

Fitting Distributions

Instructions

- Please submit your homework by:
 - copying it to "submit" directory
 - changing access rights to it: `chmod 740 submit/<yourfile>`

Links and References:

- User's Guide: <http://root.cern.ch/drupal/content/users-guide>
 - Fitting histograms: <http://root.cern.ch/download/doc/5FittingHistograms.pdf>
- Reference Guide: <http://root.cern.ch/root/html534/ClassIndex.html>
 - TH1F: <http://root.cern.ch/root/html534/TH1F>
 - TF1: <http://root.cern.ch/root/html534/TF1>

1 Fitting and Data Analysis

Fitting is one of the most frequent “measurements” one does in analyzing the data. Fitting is a procedure designed to extract parameters of a function that is hypothesized to be describing a distribution observed in data. The data would typically be represented with a histogram and you would want to fit an analytical function, which can depend on multiple parameters, to the data in order to extract the most likely set of parameters, their uncertainties and the correlation matrix, which we will discuss later. Note that in this discussion we assume that the true function describing the data is known. In a real life analysis, one would need to perform a number of studies to establish whether the data and the function you use are compatible and establish how likely it is that the data is indeed following the function you are using.

An example of a simple use case for fitting would be finding the rate of “signal events” due to resonant production over combinatorial background using the distribution of the experimentally measured invariant mass (see example in Fig. 1(left)). In this case you would want to build the fit function as a sum of two functions: $B(x)$ will be a smooth function describing the backgrounds (a simple polynomial may work well) and a bell-shape function $S(x)$ to describe the signal:

$$F(m) = B(m) + S(m) \tag{1}$$

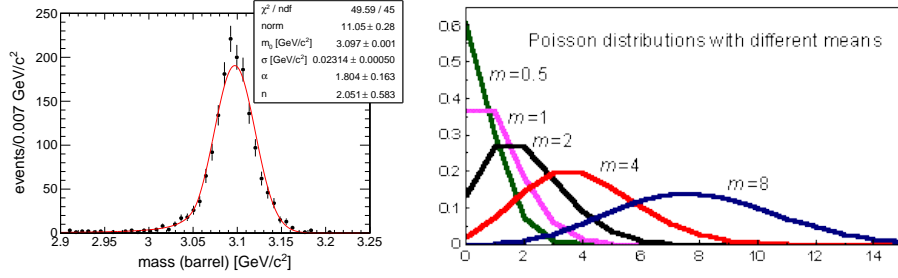


Figure 1: Left: an example of using the distribution of the experimentally measured invariant mass of two muons for fitting of J/ψ resonance production ($m_{J/\psi} = 3.097$ GeV). Right: Poisson distribution for $\nu = 1$, $\nu = 5$, $\nu = 10$.

If we choose to use a second degree polynomial and a gaussian to describe the shape, the fit function would look like this:

$$F(x) = B_0 + B_1m + B_2m^2 + S_0 \exp\left(-\frac{(m - m_0)^2}{2\sigma^2}\right) \quad (2)$$

with 6 parameters, three B_i 's describing the background distribution and three describing the signal. Note that while we only talked yet about the rate of signal events (which would be something you need to know in order to measure the cross-section) S_0 , other parameters may be no less important, e.g. in this example x_0 would be the mass of the signal resonance and σ would be the width Γ , which is directly related to the lifetime ($\tau = 1/\Gamma$).

ROOT provides a set of easy tools to perform a variety of simple fitting tasks, one can even use a GUI (called FitPanel, which you can call by clicking on the canvas displaying a histogram) to get “fast results”. Using GUI quickly becomes cumbersome as the complexity of the problems you are trying to solve grows. Except very simple cases (quick fit to a gaussian plus a constant), you would want to write a script and use fitting methods built into the basic histogram class TH1.

With more complex problems, simple built-in fitting methods are often not adequate due to the limited number of handles and knobs that one may need to use to analyze and validate the results of the fit. In that case, one would use the MINUIT package, a minimization program written originally in FORTRAN and later translated into the C++ code available in ROOT class TMinuit.

2 Fitting Mechanics

Fitting is an example of a minimization problem, where one seeks for a set of parameters that minimizes a certain function that you chose as the figure of merit to quantify the level of agreement between the function that you think

is describing the distribution data and the distribution itself. One example to construct this quantity is the χ^2 method with the metrics defined as:

$$\chi^2(\vec{p}) = \sum_{i=1}^{N_{bins}} (f(\vec{p}, x_i) - D_i)^2 / 2\sigma_i^2, \quad (3)$$

where f is the value of the function in bin i , D_i is the measurement (the data) in bin i , and σ_i is the “uncertainty” associated with f in this bin. Let’s discuss σ and for that we will come back to the example discussed earlier. If the data you are looking at is the number of events in the bins of the invariant mass distribution, then the function you are trying to measure is the yield (the true rate of the events coming integrated over the time of the measurement). If you knew the function describing the distribution perfectly well (someone who we believe unconditionally told you) and the value of that function in bin i was 10.1 (say in events per month and your data is taken over one of these 1-month long running periods), you do not expect that the data will yield exactly 10.1 events in that bin. That would not even be possible as events in data come in integers. You would expect that sometimes you will get 10, sometimes 9 or 11, less frequently 8 or 12 etc. The distribution that the data would follow is the Poisson distribution that describes the probability of observing N “events” when you expect ν events, see Fig. 1(right)). So even if ν is known perfectly well (has zero uncertainty), N is still expected to vary around ν in some range. For large ν you expect that 68% of the time N would fall within $\nu \pm \sqrt{\nu}$. The $\sqrt{\nu}$ term in this case is the “variance” which is the measure of how close you expect D_i and f_i in the formula above to be. So in this case you would use $\sigma_i = \sqrt{f_i}$ in the formula above as this would tell you if the number of the events in data is reasonably compatible with the rate ν given expected statistical fluctuations. Note that the uncertainty is the square root of the yield (or rate) and not a square root of the number of events in data, which is a frequent mistake made. The algorithm then would be to find parameters that minimize the value of χ^2 as those parameters would give you the best agreement with the data. If that’s what you do, you are using the “*chi*² minimization”. It is similar to the method of least squares you must have used a lot in your undergrad labs.

One problem with the χ^2 method is that once ν becomes small, using the Poisson variance is no longer a good approach as the range $\nu \pm \sqrt{\nu}$ is no longer correctly describing the probability of various possible outcomes for the number of events in data for a given ν . Look at Fig. 1(right) - the distribution is nowhere close to being symmetrical if the rate is less than 5. If you expect that the data or the function can come out in your experiment in “small numbers” (or you use very narrow bins so that on average you have only very few events expected per bin) you have to use a so-called “maximum likelihood” method, which evaluates Poisson probabilities of different outcomes explicitly. In this case you will be minimizing the function that looks like the following:

$$\mathcal{L} = \prod_{i=1}^{N_{bins}} P(f_i, D_i), \quad (4)$$

where

$$P(\nu, N) = \nu^N e^{-\nu} / N! \quad (5)$$

is the Poisson distribution shown in Fig. 1(right). If ν is sufficiently large, method of maximum likelihood will be equivalent to simple χ^2 minimization.

These two are the most frequently used approaches. The minimum of these functions gives you the most probable set of parameters of the function, and you can estimate the uncertainty in the parameters by finding the range of the parameters which change the metrics you use “within some reasonable range near the minimum”. The latter of course needs to be defined more rigorously than this verbal statement. We will talk about this later, but it turns out that allowing χ^2 to vary by ± 1 from the minimum corresponds to the 68% C.L. on the parameter measurements, which we usually use to quote the uncertainties.

3 Practical Considerations

In math problems where you solve a system of equations, you don’t want to have less equations than you have variables. For the same reason, when you decide how many bins you want to use in your data distribution (keep in mind that it’s always up to you, even in this class you can take the histogram we give you and re-bin it to have twice less bins, right?), you do not want to have less bins than you have parameters.

Another consideration is that the fit is no smarter than you are and so if you make too coarse bins, the fit will not be able to see where the maximum is and will end up giving you a poor measurement only because you chose bins to be too wide. You want all important features of the distribution to be “seen” by the fit, which in the example of the fit we have been talking about means that in the range where the bell function goes up and down you want to have at least a few bins (5-6 or more if possible) to be able to measure x_0 and σ .

Another consideration is you want to define the function “well”. Consider an extreme example where in the function considered earlier I would have added a new parameter:

$$F(x) = B_0 + B_1 x + B_2 x^2 + (S_0 + S_1) \times \exp\left(-\frac{(x - x_0)^2}{2\sigma^2}\right) \quad (6)$$

Would the fit be able to “measure” S_0 and S_1 well? Of course not as the split is “unphysical”, the fit will only be able to constrain the sum of the two, but it would never be able to tell you (and neither would you) what are the probable values of S_0 and S_1 are unless you include some other knowledge from some outside source. This would be an example where a poorly written fitting package will give a random answer, a little smarter one will fail and a good one will tell you that there is a large correlation of the parameters and that there is no single minimum in the function you are minimizing. The latter is very easy to see as any combination of S_1 and S_2 that gives a fixed sum will always

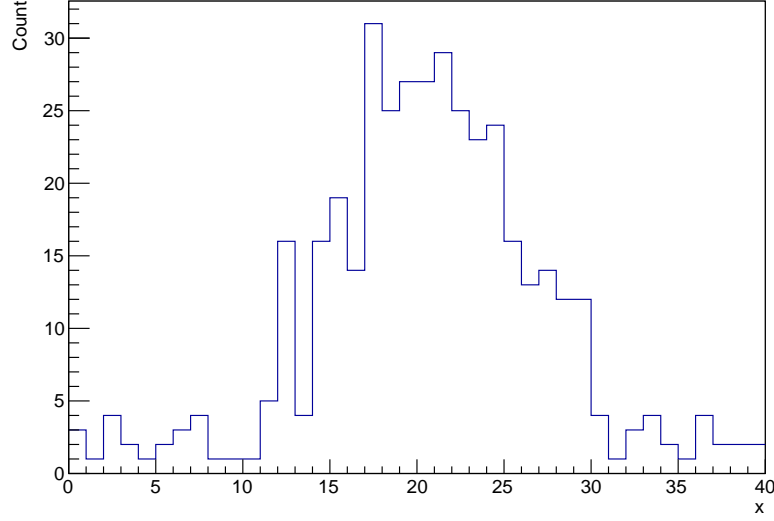


Figure 2: A histogram from Lab_Assignment_004.root

Table 1: Four simple parametric function choices with one floating parameter.

$$f = p_0^2 + p_1 \cdot \exp\left(-\frac{(x-p_2)^2}{2 \cdot p_3^2}\right)$$

	p_0^2	p_1	p_2	p_3
1	2.5	25.0	20.4	float
2	2.5	25.0	float	4.8
3	2.5	float	20.4	4.8
4	float	25.0	20.4	4.8

have the same χ^2 or \mathcal{L} , and your function will have a “valley” in the multi-dimensional space of its parameters near the minimum. Well defined function should have little correlation or you have to be prepared to deal with that later on, which is possible but in simple cases requires disproportionately large effort to accomplish and is usually not practical.

4 Lab Assignment

As always, you will need to produce your report as a LaTeX “article” with the required plots, figures and discussions documenting your research.

Table 2: Four simple parametric function choices with two floating parameter.

$$f = p_0^2 + p_1 \cdot \exp\left(-\frac{(x-p_2)^2}{2 \cdot p_3^2}\right)$$

	p_0^2	p_1	p_2	p_3
1	2.5	25.0	float	float
2	2.5	float	float	4.8
3	2.5	float	20.4	float
4	float	float	20.4	4.8

4.1 Part 1: Building and studying minimization functions

You will use a histogram given to you in the file Lab_Assignment_004.root (see Fig. 2) and four simple parameteric function choices (see Tab. 1), which we will first try to fit to the data by hand in order to evaluate the parameters.

- Calculate χ^2 as a function of the floating parameter and plot the resulting function for each function given to you; make sure the range for the parameter is broad enough). Discuss features of the distribution.
- Use your graph to evaluate the most probable value and the uncertainty for the parameter using the χ^2 method for each function. Explain how you did that and what your conclusions are.

In the above examples we have simplified the task by giving you a function where all parameters but one have been fixed. Now we will use four choices of a 2-parameter function (see Tab. 2):

- Calculate and plot χ^2 versus two floating parameters p_1 and p_2 . For visualizing 2-dimensional functions, it is convenient to draw contours that show you the regions where the function (χ^2 in this case) has a constant value. Such drawing option is available in ROOT.
- Evaluate the most probable values for the measured parameters p_1 and p_2 and their uncertainties. Discuss how you would do that in each case. If for any of your functional choices, such measurement turns out to be difficult, discuss what is the reason for that and how you think you should handle such cases.

Now repeat the same measurements using the method of maximum likelihood L . In this case you will need to explicitly calculate Poisson probabilities. Repeat all the steps above using L as your metrics. For evaluating uncertainties, in the case of χ^2 method we were looking for the point(s) where χ^2 changes by one unit from its minimal value. For the likelihood, what would be the corresponding “shift” from the minimum you will be looking for? Is it something as simple? *Hint: often, people calculate and draw $\log L$ instead of L , which simplifies evaluating uncertainties. If you encounter issues with numeric stability in calculating L , this should also give you an idea of how to circumvent those.*

4.2 Part 2: Numeric minimization (1D and 2D cases only)

So far, you have been evaluating uncertainties using visual information seen in the graph. Next, we will write a simple code searching for the minimum of a given function. One simple algorithm for doing so would be to pick a random value of the parameter, evaluate the function, then step to the left or to the right and re-evaluate the function. Once you see the direction in which the function decreases or increases, walk in that direction until the function changes its gradient. Once that happens, reduce the step size (in half?) and walk back. Keep iterating until you are satisfied with the accuracy of the position of the minimum/maximum. How do you decide what accuracy is acceptable?

To evaluate the uncertainty, you need to find value of the parameter, at which the function changes from its minimal/maximum value by a certain value, which determines the size of the uncertainty. Write code that would find that point. Start with one-dimensional example, then move to a two-dimensional case. How would you report uncertainty in a 2-dimensional case?

4.3 Part 3: Learn to Use the ROOT Fit Panel

Next, we will use the ROOT FitPanel and learn how to use it (open the TBrowser, open the histogram, point the mouse onto the histogram, right-click on the mouse and you should see a menu of options, one of them is the FitPanel).

Repeat the fits using TH1 Fit methods using the same simple functions you were given earlier. In most standard cases you can type the function according to ROOT convention and ROOT will interpret that message and do all the work for you. For example, to define " $p_0 + p_1 \cdot (x - p_2)^2$ " fit function type "[0]+[1]*(x-[2])^2" (see Fig. 3).

Next, you will use your own function TF1 with parameters that you will pass to the Fit method. Once you start working with more complex functions and want to try different functions, this is what you would most often be doing. Using "your own" functions will also simplify your transition to using the MINUIT package later on.

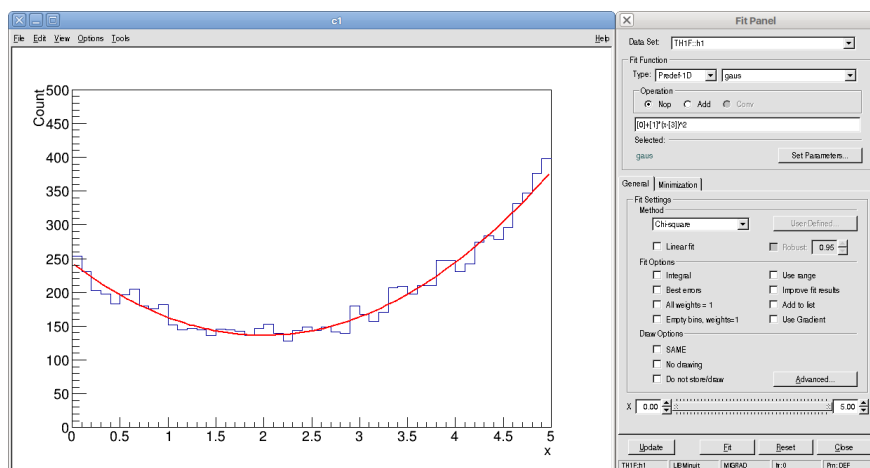


Figure 3: Example of fitting a histogram using ROOT Fit Panel.