

# *SLUO Lectures on Statistics and Numerical Methods in HEP*

## *Lecture 7: Fitting Methods*

Roger Barlow  
The University of Manchester  
28<sup>th</sup> August 2000

This is a lecture covering various techniques for function approximation, interpolation, and, most importantly, fitting.

### 1. Optimal Observables

Suppose you are fitting the data to a function of the form

$$f_0(x) + \alpha f_1(x)$$

Which is pretty common, especially including equivalents like  $(1 - \alpha)f_0 + \alpha f_1$ . Examples are signal+background situations, small extra couplings [1,3,4], tau polarisation[2], and plenty more.

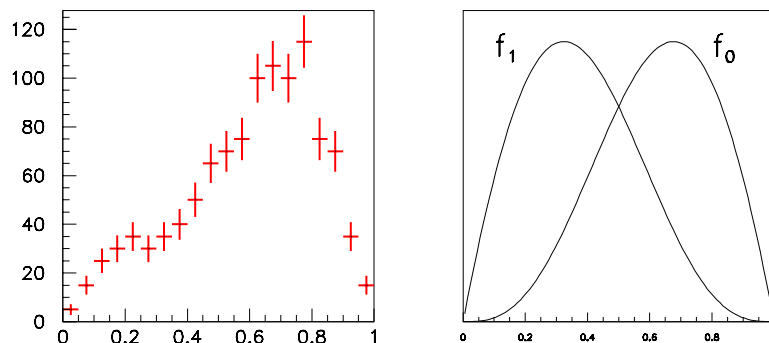


Figure 1: A common problem

Now, if you get an event at some point  $x$  in the plot, that contains information to be handled by the fit. Fine. Clearly the actual value of  $x$  is irrelevant. What matters are the values of  $f_0(x)$  and  $f_1(x)$ . Furthermore, this enters only through the *ratio* of the two. (This is easily seen in the signal/background example.) All that matters about  $x$  is  $\mathcal{O} = f_1(x)/f_0(x)$ . So instead of plotting  $x$  and fitting it, you can transform each  $x$  into the equivalent  $\mathcal{O}$ , and plot that distribution. (Or you can use  $f_0/f_1$ , or anything else trivially related. You do sometimes see different forms.)

And this works even if  $x$  is multidimensional! You can write down a whole slew of indicator variables; at any point in this multidimensional space you have the single value  $\mathcal{O} = f_1(\vec{x})/f_0(\vec{x})$ . This is a great relief, because if you were going to fit this data by histogramming it, you need to have a reasonable number of data entries in each bin.

To simplify some approaching algebra, I'm going to rewrite the function as  $a(x) + \alpha b(x)$  where  $\int a(x) dx = 1$  and  $\int b(x) dx = 0$ . The expected value of  $\mathcal{O}$ , which is  $b(x)/a(x)$ , is then

$$\langle \mathcal{O} \rangle = \int (b/a)(a + \alpha b) dx = \alpha \int (b^2/a) dx$$

so we can use this as an estimator

$$\hat{\alpha} = \frac{\overline{\mathcal{O}}}{\int (b^2/a) dx}$$

The variance of the estimator is obtained from the variance of  $\mathcal{O}$

$$V(\mathcal{O}) = \langle \mathcal{O}^2 \rangle - \langle \mathcal{O} \rangle^2 = \int (b^2/a) dx + \alpha \int (b^3/a^2) dx - \alpha^2 \left( \int (b^2/a) dx \right)^2$$

and

$$V(\hat{\alpha}) = \frac{V(\mathcal{O})}{N \left( \int (b^2/a) dx \right)^2}$$

The MVB for the problem is given by

$$MVB^{-1} = - \left\langle \frac{d^2 \ln L}{d\alpha^2} \right\rangle = -N \left\langle \frac{d^2 \ln(a + \alpha b)}{d\alpha^2} \right\rangle = N \left\langle \frac{b^2}{(a + \alpha b)^2} \right\rangle = N \int \frac{b^2}{a + \alpha b} dx$$

In the limit of small  $\alpha$  the variances are the same, showing that the estimator is efficient. So you don't even need to study the distribution of  $\mathcal{O}$ , you just take the mean.

$\mathcal{O}$  is given the name 'Optimal Observable', and you can see it deserves it. A multidimensional fitting problem has been reduced to a simple sum.

The observable is optimal (i.e. as efficient as Maximum Likelihood) in the limit of small  $\alpha$ . In many cases this is true, particularly in searches for anomalous couplings, new particles, etc. If you are given a parametrisation in which  $\alpha$  is not small, then it is easy to re-parametrise it so that it is. This is bound to work unless the errors on  $\alpha$  are large, i.e. one is forced to consider areas of parameter space where the approximation breaks down.

Exactly the same algebra is behind optimal weighting techniques [7,8]. Using suitable functions means that simple sums provide parameter estimators as good as Maximum Likelihood. Unfortunately these functions themselves involve the values of the parameters [6]. However use of incorrect values need not invalidate the method, and generally does not give an appreciable loss of efficiency[8].

Typically your  $f$  functions are defined by theory. In real life there are acceptance corrections and stuff to worry about. These means that by using the OOs as defined above they are not truly 'optimal', however unless the distortions are severe this is not a worry. More important is the fact that you can't then use the functions to get from the measured mean OO value to the parameter value. Instead you use Monte Carlo simulations (which you trust to be realistic) to calibrate your measurement. Such curves can probably be generated by some clever reweighting.

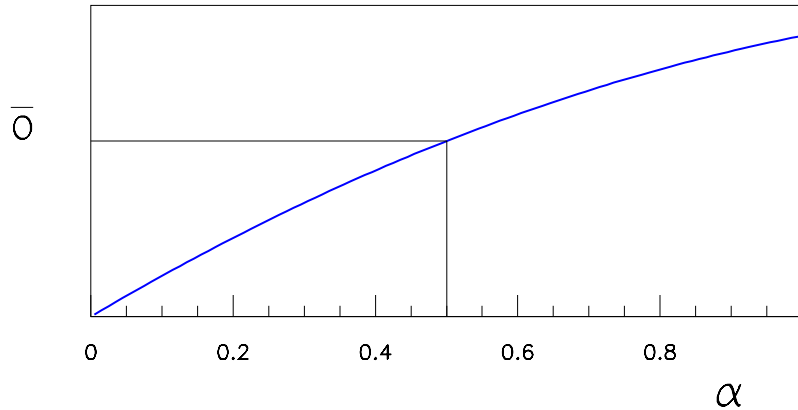


Figure 2: OO calibration

If you have several parameters to fit then each has its own OO. [3]

If the dependence is not linear but quadratic:  $f_0(x) + \alpha f_1(\vec{x}) + \alpha^2 f_2(\vec{x})$  then it can be shown that [4] all the information is contained in the optimal observables  $\mathcal{O}^{(1)} = f_1/f_0$  and  $\mathcal{O}^{(2)} = f_2/f_0$ . However it has not been shown (as far as I know) that in this case all the information is contained in the mean of the OOs, so in principle the fully effective method is to do an ML fit to these distributions. Diehl and Nachtmann [6] discuss their use to obtain improved estimates for the Three Gauge Boson couplings.

**Example 1:** In studying gauge boson interactions in  $W$  pair production, there are (after some hatchet work) 3 independent couplings ( $\Delta\kappa_\gamma$ ,  $\Delta g_1^Z$ , and  $\lambda$ ) whose values are prescribed as zero by the standard model. They are investigated using the 5 angular variables from  $W$  pair production. The 3 variables are studied through 3 linear OOs and 6 quadratic ones. The means of the values are used to give a  $\chi^2$  likelihood: study of the distribution shapes does not seem to generate any further precision.

## 2. Chebyshev polynomials

Faced with the problem of fitting a function to a set of points, where there is no theoretical guidance as to what this function might be, and the shape clearly isn't well described by a straight line, the natural guess is to try a parabola, cubic, quartic...

In doing so it may well be better to fit using a set of proprietary polynomials  $P_n(x)$  where  $P_n(x)$  is a polynomial of order  $n$ , made according to a particular recipe. Examples are the Legendre and Laguerre and Hermite polynomials. In particular, the Chebyshev polynomials  $T_n(x)$  are favourite with serious numerical analysts[5]. (The spelling is more than usually ambiguous.

Known combinations are  $\left\{ \begin{matrix} \text{Ch} \\ \text{Tch} \\ \check{\text{C}} \end{matrix} \right\} \text{eb} \left\{ \begin{matrix} \text{i} \\ \text{y} \end{matrix} \right\} \left\{ \begin{matrix} \text{ch} \\ \text{sh} \\ \text{sch} \end{matrix} \right\} \text{e} \left\{ \begin{matrix} \text{v} \\ \text{ff} \end{matrix} \right\}$  )  $x$  must be in the range -1 to +1. This means you have to scale your data, but this is generally a good idea anyway.

They are defined by

$$T_n(x) = \cos(n \cos^{-1} x)$$

but more usefully through the recurrence relation

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$$

which give

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_2(x) &= 2x^2 - 1 \\ T_3(x) &= 4x^3 - 3x \\ T_4(x) &= 8x^4 - 8x^2 + 1 \\ T_5(x) &= 16x^5 - 20x^3 + 5x \end{aligned}$$

Chebyshevs have the valuable property that any  $T_n$  never deviates from zero by more than  $\pm 1$ , (and does to at  $n + 1$  points.) This is the minimum possible deviation for a polynomial of this order with the same leading coefficient.

As an example, consider function approximation. Any function  $f(x)$  can be expanded as

$$f(x) \approx \sum_r a_r T_r(x)$$

(this generalised Fourier Series approach works for any polynomials, including the Taylor series  $x^r$ ). How does one find the  $a_r$ ? Well, these polynomials are - like many - orthogonal under a suitable weight function.  $\int \frac{T_i T_j}{\sqrt{1-x^2}} dx = 0$  for  $i \neq j$ . So you pick out the coefficient by multiplying and integrating from -1 to +1. This gives  $a_r = \frac{2}{\pi} \int \frac{f(x) T_r(x)}{\sqrt{1-x^2}} dx$  except for  $a_0$ , which has a  $\frac{1}{\pi}$  instead of  $\frac{r}{\pi}$ .

If you keep an infinite number of terms then of course all these polynomial series are equivalent. But if you truncate (as you have to) then they're not. Expansion in Chebyshevs is safe (the polynomials don't deviate strongly from zero), economical (for a given required accuracy, you need fewer Chebyshevs than any other polynomial) and fast (polynomial expansion can be combined with the recursion formula.)

As an example, consider the approximation of  $\sin 3x$  in the range -1 to 1. Only odd powers are involved. The first order Taylor approximation is  $3x$ ; The Chebyshev approximation (from the above integral) is  $0.67x$ . Over the whole range it is clearly much more reasonable.

Going to a cubic term, the Taylor  $3x - \frac{27x^3}{6}$  does OK in the middle but not at the ends, whereas the Chebyshev  $0.67T_0 - 0.629T_3$  does pretty well. And with a 5th term the two are indistinguishable to the naked eye, whereas the Taylor expansion can't manage the ends.

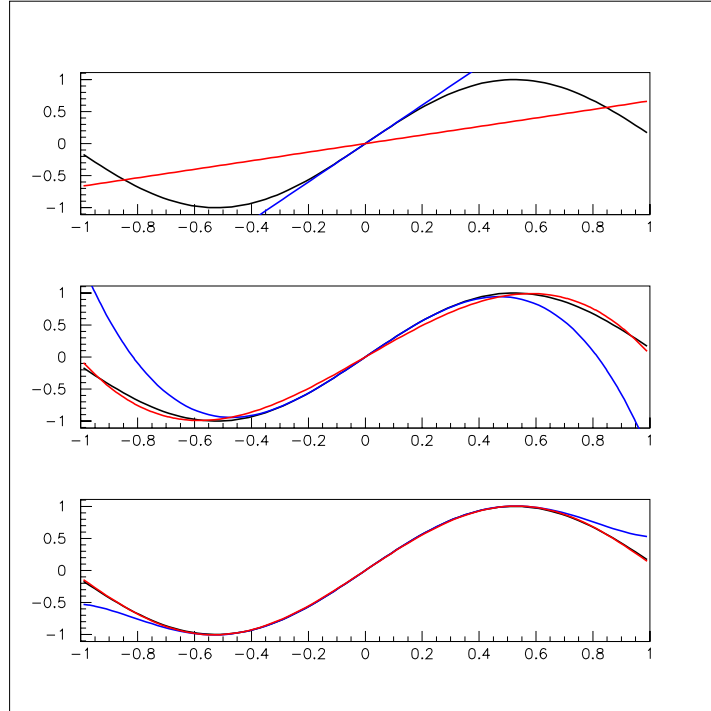


Figure 3: Chebyshev Polynomial Approximations are the best. The sine is shown in each case, with the 1st, 3rd and 5th order Taylor and Chebyshev expansions

The expansion is so rapid that the maximum deviation at any order is essentially given by the next coefficient.

### 3. Orthogonal polynomials

#### 3.1 How not to fit a curve

To fit a polynomial  $f(x) = c_0 + c_1x + c_2x^2 \dots c_nx^n$  to a set of points  $(x_i, y_i)$  using the Least Squares estimate, you minimise

$$\sum_i (c_0 + c_1x_i + c_2x_i^2 \dots c_nx_i^n - y_i)^2$$

(This assumes the errors are all the same. Extension to weighted least squares is straightforward.)

Finding the maximum by differentiating with respect to each of the  $c_i$  and setting all the resulting expressions to zero gives a set of equations that can be neatly written as a matrix

$$\begin{pmatrix} 1 & \bar{x} & \overline{x^2} & \dots & \overline{x^n} \\ \bar{x} & \overline{x^2} & \overline{x^3} & \dots & \overline{x^{n+1}} \\ \overline{x^2} & \overline{x^3} & \overline{x^4} & \dots & \overline{x^{n+2}} \\ \vdots & & & & \\ \overline{x^n} & \overline{x^{n+1}} & \overline{x^{n+2}} & \dots & \overline{x^{2n}} \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} \bar{y} \\ \overline{xy} \\ \overline{x^2y} \\ \vdots \\ \overline{x^ny} \end{pmatrix}$$

This can be solved for the coefficients by inverting the matrix. But there is another way which is *better* and *easier*.

**3.2 Defining and using orthogonal polynomials**

Let  $P_r(x)$  be the polynomial of order  $r$

$$P_r(x) = a_{0r} + a_{1r}x + a_{2r}x^2 \dots a_{rr}x^r$$

where  $a_{rr}$  is 1, and the other coefficients are to be chosen such that

$$\sum_i P_r(x_i)P_s(x_i) = 0 \quad \text{unless } r = s$$

This can easily be done iteratively. We can rewrite the expression for  $P_r(x)$  as

$$P_r(x) = b_{0r}P_0(x) + b_{1r}P_1(x) + b_{2r}P_2(x) \dots + x^r$$

The orthogonality property leads at once to

$$b_{sr} = -\frac{\sum_i x_i^r P_s(x_i)}{\sum_i P_s(x_i)^2}$$

**Example 2:** Show that the first two polynomials (for a straight line fit) are  $P_0(x) = 1$  and  $P_1(x) = x - \bar{x}$

The polynomials are specific to a data sample. This  $P_1(x)$  depends on your particular  $\bar{x}$ .

**Example 3:** Find the orthogonal polynomials up to second order.

Having got them, you can fit the data very easily to  $f(x) = c'_0 + c'_1P_1(x) + c'_2P_2(x) \dots c'_n P_n(x)$  as the equations obtained from Least Squares are

$$\overline{P_r^2} c'_r = \overline{y P_r}$$

As an example, for the histogram shown, orthogonal polynomials were used to fit a straight line, parabola... up to 6th order, using only a few lines of code.

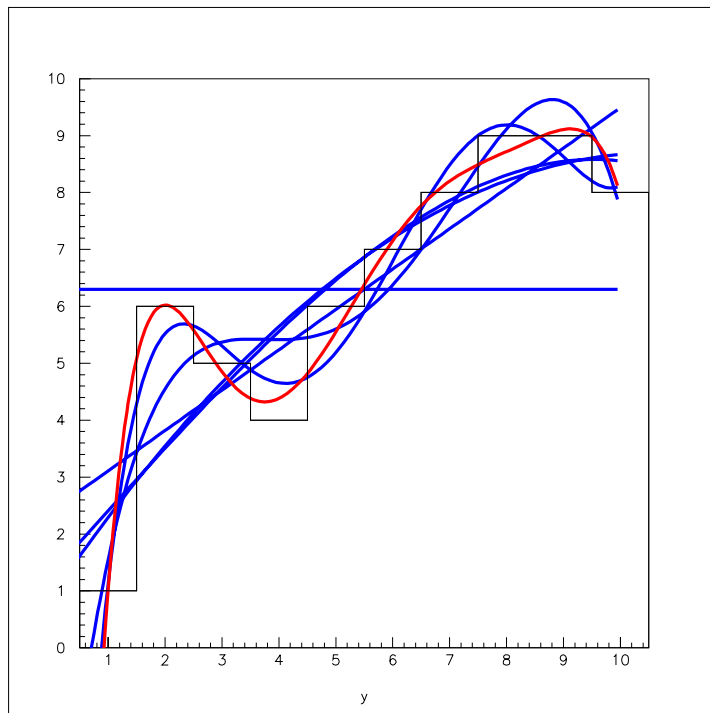


Figure 4: Fitting is easy with orthogonal polynomials

It is instructive to look at how the functions behave outside the range they have been fitted - even slightly outside the range.

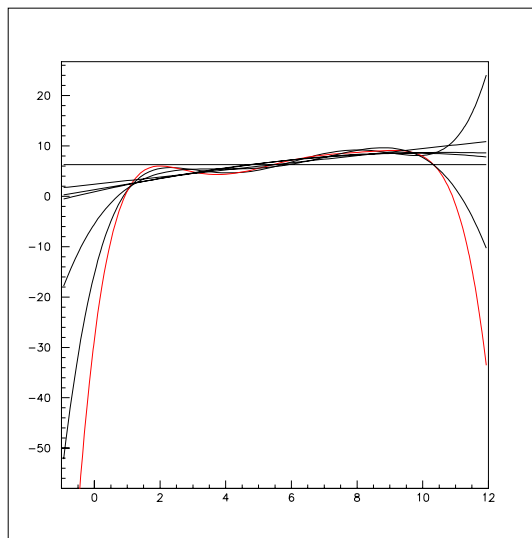


Figure 5: But don't trust them further than you can see them

Orthogonal polynomials are *better* than simple powers because the coefficients are independent. The variance of the  $c_i$  or  $c'_i$  is obtained from the Variance matrix of the vector on the right, pre and post-multiplied by the inverse of that matrix of moments. For simple powers none of these are diagonal; for orthogonal polynomials they all are.

**Example 4:** Show that the covariance matrix  $Cov(\sum_i P_r(x_i)y_i, \sum_i P_s(x_i), y_i)$  is diagonal.

This is well known even to undergraduates\*. If you fit to  $y = mx + c$  then  $m$  and  $c$  are correlated. This matters if, for example, you want to predict the value of  $y$  at some given  $x$ . You have to use  $\sigma_y^2 = x^2\sigma_m^2 + 2\rho x\sigma_m\sigma_c + \sigma_c^2$ . If you fit to  $y = m(x - \bar{x}) + c'$  then  $m$  and  $c'$  (which is actually  $\bar{y}$ ) are uncorrelated.

#### 4. Fitting Data distributions

You've got a sample  $x_1, x_2 \dots x_N$  that you believe has been generated by some pdf  $P(x; a)$  and you want to estimate  $a$  on the strength of your data.

There are four common techniques, Standard  $\chi^2$ , clever  $\chi^2$ , binned maximum likelihood and maximum likelihood. Complexity, computer time and all that increases down the list so you want to get away with the most direct. The first three involve histogramming the data. Unfortunately the earlier methods run into problems if/when the numbers of entries in each bin becomes small.

##### 4.1 Simple $\chi^2$

This minimises

$$\chi^2 = \sum \frac{(n_j - f_j)^2}{n_j}$$

---

\* At Manchester, anyway.

where  $n_j$  is the number of events ( $x$  values) in bin  $j$  and  $f_j$  is the prediction  $P(x_j; a)\Delta$ .  $x_j$  is the value at the center of the bin.  $\Delta$  is the width of the bin (and leaving it out is a very common mistake!)

This is not the real formula for  $\chi^2$  which is  $\sum \frac{(n_j - f_j)^2}{\sigma_j^2}$ . To get from one to the other you use the fact that the standard deviation of a Poisson distribution is  $\sqrt{n}$  and you approximate the true  $\sqrt{n}$  by the observed  $\sqrt{n}$ . As it's in the denominator this usually doesn't make a big difference.

Nevertheless this formula is only an approximation. It is sometimes quoted as a *definition* of  $\chi^2$  and this is wrong. But it is nice to work with in many ways. If  $P(x; a)$  is linear in  $a$  then finding the maximum from  $\frac{d\chi^2}{da} = 0$  gives you an equation that can be solved directly. If  $a$  is in fact several parameters you get a set of such equations which are amenable to matrix methods.

If  $n_j$  is zero, you generally just don't bother about that bin. This avoids division by zero, but is an indication that something's wrong.

#### 4.2 Clever $\chi^2$

If these zeros are a problem, or indeed if the denominator is small, you can use the correct form  $\chi^2 = \sum \frac{(n_j - f_j)^2}{f_j}$  and minimise that. Unfortunately linear expressions then don't give simple soluble equations when you differentiate them (as there's an  $a$  in the denominator.)

#### 4.3 Binned Maximum Likelihood

But the  $\chi^2$  estimator is valid only for Gaussian errors. Thanks to the CLT, that means it's pretty generally useful. However the distribution in the bins is not Gaussian but Poisson. (Or possibly multinomial, but the Poisson is a valid approximation to that - unless you're dealing with a small number of bins and a total number which was fixed beforehand, both of which are rare in our business.) At which point the Gaussian becomes a valid approximation to the Poisson is arguable. But below that you can use the Poisson equivalent of  $\chi^2$

$$\ln B = \sum_j \ln (e^{-f_j} f_j^{n_j} / n_j!) = \sum_j n_j \ln f_j - f_j$$

the Binned Maximum Likelihood,

#### 4.4 Maximum Likelihood

Binned ML is valid, but is throwing away information. You don't know where in the bin the  $x$  value came. If the bins are narrow then this doesn't matter. If the bins are wide (compared to the structure of the function) then maybe it does. In such a case you don't bother to histogram and use the full maximum Likelihood.

$$\ln L = \sum_{\nu} \ln P(x_{\nu}; a)$$

where the sum is over all *events*.

#### 4.5 A consumer test

As an example, each fitting method was tried on the problem of fitting the function

$$P(x; a) = \frac{1}{\sigma} x e^{-ax^2}$$

with  $a$  having the true value 1. Sample sizes of 100, 1000, and 10000 events were used, and performance evaluated by repeating the random number generation and fitting 10,000 times.

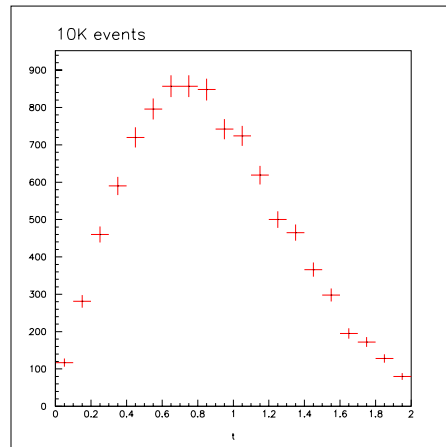


Figure 6: A typical ‘experiment’ with 10,000 values.

With 10,000 numbers, all 4 methods do equally well. The results have no bias (or a very small one), and the resolutions (or efficiencies) are indistinguishable.

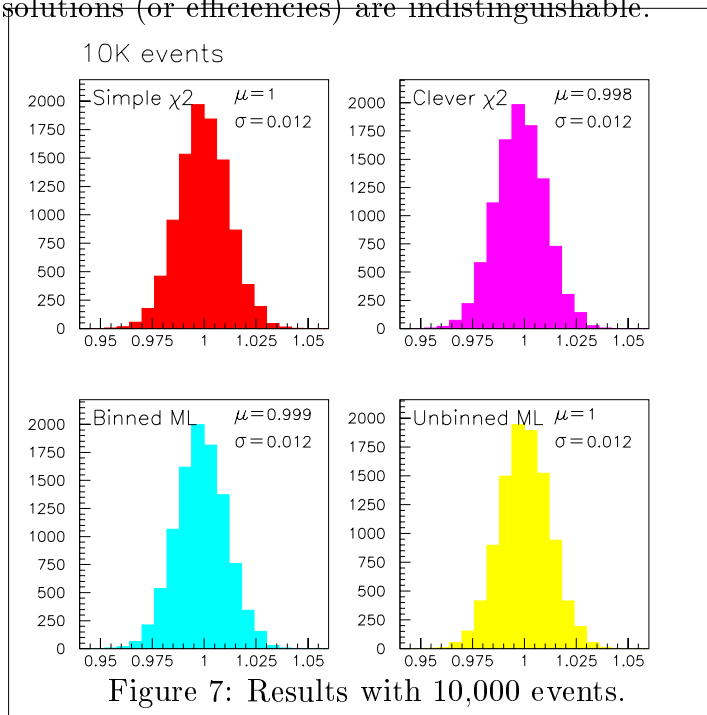
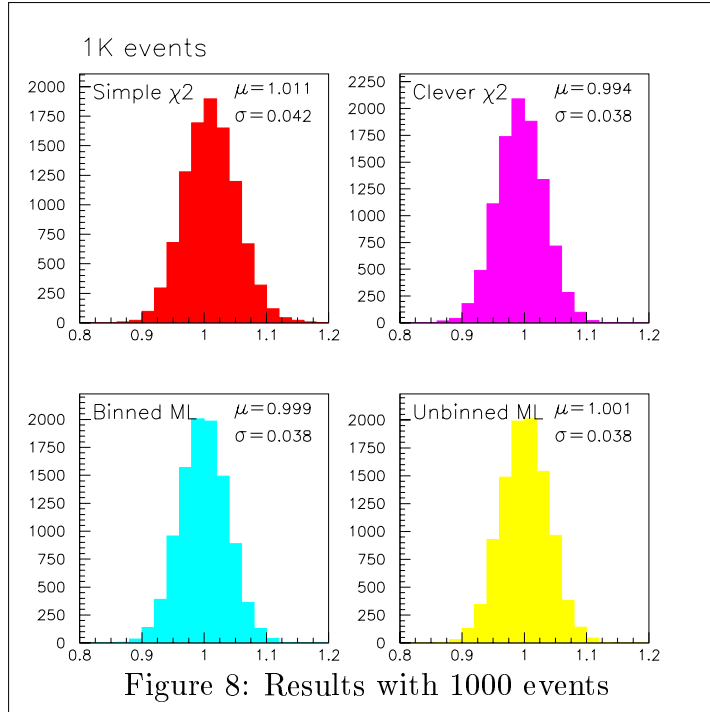
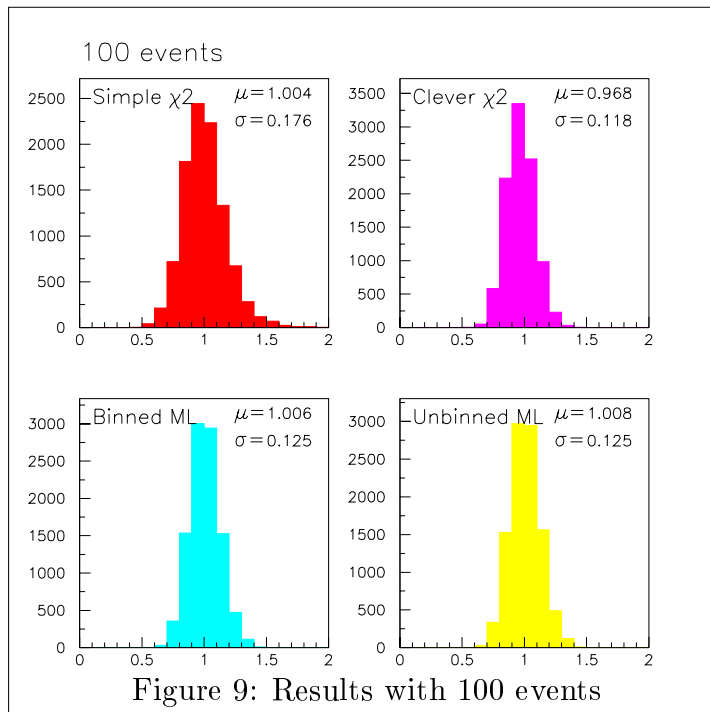


Figure 7: Results with 10,000 events.

With 1000 events things are not so bright (note the change in the scale). The Simple  $\chi^2$  estimate is systematically too high. This is typical. Many of the bins have small (ideal) values. If they fluctuate downwards,  $N$  is smaller and thus  $\sqrt{N}$  smaller than if they fluctuate upwards. Low fluctuations get a disproportionately higher weight than high fluctuations. This effectively pulls down tails, and in this case that leads to a larger value of  $\hat{a}$ . The Clever  $\chi^2$  estimate is too low, though not to the same extent. Real Poisson distributions are skew; there is a reasonable chance of a high fluctuation upwards. The Gaussian assumption gives these a high penalty-value, and tails are effectively pulled up.



Going down to 100 events, the simple  $\chi^2$  resolution has gone bad. there are now several events with 0 bins; these contain useful information but the simple  $\chi^2$  ignores it. The Clever  $\chi^2$  has developed a serious bias; the maximum likelihood methods are also showing a bias, small and upwards. The resolution of Clever  $\chi^2$  looks better than it is; if it were corrected for bias it would become as large as the ML methods' resolution.



Verdict: When you have some bins with small numbers of entries, the  $\chi^2$  method becomes

first biased and then inefficient. Clever  $\chi^2$  does a little better than simple  $\chi^2$ ; sometimes if simple  $\chi^2$  just fails, clever  $\chi^2$  is satisfactory. The Binned and Unbinned Maximum Likelihood methods do pretty much the same. There is no clear advantage for either, arguing that you should use the one which is most convenient.

### 5. Extended Maximum Likelihood

In the standard Maximum Likelihood the probability distribution is normalised to 1, on the grounds that an event has to appear somewhere in the plot. The function used for the above example was actually

$$P(x; a) = \frac{2a}{1 - e^{-4a}} x e^{-ax^2}$$

as the data only considered values between 0 and 2.

In Extended Maximum Likelihood you relax this. The integral of the unnormalised pdf  $\mathcal{P}(x)$  represents the total number of events predicted  $N_{pred}$ . The likelihood for the whole dataset then has to include the Poisson probability that you actually see  $N_{obs}$  events

$$\ln \mathcal{L} = \sum_{\nu=1}^{N_{obs}} \ln P(x_{\nu}) + \ln \left( e^{-N_{pred}} \frac{N_{pred}^{N_{obs}}}{N_{obs}!} \right)$$

The factorial is a constant and can be neglected in the minimisation business. The  $N_{obs}$  factors of  $N_{pred}$  can be included with the normalised  $P(x_i)$  to turn them into unnormalised  $\mathcal{P}(x_i)$ , leaving only the logarithm of the exponential.

$$\ln \mathcal{L} = \sum_{i=1}^{N_{obs}} \ln \mathcal{P}(x_i) - N_{pred}$$

and this is what you maximise. In the above example we could have used

$$\mathcal{P}(x; a, b) = b x e^{-ax^2}$$

In such a case it turns out that [6] the fitted value of the predicted number automatically turns out to be the observed number, and the estimate(s) of the shape parameter(s)  $a$  come out the same as the ML estimate(s). You save some algebra at the price of an extra variable to minimise, and whether you do that or not is just a matter of convenience.

If, however, you have a function where the parametrisation of the shape and size are not and cannot be separated, then you get a fitted estimate of  $N_{pred}$  different from the observed number, and a different shape estimate from the ML estimate, and these are both *better* estimates.

### 6. References

- [1] D Atwood and A Soni, Phys Rev D 45 (1992) 2405
- [2] M. Davier et al., Phys. Lett B 306 (1993) 411
- [3] M. Diehl and O. Nachtmann, Z. Phys. C 62 (1994) 397
- [4] G. K. Fanourakis et al., Nucl. Instr. Meth. A 414 (1998) 399
- [5] H Wind in Vol. 3 of 'Formulæ & Methods in Experimental Data Evaluation', CERN 1983
- [6] R Barlow, Nucl. Inst. & Meth. A297 (1990) 496
- [6] M. Diehl and O. Nachtmann, Eur. Phys. J. C 1 (1998) 177
- [7] J. F. Gunion, B. Grzadkowski and X.-G. He, Phys. Rev. Lett 77 (1996) 5172
- [8] R. Barlow Journal of Computational Physics, 72 (1987) 202