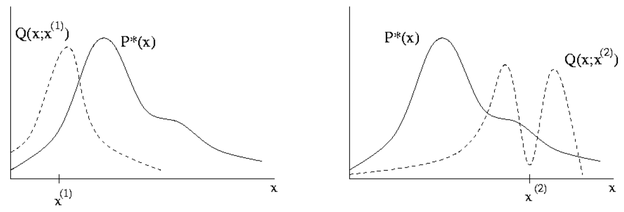
New Features in ISIS for Low- and High-Resolution Spectroscopy

Michael A. Nowak[1], David Huenemoerder[1], Victoria Grinberg[1], Lia Corrales[1], Jörn Wilms[2]

[1]MIT Kavli Institute, [2]Dr. Karl Remeis-Sternwarte and ECAP, Germany

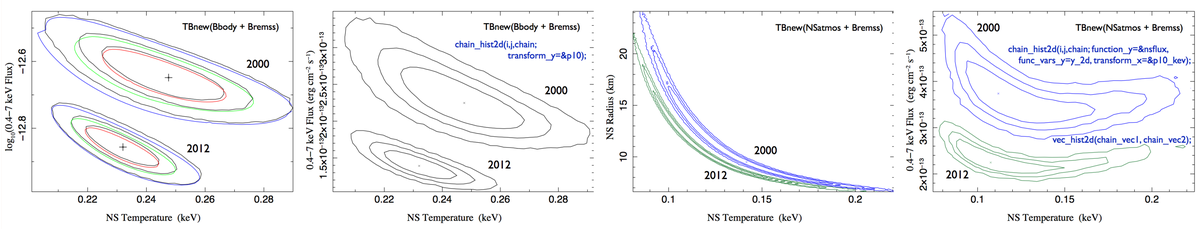
The *Interactive Spectral Interpretation System* (*ISIS*; Houck & Denicola 2000) is a highly flexible, scriptable, and extensible data analysis system. Originally developed as a tool for *Chandra High Energy Transmission Gratings* (*HETG*) analysis – it provides the core of the *Transmission Gratings Catalog* (*TGCat*) – it has become a general purpose analysis system. *ISIS* is a superset of *XSPEC*, with access to all *XSPEC* models including user local models and having analogs of all core *XSPEC* functionality. However, *ISIS* also has a wide range of tools developed for multi-wavelength spectroscopy, timing analysis, data simulation, and additional tools for high resolution spectroscopy. Furthermore, *ISIS* has well-developed tools for "transparent" user parallelization of custom codes. Here we highlight some recent *ISIS* tools that we have developed. This includes a parallelized Markov Chain Monte Carlo (MCMC) code, based upon the Goodman-Weare method, sophisticated interfaces to plasma codes allowing users access to the underlying atomic physics information, and an under development module to provide "simple" descriptions of high resolution spectroscopic data that are more sophisticated than simply adding successive gaussians to a fit, but far less involved than a full plasma-code description. The latter is being designed for simple, preliminary assessments of high resolution spectroscopic data, prior to detailed analysis with more sophisticated, but typically much slower running, code.

Markov Chain Monte Carlo (MCMC)

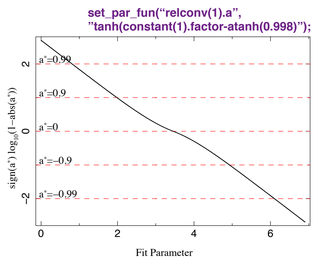


Each set of parameters can be considered a vector, x, in a multi-dimensional parameter space. Changes to this vector are generated randomly via a "proposition" $Q(x,x^{(1)})$. The probability of the model (a function of either the χ^2 or the Cash statistic,whichever has been chosen by the user) is compared between the new and old locations, along with the probability of the proposition.

- Error bar searches for X-ray model fits are often based upon $\Delta\chi^2$ searches for one or two parameters. But what if parameter space is complicated, or you want more complex parameter correlations? Ideally, we want the parameter probability distribution: $P(p_1,p_2,p_3,\ldots)$.
- Markov Chain Monte Carlo (MCMC) provides one solution. We have adapted the formalism of Goodman & Weare (2010), following the implementation by Foreman-Mackey et al. (2013), for use in *ISIS*.
- Take a calculable probability, P(x), a "proposition", $Q(x,x^{(1)})$, plus an initial distribution of "walkers", $x^{(1)}$. Here the walkers are vectors of model parameter values. Draw a new position for the walkers, $x^{(1)} \rightarrow x$. If P(x) increases, accept the position, otherwise, accept it with probability $P(x)Q(x,x^{(1)})/P(x^{(1)})Q(x^{(1)},x)$.
- The *ISIS* implementation creates an ensemble of walkers from an existing parameter file *or a set of parameter files* (for cases where there are multiple local fit statistics minima to explore), and distributes the parameters between the min/max values in the input file(s). The user controls the number of walkers and whether these initial distributions are uniform or Gaussian distributed.
- The group of walkers is divided into two sets, $x^{(0)}$ and $x^{(1)}$, and a loop over k elements is used to update: $x_k^{(0)} = x_j^{(1)} + z\,(x_k^{(0)} - x_j^{(1)})$, where j is randomly drawn and z is a scalar drawn from a probability distribution $P(z) \propto z^{-1/2}$. (The user can choose the range over which z is drawn.) A similar loop over j elements is then performed: $x_j^{(1)} = x_k^{(0)} + z\,(x_j^{(1)} - x_k^{(0)})$. These new "parameter vectors" are either accepted or rejected, and the process is repeated.
- Since these steps are independent, the process can be parallelized. *For over 8 years, ISIS has provided easy to use utilities for "transparent" parallelization of user scripts*. (Any scripted loop where the calculations can be written as a function and they are independent from step to step can be instantly parallelized via the *ISIS* `parallel_map` function; however, routines for more sophisticated parallelization exist.) Scriptable parallelization routines were first implemented in *ISIS* for standard confidence contour routines (`conf_loop`); the MCMC parallelization draws heavily from that example.
- The MCMC module also provides routines for creating 1-D and 2-D confidence regions from the resulting chains, and furthermore the mathematical nature of *S-lang* allows for straightforward manipulation of these chains to create probability products for values derived from the initial fit variables, e.g., flux or equivalent width.



Far left: A comparison of two parameter confidence contours (68%,90%, 99%) for a joint fit of two quiescent neutron star datasets. (Neutral column and neutron star radius were tied between the two datasets). The fits were performed using the `cflux` model (replacing the blackbody normalization), which has the log of the model flux in a given energy band as its parameter. One set of contours (smooth, colored contours) were generated via the more traditional $\Delta\chi^2$ search, while the other set was generated from marginalizing over uninteresting parameters from the MCMC posterior probability. Middle left: *ISIS* is scriptable in *S-lang*, which has vector based math intrinsic to the language. It is therefore very straightforward to apply mathematical transformations to the resulting probability distributions. This has been implemented as "on the fly" transformations in 1- and 2-D MCMC probability distributions. Middle right: MCMC generated confidence contours for the same datasets using a more complicated continuum model, i.e., an absorbed neutron star atmosphere plus Bremsstrahlung emission. The neutron star atmosphere model does not have a simple normalization parameter; therefore, the `cflux` model cannot be applied. It is, however, possible to write a simple *S-lang* function to calculate the resulting neutron star atmosphere flux, but it can be slow for "on the fly" use. Far right: For such slower functions, the *ISIS* MCMC implementation allows transformations to be applied to the chains themselves, and then used in specialized functions to create 1-D and 2-D confidence contours from these transformed chains.

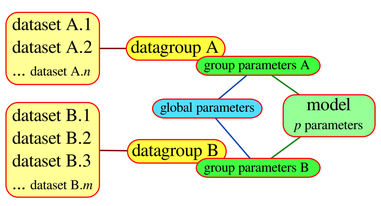


Transformation of variables is a common technique when fitting with *ISIS* that is especially useful for MCMC. Quasi-linear parameter steps that lead to comparable model changes will be easier for the walkers to follow. An example of this is black hole spin in relativistic line models where changes from a*=0.998 to 0.99 to 0.9 can make large line profile differences, but a change from a*=0.9 to 0.5 makes a small difference. In the MCMC one can use a "linearized" parameter, p, where a*=tanh(p-atanh(0.998)). This transformation can then be applied to the resulting chain to obtain the desired spin parameter.

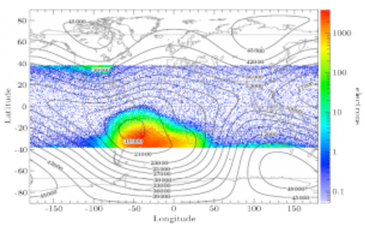
- A recent addition to the *ISIS* MCMC code is 1-D Gaussian priors for any fit parameters.
- Future enhancements will generalize these priors to a 2-D covariance matrix..
- Transparent parallelization of MCMC is for single, multi-core machines. A version implemented for multi-node MPI systems is under development.
- Code available on the Remeis *ISIS* scripts web page.

Remeis ISIS Scripts

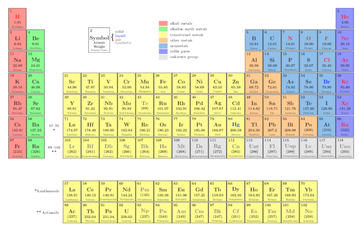
- The Remeis Observatory *ISIS* scripts repository is a collection of *S-lang* software containing nearly 1000 functions, including the MCMC and simple line fitting codes described here. It is an actively developed *git*-repository, with all changes and updates logged on the website. The collection can be downloaded directly from the website, or cloned from the repository.
- The Remeis *ISIS* scripts are the de facto repository for previously developed tools, such as the *S-lang ISIS Timing Analysis Routines* (*SITAR*), but they contain many more tools than these (including further timing analysis routines from Katja Pottschmidt), covering a wide range of functionality.
- The routines contain a full suite of functions for saving spectral analyses to, and fully restoring them from, FITS files (e.g, `fits_save_fit`). These save files include information such as: all paths to the data, the spectral grouping and noticed energy ranges, parameter values, calculated error bars, etc.
- Routines exist to manage fits to and error searches on multiple sets of data considered simultaneously (`simfit`; Kühnel et al. 2015, 2016). These routines are designed for situations more complex (see figure) than can be easily addressed via simple uses of *Isis_Active_Dataset*.
- Parallelized error bar searches *using a computer cluster*, via *MPI* and the *SLmpi* module.
- Monte Carlo calculations of model parameter significances, via simulations of the data and model (`mc_sig`).
- Sophisticated, publication quality plotting using the *SLXfig* module. All examples presented below have been created at Remeis with *SLXfig* tools.



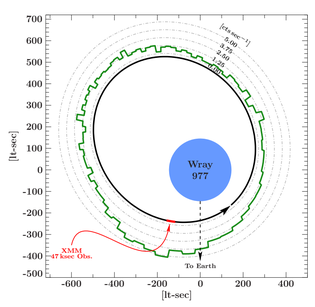
Schematic of the data management plan implemented by `simfit`. Imagine, a multi-spacecraft, multi-date dataset for a Galactic black hole binary. On a single date, many components might be fittable in common, but others may not be (e.g., the dust halo is spatially resolved by *Chandra*, but not *Suzaku*). Between dates, many model parameters might need to change, but parameters such as mass and spin remain constant. `simfit` manages these complexity levels.



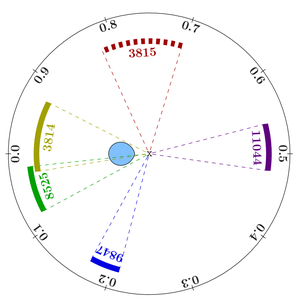
A 2-D histogram map of the South Atlantic Anomaly (SAA), overplotted on the iso-magnetic field lines of the Earth's magnetic field. (Plot by Natalie Hell.)



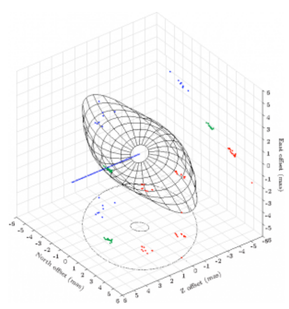
A periodic table, created with data from the "X-ray Data Booklet 2009". (Plot by Natalie Hell.)



RXTE-All Sky Monitor lightcurve (green) of the X-ray binary GX301-2, plotted along the orbit of the binary system as seen from above. (Image by Felix Fürst.)



Plot of the coverage of the orbit of the X-ray binary, Cyg X-1, with *Chandra-HETG* observations. (Image by Manfred Hanke.)



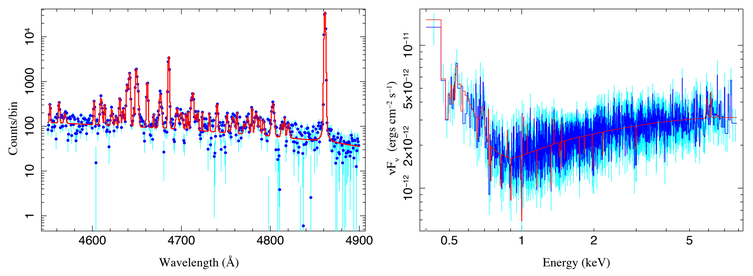
3-D plot of a maser disk, projected onto various 2-D planes. (Plot by Eugenia Litzenger.)

Simple Line Fitting

- *S-lang* employs the concept of "overloading", wherein any scripted function definition can be overwritten at any time with a new definition.
- *ISIS* employs .the concept of *Isis_Active_Dataset*, a global variable set to the numerical index of whatever dataset is being evaluated at the moment. This is a powerful tool that allows sophisticated customization of model behavior. From almost the beginning, the baseline behavior of *ISIS* has been similar to the *Sherpa* concept of "stacks"– data and models are naturally treated in groups – with *Isis_Active_Dataset* allowing one to break from this behavior in both straightforward ways [`fit_fun("powerlaw(Isis_Active_Dataset)");` creates a different model instance for each dataset] and in complex ways [`fit_fun("lines()");` discussed below creates a series of lines that can be different for each dataset and that *can redefined on the fly during analysis*].
- These abilities in concert allow for sophisticated, yet simple, manipulations of model behavior. We are building a module that creates line models, suitable for use, e.g., in high resolution spectra that is far less complex than a plasma code, but much more convenient than adding `gaussian(1)+gaussian(2)+...`
- User-defined databases can be pre-loaded, with custom names for commonly used lines (e.g., "FeKa" for a line with a baseline 6.4 keV energy), or defined on the fly. Line profiles can be any that are defined by either *XSPEC* or *ISIS* (Gaussian, Lorentzian, Voigt, redshifted or unredshifted, etc.), and the line parameters can be Angstrom or keV. Lines will be added to the model in either wavelength or energy order, depending upon the user's choice. Lines can be added either in emission or absorption, or for purposes of error bar searching, both so as to avoid hard parameter limits. Lines can be deleted from the model at any time.
- Below we give an example of the modules use. (Syntax subject to change as we develop; suggestions for functionality appreciated!)

```
isis> () = evalfile("isis_line_models");                % Load the module
isis> init_line(;line_id="FeKa",center="6.4");          % Create an Fe Kalpha line at 6.4 keV
isis> init_line(;line_id="SiKa",center=7.126,unit="a");  % Create an Si Kalpha line at 7.126 Angstrom
isis> add_line("FeKa","lines","gauss");  % Add the FeKa line to the model called "lines", using a Gaussian profile
isis> add_line("SiKa","lines","voigt");  % Add the SiKa line to the model called "lines", using a Voigt profile
isis> fit_fun("powerlaw+lines");         % Define a fit function with these lines
isis> list_free;
powerlaw+lines
 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  v_SiKa(1).norm         0       0          1e-05            0          0.1  photons/s/cm^2
   4  v_SiKa(1).energy       0       0       1.739885     1.715807     1.764648  keV
   5  v_SiKa(1).fwhm         0       0     0.03612537            0    0.3612537  keV
   6  v_SiKa(1).vtherm       0       0             50          0.1          500  km/s
   7  xg_FeKa(1).norm        0       0          1e-05            0          0.1  photons/s/cm^2
   8  xg_FeKa(1).LineE       0       0            6.4          6.3          6.5  keV
   9  xg_FeKa(1).Sigma       0       0           0.01            0          0.1  keV
isis> delete_line("SiKa","lines");
isis> () = eval_counts; list_free;
powerlaw+lines
 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  xg_FeKa(1).norm        0       0          1e-05            0          0.1  photons/s/cm^2
   4  xg_FeKa(1).LineE       0       0            6.4          6.3          6.5  keV
   5  xg_FeKa(1).Sigma       0       0           0.01            0          0.1  keV
```

- We will add methods for creating "primary" and "secondary" ties among chosen parameters. For example, one might want to create a primary tie among the wavelengths/energies for a set of lines that the user believes to come from the same physical/velocity location (e.g., ionization stages that peak near the same temperature). Secondary ties could be for line normalizations and/or energies within a given species (i.e., the α, β, γ, … lines of a given ion), which would be applied *in addition to* any primary tie. The goal is to not hard code these relations via specific models (that's what plasma codes are for…), but to leave the definitions flexible and mutable, so users can explore more phenomenological fits.
- In that spirit, we have applied the code in "blind search strategies", where lines were added serially to the modeling process. This included participating in a code comparison for optical line searches (Wesson 2016). Lines were simply named after their central wavelengths, and *ISIS* identified 45 lines (5 estimated to be spurious, at which point the line search was halted) — the next closest program found only 30 — which included line blends missed by all other codes. Currently we are using the code to perform blind searches on *Chandra-HETG* data of PG1211+143, naming the lines (possibly shifted by up to 0.2c; Pounds & Page 2006) by the statistical order in which they have been found. We will likely include several simple blind line searches in the final module.



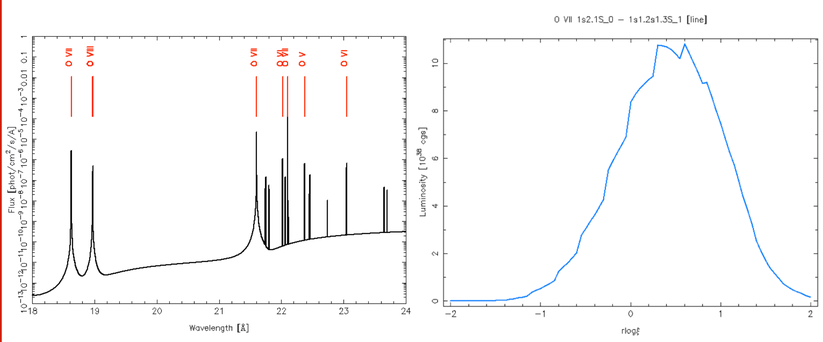
Left: The line module code applied to an optical line search (see Wesson 2016), with 45 Voigt profiles included in the spectrum. In this case, we named the lines by their central wavelength, and ordered the profiles by wavelength in the parameter file. (The next best code search found only 30 lines, and did not find all the line blends that *ISIS* did.) Right: The line module applied to *Chandra-HETG* spectra of PG1211+143. Here the Gaussian line profiles were named by the order of their statistical significance, and they were placed in energy order in the resulting parameter file.

Complex Line Fitting

- For photoionized plasmas, the *XSTAR* code (see Kallman & Bautista 2006) can provide a description of both emission (via the `photoemis` model) and absorption (via the `warmabs` model) line features. Interpreting these results, however, can be difficult . The fitted lines are only identified in the internal *XSTAR* database; absolute values of the line strengths are not directly reflected in the fit parameters, and the line strengths as a function of the model parameters can be difficult to discern.
- An *ISIS* interface to *XSTAR* has been developed by Lia Corrales and David Huenemoerder, in order to provide access to its database, including the specific line parameter values for a given fit. The *XSTARDB* module can:
  - Search the *XSTAR* database by wavelength, element, ion, transition levels, and line strength.
  - Display and save search results as an ASCII table, and mark line transitions on a plot of the model spectrum.
  - Set up and run a grid of *XSTAR* models, with model input varied over a parameter of interest.
  - Retrieve line properties (e.g., luminosities, equivalent widths, and line ratios) as a function of *XSTAR* model parameters (`rlogxi.column`)
- Module download, instructions, and examples can be found on the module home page: http://space.mit.edu/cxc/analysis/xstardb/

```
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);
isis> (x1,x2)=linear_grid(1.0,40.0,10000);
isis> y=eval_fun(x1,x2);
isis> db=rd_xstar_output("photemis_1.fits");
isis> strongest=xstar_strong(8,db;wmin=18.0,wmax=24.0);
isis> xstar_page_group(db,strongest;sort="luminosity");
isis> lstyle=line_label_default_style();
isis> lstyle.top_frac=0.85;
isis> lstyle.bottom_frac=0.7;
isis> xstar_plot_group(db,strongest,2,lstyle);
```

```
isis> (x1,x2)=linear_grid(1.0,40.0,10000);
isis> model_info=@_default_model_info;
isis> model_binning=struct{bin_lo=x1,bin_hi=x2};
isis>set_struct_fields(model_info,"photemis","rlogxi",-2.0,2.0,0.05,
    model_binning  );
isis> xstar_run_model_grid(model_info,"/my/path/";nstar=10);
isis> fgrid=glob("/my/path/photemis_*.fits");
isis> fgrid=fgrid[array_sort(fgrid)];
isis> pe_grid=xstar_load_tables(fgrid);
isis> o_vii=where(xstar_el_ion(pe_grid.mdb,O,7));
isis> xstar_page_grid(pe_grid,o_vii);
isis> o_vii_F=where(xstar_trans(pe_grid.mdb,O,7,1,2));
isis> o_vii_F_lum=xstar_line_prop(pe_grid,o_vii_F,"luminosity");
isis> rlogxi=xstar_get_grid_par(pe_grid,"rlogxi")
```



Left: After the *XSTAR* model `photemis` has been fit to a spectrum, the *XSTARDB* module is used to examine the results. The photo-emission model is plotted, and then the strongest emission lines in a user specified region are identified and labeled. The module provides similar facilities for identifying and labeling lines even when multiple components, at different redshifts, have been fit. Right: The *XSTARDB* module also provides facilities for running grids of models, and then examining the results as a function of grid parameter values. Shown here is the strength of the OVII forbidden line as a function of ionization parameter.

ISIS Resources

ISIS: http://space.mit.edu/asc/isis/manual.html
S-lang: http://www.jedsoft.org/slang/doc/html/slang.html
ISIS vs. XSPEC: http://space.mit.edu/home/mnowak/isis_vs_xspec
Remeis Scripts: http://www.sternwarte.uni-erlangen.de/isis
Remeis ISIS Wiki: http://www.sternwarte.uni-erlangen.de/wiki/doku.php?id=isis:start
SLXfig Examples: http://www.jedsoft.org/fun/slxfig/examples.html
SLXfig Examples II: http://www.sternwarte.uni-erlangen.de/wiki/doku.php?id=isis:slxfig:fancyremeisplots
XSTARDB: http://space.mit.edu/cxc/analysis/xstardb/
ISIS Workshop 2015: http://space.mit.edu/ASC/isis2015/index.html

References

Foreman-Mackey, D. et al. 2013, PASP, 125, 306
Goodman, J. & Weare, J. 2010, CAMCS, 6, 65
Houck, J. & DeNicola, L. 2000, ASP Conf. Ser. 216, 9, 591
Kalman, T. & Bautista, M. 2006, ApJ, 133, 221
Kühnel, M. et al. 2015, Acta Polytechnica, 55, 123
Kühnel, M. et al. 2016, Acta Polytechnica, 56, 41
Pounds, K.A. & Page, K.L. 2006, MNRAS, 372, 1275
Wesson, R. 2016, MNRAS, 456, 3774