# Curve Fit version 0.7e

January 1992



WARNING.  This program is an unfinished version and has had minimal beta testing.  You may encounter minor bugs as you use this package, however there are no serious bugs and you should experience no system crashes.

Because of the unfinished nature of this program, I expect no payment. However I would appreciate a post card from users of this application. By complying with this request I'll be able to gauge the amount of interest in this program and will be encouraged to add more features to it and generally improve it.  I welcome suggestions about how I might improve the program (I already have my own list of improvements to make).  Also please inform me of any bugs you may find.  If you send me instructions on how to reproduce them, I'll do my best to get a fixed version to you.  Please send post cards (I especially like ones depicting York Minster) to:

Dr. Kevin Raner
CSIRO Division of Chemicals and Polymers,
Private Bag 10, Clayton, Victoria 3149
AUSTRALIA

E-mail to:
kevin@carbon.chem.csiro.au

*Documentation*

## Introduction.

Curve Fit is a program written for scientists, science students or anyone who wants to fit user defined functions to a set of data points. (At present it does not produce nice graphs for presentation.) I originally wrote it because I wanted to fit some specific equations to my data. The popular commercial graphics packages allow you to fit linear, polynomial and logarithmic curves to data, but most packages don't let you specify your own equation. However Curve Fit allows you to fit equations such as y = ax+b+c/x. A trivial but annoying example is fitting a straight line <u>through the origin</u>. Many popular programs fit the equation y = ax + b, however they decide what value to use for b and it isn't necessarily zero. Curve Fit lets you either constrain b to zero or simply fit the equation y = ax. There is one good commercial package that lets you define your own equations but it doesn't allow the user to directly manipulate the coefficients or constrain them to chosen values. Curve Fit is very flexible in this regard and gives you control over which coefficients will be optimized and even which mathematical algorithm will be used. Curve Fit can also be used simply to plot and analyse specific mathematical functions without necessarily fitting them to any data points.

## System Requirements for Curve Fit.

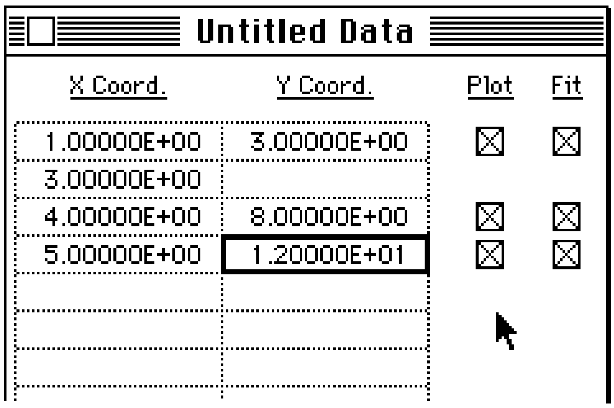
Curve Fit requires at least the 128K version of ROM which means it will run on the 512K Enhanced or higher machines. It also requires version 4.2 or higher of the System software. The program was not developed with System 7 in mind hence features such as balloon help are not available.

## An Introduction to the Program.

I'll introduce the program bit by bit. The program has three display windows: the **Data Window**, the **Function Window** and the **Plot Window**. First, I'll describe the Data Window.

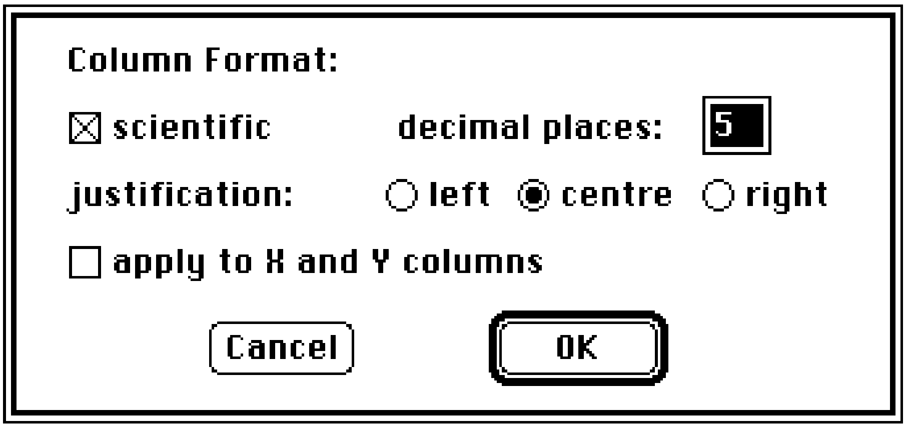
## The Data Window.

A new untitled **Data Window** may be created by selecting **New Data** from the **File** Menu. The Data Window is similar to the data windows of other programs. It is divided into two columns, labelled **X Coord.** and **Y Coord.** Up to eighteen data points may be entered into this window. (Future versions of this program may allow an unlimited number of data points.) The active cell is indicated by a thick black border and a blinking insertion point. Numbers may be typed into the cell from the keyboard and locked in by pressing the **enter** key. A new cell can be made the active cell by clicking in it. When a new cell is clicked the contents of the previous cell are automatically locked in. Adjacent cells can be selected by using the **arrow** keys (and also with the **tab**, **return** and **shift** keys).




When a row has an X and a Y value, two check boxes will appear on the right hand side of the row. The **Plot** check box indicates that this data point will be included when the plot is constructed in the Plot Window. The **Fit** check box indicates that this data point will be included when curve fitting is performed. By clearing these check boxes, data points can be omitted from the plot or can be ignored in the curve fitting procedure.

The format of the numbers in a column can be changed as follows. First indicate which column by selecting any cell in that column and then select **Column Format…** from the **Windows** Menu. The following dialog box will appear.

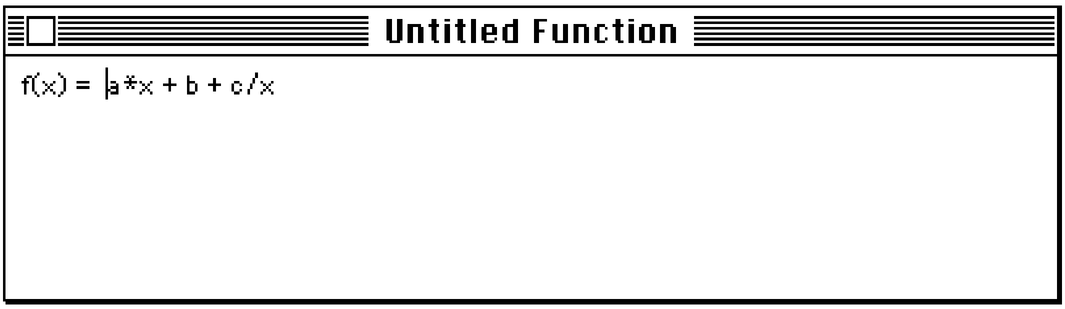

Column Format:
scientific
decimal places:
5
justification:
left
centre
right
apply to X and Y columns
Cancel
OK

The **scientific** check box controls whether the numbers will appear in ordinary or scientific format. The number of digits to the right of the decimal point can be specified in the **decimal places** field. Valid values should be in the range 0 - 5. The **justification** radio buttons allow you to nominate left, centre, or right justification of the numbers in the cells. If you want the same format to apply to both the X and the Y columns, then check the **apply to X and Y columns** option.

The Data Window can be hidden at any time by clicking in the window's go-away box. Note that this does not close the file but merely makes the window invisible. The Data Window can be brought back by selecting **Data Window** in the **Windows** Menu.

## The Function Window.

A new untitled **Function Window** may be created by selecting **New Fn** from the **Curve Fit** Menu. This displays the current function to be used in the curve fitting. The function can be typed directly from the keyboard and then locked in by pressing the **enter** key.




The function must conform to a set of syntax rules. The symbols for the allowed arithmetic operators are:

| | |
|---|---|
| + | addition |
| - | subtraction, unary minus |
| * | multiplication |
| / | division |
| ^ | exponentiation |

The recognized operands are **x**, **a**, **b**, **c**, **d**, **e**, **pi**, **π** and numeric constants. **X** is the function's argument and **a**, **b**, **c**, **d** and **e** are variable coefficients. Constants can also be used, e.g.

$$f(x) = 2.3*x + \pi$$

In the above function, **2.3** is a numeric constant. The numeric constants can also be entered in scientific format, i.e. **2.3e+0**. Pi can either be typed as '**π**' or as '**pi**'.

The function you define can also include other standard mathematical functions. The functions that are recognized are:

| | |
|---|---|
| sin() | sine |
| cos() | cosine |
| tan() | tangent |
| asin() | arcsine |
| acos() | arccosine |
| atan() | arctangent |

| | |
|---|---|
| ln() | natural logarithm (base e) |
| log() | common logarithm (base 10) |
| exp() | natural exponent (e^x) |
| | |
| sinh() | hyperbolic sine |
| cosh() | hyperbolic cosine |
| tanh() | hyperbolic tangent |
| asinh() | inverse hyperbolic sine |
| acosh() | inverse hyperbolic cosine |
| atanh() | inverse hyperbolic tangent |
| | |
| sqrt() | square root |

These functions can be used to fit data to some more complex functions, e.g.

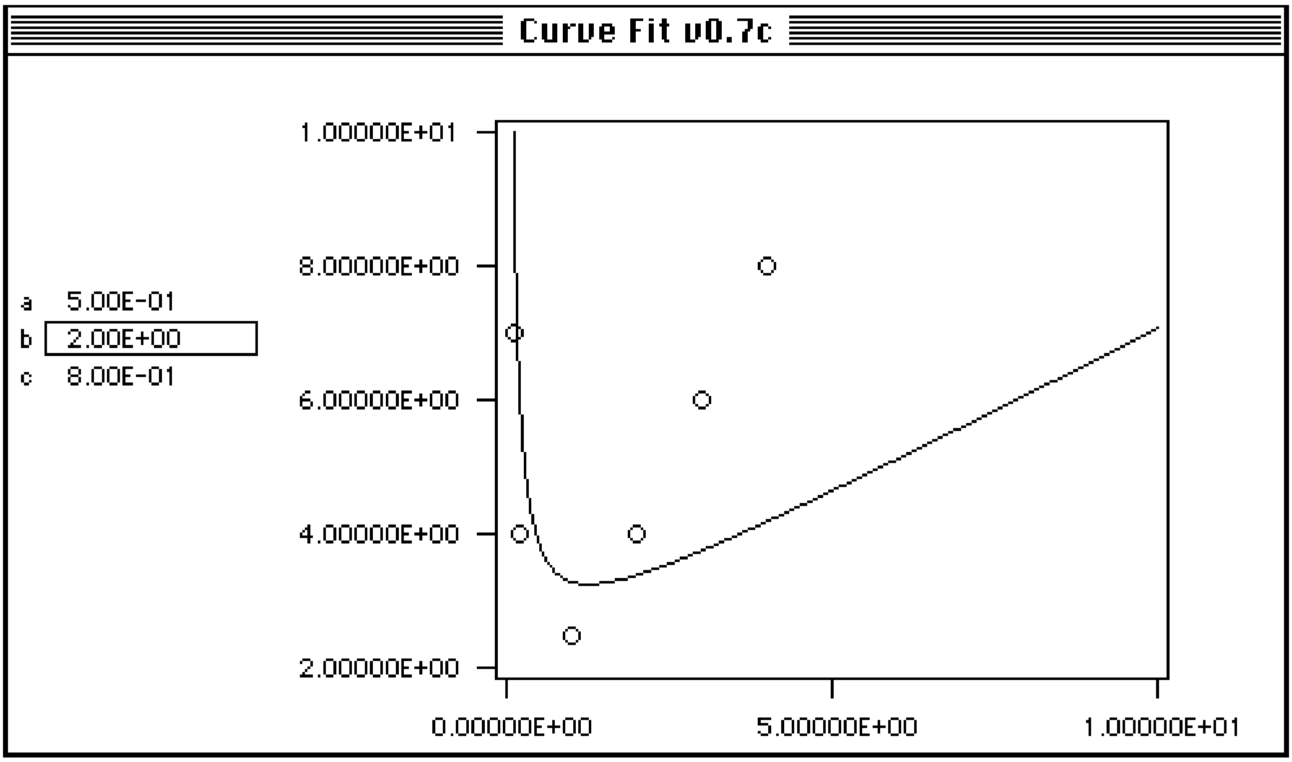
f(x) = a*x + b*x/exp(1-x*x);     a comment may be appended here

**Tip**. In the above example, $\mathbf{x^2}$ is typed as '**x*x**' rather than '**x^2**'. For reasons I won't go into, the multiplication operation is much faster than the exponentiation operation. Using 'x*x' rather than 'x^2" will significantly speed up plotting and curve fitting.

Curve Fit is not case sensitive when it reads the user defined function, hence a combination of upper and lower case characters can be used. The function can occupy more than one line of text, in which case a line is terminated by typing a return. All white space characters (i.e. tab, return, space etc.) between operands and operators are ignored. The above example also illustrates how a comment may be appended to the function by using the semicolon. All characters typed after a semicolon are ignored. After the function has been typed, it is locked in by pressing the **enter** key.

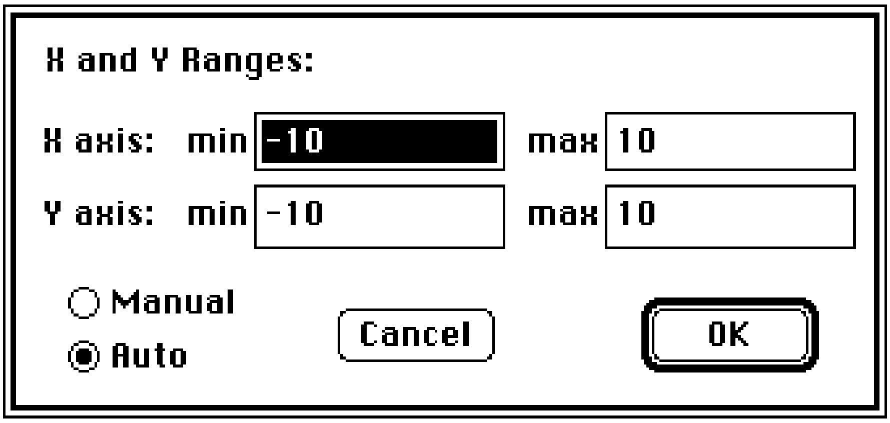
The Function Window can be hidden at any time by clicking in the window's go-away box. Note that this does not close the file but merely makes the window invisible. The Function Window can be brought back by selecting **Function Window** in the **Windows** Menu.

**The Plot Window.**



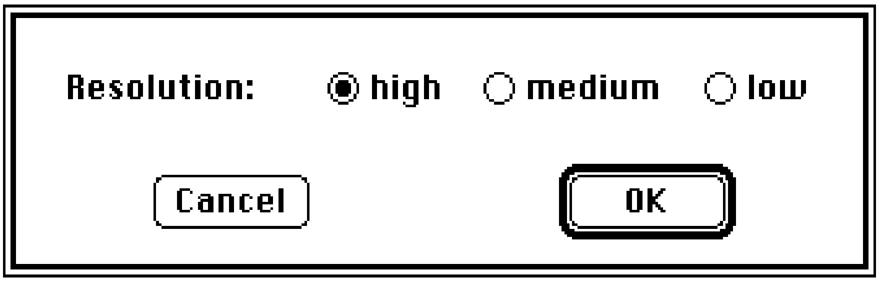
The **Plot Window** displays a plot showing the selected data points and the curve fitting function. It also displays the coefficients that are used in the current function. The plot will be updated whenever **Update Plot** is chosen from the **Plot** Menu. If the **Auto Update** option in the **Plot** Menu is checked, then the plot will be updated automatically. For example, the plot will require updating whenever a new data point or a new function is locked in. It is best to disable Auto Update when many data points need to be keyed in, as the constant replotting will slow down the data entry.
The values of the coefficients can be changed by clicking on the appropriate coefficient and typing a new value. This value can then be locked in by pressing the **enter** or **return** key. The coefficient format can also be changed by choosing **Column Format…** from the **Windows** Menu whenever the Plot Window is the front window.

The X and Y ranges of the plot can be changed by selecting **X, Y Range…** from the **Plot** Menu, the following dialog box will appear.




The X and Y ranges can be typed directly into the fields in the dialog box. Selecting the auto option will set the X and Y ranges to the smallest range that includes all of the data points (including those that are invisible in the plot).

The resolution of the plot can be altered by choosing **Resolution…** from the **Plot** Menu. The following dialog box will appear.




When the function is plotted, the Y value of the function is plotted at regular intervals along the X axis, the actual plotting is a join-the-dots exercise. If high resolution is selected, then the Y value is computed at every X pixel and the plot will be composed of 251 points. With the medium option, Y is calculated every second X pixel and the plot is constructed from 126 points. A low resolution plot calculates Y at every fifth X pixel to produce a 51 point plot. High resolution gives a superior plot, however medium or even low resolution may be used if the plotting time needs to be shortened.

The formats of the numbers along the X and Y axes are dictated by the formats in the X and Y columns of the Data Window. If necessary, the program may increase the number of decimal places.

### Editing

The contents of the active data cell, coefficient cell or the Function Window can be edited by any of the standard Macintosh **Edit** Menu commands, **Cut**, **Paste** etc. Text can be imported, via the clipboard, from other applications. All of these editing commands are reversible by choosing **Undo** from the Edit Menu. In this version of the program, editing in the Data Window is limited to the contents of a single data cell, i.e. groups of cells cannot be selected. The plot can be placed on the clipboard by choosing **Copy Graph** from the **Edit** Menu.

### Saving and Reading Data Files and Function Files.

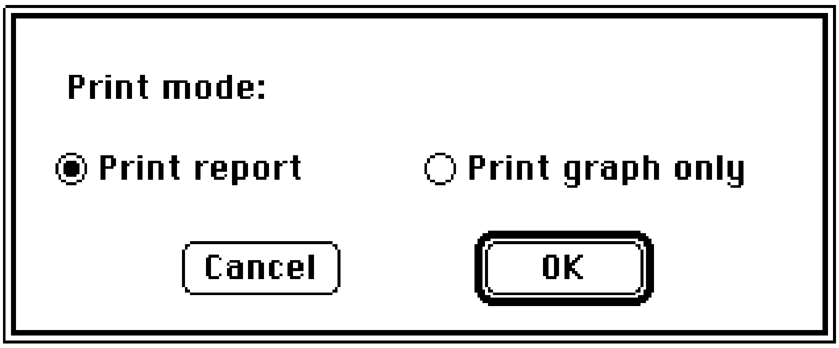
The contents of the Data Window can be saved to a disk file by selecting **Save Data…** or **Save Data As…** from the **File** Menu. These menu commands first lock in the contents of the active data cell and then call up the Macintosh standard file save dialog box. File saving is carried out in the familiar Macintosh way. In addition to the data points, the values of any coefficients are stored in the Data File along with information about the type of curve fitting employed. To open a Data File, simply select **Open Data…** from the **File** Menu. The Macintosh standard file open dialog box will appear and the Data File can be selected. After clicking the Open button the Data File will be read into memory and displayed in the Data Window. In this version of the program, only one Data File can be open at any time. Opening another Data File will result in automatic closure of the current Data Window. If the file has changed since it was opened, you will be prompted to save the file.

The function can be stored as a Function File by selecting **Save Fn…** or **Save Fn As…** from the **Curve Fit** Menu. This will invoke the standard file save dialog box. A Function File can be opened by selecting **Open Fn…** from the **Curve Fit** Menu, this will call up the standard file open dialog box. Again, only one Function File can be open at any one time and opening another Function File will automatically close the previous one.

User defined functions are linked to the Data File. When the Data File is saved the details of the current curve fit and Function File are automatically stored so that when the Data File is reopened, the Function File will also open. This saves you from having to go through two standard file dialog boxes to open the data and the function.

### Printing

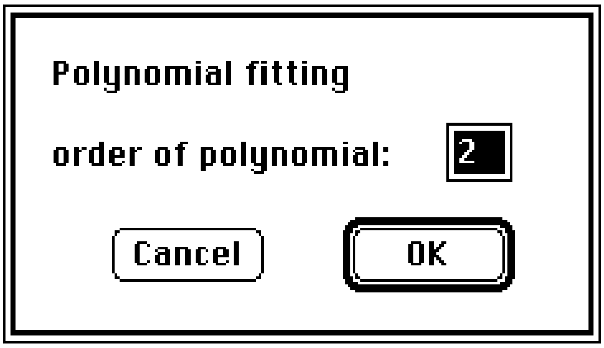
There are two types of printout produced by Curve Fit. The type can be specified by choosing **Page Setup…** from the **File** Menu, the following dialog box will appear.



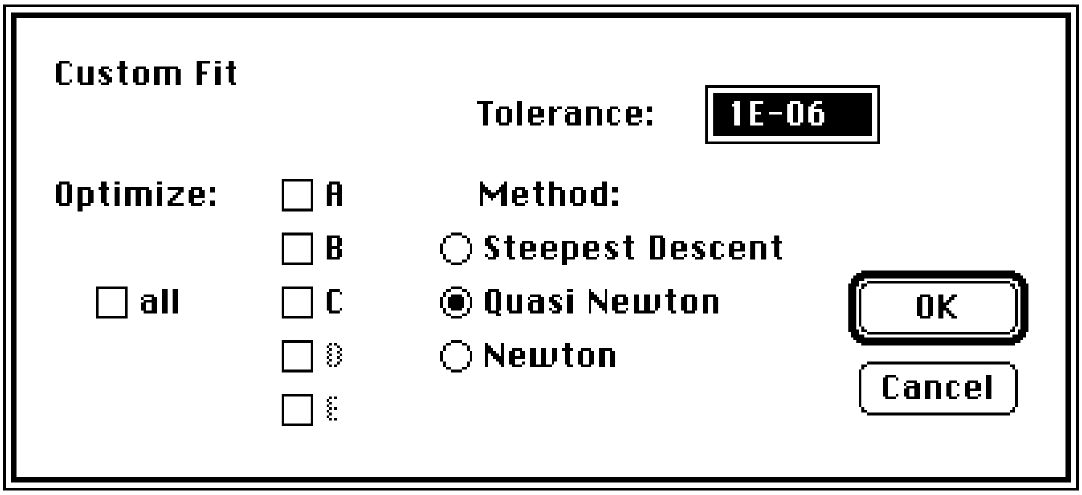

The **Print report** radio button tells the program to print the contents of the Data Window, the Function Window and the Plot Window. If **Print graph only** is selected, then the program prints the Plot Window only.  After this dialog box has been dismissed, the standard Macintosh Page Setup dialog box appears.  When **Print…** is chosen from the **File** menu the standard Macintosh Print Dialog box will appear.  Once started, the printing process can be terminated by typing command-period.

### Curve Fitting

Curve Fit version 0.7e has three modes of curve fitting: Linear, Polynomial and Custom.  By selecting **Linear** from the **Curve Fit** Menu, the least squares line of best fit is computed.  The Function Window is changed to reflect the linear function (f(x) = a*x + b) and the Plot Window is updated with a new plot and the computed values of the coefficients a and b.  When **Polynomial…** is selected from the **Curve Fit** Menu, a dialog box is displayed.  The desired order of the polynomial fit can be entered into this dialog box and then the polynomial coefficients, that correspond to the best least squares fit, are calculated.




If a user defined equation has been set up in the Function Window, then **Custom…** can be chosen from the **Curve Fit** Menu.  This allows the user to optimize any or all of the coefficients in the function to get the best least squares fit.  The custom fit dialog box has a set of check boxes which allows you to specify which coefficients will be optimized.  You can select which optimization method to use by clicking one of the radio buttons.  The tolerance field allows you to specify when the curve fitting algorithm will terminate.




To understand how to use the custom curve fitting algorithms it is necessary to explain one or two things about the mathematics.  The closeness of the curve fit is quantified by the **sum of squares.**  This is defined as

$$SS = \sum ( y_i - f(x_i, a, b, c, d, e) )^2$$

where $(x_i, y_i)$ are the set of data points, f() is the curve fitting function and a, b, c, d, e are the coefficients. If the fit was perfect then the function f() would pass through all of the data points and hence $f(x_i, a, b, c, d, e)$ would equal $y_i$ for all i and therefore SS would equal zero. Therefore the aim of curve fitting is to find values of a, b, c, d, e so that SS is zero or the minimum possible value. The **Plot** Menu allows you to display two indicators of the closeness of the fit. If **Sum of Squares** is checked then the Plot Window displays the sum of squares (SS) value computed from the current data points, function and coefficients. If Corr. Coefficient is checked then the square of the correlation coefficient ($R^2$) is displayed in the Plot Window. This is defined as

$$R^2 = 1 - \frac{\sum_i ( y_i - f(x_i, a, b, c, d, e) )^2}{\sum_i y_i^2 - (\sum y_i)^2} = 1 - \frac{SS}{\sum_i y_i^2 - (\sum y_i)^2}$$

If the curve fit is perfect then SS equals zero and hence $R^2$ is unity; as the fit worsens, $R^2$ decreases. The curve fitting algorithms used by Curve Fit adjust the coefficients such that the sum of squares decreases (and hence $R^2$ approaches unity).

### Steepest Descent Curve Fitting

All of the curve fitting algorithms rely on the user to supply a starting set of coefficient values to be optimized. The success of the curve fitting depends on how close the starting set of values are to the optimal values. The Steepest Descent method is a simple method and can be quite slow when many coefficients are being optimized. However this method is a good starting point since it can perform well even when the starting set of coefficients are a long way of the optimal values.

The Steepest Descent method can be visualised as follows. Consider a contour map of a collection of mountains and valleys. Any point on the map can be characterized as having two position coordinates, longitude and latitude, and a third value indicating height. If we want to get to the lowest point on the map we need to walk down the mountains and into the valleys. The contour map indicates which way to go. This corresponds to a situation where we are optimizing two coefficients to fit a function to a set of data points. The two coefficients, a and b, are coordinates on a contour map which indicates the sum of squares fit (SS) at that point. To get the best fit we want to minimize the sum of squares, i.e., find the lowest point on the map. The Steepest Descent method is a two step reiterative process. The first step is to take a small step in the ‘a’ direction to test the steepness in this direction and then take a small step in the ‘b’ direction. This enables the calculation of which direction provides the steepest descent. The second step is conduct a line search, i.e., to proceed along this direction until the lowest point is found. Since this new point is lower than the starting point, the sum of squares has been decreased and a better fit has been found. This new point is then used as a starting point for the next iteration.

### Newton Method

The Newton method is a rapidly converging algorithm and is the method of choice for systems that are already close to the best fit. Unfortunately, this method does not perform at all well when you are a long way from the best fit, under these conditions the method may not converge at all.

The Steepest Descent method doesn’t converge rapidly because it is essentially short sighted. It works out the best direction to go based on the gradient at the current point. The steepest direction found in this manner doesn’t necessarily continue to be steep as you move along that way. Furthermore, the method has to find out the step length by trial and error. However, the Newton method calculates not only the gradient vector (steepness) but also the Hessian matrix (curvature) at the current point. By assuming that the contour surface is quadratic, it can calculate not only which direction to go but how far it should go. Unfortunately, the basic assumption breaks down when you are a long way from the minimum sum of squares and the algorithm doesn't converge. One further point to mention is that the algorithm doesn't actually look for a minimum in the sum of squares. It proceeds in a direction towards a point where the gradient vector is zero, i.e., it may converge to a maximum sum of squares thus giving a worse fit.

### Quasi-Newton Method

The Quasi-Newton method used by this program is the Davidon-Fletcher-Powell algorithm. It is a compromise between the Steepest Descent method and the Newton method. It has the stability of the former and the rapid convergence of the latter method, for this reason it is the default method in the custom curve fit dialog box.

The method involves two steps with the first being the calculation of a good direction to go. This direction is not necessarily the direction of Steepest Descent. The second step is to conduct a line search and hence find the lowest point along this direction.

For a good introduction to Nonlinear Optimization, read "Practical Methods of Optimization. Volume 1, Unconstrained Optimization" by R. Fletcher (Wiley) 1980-81.

### Using Curve Fit to Fit a User Defined Function to a Set of Data Points

*Example 1*

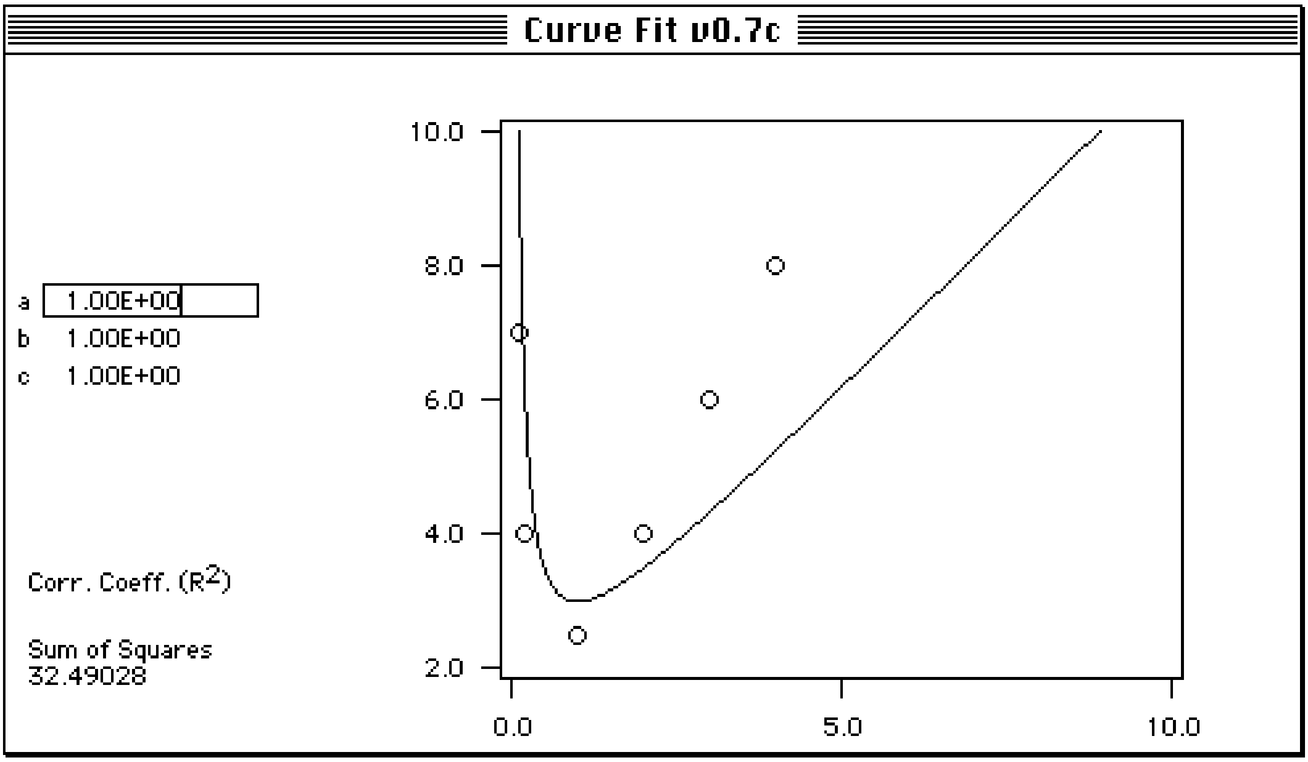
Let's fit the function

f(x) = a*x + b + c/x

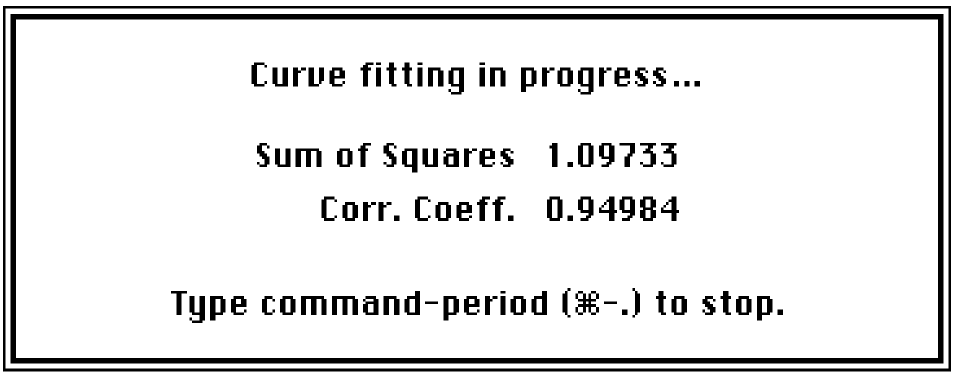
to the following set of points

| | |
|---|---|
| 0.1, | 7.0 |
| 0.2, | 4.0 |
| 1.0, | 2.5 |
| 2.0, | 4.0 |
| 3.0, | 6.0 |
| 4.0, | 8.0 |

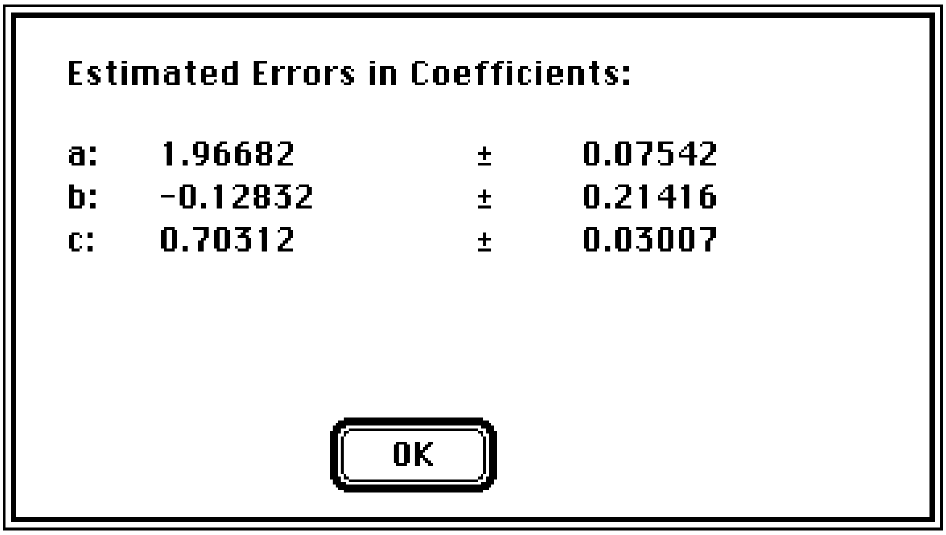
First type the data points into a new Data Window and set the format of the X and Y columns to non-scientific format with one decimal place. Then enter "a*x + b +c/x" into a new Function Window. Select the Plot Window and enter "1" into all the coefficient fields. Check the **Corr. Coeff.** and **Sum of Squares** items under the **Curve Fit** Menu. With a=b=c=1, the computed sum of squares will be 32.49028 and value of $R^2$ will not be displayed as the fit is very poor. Change the plot ranges to min X = 0, max X = 10, min Y = 2 and max Y = 10. The plot will look as shown below.



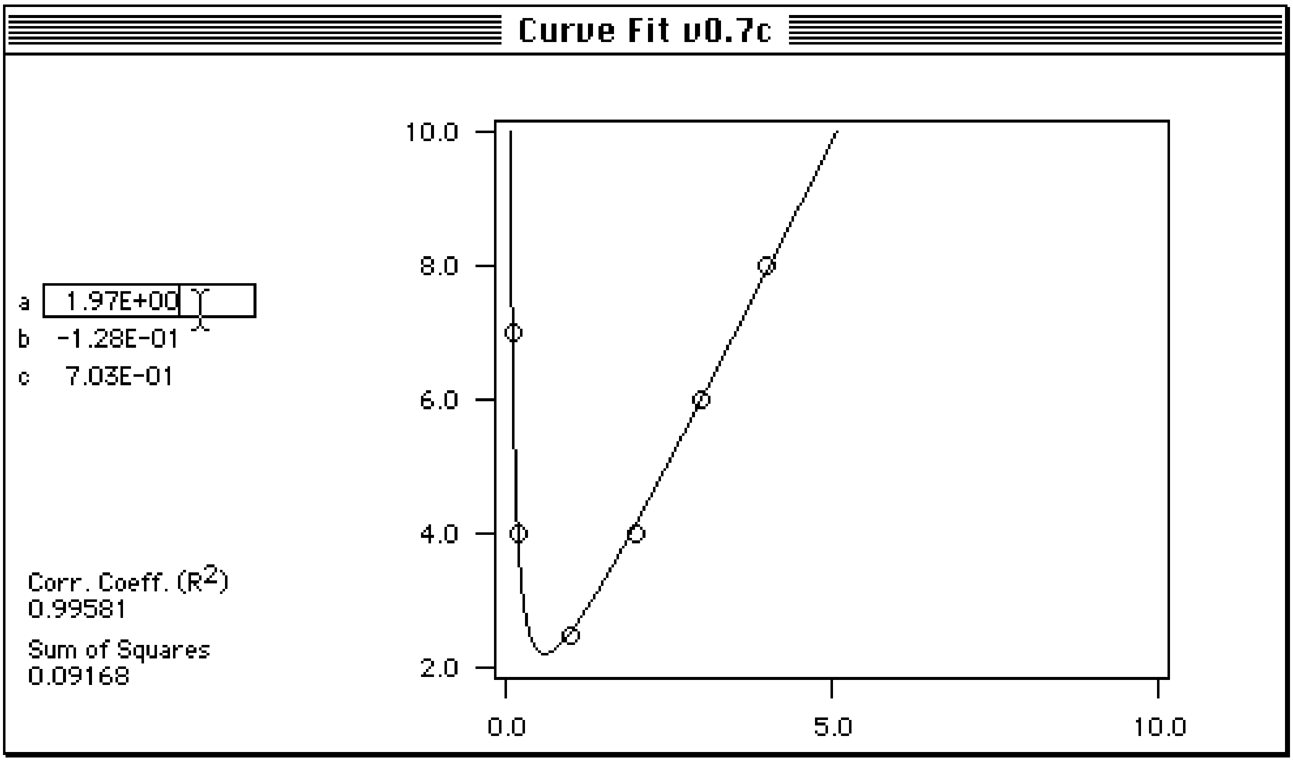

Then select **Custom…** from the **Curve Fit** Menu and check the **all** box under the heading **coefficients:**. Click the **OK** button and this will start the curve fitting using the Quasi Newton method with a tolerance of 1.0 E-06. The reiterative method will continue until the decreases in the sum of squares is less than the tolerance of 1.0 E-06. For accurate curve fitting use a small tolerance and for rough curve fitting use a larger value. As the curve fitting proceeds, a dialog box shows the current progress by displaying the sum of squares and the square of the correlation coefficient. The curve fitting can be terminated prematurely by typing command-period.




When the curve fitting process finishes, a dialog box will be displayed. This shows the best values of the coefficients along with an estimate of the uncertainties. If one or more coefficients were left constant through the fitting process, then the uncertainty is displayed as zero. After dismissing this dialog, it can be reviewed later by selecting **Coeff. Errors…** from the **Plot** menu. However if a coefficient has been altered manually since the last automated fitting process then this menu item will be dimmed and the uncertainties may not be viewed.




When above dialog box has been dismissed, the coefficients will be updated and the plot will be as shown below. The coefficient fields now show the values corresponding to the best fit and the closeness of the fit is indicated by the Corr. Coeff. and the Sum of Squares.




*Example 2*

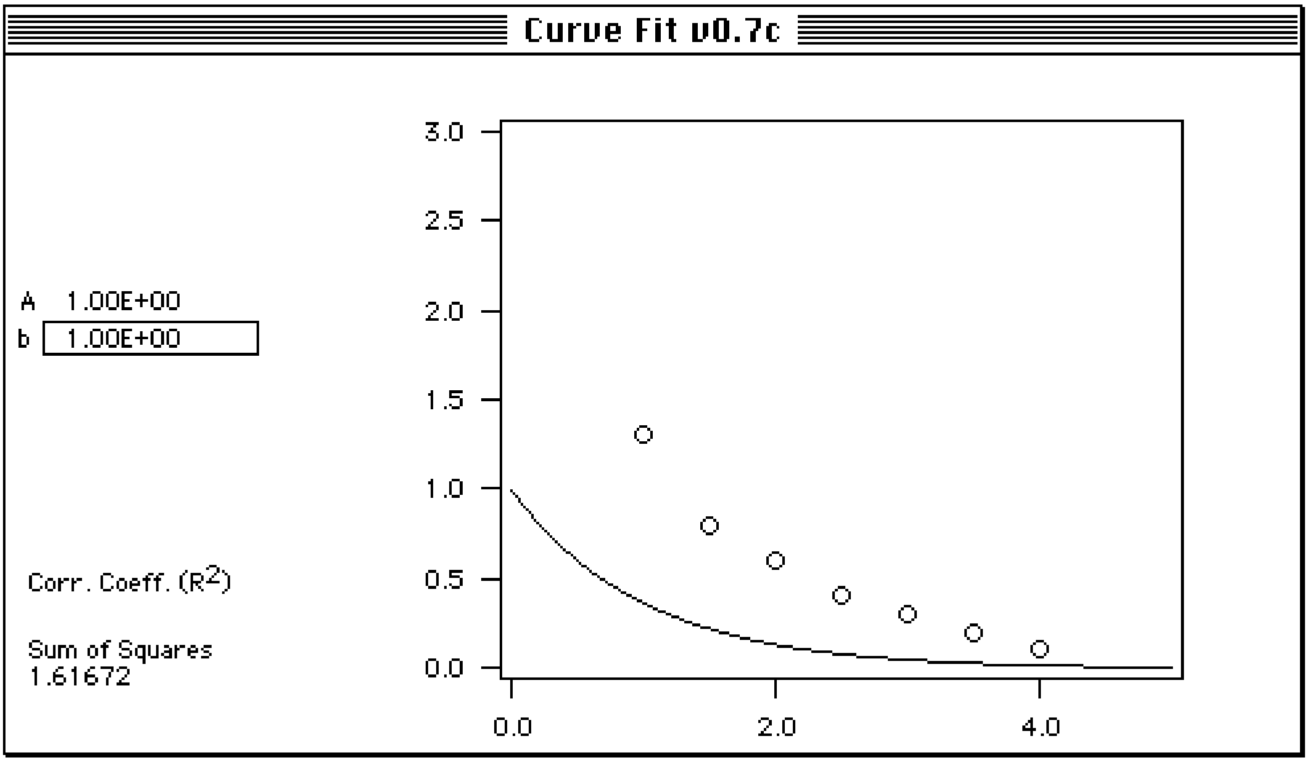
Fitting an exponential decay, i.e.,

f(x) = A*exp(-b*x)

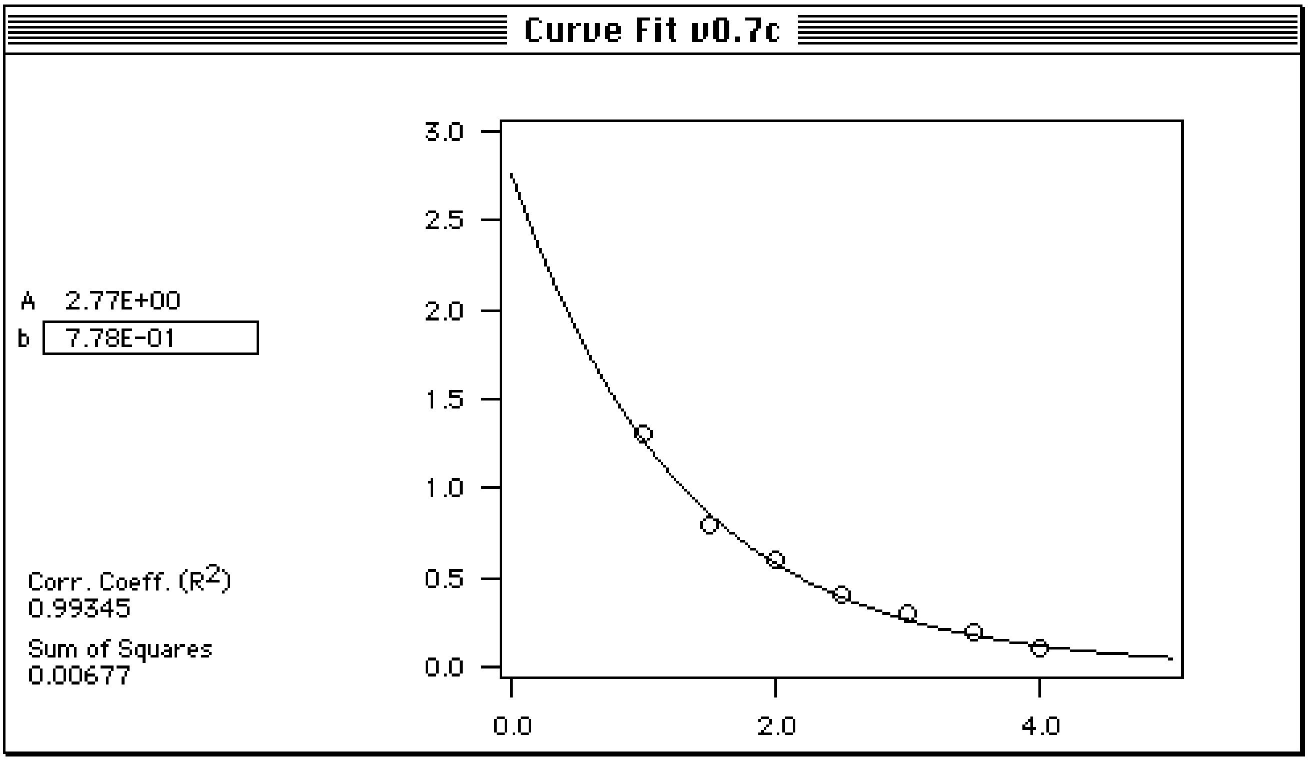
to the following set of data points:

| | |
|---|---|
| 1.0, | 1.3 |
| 1.5, | 0.8 |
| 2.0, | 0.6 |
| 2.5, | 0.4 |
| 3.0, | 0.3 |
| 3.5, | 0.2 |
| 4.0, | 0.1 |

Enter the above function and data into the program and then set the coefficients A and b to both equal “1”. Change the X and Y plot ranges to min X = 0, max X = 5, min Y = 0 and max Y = 3. The updated plot window will look like this.




In Example 1, the function was well behaved and the curve fitting would have still been successful if the Newton method was used. However the function in this example, is less well behaved and the Newton method will not converge from this starting point. By trying all three methods on this example you will find that the Steepest Descent method is slow to converge but a successful method. The Quasi Newton algorithm converges rapidly to the optimum values of A and b, shown below, and the Newton method does not converge at all. However the Newton method will converge rapidly if better starting values are chosen, such as A = 3 and b = 0.8.




In general use the Quasi Newton or Newton methods, however if they will not converge then use the Steepest Descent method to get a rough fit. If the starting values of the coefficients are a long way from the optimal values then you may need to fit the coefficients one at a time or even manually. However once a rough solution is found the Quasi Newton and Newton algorithms will converge quickly. Bear in mind that these algorithms find the closest local minimum for the sum of squares, hence the requirement for a rough solution so that the methods converge to the global minimum.

### A Final Note

I am currently writing a new version of this program. In the process I'm getting acquainted with Symantec's Think Class Library. The new version will allow up to 1000 columns of data and 32,767 rows. Multiple files may be open at once and plots may include more than one set of data points. I'm also going to produce nicer looking plots and in general the program will be greatly improved. This will take time to get together but, by mid 1992, I expect to have a much snazzier program. If you register you will be notified when the super-dooper version is released.