0
 photemis/warmabs v2.22
 xstar version 2.2.1bn18
 Loading Atomic Database...
  Loading Atomic Database...
 Atomic Data Version: 2014-06-09T20:20:15
 initializng database...
  initializng database...
```

Starting up XSTARDB

```
isis> (x1, x2) = linear_grid( 1, 40, 1000 );
isis> y = eval_fun( x1, x2 );
 doing setup:           0           1           0           0           0           0           0
 photemis/warmabs v2.22
 xstar version 2.2.1bn18
 Loading Atomic Database...
  Loading Atomic Database...
 Atomic Data Version: 2014-06-09T20:20:15
 initializng database...
  initializng database...
```

...

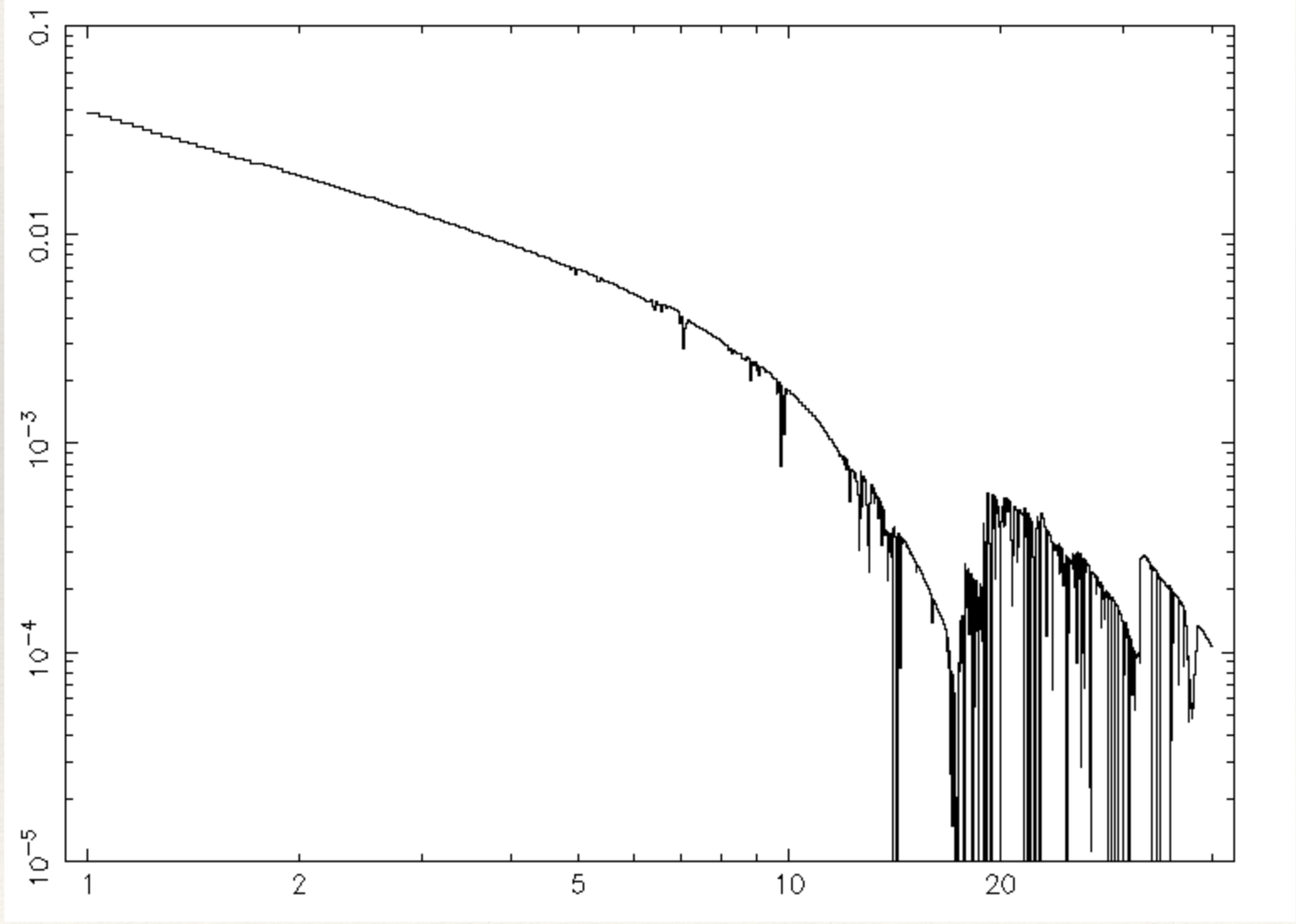
```
 read complete (pops.fits)
isis> xlog; ylog;
isis> yrange(1.e-5, 0.1);
isis> hplot( x1, x2, y );
```

Starting up XSTARDB

```
isis> (x1, x2) = linear_grid( 1, 40, 1000 );
isis> y = eval_fun( x1, x2 );
 doing setup:           0           1           0           0           0           0           0
 photemis/warmabs v2.22
 xstar version 2.2.1bn18
 Loading Atomic Database...
  Loading Atomic Database...
 Atomic Data Version: 2014-06-09T20:20:15
 initializng database...
  initializng database...
```

...

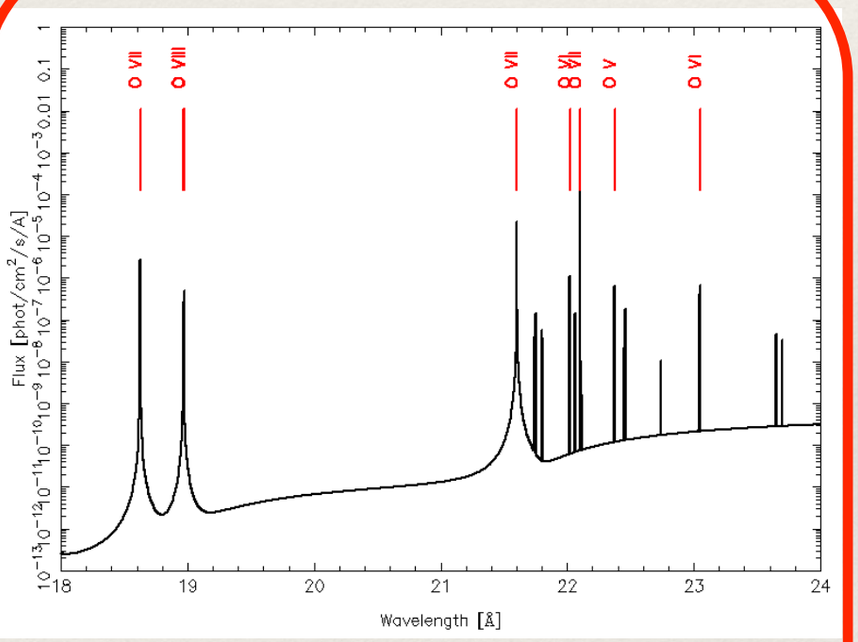
```
 read complete (pops.fits)
isis> xlog; ylog;
isis> yrange(1.e-5, 0.1);
isis> hplot( x1, x2, y );
```



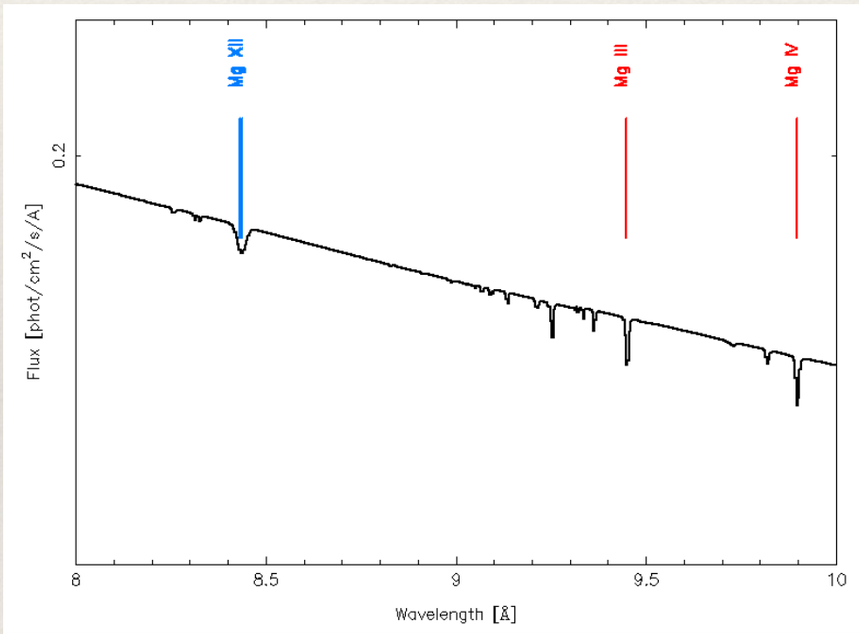
http://space.mit.edu/cxc/analysis/xstardb/

Three essential examples

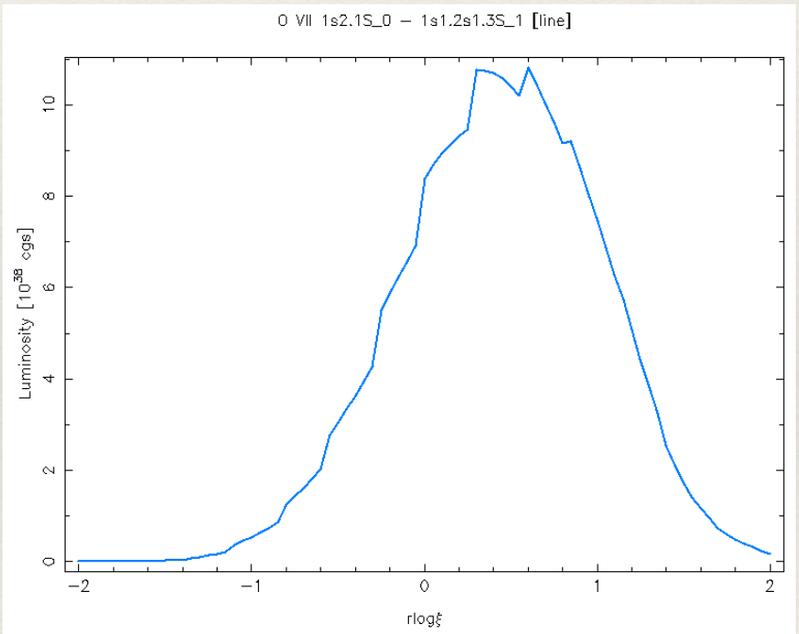
http://space.mit.edu/cxc/analysis/xstardb/examples.html



single photemis model



multiple warmabs models



grid of photemis models
(for line luminosity vs rlogxi)

Example 1: A single *photemis* model

```
isis> fit_fun( "photemis2(1)" );
isis> set_par( "photemis2(1).write_outfile", 1 );
isis> set_par( "photemis2(1).autoname_outfile", 1 );
```

Example 1: A single *photemis* model

```
isis> list_par;
powerlaw(1) * warmabs2(1)
 idx  param                    tie-to  freeze        value         min         max
  1  powerlaw(1).norm             0       0            1           0       1e+10
  2  powerlaw(1).PhoIndex         0       0            1          -2           9
  3  warmabs2(1).column           0       0            0          -3           2  cm^-2
  4  warmabs2(1).rlogxi           0       0            0          -4           5
  5  warmabs2(1).Cabund           0       1            1           0        1000
  6  warmabs2(1).Nabund           0       1            1           0        1000
  7  warmabs2(1).Oabund           0       1            1           0        1000
  8  warmabs2(1).Fabund           0       1            1           0        1000
  9  warmabs2(1).Neabund          0       1            1           0        1000
 10  warmabs2(1).Naabund          0       1            1           0        1000
 11  warmabs2(1).Mgabund          0       1            1           0        1000
 12  warmabs2(1).Alabund          0       1            1           0        1000
 13  warmabs2(1).Siabund          0       1            1           0        1000
 14  warmabs2(1).Pabund           0       1            1           0        1000
 15  warmabs2(1).Sabund           0       1            1           0        1000
 16  warmabs2(1).Clabund          0       1            1           0        1000
 17  warmabs2(1).Arabund          0       1            1           0        1000
 18  warmabs2(1).Kabund           0       1            1           0        1000
 19  warmabs2(1).Caabund          0       1            1           0        1000
 20  warmabs2(1).Scabund          0       1            1           0        1000
 21  warmabs2(1).Tiabund          0       1            1           0        1000
 22  warmabs2(1).Vabund           0       1            1           0        1000
 23  warmabs2(1).Crabund          0       1            1           0        1000
 24  warmabs2(1).Mnabund          0       1            1           0        1000
 25  warmabs2(1).Feabund          0       1            1           0        1000
 26  warmabs2(1).Coabund          0       1            1           0        1000
 27  warmabs2(1).Niabund          0       1            1           0        1000
 28  warmabs2(1).Cuabund          0       1            1           0        1000
 29  warmabs2(1).Znabund          0       1            1           0        1000
 30  warmabs2(1).write_outfile    0       1            1           0           1
 31  warmabs2(1).outfile_lax      0       1            0           0           1
 32  warmabs2(1).vturb            0       1            0           0       10000  km/s
 33  warmabs2(1).Redshift         0       1            0           0          10
 34  warmabs2(1).autoname_outfile 0       1            1           0           1
```

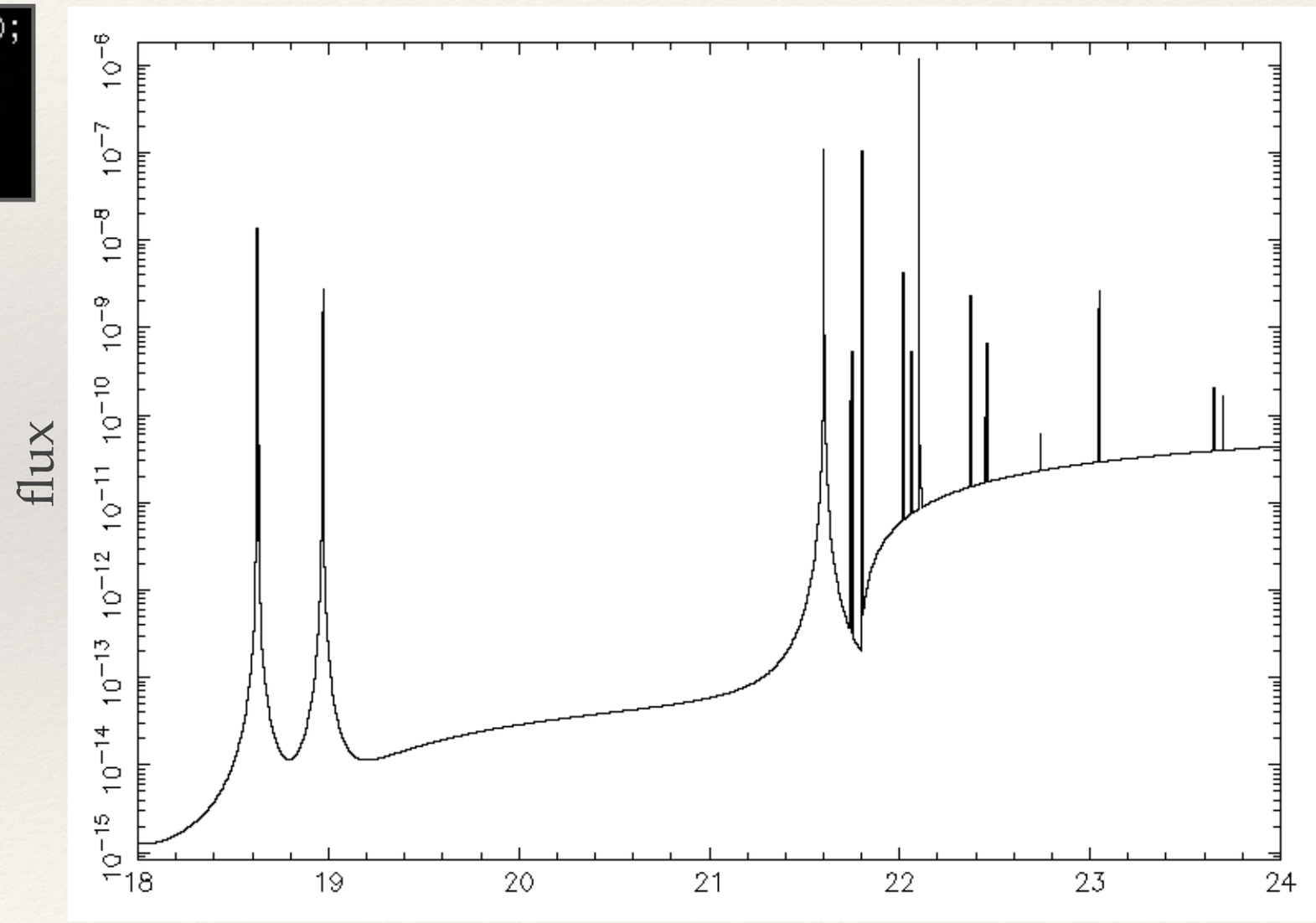
Example 1: A single *photemis* model

```
isis> fit_fun( "photemis2(1)" );
isis> set_par( "photemis2(1).write_outfile", 1 );
isis> set_par( "photemis2(1).autoname_outfile", 1 );
isis> set_par( [3:27], 0 );
isis> set_par( "photemis2(1).Oabund", 1.0 );
```

Example 1: A single *photemis* model

```
isis> fit_fun( "photemis2(1)" );
isis> set_par( "photemis2(1).write_outfile", 1 );
isis> set_par( "photemis2(1).autoname_outfile", 1 );
isis> set_par( [3:27], 0 );
isis> set_par( "photemis2(1).Oabund", 1.0 );
```

```
isis> (x1, x2) = linear_grid( 1, 40, 10000 );
isis> y = eval_fun( x1, x2 );
isis> xlin; ylog;
isis> xrange( 18.0, 24.0 ); yrange();
isis> hplot( x1, x2, y );
```



Example 1: A single *photemis* model

```
isis> db = rd_xstar_output( "photemis_1.fits" );
isis> strongest = xstar_strong(8, db; wmin=18.0, wmax=24.0);
XSTAR_STRONG: Emission model found, sorting by luminosity
```

```
xstar_page_group( db, strongest; sort="luminosity" );
```

```
isis> xstar_page_group( db, strongest; sort="luminosity" );
#     id       ion   lambda    A[s^-1]          f  gl  gu      tau_0        W(A)  L[10^38 cgs]      type                    label
    7154   0   VII  22.1012  1.040e+03  2.284e-10   1   3  0.000e+00 -5.946e-14    6.723e+00      line      1s2.1S -   1s1.2s1.3S
    7197   0   VII  21.6020  3.303e+12  6.930e-01   1   3  0.000e+00 -5.834e-15    6.448e-01      line      1s2.1S -   1s1.2p1.1P
    7198   0   VII  21.8044  3.388e+07  2.173e-05   1   9  0.000e+00 -5.366e-15    5.986e-01      line      1s2.1S -   1s1.2p1.3P
    7187   0   VII  18.6270  9.333e+11  1.456e-01   1   3  0.000e+00 -9.538e-16    9.090e-02      line      1s2.1S -   1s1.3p1.1P
    7329   0  VIII  18.9671  2.566e+12  2.767e-01   2   4  0.000e+00 -1.967e-16    1.908e-02      line   1s1.2S_1/2 - 1s0.2p1.2P_3/2
    7328   0  VIII  18.9725  2.566e+12  1.384e-01   2   2  0.000e+00 -1.903e-16    1.847e-02      line   1s1.2S_1/2 - 1s0.2p1.2P_1/2
    6909   0    VI  23.0500  1.850e+11  7.365e-03   4   2  0.000e+00 -1.462e-16    1.724e-02      line 2s0.2p1.2P_3/2 - 1s1.2s2.2S_1/2
    6884   0    VI  22.0194  3.390e+12  4.926e-01   2   4  0.000e+00 -1.360e-16    1.532e-02      line   2s1.2S_1/2 - 1s1.2s1.2p1.2P_3/2
```

wavelength

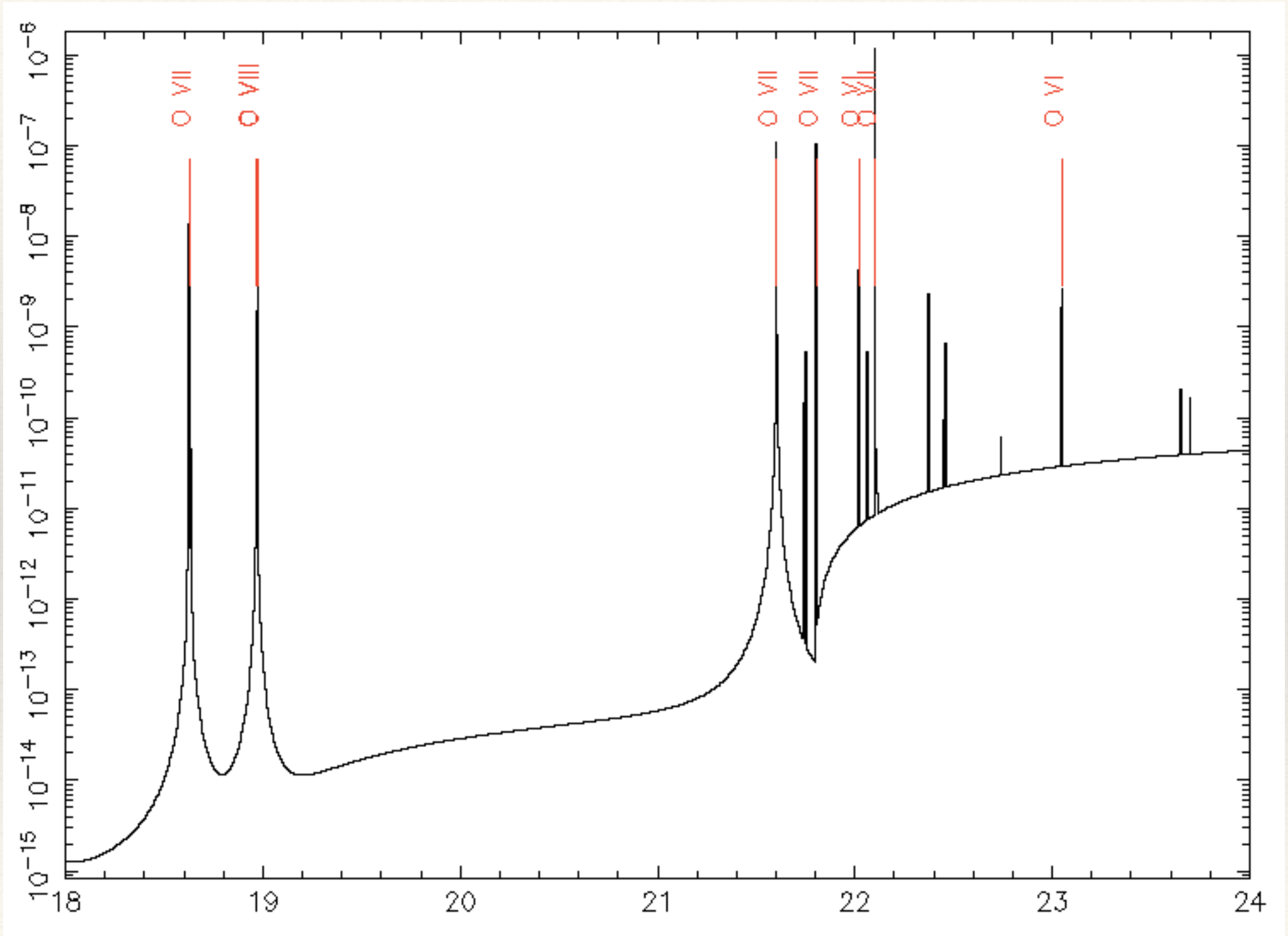
Einstein A

oscillator strength

equiv width

Example 1: A single *photemis* model

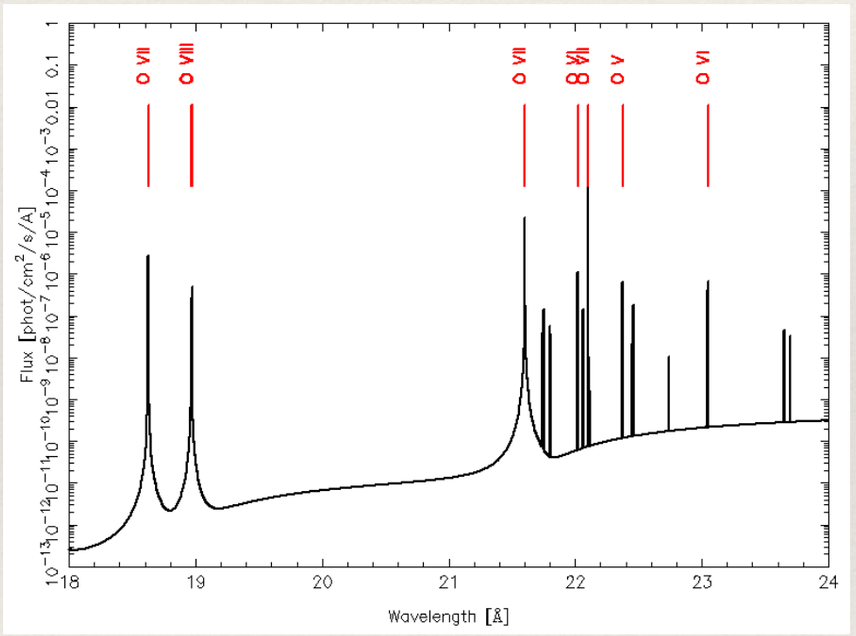
```
xstar_plot_group( db, strongest );
```



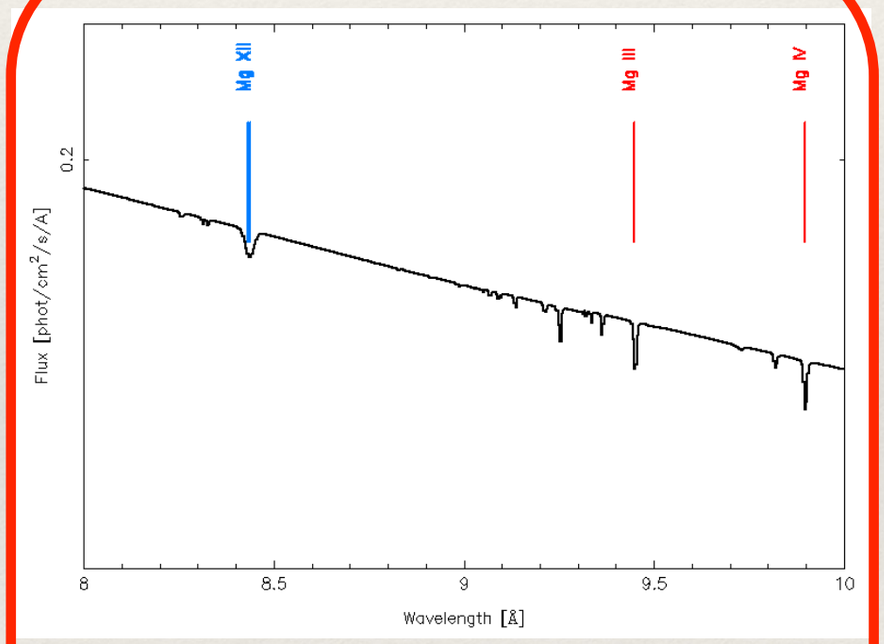
http://space.mit.edu/cxc/analysis/xstardb/

Three essential examples

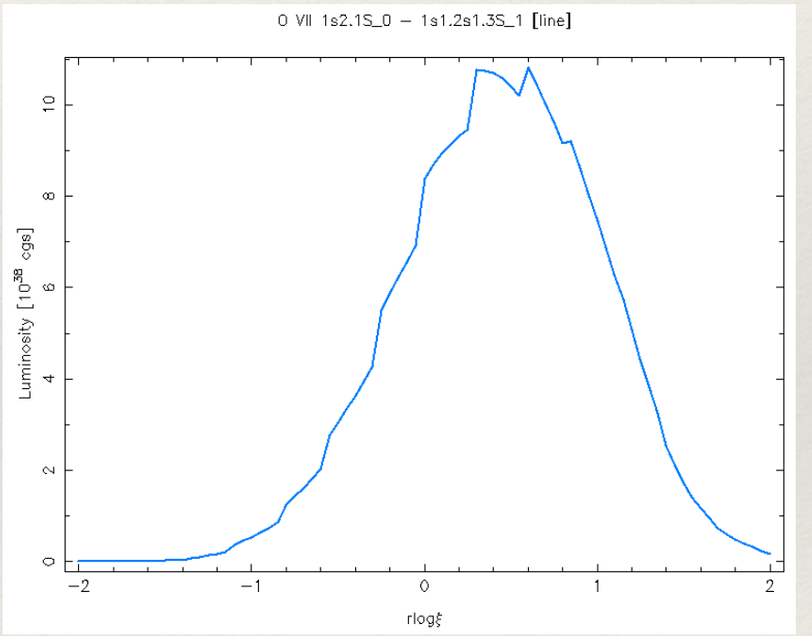
http://space.mit.edu/cxc/analysis/xstardb/examples.html



single photemis model



multiple warmabs models



grid of photemis models
(for line luminosity vs rlogxi)

Example 2: Multiple *warmabs* models

Example of two warm absorbers in spectrum of a Seyfert I galaxy
Holczer et al. (2010) ApJ 708, 981

```
isis> fit_fun( "powerlaw(1) * (warmabs2(1) + warmabs2(2))" );
isis> set_par("warmabs2(*).write_outfile", 1);
isis> set_par("warmabs2(*).autoname_outfile", 1);
isis> set_par( "warmabs2(1).column", -1 );
isis> set_par( "warmabs2(1).rlogxi", -1.0 );
isis> set_par( "warmabs2(1).vturb", 100.0 );
isis> set_par( "warmabs2(1).Redshift", 0.0 );
isis> set_par( "warmabs2(2).column", log10(8.1) );
isis> set_par( "warmabs2(2).rlogxi", 3.8 );
isis> set_par( "warmabs2(2).vturb", 500.0 );
isis> set_par( "warmabs2(2).Redshift", 0.1 );
isis> y = eval_fun( x1, x2 );
```

For sake of example,
I set this to a high value
compared to paper

Example 2: Multiple *warmabs* models

```
isis> z = [ get_par("warmabs2(1).Redshift"), get_par("warmabs2(2).Redshift") ];
isis> db_m = xstar_merge( [ "warmabs_1.fits", "warmabs_2.fits" ] );
isis> slines = xstar_strong( 4, db_m; wmin=8.0, wmax=10.0, redshift=z );
XSTAR_STRONG: Absorption model found, sorting by equivalent width
```

Example 2: Multiple *warmabs* models

```
isis> z = [ get_par("warmabs2(1).Redshift"), get_par("warmabs2(2).Redshift") ];
isis> db_m = xstar_merge( [ "warmabs_1.fits", "warmabs_2.fits" ] );
isis> slines = xstar_strong( 4, db_m; wmin=8.0, wmax=10.0, redshift=z );
XSTAR_STRONG: Absorption model found, sorting by equivalent width
```

Example 2: Multiple *warmabs* models

```
isis> z = [ get_par("warmabs2(1).Redshift"), get_par("warmabs2(2).Redshift") ];
isis> db_m = xstar_merge( [ "warmabs_1.fits", "warmabs_2.fits" ] );
isis> slines = xstar_strong( 4, db_m; wmin=8.0, wmax=10.0, redshift=z );
XSTAR_STRONG: Absorption model found, sorting by equivalent width
```

Example 2: Multiple *warmabs* models

```
isis> z = [ get_par("warmabs2(1).Redshift"), get_par("warmabs2(2).Redshift") ];
isis> db_m = xstar_merge( [ "warmabs_1.fits", "warmabs_2.fits" ] );
isis> slines = xstar_strong( 4, db_m; wmin=8.0, wmax=10.0, redshift=z );
XSTAR_STRONG: Absorption model found, sorting by equivalent width
```

```
xstar_page_group( db_m, slines );
```

```
isis> xstar_page_group( db_m, slines );
#     id      ion   lambda    A[s^-1]          f  gl  gu      tau_0      W(A) L[10^38 cgs]      type                    label                origin
   13674  Mg   XII   8.4192   .299e+13  2.760e-01   2   4  2.596e-02  1.485e-01    0.000e+00      line   1s1.2S_1/2 - 1s0.2p1.2P_3/2   warmabs_2.fits
   13673  Mg   XII   8.4246   .299e+13  1.382e-01   2   2  1.276e-02  7.471e-02    0.000e+00      line   1s1.2S_1/2 - 1s0.2p1.2P_1/2   warmabs_2.fits
   11380  Mg   III   9.4470   .000e+12  4.012e-02   1   3  7.341e-02  1.091e-01    0.000e+00      line     2p6.1S_0 - 1s1.2s2.2p6.3p1.1P_1    warmabs_1.fits
   11574  Mg    IV   9.8951   .160e+13  8.511e-02   4   2  7.859e-02  1.033e-01    0.000e+00      line   2p5.2P_3/2 - 1s1.2s2.2p6.2S_1/2    warmabs_1.fits
```

wavelength
(rest frame)

notice extra column:
warmabs system to which line belongs

Example 2: Multiple *warmabs* models

```
isis> z = [ get_par("warmabs2(1).Redshift"), get_par("warmabs2(2).Redshift") ];
isis> db_m = xstar_merge( [ "warmabs_1.fits", "warmabs_2.fits" ] );
isis> slines = xstar_strong( 4, db_m; wmin=8.0, wmax=10.0, redshift=z );
XSTAR_STRONG: Absorption model found, sorting by equivalent width
```

```
isis> print( db_m );
{transition=Integer_Type[3248],
 type=String_Type[3248],
 ion=String_Type[3248],
 wavelength=Double_Type[3248],
 tau0=Float_Type[3248],
 tau0grid=Float_Type[3248],
 ew=Float_Type[3248],
 luminosity=Float_Type[3248],
 lower_level=String_Type[3248],
 upper_level=String_Type[3248],
 a_ij=Float_Type[3248],
 f_ij=Float_Type[3248],
 g_lo=Float_Type[3248],
 g_up=Float_Type[3248],
 ind_lo=Integer_Type[3248],
 ind_up=Integer_Type[3248],
 ind_ion=Integer_Type[3248],
 model_name=String_Type[2],
 params=Struct_Type[2],
 Z=Integer_Type[3248],
 q=Integer_Type[3248],
 transition_name=String_Type[3248],
 filename=String_Type[2],
 origin_file=Integer_Type[3248]}
```

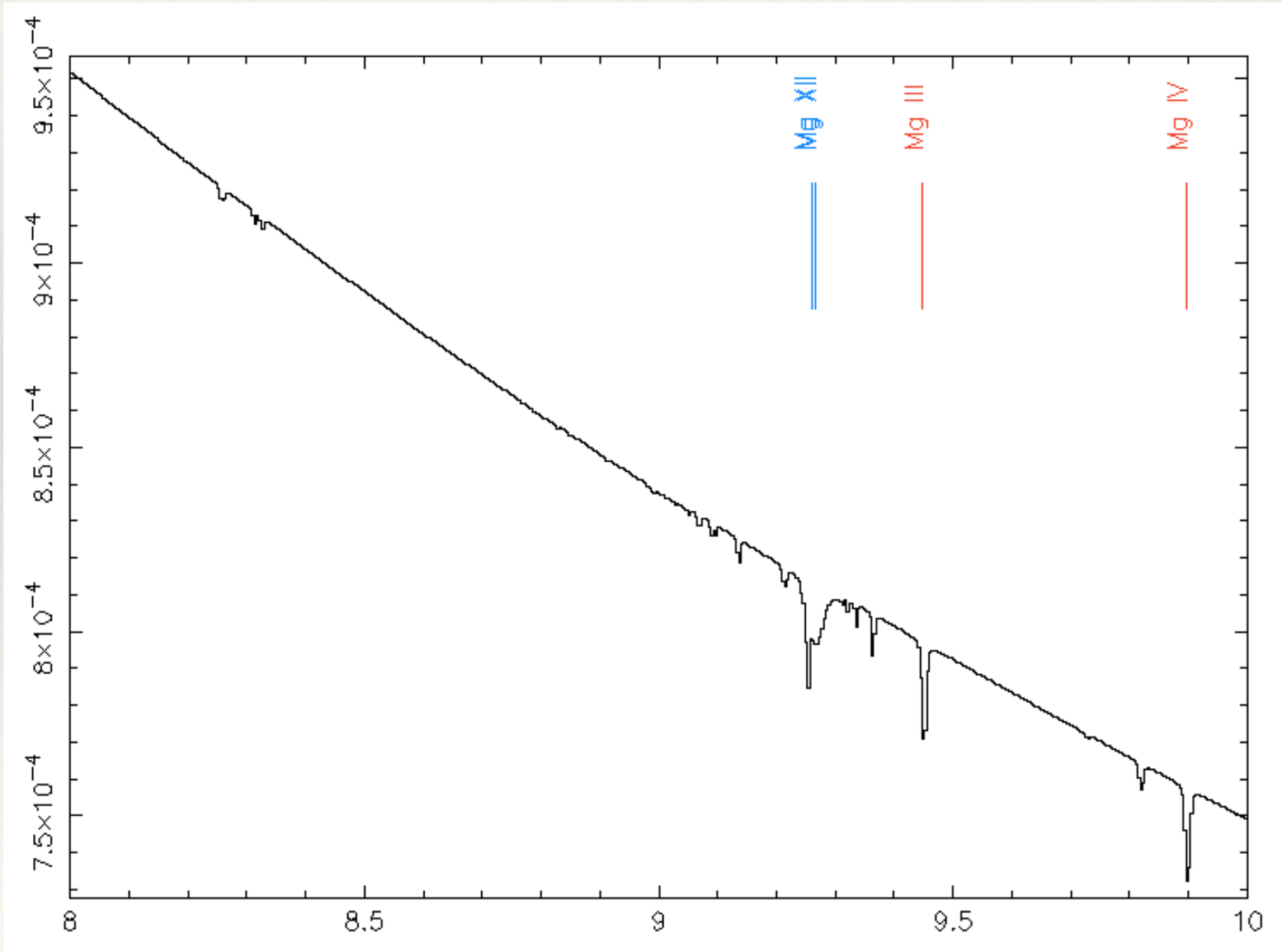
keeps track of which file
each line belongs to (integer array)

Example 2: Multiple *warmabs* models

```
isis> l1 = slines[ where(db_m.origin_file[slines] == 0) ];
isis> l2 = slines[ where(db_m.origin_file[slines] == 1) ];
```

Example 2: Multiple *warmabs* models

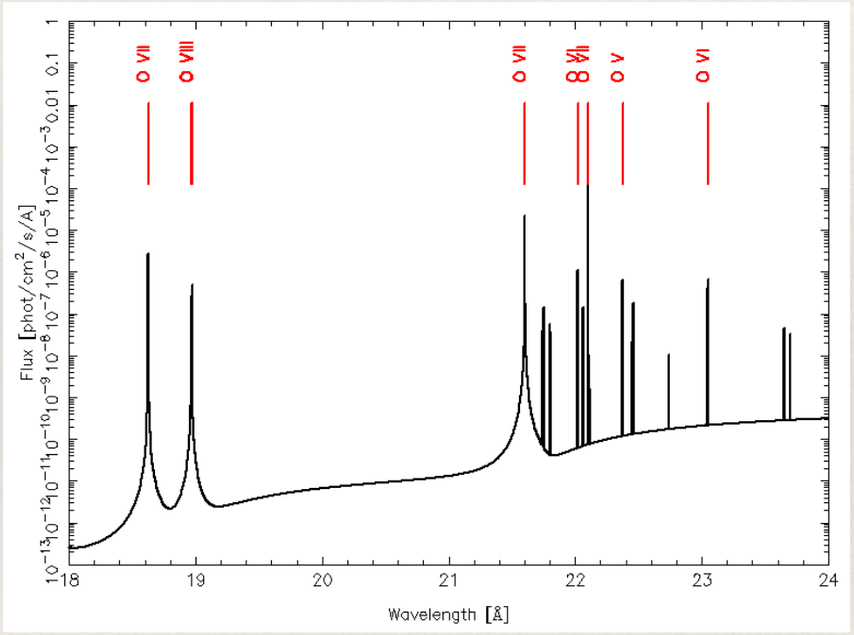
```
isis> l1 = slines[ where(db_m.origin_file[slines] == 0) ];
isis> l2 = slines[ where(db_m.origin_file[slines] == 1) ];
isis> lstyle = line_label_default_style();
isis> lstyle.bottom_frac = 0.7;
isis> lstyle.top_frac = 0.85;
isis> xstar_plot_group( db_m, l1, 2, lstyle, z[0] );
isis> xstar_plot_group( db_m, l2, 11, lstyle, z[1] );
```



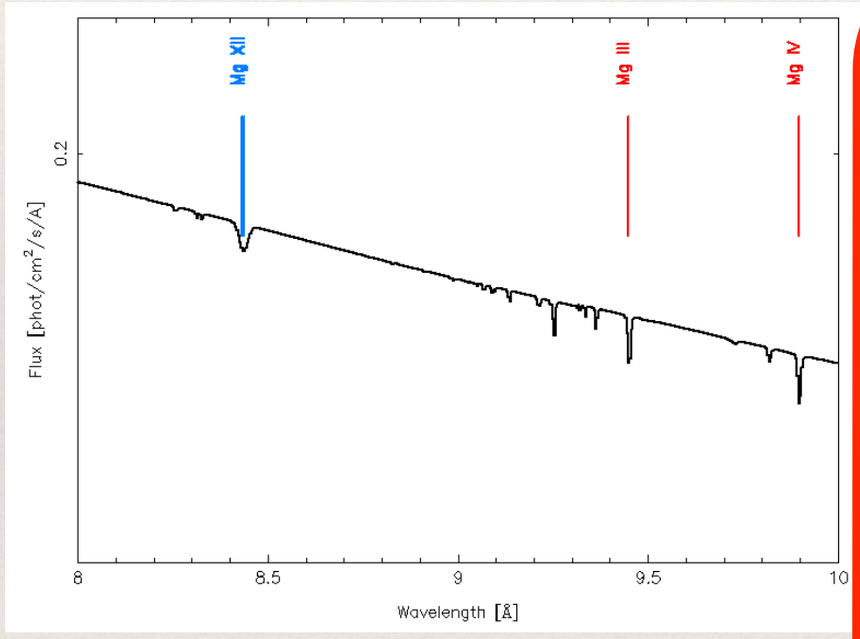
http://space.mit.edu/cxc/analysis/xstardb/

Three essential examples

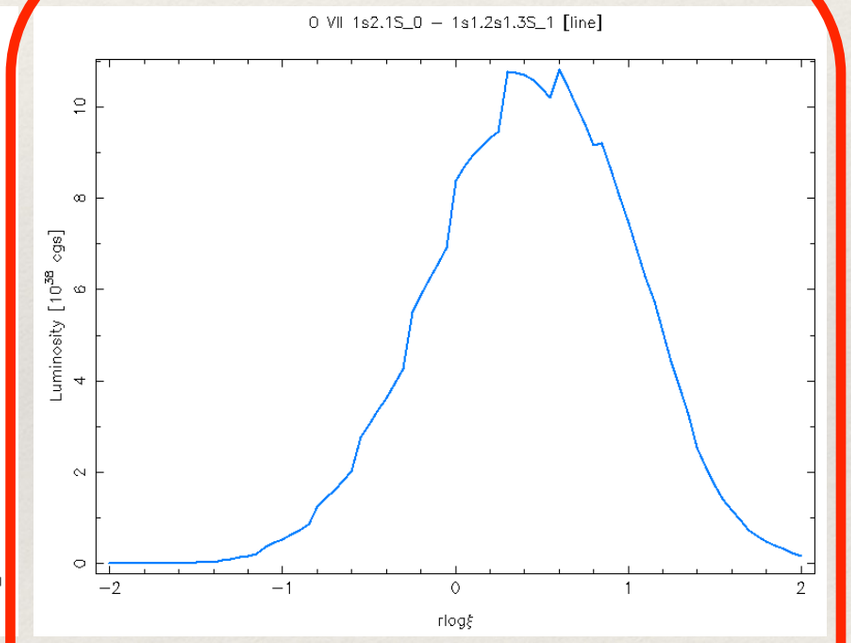
http://space.mit.edu/cxc/analysis/xstardb/examples.html



single photemis model



multiple warmabs models



grid of photemis models
(for line luminosity vs rlogxi)

Example 3: A grid of models

```
(x1, x2)       = linear_grid( 1, 40, 10000 );
model_binning = struct{ bin_lo=x1, bin_hi=x2 }
```

Example 3: A grid of models

```
(x1, x2)       = linear_grid( 1, 40, 10000 );
model_binning = struct{ bin_lo=x1, bin_hi=x2 }
model_info     = @_default_model_info;
set_struct_fields( model_info, “photemis”, “rlogxi”,
                   -2.0, 2.0, 0.05, model_binning );
```

Example 3: A grid of models

```
(x1, x2)       = linear_grid( 1, 40, 10000 );
model_binning = struct{ bin_lo=x1, bin_hi=x2 }
model_info     = @_default_model_info;
set_struct_fields( model_info, “photemis”, “rlogxi”,
                   -2.0, 2.0, 0.05, model_binning );
```

```
isis> print(model_info);
{mname="photemis",
 pname="rlogxi",
 min=-2.0,
 max=2.0,
 step=0.05,
 bins=Struct_Type with 2 fields}
```

Example 3: A grid of models

```
(x1, x2)       = linear_grid( 1, 40, 10000 );
model_binning = struct{ bin_lo=x1, bin_hi=x2 }
model_info     = @_default_model_info;
set_struct_fields( model_info, “photemis”, “rlogxi”,
                   -2.0, 2.0, 0.05, model_binning );
```

```
xstar_run_model_grid( model_info, “/my/rlogxi_example” );

fgrid   = glob( “/my/rlogxi_example/photemis_*.fits” );
fgrid   = fgrid[ array_sort(fgrid) ];
pe_grid = xstar_load_tables( fgrid );
```

Example 3: A grid of models

```
o_vii = where( xstar_el_ion(pe_grid.mdb, O, 7) );

xstar_page_grid( pe_grid, o_vii );
```

Example 3: A grid of models

```
o_vii = where( xstar_el_ion(pe_grid.mdb, O, 7) );

xstar_page_grid( pe_grid, o_vii );
```

# uid		ion	lambda	A[s^-1]	f	gl	gu	type	label
350010241	0	VII	16.7705	0.000e+00	0.000e+00	1	0	edge/rrc	1s2.1S_0 - continuum
350010240	0	VII	16.7805	3.103e+06	3.928e-07	1	3	line	1s2.1S_0 - 1s1.200p1.1P
350010040	0	VII	17.3960	1.963e+11	2.671e-02	1	3	line	1s2.1S_0 - 1s1.5p1.1P_1
350010029	0	VII	17.7680	3.867e+11	5.488e-02	1	3	line	1s2.1S_0 - 1s1.4p1.1P_1
350010017	0	VII	18.6270	9.333e+11	1.456e-01	1	3	line	1s2.1S_0 - 1s1.3p1.1P_1
350010007	0	VII	21.6020	3.303e+12	6.930e-01	1	3	line	1s2.1S_0 - 1s1.2p1.1P_1
350010003	0	VII	21.8044	3.100e+04	2.209e-09	1	1	line	1s2.1S_0 - 1s1.2p1.3P_0
350010005	0	VII	21.8044	3.100e+04	1.104e-08	1	5	line	1s2.1S_0 - 1s1.2p1.3P_2
350010004	0	VII	21.8070	3.100e+04	6.628e-09	1	3	line	1s2.1S_0 - 1s1.2p1.3P_1
350010002	0	VII	22.1012	4.000e+01	8.784e-12	1	3	line	1s2.1S_0 - 1s1.2s1.3S_1

Example 3: A grid of models

```
o_vii = where( xstar_el_ion(pe_grid.mdb, O, 7) );

xstar_page_grid( pe_grid, o_vii );
```

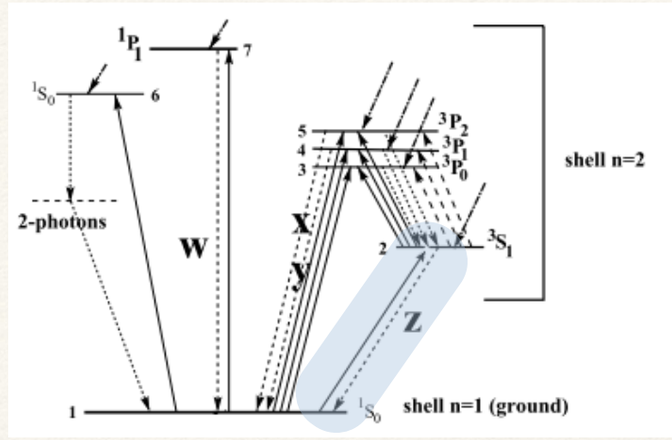
#	uid		ion	lambda	A[s^-1]	f	gl	gu	type	label
	350010241	0	VII	16.7705	0.000e+00	0.000e+00	1	0	edge/rrc	1s2.1S_0 - continuum
	350010240	0	VII	16.7805	3.103e+06	3.928e-07	1	3	line	1s2.1S_0 - 1s1.200p1.1P
	350010040	0	VII	17.3960	1.963e+11	2.671e-02	1	3	line	1s2.1S_0 - 1s1.5p1.1P_1
	350010029	0	VII	17.7680	3.867e+11	5.488e-02	1	3	line	1s2.1S_0 - 1s1.4p1.1P_1
	350010017	0	VII	18.6270	9.333e+11	1.456e-01	1	3	line	1s2.1S_0 - 1s1.3p1.1P_1
	350010007	0	VII	21.6020	3.303e+12	6.930e-01	1	3	line	1s2.1S_0 - 1s1.2p1.1P_1
	350010003	0	VII	21.8044	3.100e+04	2.209e-09	1	1	line	1s2.1S_0 - 1s1.2p1.3P_0
	350010005	0	VII	21.8044	3.100e+04	1.104e-08	1	5	line	1s2.1S_0 - 1s1.2p1.3P_2
	350010004	0	VII	21.8070	3.100e+04	6.628e-09	1	3	line	1s2.1S_0 - 1s1.2p1.3P_1
	350010002	0	VII	22.1012	4.000e+01	8.784e-12	1	3	line	1s2.1S_0 - 1s1.2s1.3S_1

To keep track among fits files, features are assigned an id based on ion and level id from **xstarlevels.txt**

```
35   = ion number (e.g. 01 for HI, 02 for HII)
001  = lower level id
0241 = upper level id
```

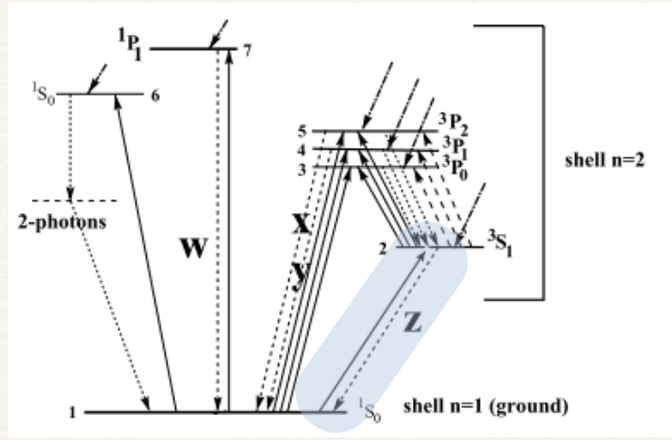
ftp://legacy.gsfc.nasa.gov/software/plasma_codes/xstar/rates/xstarlevels.text

Example 3: A grid of models



# uid		ion	lambda	A[s^-1]	f	gl	gu	type	label
350010241	0	VII	16.7705	0.000e+00	0.000e+00	1	0	edge/rrc	1s2.1S_0 - continuum
350010240	0	VII	16.7805	3.103e+06	3.928e-07	1	3	line	1s2.1S_0 - 1s1.200p1.1P
350010040	0	VII	17.3960	1.963e+11	2.671e-02	1	3	line	1s2.1S_0 - 1s1.5p1.1P_1
350010029	0	VII	17.7680	3.867e+11	5.488e-02	1	3	line	1s2.1S_0 - 1s1.4p1.1P_1
350010017	0	VII	18.6270	9.333e+11	1.456e-01	1	3	line	1s2.1S_0 - 1s1.3p1.1P_1
350010007	0	VII	21.6020	3.303e+12	6.930e-01	1	3	line	1s2.1S_0 - 1s1.2p1.1P_1
350010003	0	VII	21.8044	3.100e+04	2.209e-09	1	1	line	1s2.1S_0 - 1s1.2p1.3P_0
350010005	0	VII	21.8044	3.100e+04	1.104e-08	1	5	line	1s2.1S_0 - 1s1.2p1.3P_2
350010004	0	VII	21.8070	3.100e+04	6.628e-09	1	3	line	1s2.1S_0 - 1s1.2p1.3P_1
350010002	0	VII	22.1012	4.000e+01	8.784e-12	1	3	line	1s2.1S_0 - 1s1.2s1.3S_1

Example 3: A grid of models

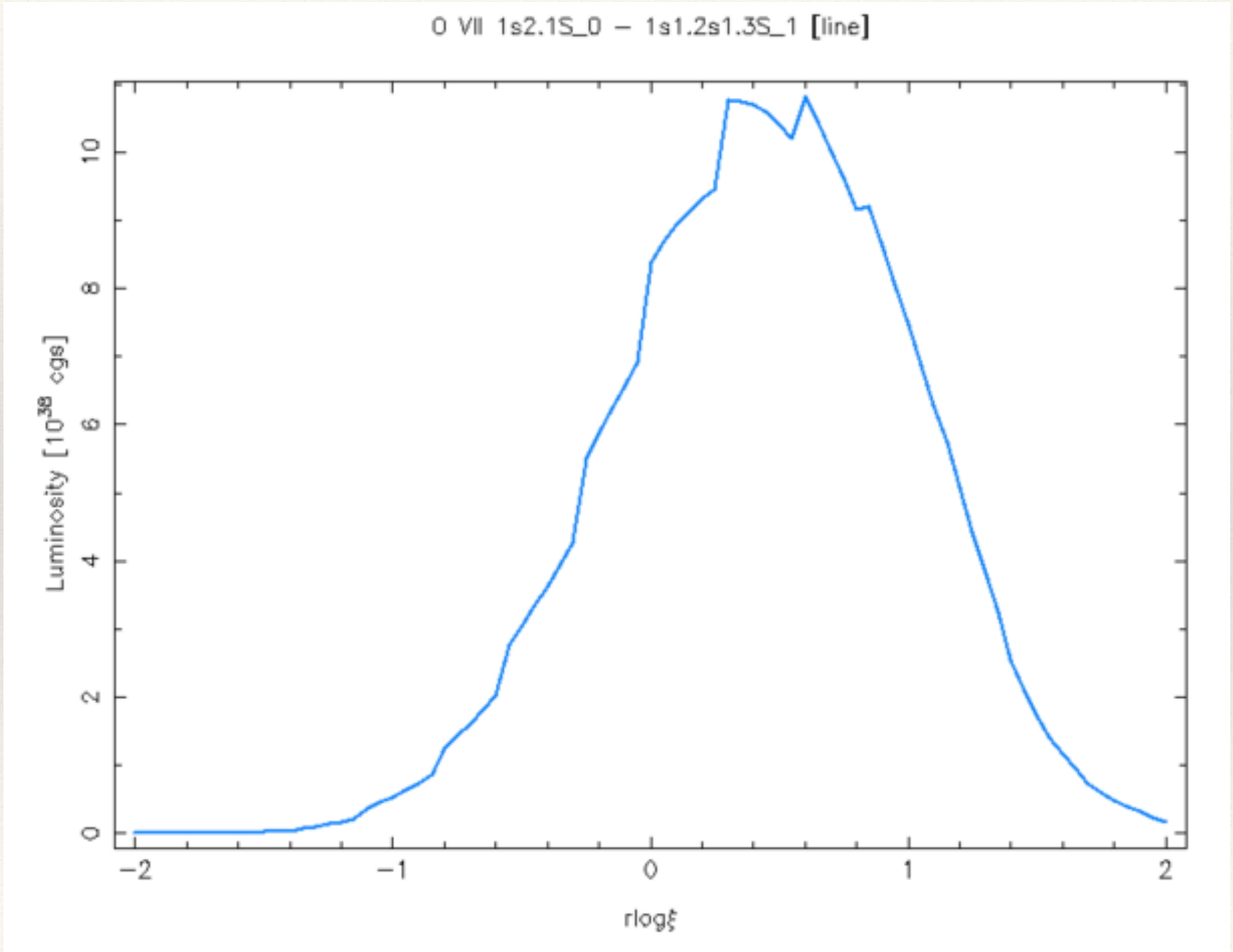


# uid		ion	lambda	A[s^-1]	f	gl	gu	type	label
350010241	0	VII	16.7705	0.000e+00	0.000e+00	1	0	edge/rrc	1s2.1S_0 - continuum
350010240	0	VII	16.7805	3.103e+06	3.928e-07	1	3	line	1s2.1S_0 - 1s1.200p1.1P
350010040	0	VII	17.3960	1.963e+11	2.671e-02	1	3	line	1s2.1S_0 - 1s1.5p1.1P_1
350010029	0	VII	17.7680	3.867e+11	5.488e-02	1	3	line	1s2.1S_0 - 1s1.4p1.1P_1
350010017	0	VII	18.6270	9.333e+11	1.456e-01	1	3	line	1s2.1S_0 - 1s1.3p1.1P_1
350010007	0	VII	21.6020	3.303e+12	6.930e-01	1	3	line	1s2.1S_0 - 1s1.2p1.1P_1
350010003	0	VII	21.8044	3.100e+04	2.209e-09	1	1	line	1s2.1S_0 - 1s1.2p1.3P_0
350010005	0	VII	21.8044	3.100e+04	1.104e-08	1	5	line	1s2.1S_0 - 1s1.2p1.3P_2
350010004	0	VII	21.8070	3.100e+04	6.628e-09	1	3	line	1s2.1S_0 - 1s1.2p1.3P_1
350010002	0	VII	22.1012	4.000e+01	8.784e-12	1	3	line	1s2.1S_0 - 1s1.2s1.3S_1

```
o_vii_F     = where(xstar_trans( pe_grid.mdb, O, 7, 1, 2 ));
o_vii_F_lum = xstar_line_prop( pe_grid, o_vii_F, "luminosity" );
```

Example 3: A grid of models

```
rlogxi = xstar_get_grid_par( pe_grid, "rlogxi" );
plot( rlogxi, ovii_F_lum, 11 );
```



Caveats

- Need XSTAR version 2.22 or higher
- Not responsible for performance of XSTAR models
- RRC / Edges are computed from templates, so strong features from non He-like or H-like atoms (e.g. OV K-alpha) do not appear in line list
- Can supply your own ion population file (pops.fits), but cannot change between different files within a single ISIS session

Caveats

- Need XSTAR version 2.22 or higher
- Not responsible for performance of XSTAR models
- RRC / Edges are computed from templates, so strong features from non He-like or H-like atoms (e.g. OV K-alpha) do not appear in line list **`ismabs + ismdust`**
- Can supply your own ion population file (pops.fits), but cannot change between different files within a single ISIS session

Resources

http://space.mit.edu/cxc/analysis/xstardb/

http://space.mit.edu/cxc/analysis/xstardb/examples

- Master list of ISIS talks and tutorials, including XSTARDB -

http://space.mit.edu/cxc/isis2015/

- Source Code -

https://github.com/eblur/xstardb

^ see documentation folder for tutorial ISIS scripts

lia@space.mit.edu
dph@space.mit.edu