A Typical Sherpa Session
(The boiled-down version.)

The user:

- reads in data (and sets filters, etc.);
- builds model expressions;
- chooses a statistic;
- fits the model expressions to the data, one at a time;
- compares the results of the fits in order to select a best-fit model; and
- estimates the errors for the best-fit model parameters.
Choosing a Statistic

(So many choices, so little guidance.)

A key feature of Sherpa is its large array of statistics appropriate for analyzing Poisson-distributed (i.e. counts) data.

- Statistics based on $\chi^2$:
  - CHI GEHRELS
  - CHI DVAR
  - CHI MVAR
  - CHI PARENT
  - CHI PRIMINI

- Statistics based on the Poisson likelihood $\mathcal{L}$:
  - CASH
  - BAYES

If the data are not Poisson-distributed (e.g. fluxes), then alternatives include:

- least-squares fitting: setting all variances to one; or
- providing errors in an input file.
**$\chi^2$-Based Statistics**

The $\chi^2$ statistic is

$$\chi^2 \equiv \sum_i \frac{(D_i - M_i)^2}{\sigma_i^2},$$

where

- $D_i$ represents the observed datum in bin $i$;
- $M_i$ represents the predicted model counts in bin $i$; and
- $\sigma_i^2$ represents the variance of the sampling distribution for $D_i$.

<table>
<thead>
<tr>
<th>$\chi^2$ Statistic</th>
<th>$\sigma_i^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>GEHRELS</td>
<td>$[1 + \sqrt{D_i + 0.75}]^2$</td>
</tr>
<tr>
<td>DVAR</td>
<td>$D_i$</td>
</tr>
<tr>
<td>MVAR</td>
<td>$M_i$</td>
</tr>
<tr>
<td>PARENT</td>
<td>$\frac{\sum_{i=1}^{N} D_i}{N}$</td>
</tr>
<tr>
<td>PRIMINI</td>
<td>$M_i$ from previous best-fit</td>
</tr>
</tbody>
</table>
Likelihood-Based Statistics

The CASH statistic is

\[ C \equiv 2 \sum_i [M_i - D_i \log M_i] \propto -2 \log \mathcal{L}, \]

where

- \( D_i \) represents the observed datum in bin \( i \);
- \( M_i \) represents the predicted model counts in bin \( i \); and
- \( \mathcal{L} = \prod_i \frac{M_i^{P_i}}{D_i!} \exp(-M_i) \).
Statistics: Caveats
(Potholes on the road to publication.)

Things to remember when using $\chi^2$:

- $\chi^2$ is an approximation of $\log \mathcal{L}$ in the Gaussian (high-counts) limit. So...

- All estimations of variance (except GEHRELS) assume a Gaussian sampling distribution, not Poisson. Hence the number of counts in each bin should be $\geq 5$.

- CHI GEHRELS works with low-count data, but does not generally follow the $\chi^2$ distribution: best fits are often “too good.”

- And $\chi^2$ is a biased estimator.

Things to remember when using CASH or BAYES:

- In the limit of high counts, $\Delta C \sim \Delta \chi^2$.

- Likelihood estimators are unbiased. But...

- Background subtraction is not allowed.

- There is no “goodness-of-fit” measure.

- And negative model amplitudes are not allowed.
A Demonstration of Bias

- Using the Sherpa utility FAKEIT, we simulated 500 datasets from a constant model with amplitude 100 counts.

- We then fit each dataset with a constant model, recording the inferred amplitude.

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Average Amplitude</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHI GEHRELS</td>
<td>99.05</td>
</tr>
<tr>
<td>CHI DVAR</td>
<td>99.02</td>
</tr>
<tr>
<td>CHI MVAR</td>
<td>100.47</td>
</tr>
<tr>
<td>CHI PARENT</td>
<td>99.94</td>
</tr>
<tr>
<td>CHI PRIMINI</td>
<td>99.94</td>
</tr>
<tr>
<td>CASH</td>
<td>99.98</td>
</tr>
</tbody>
</table>
Optimization in Sherpa

Optimization is the action of minimizing $\chi^2$ or $-\log L$ by varying the thawed parameters of the model. The user may choose between several optimization methods in Sherpa, including ones which:

- Find the local minimum.
  - POWELL
  - SIMPLEX
  - LEVENBERG–MARQUARDT

These algorithms are not computationally expensive, but they are also not appropriate for finding the global minimum of a complex statistical surface when starting from a random point.

- Attempt to find the global minimum.
  - GRID and GRID–POWELL
  - MONTE and MONTE–POWELL
  - SIMULATED ANNEALING

These are computationally intensive algorithms which are useful for searching complex statistical surfaces, starting from a random point.
Optimization: Powell

POWELL is Sherpa’s default optimizer.

• It is a direction-set method in which initially, the chosen statistic is minimized by varying each parameter in turn while holding all other parameter values fixed.

• Advantages:
  – no gradient calculation
  – robust
    * can find local minima even on complex surfaces
    * can be used with all statistics

• Disadvantage:
  – relatively slow
Optimization: Simplex

- The vertices of a simplex are reflected and/or contracted until the local minimum is bracketed.

- Advantages:
  - no gradient calculation
  - can find local minima even on complex surfaces
  - faster than POWELL

- Disadvantage:
  - exhibits a tendency to converge before reaching minima
Optimization: Levenberg-Marquardt

• Approach the minimum taking steps of size $\delta \theta$, computed by solving the set of linear equations:

$$
\sum_{j=1}^{n} \alpha_{i,j} (1 + \lambda_{i,j}) \delta \theta_j = \beta_i,
$$

where

$$
\alpha_{i,j} = \sum_{k=1}^{n} \frac{1}{\sigma_k^2} \left[ \frac{\partial M(\hat{\theta})}{\partial \theta_i} \frac{\partial M(\hat{\theta})}{\partial \theta_j} \right],
$$

$$
\beta_i = -\frac{1}{2} \frac{\partial \chi^2}{\partial \theta_i},
$$

and $\lambda_{i,j}$ is a numerical factor, non-zero when $i = j$.

• Advantage:
  – fast

• Disadvantages:
  – requires gradient calculation
  – less robust in complex parameter spaces
  – appropriate for use with $\chi^2$ statistics only

• Enhancements made in CIAO 2.1:
  – works correctly during simultaneous fits of source and background data
  – works correctly with double-precision data
Confidence Intervals and Regions

(What are the errors on my parameters?)

- In frequentist statistics, the data are the random variables. Thus to estimate confidence intervals, new datasets need to be repeatedly simulated, either from the best-fit model or from the data themselves.

- A distribution of parameter values is generated by fitting the model to each simulated dataset.

- The central 68% of the parameter values can be called the 1σ confidence interval.

- Simulations are computationally expensive. If:
  - the $\chi^2$ or $\log \mathcal{L}$ surface in parameter space is approximately shaped like a multi-dimensional paraboloid, and
  - the best-fit point is sufficiently far from parameter space boundaries,

then we may achieve good estimates of confidence intervals by examining the $\chi^2$ or $\log \mathcal{L}$ surface itself.
Confidence Intervals and Regions: Uncertainty

• Vary a parameter’s value, while holding the values of all other parameters to their best-fit values, until the fit statistic increases by some preset amount from its minimum value (e.g. $\Delta \chi^2 = 1$ for $1\sigma$).

• Gives correct results if and only if:
  – the statistic surface is “well-behaved”
  – there are no correlations between parameters

• Advantage:
  – fast

• Disadvantage:
  – errors are generally underestimated

• The user can visualize fit statistics as a function of parameter value using INTERVAL-UNCERTAINTY.

• The user can visualize two-dimensional confidence regions using REGION-UNCERTAINTY.
Confidence Intervals and Regions: Projection

• Vary a parameter’s value, while allowing the values of all other parameters to float to new best-fit values, until the fit statistic increases by some preset amount from its minimum value (e.g. \( \Delta \chi^2 = 1 \) for 1\( \sigma \)).

• Gives correct results if and only if:
  – the statistic surface is “well-behaved”

• Advantages:
  – more accurate than UNCERTAINTY
  – provides a relatively inexpensive means of surface visualization

• Disadvantages:
  – no more accurate than the faster COVARIANCE

• The user can visualize fit statistics as a function of parameter value using INTERVAL–PROJECTION.

• The user can visualize two-dimensional confidence regions using REGION–PROJECTION.
Confidence Intervals and Regions: Covariance

- $1\sigma$ confidence intervals are given by $\sqrt{C_{i,i}}$, where

$$C_{i,j} = I_{i,j}^{-1},$$

and $I$, the information matrix computed at the best-fit point, is

$$I_{i,j} \equiv \frac{1}{2} \frac{\partial^2 \chi^2}{\partial p_i \partial p_j} \text{ or } \frac{1}{2} \frac{\partial^2 C}{\partial p_i \partial p_j} \text{ or } \frac{\partial^2 B}{\partial p_i \partial p_j}.$$

- Gives correct results if and only if:
  - the statistic surface is “well-behaved”

- Advantage:
  - fast

- Disadvantages:
  - the only computations are near the best-fit point, so not useful for surface visualization
  - involves matrix inversion, which can fail
Example with a Well-Behaved Parameter Space

sherpa> fit
powll: v1.2
powll: initial function value = 8.22297E+01
powll: converged to minimum = 6.27050E+01 at iteration = 7
powll: final function value = 6.27050E+01
    p.c0  56.2579
    p.c1  0.11117
    p.c2 -0.00119999

sherpa> uncertainty
Computed for uncertainty.sigma = 1

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Best-Fit Lower Bound</th>
<th>Upper Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>p.c0</td>
<td>56.2579</td>
<td>+0.864461</td>
</tr>
<tr>
<td>p.c1</td>
<td>0.11117</td>
<td>+0.0148038</td>
</tr>
<tr>
<td>p.c2</td>
<td>-0.00119999</td>
<td>+0.000189222</td>
</tr>
</tbody>
</table>

sherpa> projection
Computed for projection.sigma = 1

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Best-Fit Lower Bound</th>
<th>Upper Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>p.c0</td>
<td>56.2579</td>
<td>+2.64497</td>
</tr>
<tr>
<td>p.c1</td>
<td>0.11117</td>
<td>+0.120703</td>
</tr>
<tr>
<td>p.c2</td>
<td>-0.00119999</td>
<td>+0.00114976</td>
</tr>
</tbody>
</table>

sherpa> covariance
Computed for covariance.sigma = 1

<table>
<thead>
<tr>
<th>Parameter Name</th>
<th>Best-Fit Lower Bound</th>
<th>Upper Bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>p.c0</td>
<td>56.2579</td>
<td>+2.64786</td>
</tr>
<tr>
<td>p.c1</td>
<td>0.11117</td>
<td>+0.121023</td>
</tr>
<tr>
<td>p.c2</td>
<td>-0.00119999</td>
<td>+0.00115675</td>
</tr>
</tbody>
</table>
Confidence Region – Uncertainty

Confidence Region – Projection
Credible Intervals and Regions

(Bayesian methodology in the tiniest of nutshells.)

- In Bayesian methodology, credible intervals and regions are computed directly from the $\chi^2$ or $\log\mathcal{L}$ surface, using Bayes’ theorem:

$$p(\vec{\theta}|D) = p(\vec{\theta}) \frac{p(D|\vec{\theta})}{p(D)},$$

where

- $p(D|\vec{\theta})$ is the likelihood of the data $D$ given $\vec{\theta}$, the vector of model parameter values
  
  (i.e. $\log\mathcal{L}$ or $\exp(-\chi^2/2)$)

- $p(\vec{\theta})$ is the prior for $\vec{\theta}$

- $p(\vec{\theta}|D)$ is the posterior for $\vec{\theta}$

- $p(D)$ is an ignorable normalization constant

- The ability to specify priors is not yet included in Sherpa.
Credible Intervals and Regions

- To estimate credible intervals, one marginalizes over *nuisance* parameters, *e.g.*:

\[ p(\theta_1|D) = \int_{\theta_2} d\theta_2 \cdots \int_{\theta_n} d\theta_n \ p(\bar{\theta}|D). \]

- The central 68% of the distribution \( p(\theta_1|D) \) is the 1\( \sigma \) credible interval.

- The computation of credible intervals and regions can be computationally intensive if there are many free parameters.

- However, approximate techniques such as adaptive integration are coded in freely available software, such as **BAYESPACK** (by Genz).
Likelihood-Based Statistics

The **BAYES** statistic is the posterior distribution for the source model parameters $\mathbf{\theta}_S$, with the background amplitudes in each (energy) bin $\theta_{B,i}$ marginalized out:

$$
B \equiv -p(\mathbf{\theta}_S|D) = -\sum_i \int_{\theta_{B,i}} d\theta_{B,i} p(\mathbf{\theta}_S, \theta_{B,i}|D)
$$

If $\theta_{B,i}$ is *constant* as a function of spatial location and/or time, then an analytic expression (not reproduced here) replaces the summation of integrals.

**NOTE:** $\theta_{B,i}$ are *implicit* parameters, not user-defined!

---

**How is this statistic different from CASH?**

1. **CASH** makes no assumptions about the behavior of the background as a function of spatial location and/or time.
2. **CASH** performs no implicit marginalization.
New Methods of Parameter Estimation

(Or, what might go into CIAO 4.0...)

Markov Chain Monte Carlo (MCMC) is a well-developed method that works as both an optimizer and a parameter estimator.

- A Markov Chain is an ordered sequence of random variables $\Theta$; the probability of sampling variable $\Theta_i$ depends only upon $\Theta_{i-1}$.
- The Monte Carlo aspect is how possible $\Theta_i$ are chosen: randomly.

To use MCMC, a Sherpa user would:

- specify a rule for how possible $\Theta_i$ are chosen (e.g. select new random values for a subset of the thawed parameters);
- specify a rule for whether $\Theta_i$ is used, or disregarded (e.g. the Metropolis algorithm: given a randomly selected number $r$, $0 \leq r \leq 1$, keep $\Theta_i$ if

$$ r < \min \left[ 1, \frac{\mathcal{L}(\Theta_i)}{\mathcal{L}(\Theta_{i-1})} \right]; $$

- and specify a stopping rule.

The central 68% of the selected parameter values define the $1\sigma$ credible/confidence interval.
Model Comparison Tests

(Which of my models is the best one?)

These do not yet exist in Sherpa. They compare directly compare two models, $M_0$ and $M_1$, to yield either:

- The frequentist test significance, $\alpha$, that represents the probability of selecting the alternative (more complex) model $M_1$ when in fact the null hypothesis $M_0$ is correct; or

- The Bayesian odds, which is the ratio of model posterior probabilities for $M_1$ and $M_0$:

$$O_{10} = \frac{p(M_1|D)}{p(M_0|D)}$$

In simple situations, the model posterior probability is determined by determining the integral of $L$ over all parameter space.
Model Comparison Tests

Standard model comparison tests include:

- The Maximum Likelihood Ratio (MLR) test:
  \[ \alpha_{\chi^2_{\text{MLR}}} = \int_{\Delta\chi^2} d\chi^2 p(\Delta\chi^2|\Delta N_\theta), \]
  where \( \Delta N_\theta \) is the number of additional thawed model parameters in model \( M_1 \).

- The F-test:
  \[ \alpha_F = \int_F^{\infty} dF p(F|\Delta N_\theta, n - N_{\theta,1}) \]
  \[ = I_{\frac{n-N_{\theta,1}}{n-N_{\theta,1}+(\Delta N_\theta)^2}} \left( \frac{n - N_{\theta,1}}{2}, \frac{\Delta N_\theta}{2} \right), \]
  where \( n \) is the number of bins in the fit and \( N_{\theta,1} \) is the total number of thawed parameters in model \( M_1 \), \( I \) is the incomplete beta function, and \( F \) is the \( F \)-statistic
  \[ F = \frac{\Delta\chi^2}{\Delta N_\theta} / \frac{\chi^2_1}{(n - N_{\theta,1})}. \]

- Computation of the Bayesian odds using the Laplace approximation, valid for “well-behaved” surfaces. This approximation yields an analytic formula (not reproduced here) that allows the odds to be computed from \( \Delta \log \mathcal{L} \), \( \Delta N_\theta \), the covariance matrices associated with both models, and the value of the priors at the best-fit points.
Other Future Enhancements to Sherpa

• In convolution and optimization:
  – Treating pile-up.
  – Adding a convolution operator.
  – Adding the ability to use responses directly input from Fits Embedded Function (FEF) files when fitting models.

• In two-dimensional image analysis:
  – Being able to simultaneously fit source and background regions without inputting the background as a separate dataset.
  – Adding the ability to use exposure maps.
  – Extending flux calculations to two dimensions.

• In higher-dimensional data analysis:
  – Improving multi-axis fitting with functionals.
  – Adding visualization of data projected to one or two dimensions.

• And:
  – Enhancing the capabilities of GUIDE to make it easier both to fit a sequence of individual lines and to perform differential emission measure fits.