X-ray Timing Analysis

*Michael Nowak - Chandra X-ray Science Center/MIT Kavli Institute

Some Questions That We'd Like to Answer:

- Does My Source Vary?
- On What Time Scales Does it Vary?
- Are the Variations Periodic or Aperiodic?
- How Do Different Energy Bands Relate to One Another?

*(With Some Judicious Stealing of Slides from Z. Arzoumanian's 2003 X-ray Astronomy School Talk)

Characteristic Time Scales:

$$\tau \geq R/V, \quad V \leq c, \quad R \geq 2\,GM/c^2$$

$\tau \geq 1000$ sec $\qquad 10^8\ M_\odot$ (AGN)

$\tau \geq 100\ \mu$sec $\qquad 10\ M_\odot$ (BHC)

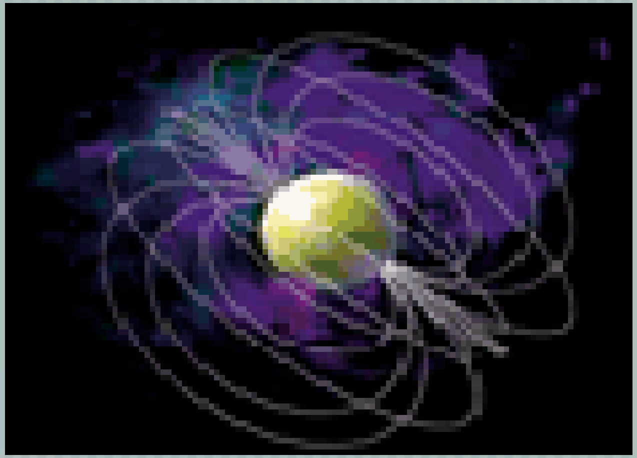
$\tau \geq 15\ \mu$sec $\qquad 1.4\ M_\odot$ (NS)

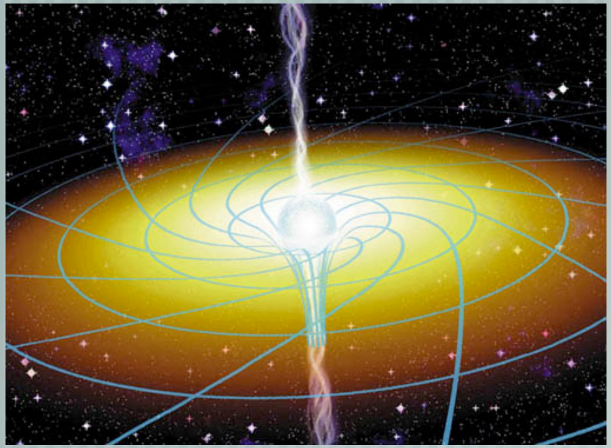
These are the Fastest Achievable Time Scales. In Reality, There Can be Variability on a Range of Time Scales.

Rotational Periods:

msec - sec for NS/WD

hr - days for Stars





Accretion Time Scales:

Dynamical, Thermal, Viscous
Time Scales

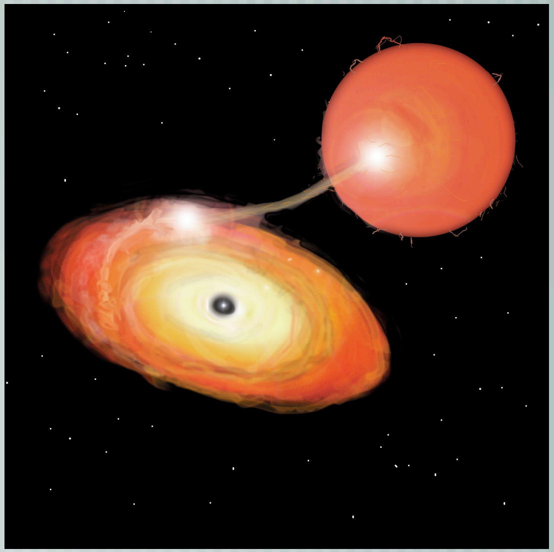
msec - days for NS/BHC

minutes - years for AGN

Orbital Time Scales:

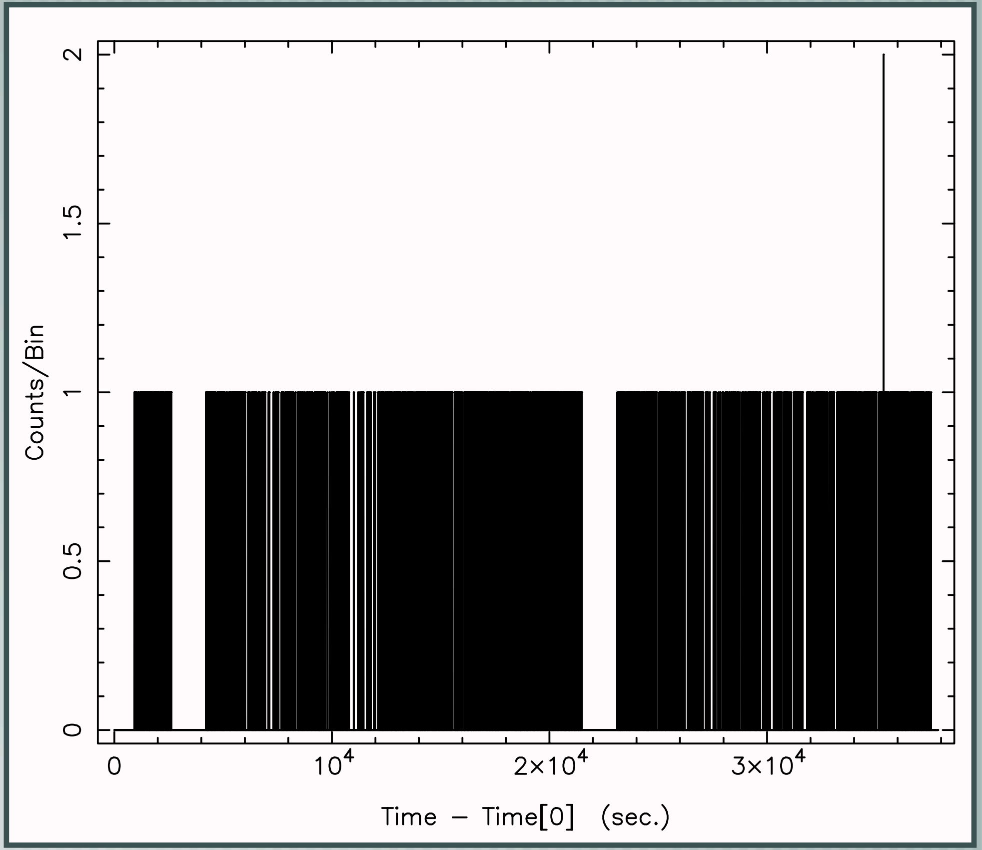
minutes to days for NS/BHC

Suber-orbital periods:
weeks to months



Timing Starts with a Lightcurve

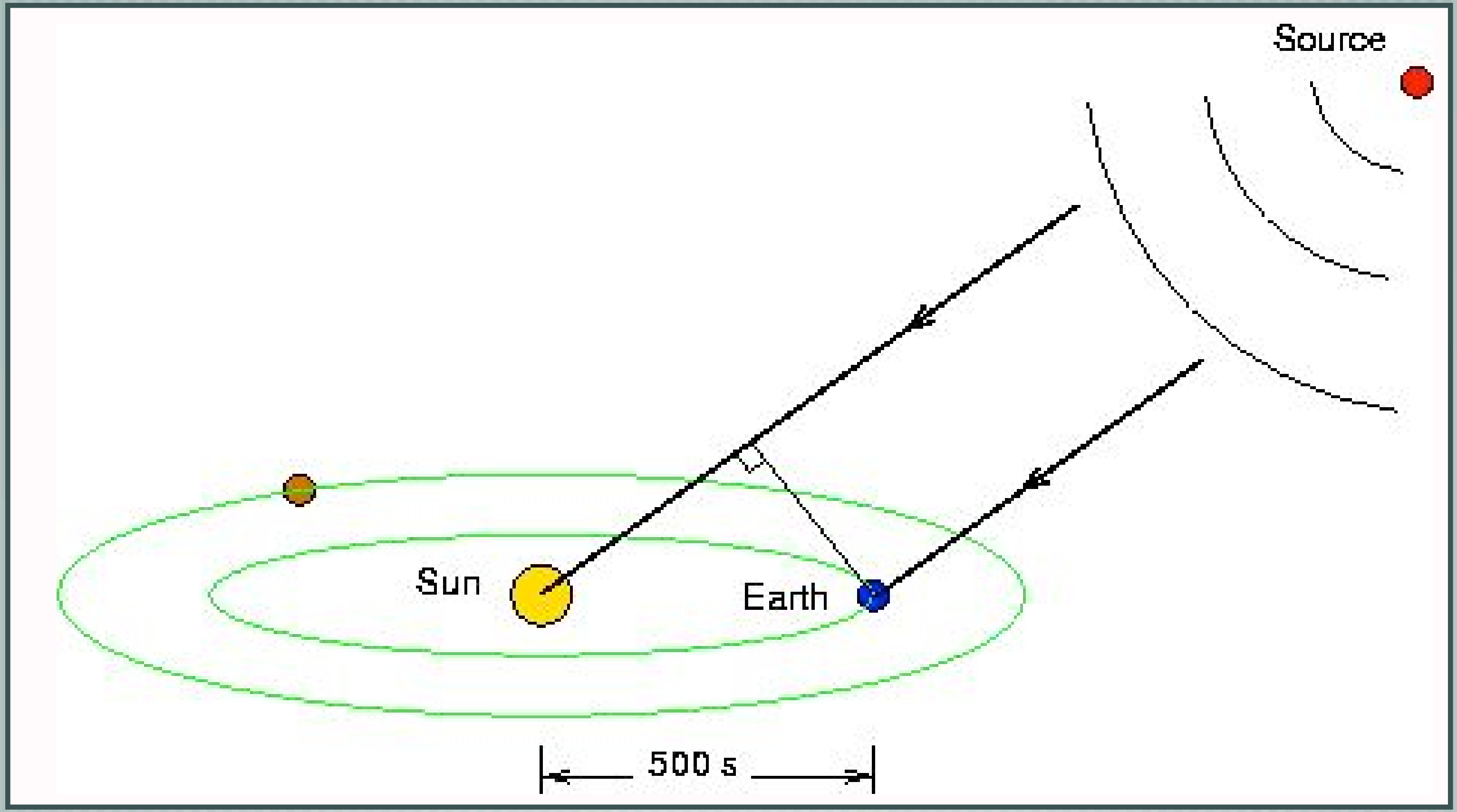
- Different Spacecraft can have different tools for creating Lightcurves
- DMTOOLS, FTOOLS, Xselect, S-lang or Python scripts using cfitsio library.
- Always choose integer multiple of “natural” time unit for binning
- Don’t bin any more than you have to - save it for subsequent analysis



Precision Absolute & Relative Timing

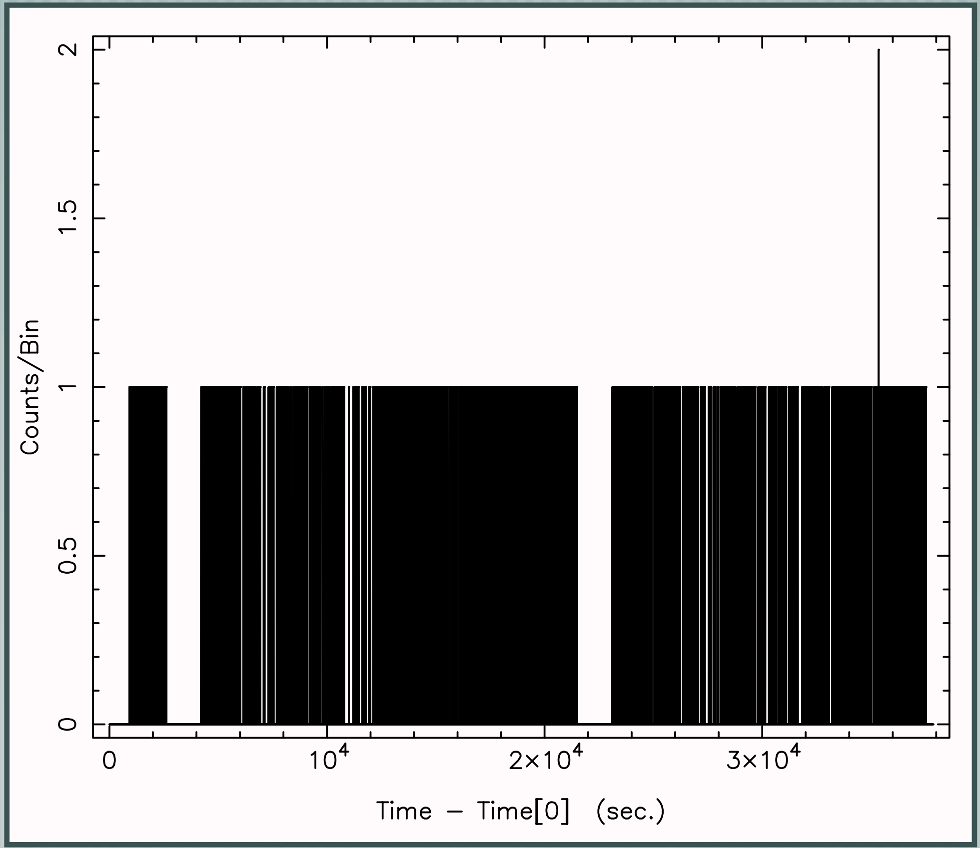
- Barycenter the data with axbary (CIAO) or fxbary (FTOOLS)!

```
axbary  4u2129_chandra.fits  orbit_file.fits  4u2129_barycenter.fits
```



CIAO: DMEXTRACT

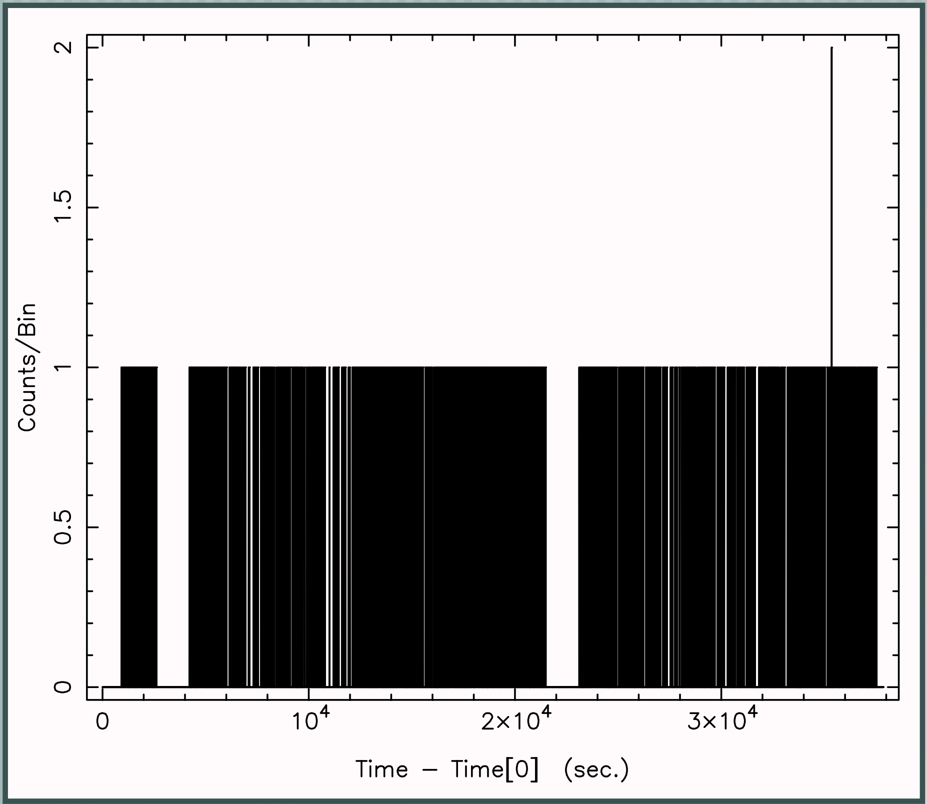
dmextract infile="4u2129_chandra.fits [EVENTS] [sky=region(source.reg)][bin time=::1.14104]" outfile=4u2129_ps.fits opt=ltc1



Could have used FTOOLS or Xselect (wrapper around FTOOLS)

CIAO: Or Scripts (ISIS, Sherpa)

Perhaps first do a dmcopy to isolate the source(s) of interest.

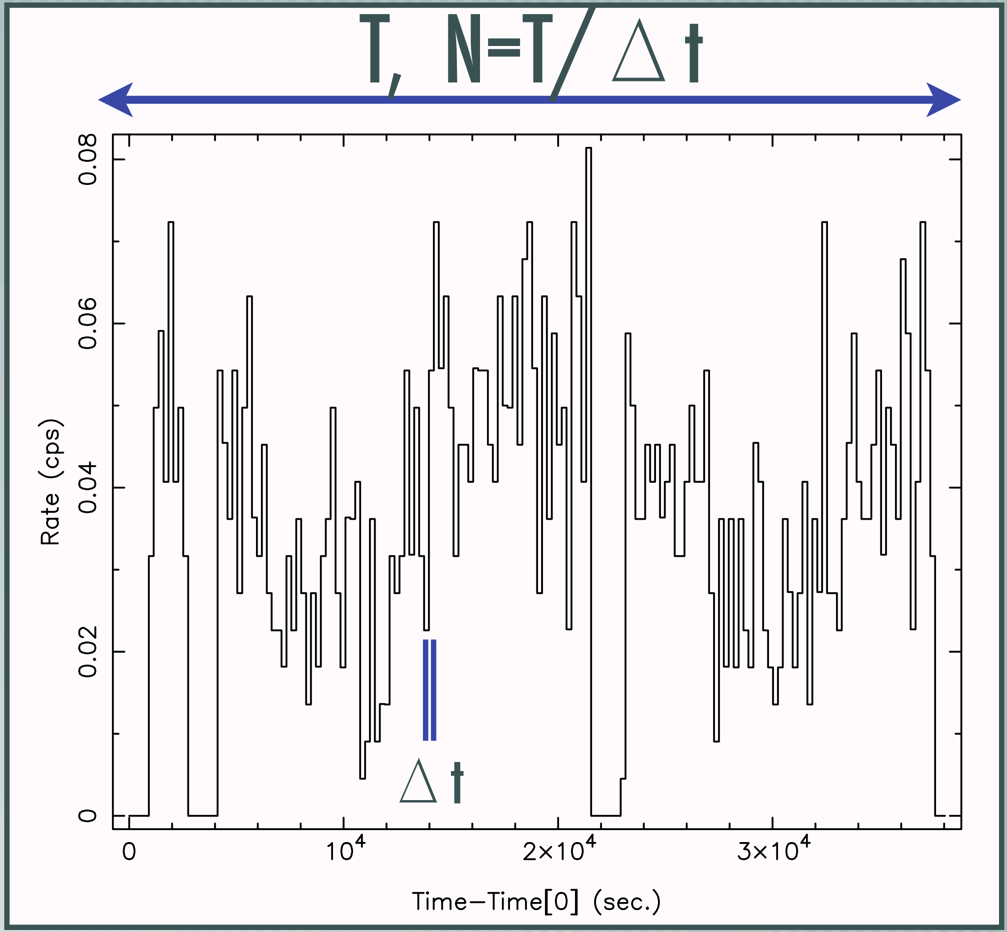



```
isis> (tmin,tmax) = fits_read_header("4u2129_chandra.fits",
"tstart","tstop");
isis> tlo=[tmin:tmax:1.14104];
isis> thi=make_hi_grid(tlo);
isis> t=fits_read_col("4u2129_chandra.fits","time");
isis> counts=histogram(t,tlo,thi);
isis> hplot(tlo,thi,counts);
```

(Some more work if one wanted to intersect Good Time Intervals=GTI)

Length & Binning Determine Limits

- Lowest Frequency: $f_{long} = 1/T$
- Highest Frequency: Nyquist Frequency, $f_{Nyq} = 1/(2\,\triangle t)$
- Basic Question, is the Variance: $\sigma^2 = \langle x^2 \rangle - \langle x \rangle^2$ Greater than Expected from Poisson Noise?
- σ = Root Mean Square Variability



What are the Tools of the Trade?

- Timing: Xronos - Some use, but less "universal" than spectra
- More CIAO Tools (e.g., from Chandra Catalog) will be coming
  - gl_vary (Bayesian lightcurve), dither_region (area vs. time)
- Most People "Roll Their Own"
  - Custom Fortran/C/C++ Code
  - IDL or MATLAB or Python or Ruby or ...
  - Me: S-lang (http://space.mit.edu/CXC/analysis/SITAR)

Variability Test I: Excess Variance

Binned Lightcurve with Values: $X_i \pm \sigma_i$ and mean: μ

$$\sigma^2_{\rm rms} = \frac{1}{N\mu^2} \sum_{i=1}^{N} \left[(X_i - \mu)^2 - \sigma_i^2 \right]$$

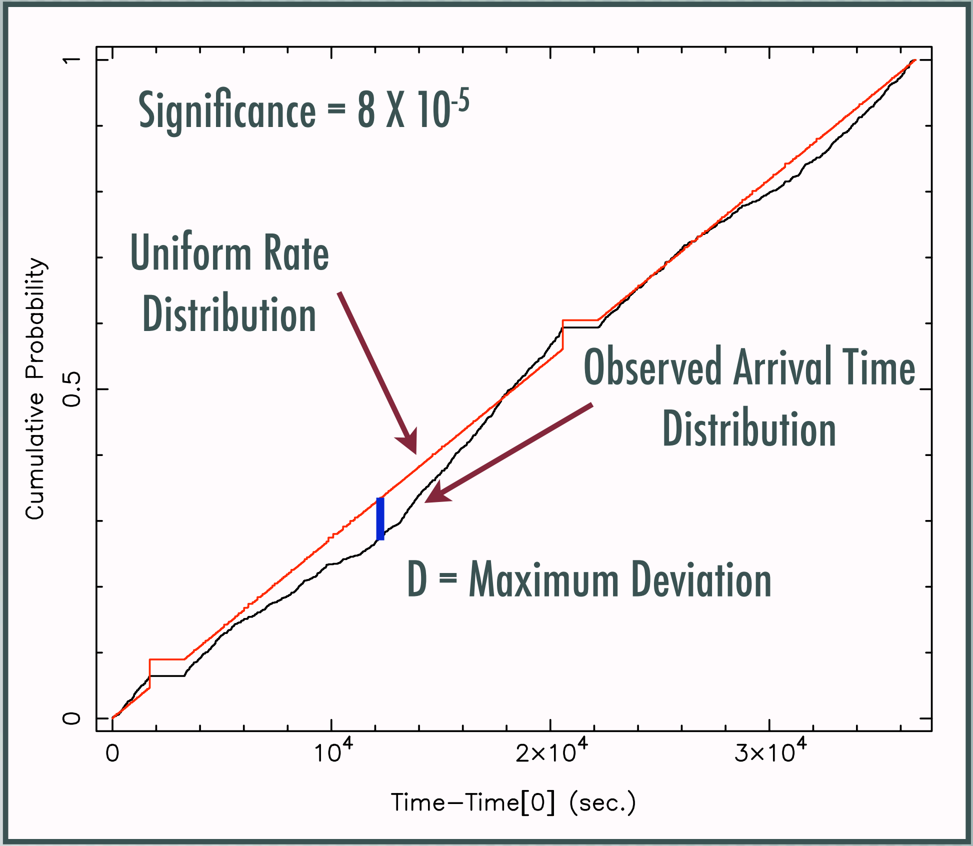
$$\Delta\sigma^2_{\rm rms} = s_D / (\mu^2 \sqrt{N})$$

$$s_D^2 = \frac{1}{N-1} \sum_{i}^{N} \left(\left[(X_i - \mu)^2 - \sigma_i^2 \right] - \sigma^2_{\rm rms}\mu^2 \right)^2$$

See Turner et al. 1999, ApJ, 524, p. 667 ; Nandra et al. 1997, ApJ, 476, p. 70

Test II: Kolmogorov-Smirnov

- Is cumulative arrival time consistent with constant rate? -or- Is distribution of times inbetween events consistent?
- See Press et al., "Numerical Recipes", plus lots of other better statistics sources.
- CIAO Tools for K-S and Kuiper variant, eventually from Chandra Source Catalog.
- Also available in many script forms (S-lang script available upon request).
- K-S/Kuiper test variability, but don't characterize it.



Test II: Kolmogorov-Smirnov

```
isis> require("stats");                         % S-lang statistics model with KS/Kuiper tests
isis> modl = (t-tstart)/(tstop-tstart);   % Fraction of observing time vs. event
isis> print(ks_test(modl));                     % Kolmogorov- Smirnov probability
isis> print(kuiper_test(modl));               % Kuiper test probability
```

The above presumes uniform effective area vs. time, no dead time intervals, etc.

Straightforward to modify to include such effects. The model must become, e.g.:

$$\frac{\int_{t_{start}}^{t[i]} Area(t)\ \theta(t_{GTI})\ dt}{\int_{t_{start}}^{t_{stop}} Area(t)\ \theta(t_{GTI})\ dt}$$

Test III: Direct Fitting, ISIS

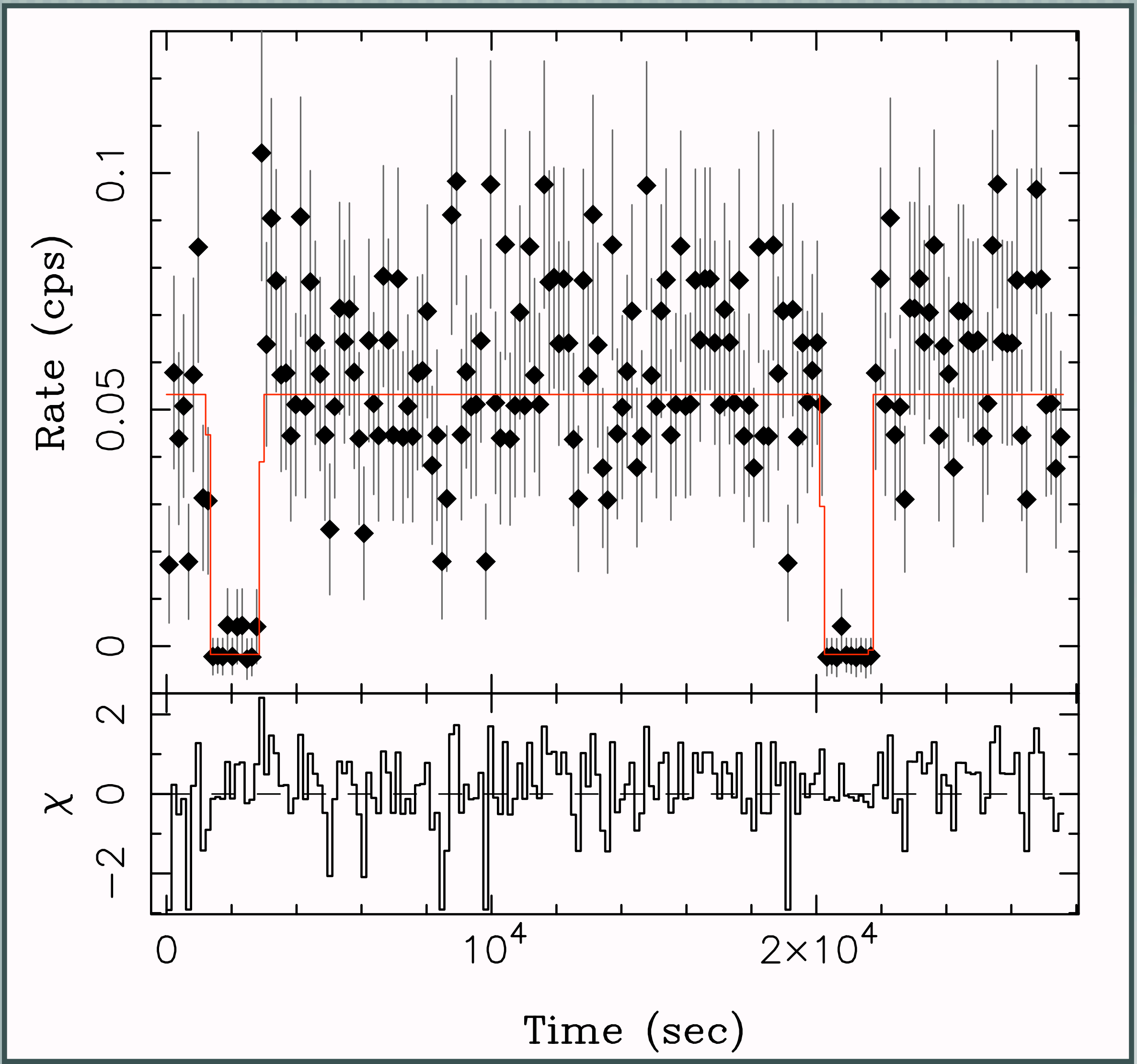
- Can be done in ISIS, Sherpa, or XSPEC (most difficult in XSPEC, since one has to create fake response matrices first - not true in ISIS or Sherpa)
- You can write your own fit functions - remember to account for finite bin widths!

```
isis> (tlo,r,e) = fits_read_col("lc.fits","time","rate","error");
isis> thi = make_hi_grid(tlo);
isis> define_counts(tlo,thi,r,e);
isis> fit_fun("orbit(1)");        % Orbit is a custom written S-lang function
isis> () = fit_counts;
```

Test III: Direct Fitting, ISIS

```
define orbit_fit(lo,hi,par)
{ % par[0] = Time of phase 0 (midpoint of eclipse)
  % par[1] = Period
  % par[2] = Width of Ingress (seconds)
  % par[3] = Width of Eclipse (seconds)
  % par[4] = Width of Egress  (seconds)
  variable plo, phi, ipneg, pa, pb, pw, pmid, pumid, orb = @lo;
  % Times converted to phases
  plo = ((lo - par[0]) mod par[1])/par[1];
  phi = ((hi - par[0]) mod par[1])/par[1];
  ipneg = where(plo < 0);
  plo[ipneg]=plo[ipneg]+1;
  ipneg = where(phi < 0);
  phi[ipneg]=phi[ipneg]+1;
  % Time widths converted to phase widths
  pa = par[2]/par[1];
  pw = par[3]/par[1];
  pb = par[4]/par[1];
  variable i=0;
  loop(length(lo))
  {  % Lo bin is in eclipse, but hi bin is ...
    if( plo[i]>=0 and plo[i]<pw/2 )
    {  % in eclipse ...
      if(phi[i] <= pw/2){ orb[i]=0.; }
     % or in egress ...
      else if(phi[i]<=pw/2+pb)
       {pmid=pw/2;
        orb[i]=((phi[i]*(phi[i]-pw)-pmid*(pmid-pw))/2/pb)
               /(phi[i]-plo[i]);}
      % or is eclipsed
      else
      {  pumid=1-pw/2;
         pmid=1-pw/2-pa;
         orb[i]=((pmid-plo[i])+(pumid*(2-pw-pumid)-
              pmid*(2-pw-pmid))/2/pa)/(phi[i]-plo[i]);  }
    }
    % Lo bin is in egress, but hi bin is ...
    else if( plo[i]>=1-pw/2-pa and plo[i]<1-pw/2 )
    {  % is in egress ...
      if(phi[i]<=1-pw/2)
      { orb[i]=(phi[i]*(2-pw-phi[i])- plo[i]*(2-pw-plo[i]))
               /2/pa/(phi[i]-plo[i]); }
      % or is eclipsed
      else
      { pmid=1-pw/2;
        orb[i]=(pmid*(2-pw-pmid)-plo[i]*(2-pw-plo[i]))
               /2/pa/(phi[i]-plo[i]); }
    }
    % Lo bin is eclipsed (and hi bin by assumption)
    else
    { orb[i]=0.; }
    i++; }   return orb; }
```

Test III: Direct Fitting, ISIS



XMM-Newton data of previous Chandra observed source.

Model is constant, with finite ingress & egress, plus eclipse.

Can add other components (e.g., sinusoid), and do confidence tests as for spectra.

Test III: Direct Fitting, Sherpa

- Example of fitting data from an ASCII file:

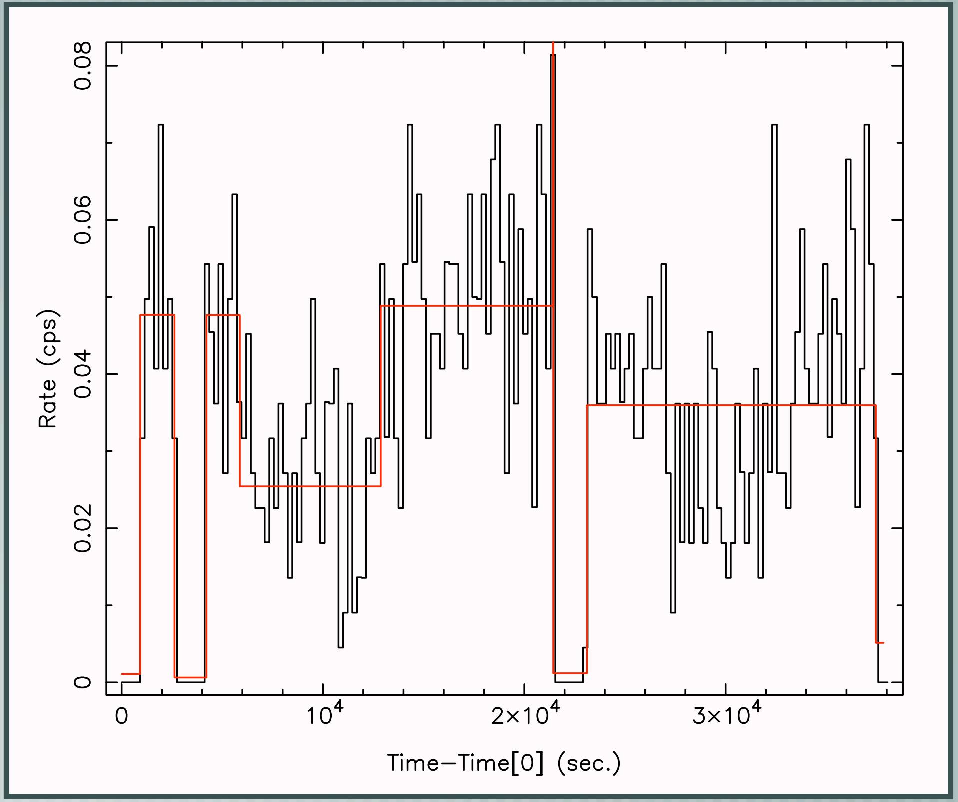
```
sherpa> load_data(1,"lightcurve.dat",4,["XLO","XHI","Y","STATERROR"]);
sherpa> set_model("custom_model");
sherpa> fit(1);
```

- Could also have read data with a single X-column.
  - Histogram data is a little bit easier to deal with, since defining the average over a time bin is less ambiguous.

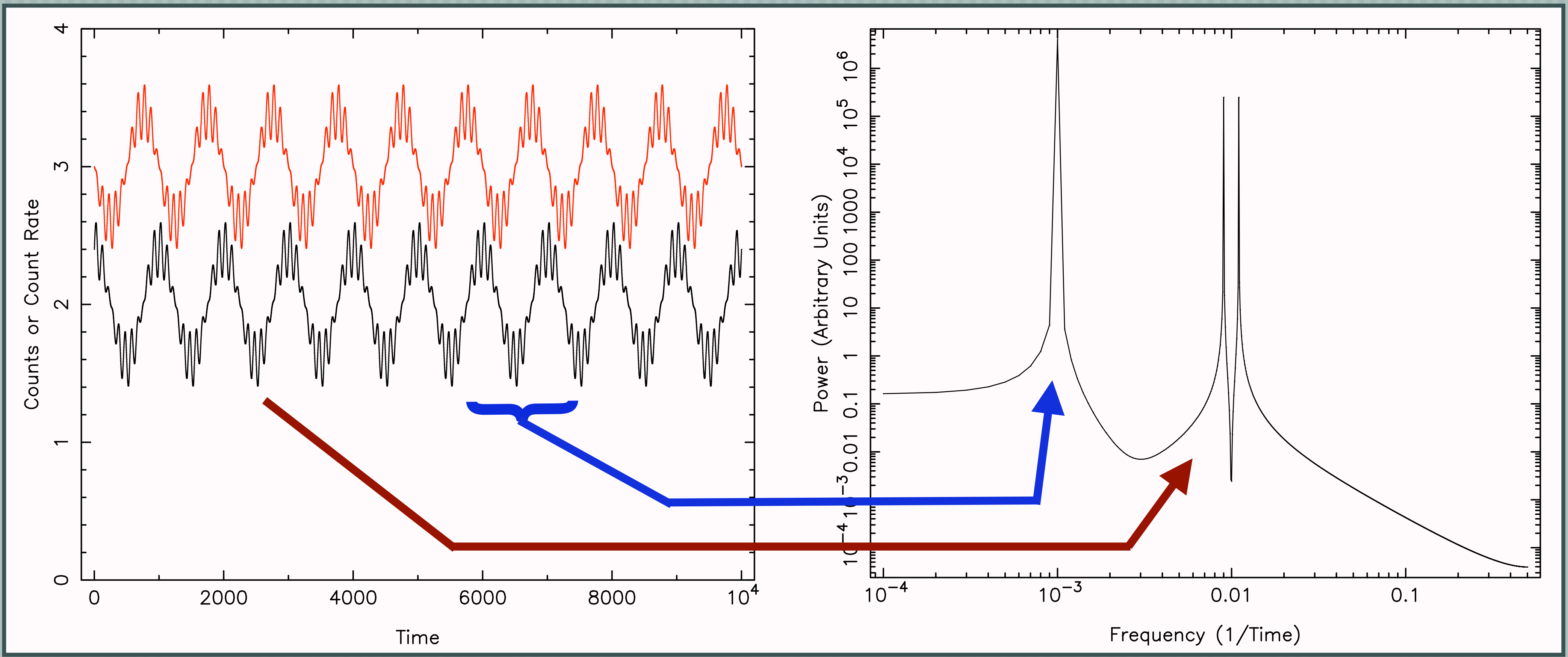
Ninja Topic: Bayes Stats

- Bayesian Methods Don't Require Binning (Case below: event times only!)
- Gregory & Loredo (1992, ApJ, 398, p. 146) - Determines Optimal Uniform Binning. (CIAO version, gl_vary, soon; S-lang script available upon request)
- Bayesian Blocks (J. Scargle, 20XX) - Determines Optimal Non-uniform Binning. (S-lang Version on SITAR page)

```
 t = fits_read_col("4u2129_chandra.fits","time");
cell = sitar_make_data_cells(t,2,0.7,1.14104,min(t),max(t));
ans = sitar_global_optimum(cell,3.5,2);
```



Fourier Transform Methods



- A Workhorse of the Timing World
- How is Variability Power Distributed as a Function of Frequency?

Fast Fourier Transform (FFT)

$$X_j \equiv \sum_{k=0}^{N-1} x_k \exp(2\pi i j k/N) \quad , \quad j = [-N/2, \ldots, 0, \ldots, N/2]$$

$$P_j = 2|X_j|^2/(\mathrm{Rate}^2 \times \mathrm{T}_{\mathrm{total}})$$ ("One Sided" RMS Normalization)

$$P_j = 2|X_j|^2/(\mathrm{Rate} \times \mathrm{T}_{\mathrm{total}})$$ ("One Sided" Leahy Normalization)

- Lightcurve with: N bins, comprised of counts, x_i, becomes power spectrum, with N/2+1 independent amplitudes, and N/2-1 independent complex phases (real data)
- Good FFTs usually optimized for N = power of 2
- Power Spectrum is the squared Fourier amplitude, properly normalized
- Power Spectrum is throwing out information! Not unique!

FFT Normalizations

- Leahy: Poisson Noise level = 2, intrinsic Power scales as rate
- RMS: intrinsic Power independent of rate, Noise level = 2/rate
- Integral of PSD is measure of Root Mean Square variability

$$A = \int P_{\rm rms} df = \sum_j P^j_{\rm rms} \Delta f \quad , \quad \Delta f = 1/T$$

$$\sqrt{A} = {\rm rms/mean} = \left(\frac{\langle x^2 \rangle - \langle x \rangle^2}{\langle x \rangle^2} \right)^{1/2}$$

Pulsed Fraction (coherent oscillation): $f_p = \sqrt{\frac{2(P_{\rm Leahy} - 2)}{\rm Rate}}$

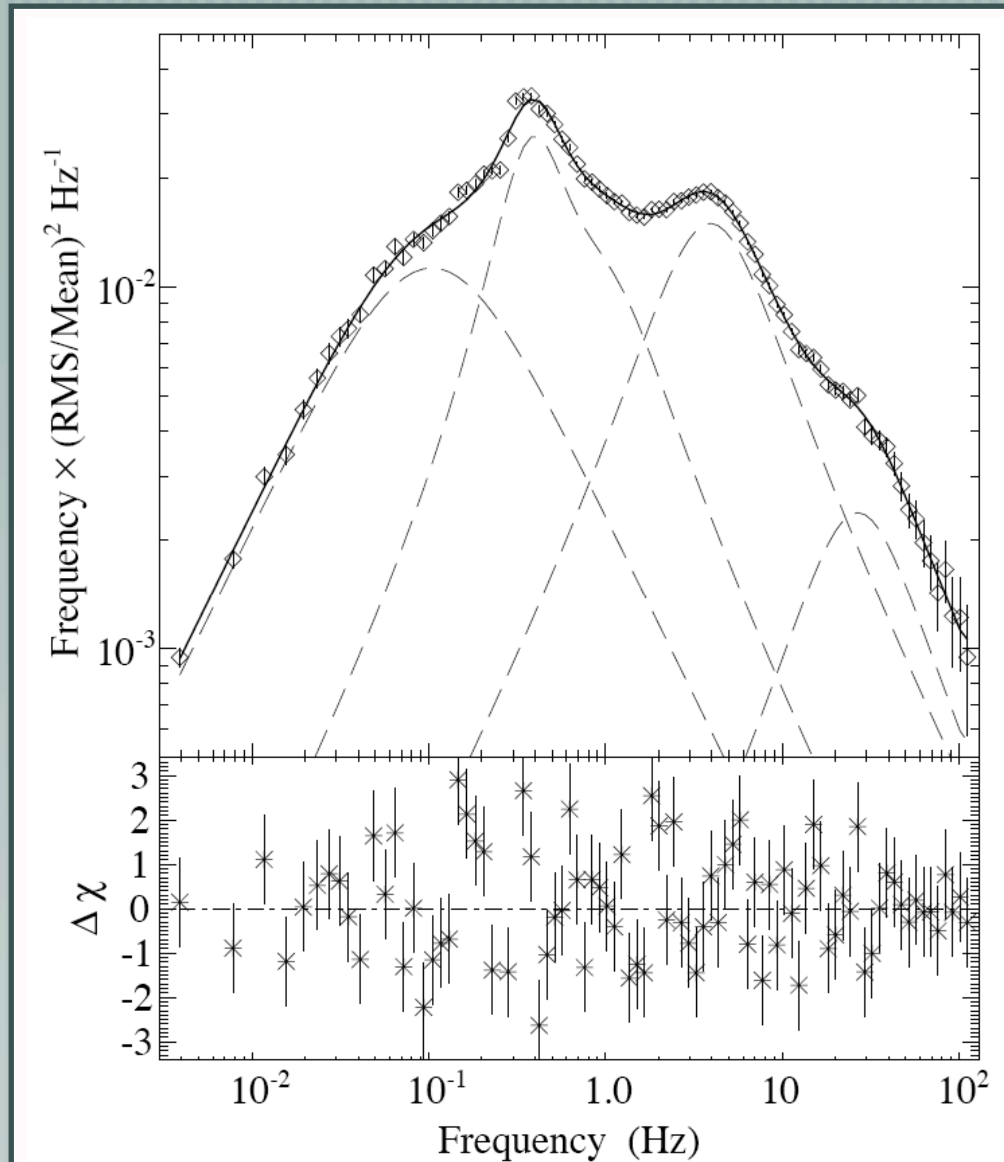
PSD Normalizations are often plotted as $(\mathrm{RMS})^2/\mathrm{Hz}$

PSD Statistics

- Leahy noise level is 2 +/- 2 (distributed as χ^2 with 2 DoF)
- Increasing lightcurve length doesn't help - distributes noise among more frequency bins!
- "Statistically Stationary Processes" have Power = P_i +/- P_i
- Reduce noise by averaging PSD from individual lightcurve segments, as well as over (usually logarithmically spaced) adjacent frequency bins
- Errors reduced by factor of: $\sqrt{N_{\mathrm{avg}}}$

With: $P'_j = (P_j - P_{\rm noise}) \pm P_j/\sqrt{N_{\rm avg}}$ You can fit models

Note: Total RMS = Incoherent sum of components, i.e., $\left(\sum_i {\rm RMS}_i^2\right)^{1/2}$



Advice: fit models that average over frequency bin widths

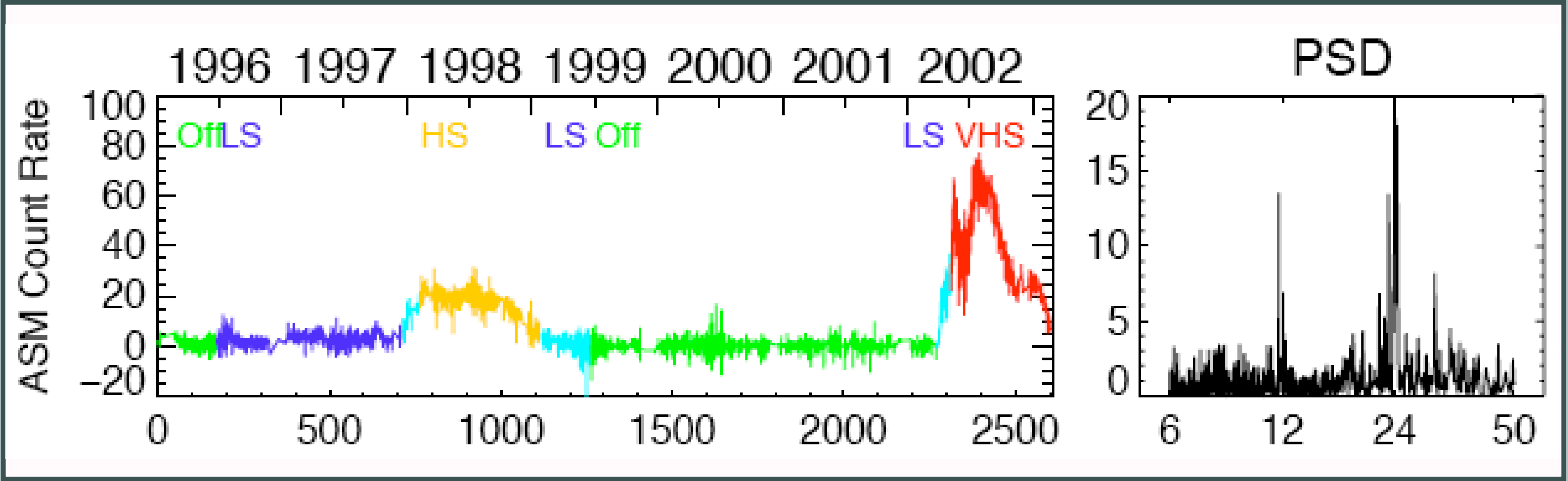
Fit PSDs Just Like Lightcurves

- ISIS example, with helper functions from SITAR

```
isis> (t,cts) = fits_read_col("events_18_39_a.lc","time","counts");
isis> (f,psd,n,cts) = sitar_avg_psd(cts,65536,1./2^12,t);
isis> (aflo,afhi,apsd,nf) = sitar_lbin_psd(f,psd,0.01);
isis> id = sitar_define_psd(aflo,afhi,apsd,apsd/sqrt(na*nf));
isis> fit_fun("constant(1)+qpo(1)");                    % QPO is a custom S-lang function
isis> () = fit_counts;
```

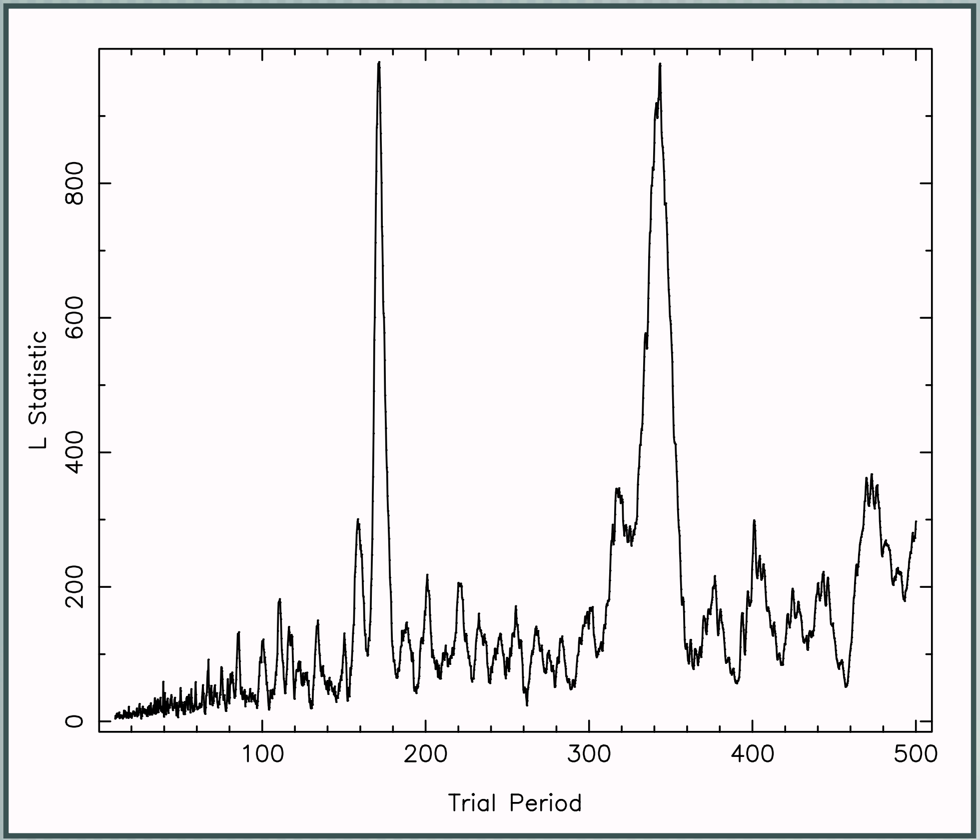
Ninja Topic: Aliasing!

- Signals appear at sum/difference frequencies of primary signals, whether signals are “real” or “fake” (e.g., sampling periods)
- Beware characteristic times! Spacecraft orbits, dither time scale, 1 year, ...
- Example: RXTE-All Sky Monitor - Many sources show periods at 24 hours +/- a small bit = 1/Years Secular Change with a 24 hour sample period (e.g., from AGN monitoring).



Epoch Folding & Period Searches

- Good for non-sinusoidal variations
- Good for when there are data gaps or complicated window functions
- Not good for aperiodic variability




```
isis> event = sitar_readasm("xa_x1820-303_d1",,,1.2);
isis> fold = sitar_epfold_rate(event.time,event.rate,
10,500,20,2000);
isis> xlabel("Trial Period"); ylabel("L Statistic");
isis> plot(fold.prd,fold.lstat);
```

Xronos has epoch folding, various IDL routines can be found on the web.

Read the literature on significance levels!

Reiterating Words of Advice:

- CIAO tools, dmextract & axbary, can be used to create lightcurves, or ...
- ... create directly via scripts. (Tools require less customization for, e.g., GTI.)
- Bin lightcurves on integer multiples of “natural” time scales
- Lightcurves can be directly fit in ISIS, Sherpa, or XSPEC (latter most difficult).
- Do FFTs with evenly spaced bins (Lomb-Scargle for unevenly spaced bins), and avoid data gaps (see literature if dealing with gaps). PSD can be fit in ISIS, Sherpa, ...
- Beware of signals on “characteristic time scales” (spacecraft, Earth, ...)
- Large literature with many techniques & statistics.

References for Further Reading

- van der Klis, M. 1989, “Fourier Techinques in X-ray Timing”, in Timing Neutron Stars, NATO ASI 282, Ögelman & van den Heuvel eds., Kluwer
- Press et al., “Numerical Recipes” (Discussions only! Better code exists on the web!)
- Leahy et al. 1983, ApJ, 266, p. 160 (FFT & PSD Statistics)
- Leahy et al. 1983, ApJ, 272, p. 256 (Epoch Folding)
- Davies 1990, MNRAS, 244, p. 93 (Epoch Folding Statistics)
- Vaughan et al. 1994, ApJ, 435, p. 362 (Noise Statistics)