Second Lab on Muon Lifetime. Data Analysis

Will Johns, Paul Sheldon, and Med Webster

January 21, 2001

Introduction

The run has completed and the pertinent data from the run has been assembled in the accompanying plots and in a computer file. The instructions for the eyeball fit method using the plot provided should give you some feel for the process of obtaining a value for the lifetime from the data and serve as an introduction to the improved method outlined below. The improved method involves a least squares fit. Since some students are familiar with least squares only as a "black-box" option on spread sheets, a brief introduction to least squares is provided as an appendix. You are cautioned that many least squares procedures use unweighted data. The errors on the data points obtained in this experiment are the square root of the number of hits in the bin and hence vary a great deal: it is just plain wrong to use unweighted fitting for these data.

The data file is a simple ascii (text) file with the (Raw) data you took. Each number is the time read from the scope after the trigger initiated a readout. To perform the fit to the data, proceed as follows:

1 Make a copy of your data file (just to be safe)

2 Stop your data acquisition (Do NOT NOT NOT start again!)

3 Print out a copy of the histogram (plot) of the data

Do this by clicking on the left uppermost rectangle in the plot window and choosing "Plot". You can change the scale by modifying the "Properties" (same menu bar) before you print it out.

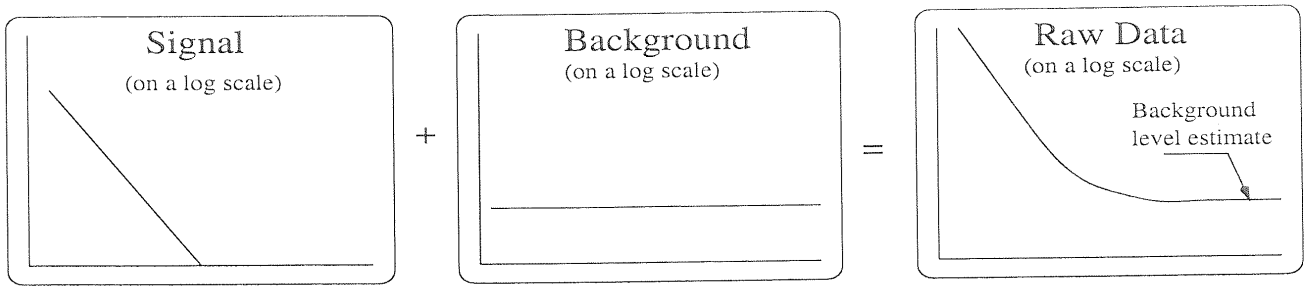
4 Start the "Muoan" program by double clicking on it.

5 Double click on "Read Data From File"

6 Choose your data file (you can shrink this box now)

7 Choose a Background Level, enter it in "Background Level or B"

This background level is the number of events in each bin (see Appendix B for a discussion on how the histogram and the fit variables are formed) you think come from random sources. The easiest way to determine this is to look at the very long lifetimes in the plot you printed out in "3" above. (Appendix C has a more accurate method.)



8 Click "CLICK HERE TO BEGIN ANALYZING DATA"

The calculation is a linear least squares fit to the natural log of the counts per bin (after background subtraction). This fit minimizes the sum, over the bins used, of the square of the difference between fit prediction and the log of the observed number of events divided by the error on the log of the observed number of events. Development of the linear least square formulas is shown in most elementary statistics books and in Appendix A. Straight forward application of the usual derivative procedure for propagation of error shows that the error on the log of the number of events is simply the reciprocal of the square root of the number of events.

Here are the formulas calculated in "MUOAN" and what they mean:
Form A - The bin centers
Form B - The level of "Noise" or Background
Form C – the bin contents (useful for the error too).
Form D – natural log of (C-B). (This is what is plotted as a check.)
(note: D is programmed to have a minimum bin content of 1)
Form E – contents of C * contents of D.
Form F – A*C
Form G – A*C*D or better A*E
Form H – A*A*C or better A*F
C – The sum of the "Form C" elements
E – The sum of the "Form E" elements
F – The sum of the "Form F" elements
G – The sum of the "Form G" elements
H – The sum of the "Form H" elements

NOTE: If you see a lot of "1" bins in the histogram of your modified data, you should rethink the value you used in the "Maximum Time to accept" box (see Appendix B).

The variables created by the program are the coefficients for a pair of equations for the least squares estimate of the vertical intercept (of the log) and the slope of the fit. In a notation where the capital letters stand for the variable name in the program:

$$\mathrm{E} = \mathrm{C}\,\mathrm{b} + \mathrm{F}\,\mathrm{m} \quad \text{and} \quad \mathrm{G} = \mathrm{F}\,\mathrm{b} + \mathrm{H}\,\mathrm{m}$$

or, in matrix notation:

$$\begin{pmatrix} E \\ G \end{pmatrix} = \begin{pmatrix} C & F \\ F & H \end{pmatrix} \begin{pmatrix} b \\ m \end{pmatrix}$$

with the solution

$$\begin{pmatrix} b \\ m \end{pmatrix} = \begin{pmatrix} C & F \\ F & H \end{pmatrix}^{-1} \begin{pmatrix} E \\ G \end{pmatrix}$$

where the inverse matrix is

$$\begin{pmatrix} C & F \\ F & H \end{pmatrix}^{-1} = \frac{\begin{pmatrix} H & -F \\ -F & C \end{pmatrix}}{\det \begin{pmatrix} C & F \\ F & H \end{pmatrix}}$$

or m = (CG-EF)/(CH-FF). Solving these equations for m and taking its negative reciprocal gives the estimate of the muon lifetime in nanoseconds. Using the matrix inversion method is helpful in calculating the errors. The statistical error on the slope, m, is the square root of the 2 2 element of the inverted matrix, C/(CH-FF). The lifetime is the reciprocal of m and the error on the reciprocal is the error divided by the square of the value.

You should determine how sensitive your calculation is to the limits you chose for the maximum and minimum lifetime. Be careful how you choose your values, the program is not sophisticated enough to know that you want each bin to be the same. In other words, make choices in which

$$(Maximum - Minimum)/(Step)$$

is a whole number.

Your errors may be slightly smaller than those from the more complicated fit to background along with slope and intercept. That method takes uncertainty in the background into account. You should vary the amount of background which you subtract and see how much that changes your answer and then increase your error to account for this additional uncertainty. See Appendix C for a method to estimate a background level and a background level error.

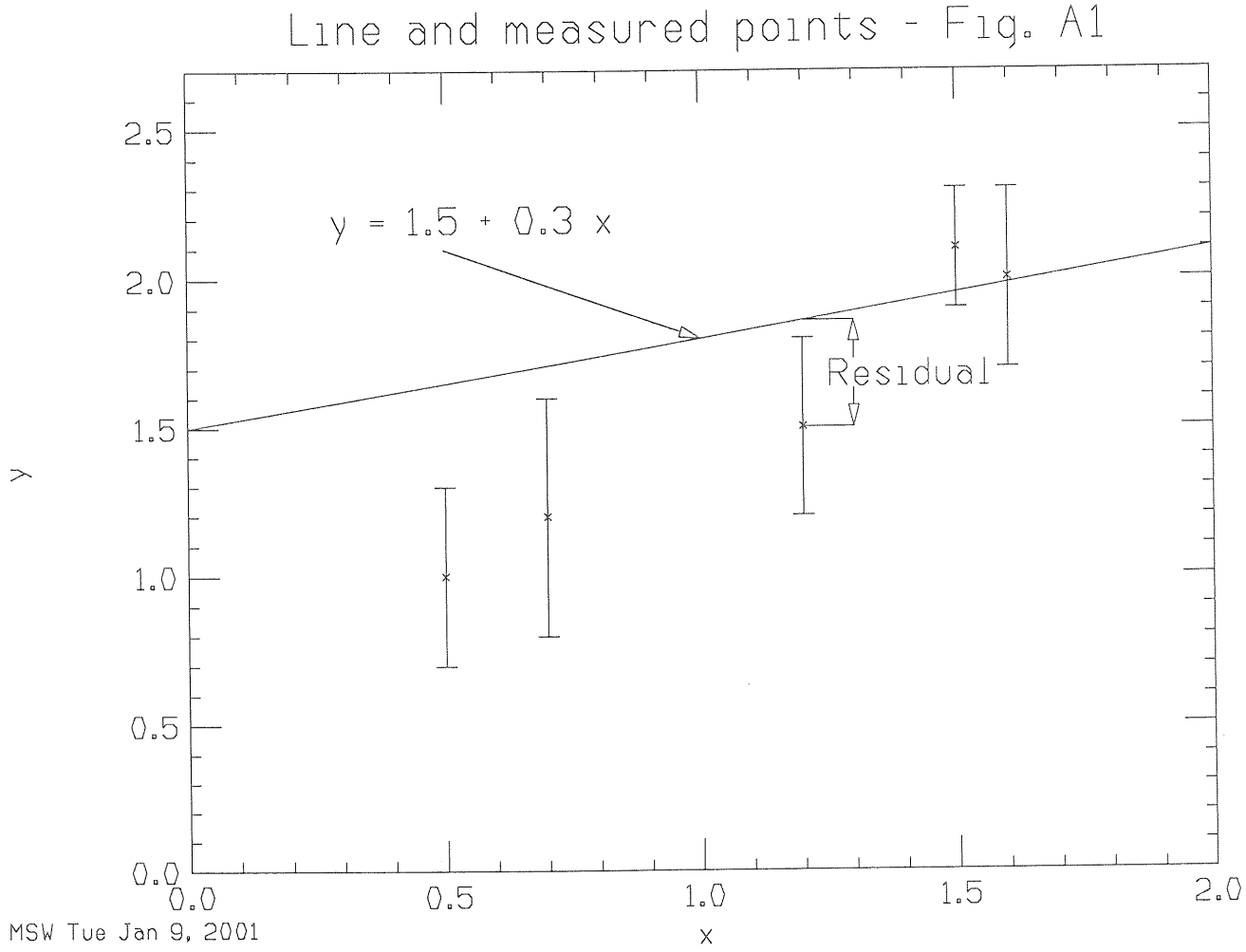
Only if there is time do this part. There is a utility you can use to check your fit. The utility, called "muogen", will generate a fake data set with the input parameters that you designate. A file will be output that you can read in with the fitting routine. By doing this activity 20 times or so, you can plot the results of the fit to the 20 fake data sets and see if the errors returned by the fit are reasonable. High energy physicists call this the "monte carlo" method (because you are rolling the dice). An explanation of the procedure can be found in Appendix D.

Question: Do we need time dilation?
a) If there were no time dilation, how far would a muon moving at very nearly the speed of light go in one lifetime?
b) Airplanes sometimes fly as high as 30,000 feet above sea level. How many of the lifetimes you calculated in a) is 30,000 feet?

c) If the flux at sea level is 50 mrem/year, what is the flux at 30,000 ft?
d) The (short term) radiation dose which kills half the people exposed is 500 rem. How long would an airplane have to remain at 30,000 feet to expose the passengers to the half lethal dose if the flux were as calculated in c)?

Appendix A, Weighted Least Square fit

Suppose we have a variable y, which we expect to be linearly related to a second variable, x. That is we expect $y = c_1 + c_2 x$. Further suppose that we can measure y_i with an error σ_i for a setting of x_i (which has negligible error), and that we have done this measurement for n values of i, i=1,n. In this lifetime application, y_i is the log of the events in bin i (after background subtraction), σ_i is the error on y_i which is the square root of the number of events in the bin before background subtraction, and x_i is the time at the midpoint of bin i. Figure A1 is a plot of 5 data points with errors and of the line with $c_1 = 1.5$ and $c_2 = 0.3$. The line shown is obviously a bad fit to the data and the goal is to choose values of c_1 and c_2 which give the best possible fit. The residual, r_i, is the distance (vertically) from the data point to the line and the probability that the measurement misses the line by this much is related to r_i/σ_i. The definition of a least squares fit is the choice of c_1 and c_2 which minimizes the sum of the squares of r_i/σ_i. The squares are used because they are related to the likelihood of a fluctuation, just as it is squares which occur in the exponent of a Gaussian distribution. While a linear fit with only two unknown coefficients is the theoretically expected form for the curve in the lifetime example, the method is easily generalized to the case of the m coefficients of a polynomial of degree m-1.



The polynomial can be written:

$$y = \sum_{j=1}^{m} c_j \, x^{j-1}$$

and the sum of the squares of the residuals over the errors is

$$R = \sum_{i=1}^{n} \left(\frac{y_i - \sum_{j=1}^{m} c_j \, x_i^{j-1}}{\sigma_i} \right)^2$$

Note that the i index runs over the (x_i,y_i) data points and that the j index runs over the coefficients and power of x for evaluation of the polynomial at the i data point. The least squares fit is the choice of the m coefficients, c_j, which minimizes R. These coefficients are found by taking the derivative of R with respect to each of the c's in turn and setting the derivative to zero. This gives m linear equations for the m unknowns and is readily solved by the method of determinants. We will formulate the solution in matrix notation because that method also yields an estimate of the error on each of the coefficients. After a modest rearrangement and reversing the order of the summations, the equation for the derivative with respect to c_k becomes:

$$\sum_i \frac{x_i^{k-1} \, y_i}{\sigma_i^2} = \sum_j \left(\sum_i \frac{x_i^{j+k-2}}{\sigma_i^2} \right) c_j \qquad k = 1, m$$

or S = M C where S, M, and C are the matrices:

$$S = \begin{pmatrix} \sum_i \frac{y_i x_i^0}{\sigma_i^2} \\ \sum_i \frac{y_i x_i^1}{\sigma_i^2} \\ \vdots \\ \sum_i \frac{y_i x_i^{m-1}}{\sigma_i^2} \end{pmatrix} \qquad M = \begin{pmatrix} \sum_i \frac{x_i^0}{\sigma_i^2} & \sum_i \frac{x_i^1}{\sigma_i^2} & \cdots & \sum_i \frac{x_i^{j-1}}{\sigma_i^2} \\ \sum_i \frac{x_i^1}{\sigma_i^2} & \sum_i \frac{x_i^2}{\sigma_i^2} & \cdots & \sum_i \frac{x_i^j}{\sigma_i^2} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_i \frac{x_i^{m-1}}{\sigma_i^2} & \sum_i \frac{x_i^m}{\sigma_i^2} & \cdots & \sum_i \frac{x_i^{2m-2}}{\sigma_i^2} \end{pmatrix} \qquad C = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \end{pmatrix}$$

Multiplying on the left by M^{-1} gives

$$M^{-1}S = M^{-1}MC = C \quad \text{or} \quad C = M^{-1}S$$

In order to show how to calculate the inverse matrix, it is helpful to define M_{ij}, the ij minor of the matrix M as the matrix (of order m-1) which is obtained by deleting the ith row and the jth column of M. With this definition, the elements of M^{-1} are

$$(m^{-1})_{ij} = \frac{-1^{i+j} m_{ij} \det(M_{ij})}{\det(M)}$$

The minor of a second order matrix is simply the diagonally opposite element, so this inversion is particularly simple for second order matrices and the results are shown explicitly

in the body of the writeup. The use of matrices as shown here is treated in linear algebra courses and you should consider taking such a course if you have not done so already.

If the problem of determining the best polynomial is formulated as a maximum likelihood problem, the same function, R, is minimized and the significance of the inverse matrix becomes clear: The diagonal elements of the inverse matrix are the squares of the errors of the calculated coefficients and the off-diagonal elements are the correlation coefficients of the errors. For this lab we are concerned only with the error on the slope which is the square root of the 2-2 element of the inverted matrix. The error on the lifetime, which is the negative of the reciprocal of the slope, is calculated from the usual formula for the propagation of small errors:

$$\sigma_{\mathrm{y}} = \left|\frac{\partial \mathrm{y}}{\partial \mathrm{x}}\right| \sigma_{\mathrm{x}}$$

or

$$\sigma_{\mathrm{lifetime}} = \left|\frac{\partial 1/\mathrm{m}}{\partial \mathrm{m}}\right| = \frac{1}{\mathrm{m}^2}\,\sigma_{\mathrm{m}}$$

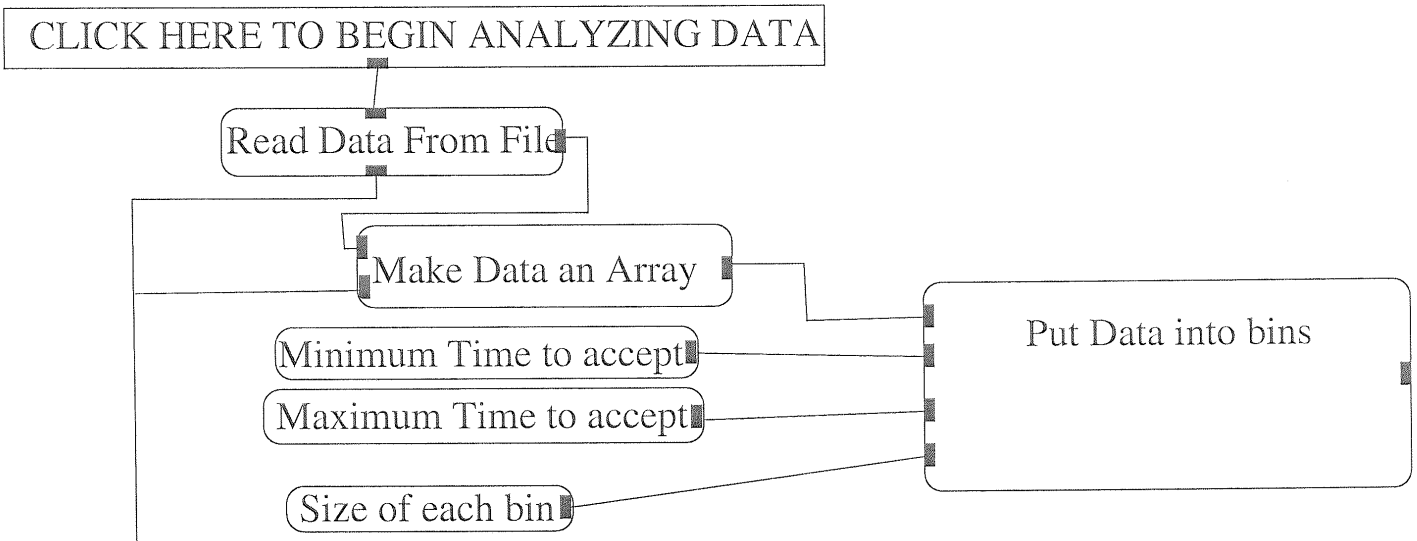
where m is the fitted slope.

Appendix B: The Muoan Program

The MUOAN program is a lot like the program you used to take data in the first part of this lab.

Take a look at the structure:

Step 1: Read in the data, form the data into a histogram.



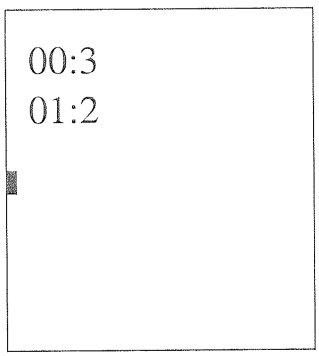
After we read in the data (you can double click on that box to change the name) we form the data into an array of results (really a vector) that is as long as the number of events we read in.

Then, we use a function that organizes the data into bins according to our specifications. For instance, in the program, we ask the function to organize the data in bins that are 100ns wide and go from 0.4 micro-seconds to 10 microseconds. This should give us 9.6/0.1 = 96 bins. Now we have an array that tells us how many of our data points fell between 400ns and 500ns, 500ns and 600ns,...., 9.9micro-seconds and 10.0 micro-seconds, you get the idea. The output from this function is Number of Data Points in a Bin and Bin number.

Example:

Data	Bin data from 0.5u to 10.5u in 5u bins
0.2u	
0.9u	3 array members in bin 1 (0.9, 1.1 and 2.3)
19.0u	
8.4u	2 array members in bin 2 (8.4 and 6.4)
2.3u	
6.4u	
1.1u	

If you were to look at the bin data by connecting a window to the output of the "Put the data into bins" box, you'd see:



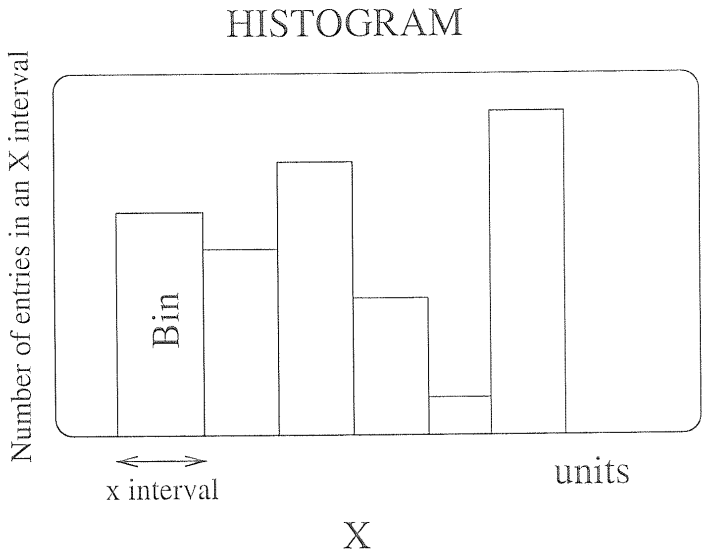
To do this, click on Display > AlphaNumeric,
Then place the window at a convenient spot, and click on the data output pin of "Put the data into bins" and connect the little wire you got to the input pin of the AlphaNumeric box and run your program.

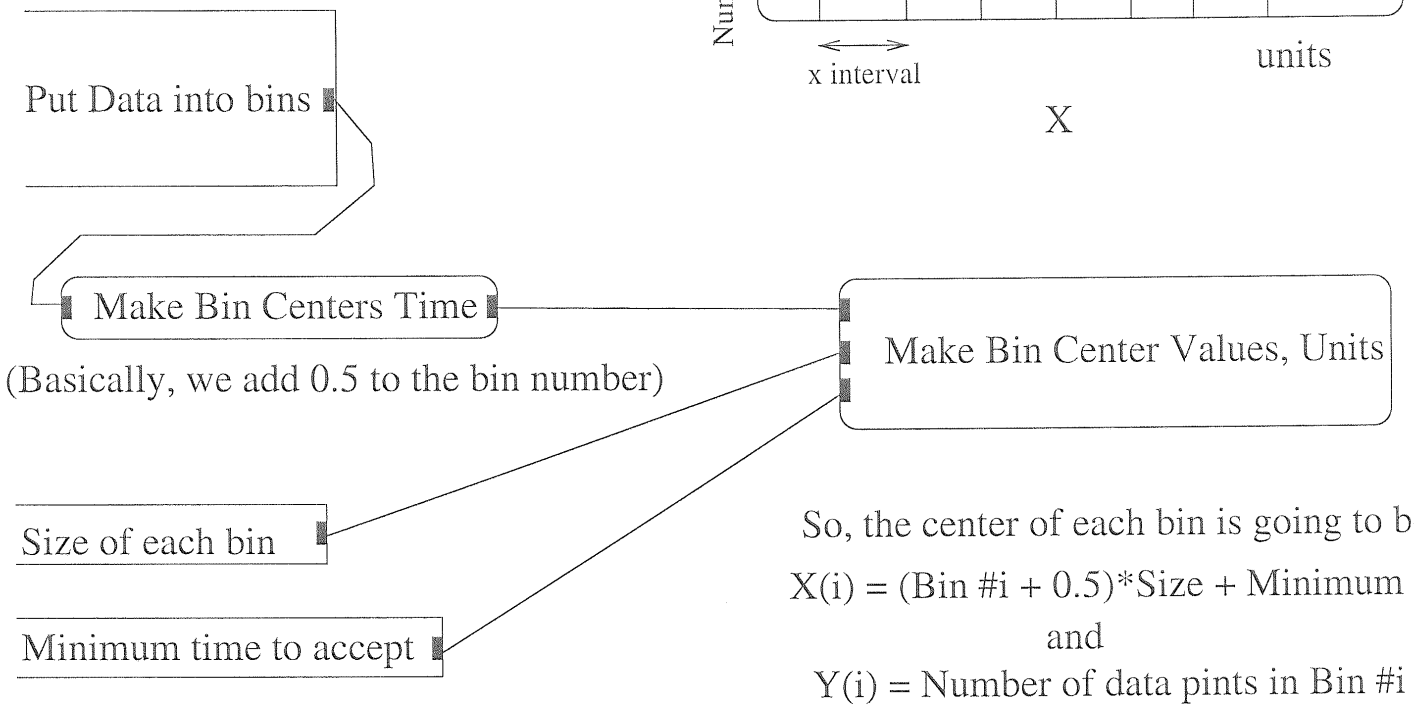
(Next Step: Convert the bin number to time.)

Step 2: Convert bin centers to time.

The next step is to create a usable x coordinate. Right now our Y coordinate is how often the data fell into a particular bin of our histogram.

To make the x coordinate, we are going to assume that the bin center is a good approximation for the average x (in our case the time) for the data in a bin.





So, the center of each bin is going to be:

$X(i) = (\text{Bin \#}i + 0.5) * \text{Size} + \text{Minimum time}$

and

$Y(i) = \text{Number of data pints in Bin \#}i$

Step 3: Calculate!

We are trying to perform the least squares fit to the data. We presume that we can fit the data to a straight line if we can just get rid of that "noise" that appears at the high end of the histogram you made when you took data.

Lifetime data Bin = Raw Data Bin - Noise in Bin and

ln(lifetime bins) vs. X(lifetime) is a straight line (we hope)

The rest of the program is forming the variables you need for the calculations described in Appendix A, and demonstrated in the text as a set of equations to solve.

Appendix C: Determining the Background from your Data

From the plot on the bottom of page 1, you see that there is a flat spot at high lifetimes where the exponential has died off and we are left with what we think is random contaminations from cosmic rays that arrive at uncorrelated times. Rather than just draw a line through these points and guess how much is reasonable for that line to vary, we can use the data when we read it into ''Muoan''. Basically, we will sum over bins of the raw data in a range that we think is flat and find the average value of the number of events in a single bin. It is important in this process to maintain the same bin width that you used in your original calculation of the lifetime.

$$\text{Average Background in a bin} = \frac{\text{Sum of all the events in the flat region of the histogram}}{\text{Number of bins you chose to be in the flat part}}$$

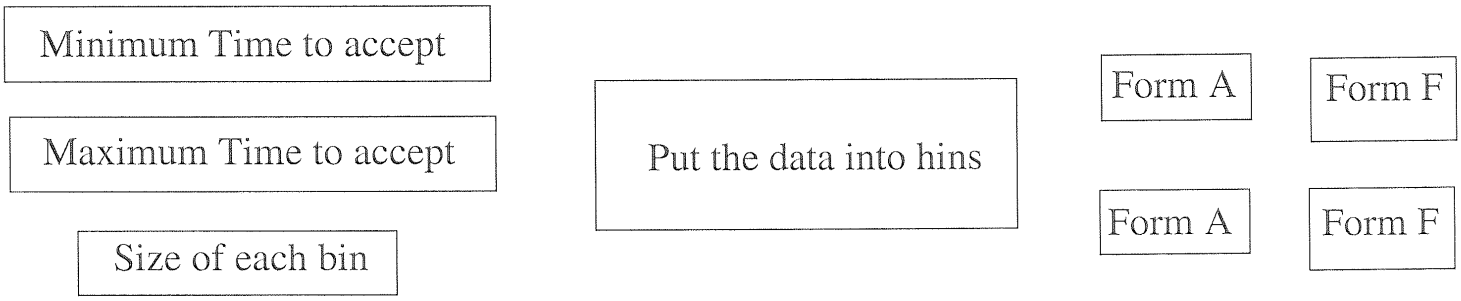
And the error on the sum is the square root of the sum, so the error on the Average Background is

$$\frac{\sqrt{\text{Sum of all the events in the flat region of the histogram}}}{\text{Number of bins you chose to be in the flat part}}$$

We just need to calculate this. You have most of the components already. You are going to construct another histogram with different limits than you used before and perform some mathematical operations on the contents.

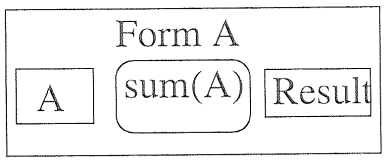
1)Save a copy of the ''muoan'' program (use Save As in the Edit menu) as muoan2

2)Modify muoan2. Copy and paste these windows into a clear area of the Main window:



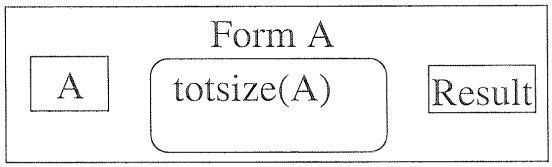
Then click on the Display menu and choose Alphanumeric. When you click again, this box will be placed in the Main window where you clicked. Get 2 of these.

Modify one of the Form A's to be (you may need to double click)



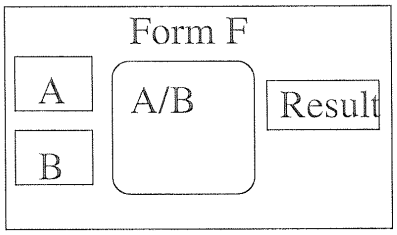
(sum of bin entries)

Modify the other as



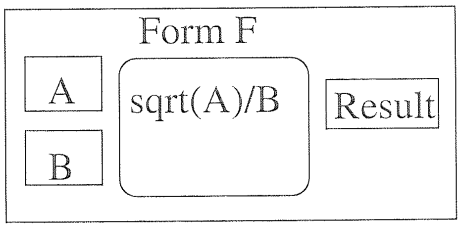
(number of bins)

Modify one Form F this way:



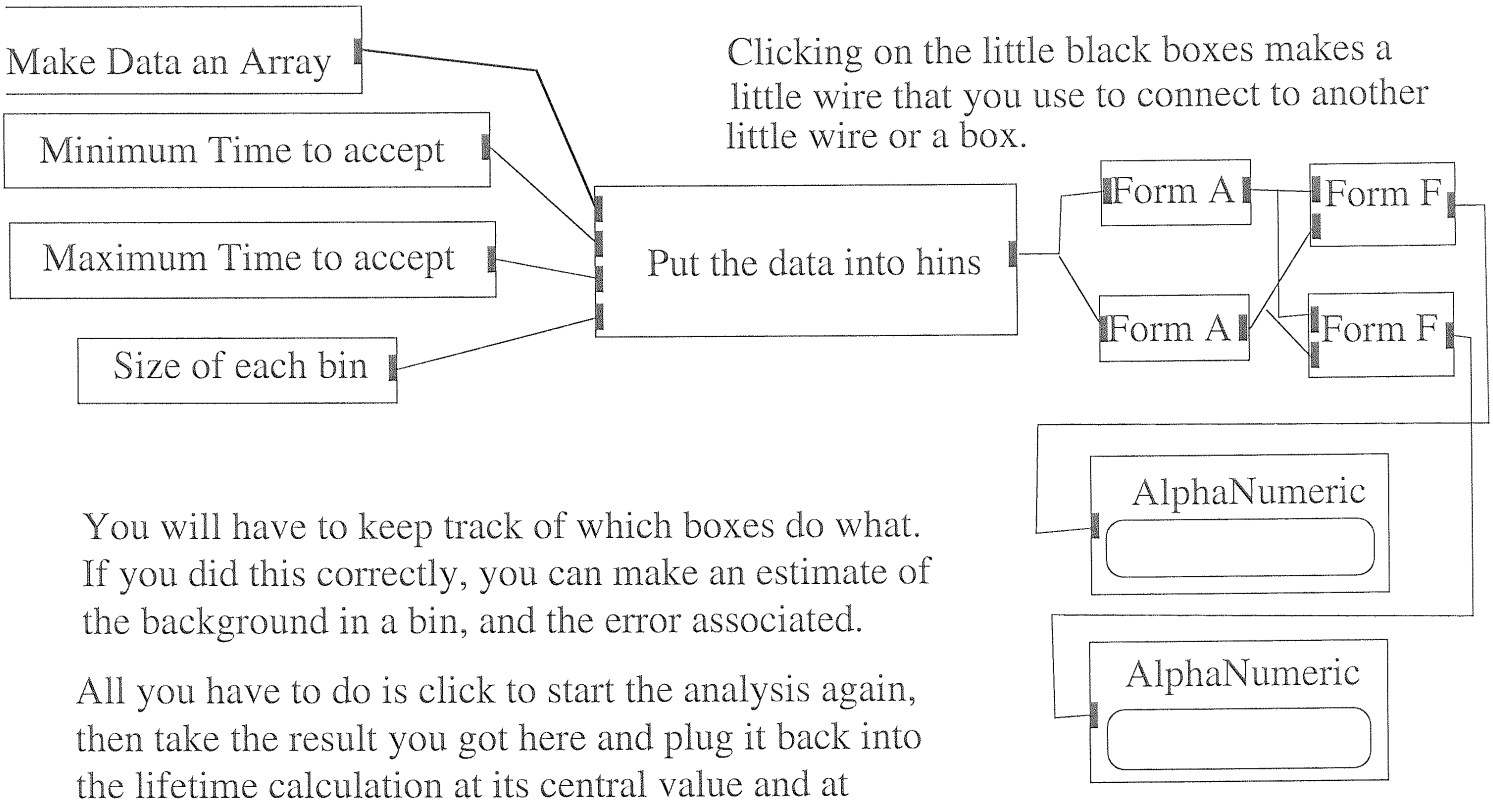
(this is the average)

And the other one this way:



(this is the error on the average)

Now hook them up (this is the fun part)



Clicking on the little black boxes makes a little wire that you use to connect to another little wire or a box.

You will have to keep track of which boxes do what. If you did this correctly, you can make an estimate of the background in a bin, and the error associated.

All you have to do is click to start the analysis again, then take the result you got here and plug it back into the lifetime calculation at its central value and at the central value +/- the error to see the effect on the lifetime.

Appendix D: The Mini Monte Carlo method

You will be creating about 10 different files, each of them having a different:
1) File name
2) Random Number Seed (the number that starts the Random Number Generator)

The first thing you need to determine is how much of your data is background. Take the total number of events you wrote on your raw data tape and the number of background events you calculated using the method of Appendix C. Divide the number of background events by the total. This is the probability that you read a background event. It will also be the probability that you generate a background event.
Open the HPVEE program ''muogen'' and double click on the little box

GenRandomEvent1

Enter the probability you calculated (typically less than 0.10) into the Real64 box that is attached to the If/Then/Else box in the center part of the GenRandomEvent1 box. (you can minimize this now)

Enter the result you got from the fit to the lifetime in the Real64 box that is attached to the A*B box at the top of the GenRandomEvent1 box. You will need to enter this as a negative number in microseconds. E.g. -2.2u

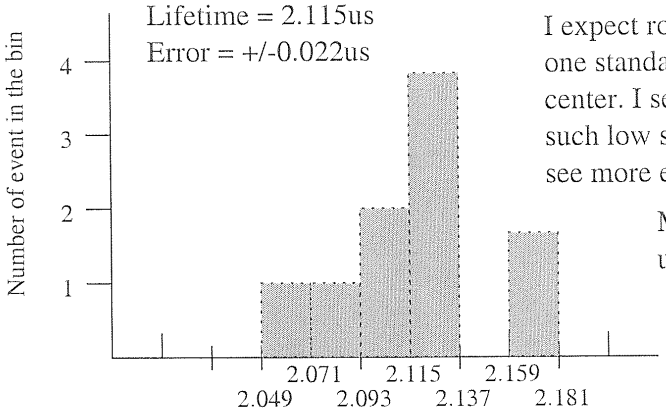
Now, modify the output file name to something like mygen1, and modify the number in ''Change this number for new fake data!'' to 1. Click the start box.

Repeat this, each time changing the file name and the number for fake data.

Analyze each file as if it were data. (You don't need to vary the background value by its error, but you do need to use a new background number for each file.)

Record your results.

Make a histogram of the results. Make your bins as wide as your lifetime error you got from the data. Does the spread of the generated data lifetimes roughly follow what you expected? Results from an event sample of 14000 events:



I expect roughly 7 events within one standard deviation of the center. I see 6. Not too bad for such low statistics, thouugh I see more events beyond 2 s.d.

More trials would be useful.