

---

# 13

## Adding Error Bars and Curve Fits

---

This chapter gives you basic information on how to apply and define Error Bars and Curve Fits. It also explains how to define your own curve fit function.

This chapter covers the following:

- Adding error bars to a chart
- Understanding the types of curves
- Applying a curve fit
- Assigning curve specifications to the curve fit
- Applying a Spline curve
- Creating user-defined curve fits

### Adding Error Bars to a Chart

Error bars allow you to graphically illustrate actual errors, the statistical probability of errors, or a general approximation or “spread” in your data. Examples might include experimental errors in measurement or atypical data points in comparison to the rest of the data.

You can add error bars to the following 2-D charts: Bar, Column, Bar and Column Segmentation, Floating Bar and Column, Line, Step, XY Column, Scatter, Paired Scatter, XY Line, and Paired XY Line. You can choose error bar options for *all* series or for an *individual* series with different parameters for the upper and lower error bars.

Error bars appear as two short dashes for the upper and lower values of the spread with a line connecting the two dashes at their centers. These three parts of the error bar are treated as one chart element in the chart, so you can, for example, make an all-red error bar, but you cannot make red dashes with a blue connecting line.

**NOTE** 

If you need more graphic formatting control, and the underlying data are best represented as a line chart, use the “Formula Builder” command on the Data menu, or do the calculations outside of DeltaGraph and make a Range chart with high and low components instead of a Line chart with error bars.

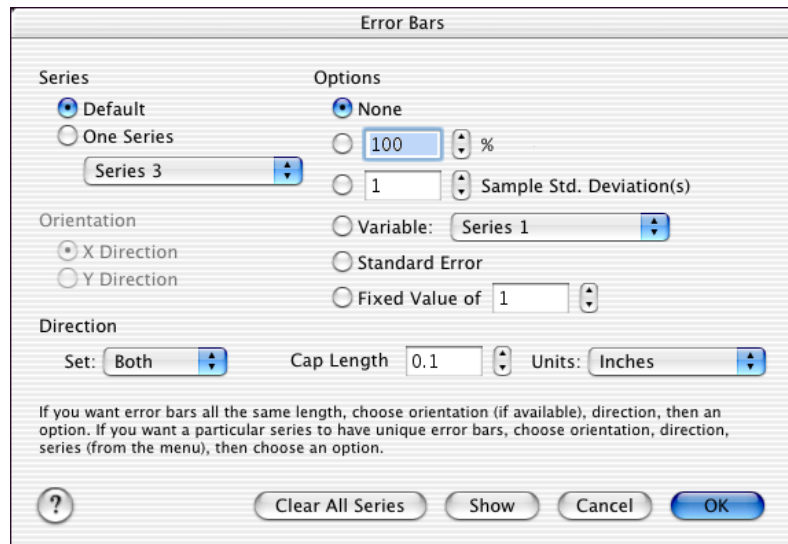
To add error bars to a chart, do the following:

1. Select the chart to which you want to add error bars.

Error bars can be added to Bar, Column, Bar and Column Segmentation, Floating Bar and Column, Line, Step, XY Column, Scatter, Paired Scatter, XY Line, and Paired XY Line.



2. Click the Error Bars icon on the extended Command bar, or choose **Error Bars** from the Chart menu. A variation of the “Error Bars” dialog box appears.



3. Make your selections and enter data as needed. The ends of each error bar are determined by making selections or typing values in the “Error Bar” dialog box. You have the following options:

**Series**

Sets error bars for all or one data series. “Default” affects all data series. To set error bars for an individual data series, select the series name from the pop-up menu below “One Series.” This automatically selects the “One Series” option.

Once you have specified different parameters for a particular series, changes you make while you’re in the default mode will not affect it. Changes made in the “Options” and “Direction” sections affect only the series that are selected. Help for the “Series” section appears below the “Options” selections.

**Orientation**

(Scatter, XY Column, and XY Line charts only) When using error bars with a Scatter chart, each X component and Y component can independently have error bars of any variety, including “None.” Switch back and forth between components by clicking the buttons under “Orientation.” When analyzing the data to determine where the error bars are placed, the above explanations are valid as long as “data value” is replaced with “X component of the data value” or “Y component of the data value,” as appropriate.

**Direction**

Sets the orientation of the error bars (Both, Plus, or Minus). You can have different error bar parameters for the Plus or Minus bar. For example you might use the standard error for the Plus but use a percentage for the Minus. You can also use the direction settings in combination with individual series settings.

**Options group box**

Lists the types of error bars you can add to your chart as follows:

**None**

No error bars are drawn.

**Standard Error**

The placement of each end of the error bar corresponds to the standard error of the data point. The standard error is computed as follows:

$$\text{StandardError} = \sqrt{\text{sumSquare} \div ((\text{count} - 1)(\text{count}))}$$

where:

sumSquare = sum of the squares of the data values minus the  
data series average

count = number of valid data values plotted

**%**

Each end of the error bar is placed at the specified percentage times the value of the data point. The percentage number is based on the scale of the axis. You can type in any value greater than or equal to zero or use the direction arrow to scroll the options.

**Fixed Value**

Each end of the error bar is placed a distance from the data value equal to the number entered. That number is based on the scale of the axis. You can type in any value greater than or equal to zero or use the direction arrow to scroll the options.

**Sample Standard Deviation(s)**

Each end of the error bar is placed at a distance  $N \cdot SD$  in user scale units from the computed value.  $M \cdot N$  can be any number greater than or equal to zero. You can type in the appropriate value for  $N$  or use the direction arrows to scroll the options.  $SD$  is the sample standard deviation computed as follows:

$$SD = \sqrt{\text{sumSquare} \div (\text{count} - 1)}$$

where: sumSquare and count are defined as above and

$$M = \frac{\text{sum of all valid data values for all series selected}}{\text{divided by count (i.e., the arithmetic mean)}}$$

This computation places the mean of all the series at the center of each error bar, rather than placing the data values at the center of each error bar. So the position of the ends of each error bar measured along the Value axis is the same for each category (such as a Line chart or Step chart) or each data value (such as a Scatter chart). That is, all the error bars “line up” instead of being “staggered” with the data.

**Variable**

Each end of the error bar is placed at a certain distance from the data value, the distance being different for each point of the series. The variable errors must be placed in a column of the Data page where the data to be plotted reside. Choose the name of the column from the “Variable” pop-up menu. For more information on creating variable error bars, see “Creating Variable Error Bars” on page 13-5.

**Cap Length**

Sets the length of the end caps (whiskers) on the error bars. The default is set to 0.1 inches, but can also be set to picas, points, centimeters, or cicerros. This will affect all error bars in the selected series.

**Clear One Series/Clear All Series**

Changes the settings of the selected series back to the default setting. Select the series, then click the **Clear** button.

4. Click **Show** to preview your changes without exiting the dialog box. This makes it easy to experiment with different effects. You can move the dialog box out of the way by dragging the Title bar.
5. When you have the results you want, click **OK** to implement the changes and return to the Chart view. To exit the dialog box without changing the chart, click **Cancel**.

## Creating Variable Error Bars

Variable error bars are used when you want an individual error for each point on a series. If you have your own calculated errors, imported data, or data calculated by the Formula Builder, create variable error bars.

To create variable error bars, do the following:

1. Input both the data and variable error data in a Data page.

	Label	A	B	C	D
Label		Series 1	Series 2	Error 1	Error 2
1		10	5	0.5	0.48
2		20	10	0.65	0.15
3		30	15	0.418	0.7
4		40	20	0.13	5
5		50	25	0.32	0.156

2. Select and plot only the data for the chart, not the variable error data.
3. Click the chart you plotted.
4. Click the Error Bars icon on the Command bar, or choose **Error Bars** from the Chart menu. The “Error Bars” dialog box appears.

5. Click the pop-up menu in the “Series” section of the dialog box, and select the first data series (Series 1 in the example).
6. Click the “Variable” pop-up menu in the “Options” section of the dialog box, and select the column from the Data page that contains the corresponding error (Error 1 in the example).
7. Repeat steps 5 and 6 until all of the series are matched up.
8. Click **OK**. The chart appears with error bars drawn in the chart.

## Curve Fitting



Here we shall look at the general theory of curve fitting and at power, exponential, and logarithmic curves. For more details on these curve fits and for an explanation of the more complex theory behind polynomial curves, see the excellent treatment of numerical methods in *Numerical Recipes in C* by Cambridge Press, upon which some of the theory in this section is based. The brief tour given here is not intended to be an exhaustive explanation of curve fitting theory.

There are four basic types of curve fits, as follows:

- Power curves
- Exponential curves
- Logarithmic curves
- Polynomial curves

---

**NOTE** 

Spline fits are not true curve fits but are an illustrative tool.

---

## General Theory

Before we delve into the mathematics of curve fitting, we should define what curve fitting is. It is a mathematical process by which a curve is determined that minimizes the squares of the distance from the curve to the data points. Thus, it is the attempt to find an equation that represents the data being fit. The better the fit,

the smaller the distances are from the curve to the actual data points. However, unless the data are known to fit a particular equation exactly, the curve will not pass directly through all the data points.

Consider the following case, where we wish to determine the best line through a set of data. The equation used to model the line is of the following form:

$$y = f(x) = a + bx$$

To measure how well the linear model agrees with the data, we use the chi-square merit function, which in this case is as follows:

$$\chi^2(a,b) = \sum_{i=1}^n \left( \frac{y_i - y(x_i, a)}{\sigma_i} \right)^2$$

Substituting our linear equation, we get the following:

$$\chi^2(a,b) = \sum_{i=1}^n \left( \frac{y_i - a - bx_i}{\sigma_i} \right)^2$$

The equation is minimized to determine  $a$  and  $b$ . At its minimum, derivatives of  $\chi^2(a,b)$  with respect to  $a$  and  $b$  vanish.

$$0 = \frac{\partial \chi^2}{\partial a} = -2 \sum_{i=1}^n \frac{y_i - a - bx_i}{\sigma_i^2}$$

$$0 = \frac{\partial \chi^2}{\partial x} = -2 \sum_{i=1}^n \frac{x_i(y_i - a - bx_i)}{\sigma_i^2}$$

These conditions can be rewritten in a convenient form if we define the following sums:

$$S = \sum_{i=1}^n \frac{1}{\sigma_i^2}, S_x = \sum_{i=1}^n \frac{x_i}{\sigma_i^2}, S_y = \sum_{i=1}^n \frac{y_i}{\sigma_i^2}, S_{xx} = \sum_{i=1}^n \frac{x_i^2}{\sigma_i^2}, S_{xy} = \sum_{i=1}^n \frac{x_i y_i}{\sigma_i^2}$$

With these definitions we have the following:

$$aS + bS_x = S_y$$

$$aS_x + bS_{xx} = S_{xy}$$

The solution of these two equations in two unknowns is calculated as follows:

$$\Delta = SS_{xx} - (S_x)^2$$

$$a = \frac{SS_{xx}S_y - S_xS_{xy}}{\Delta}$$

$$b = \frac{SS_{xy} - S_xS_y}{\Delta}$$

These are the general forms of solutions for DeltaGraph's curve fitting. The difference between the linear, power, exponential and logarithmic curve fits is confined mainly to linearizing the data prior to using the above equations to determine  $a$  and  $b$ . We will consider each of the remaining cases individually.

## Power Curve

For a power curve fit, we use the following assumption about the model:

$$y = f(x) = ax^b$$

Using the following algebra, we can linearize the equation:

$$\begin{aligned} \ln(y) &= \ln(ax^b) \\ &= \ln(a) + \ln(x^b) \\ &= \ln(a) + b\ln(x) \\ &= A + bx' \\ &= g(x) = A + bx' \end{aligned}$$

where:  $A = \ln(a)$  or  $e^A = a$  and  $\ln(x) = x'$

This linearized equation can use the above solution by assuming the error is of the following form:

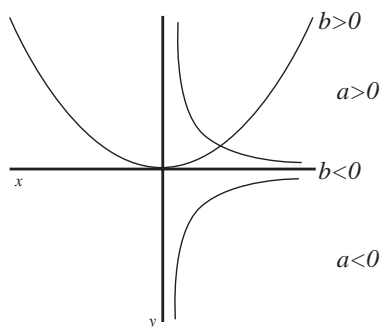
$$\chi^2(a,b) = \sum_{i=1}^n \left( \frac{\ln(y_i) - g(\ln(x_i))}{\sigma_i} \right)^2 = \sum_{i=1}^n \left( \frac{\ln(y_i) - A - B \ln(x_i)}{\sigma_i} \right)^2$$

As a result, we can use the  $\ln(y)$  and the  $\ln(x)$  to calculate  $A$  and  $B$  for the power equation. Using the natural logarithms in effect linearizes the data, so the same theory can be used to determine the line. The results are then used in the original equation to show the best fit.

When using a power curve fit, you must know the range over which the model is valid. As seen in the next figure, when the power ( $b$ ) is greater than 0 (zero) the curve is shaped like a “U” centered around  $x = 0$ .

Changing the sign of  $a$  flips the “U” upside down. This means that, for data sets which appear like a portion of the  $b > 0$  family of curves, there is an  $a$  and  $b$  which is valid.

Conversely, when  $b < 0$  the curve appears like the lower left half of an “O”. As an example, consider  $b = -1$ , which is  $y = f(x) = a/x$ . As  $x$  tends toward 0, the value of  $y$  goes to  $\infty$  and there is no curve on the  $x < 0$  side of the graph. Hence, trying to fit data sets with a downward sloping trend for  $x < 0$  will often result in no solution.



## Exponential Curve

For an exponential curve fit, we use the following assumption about the model:

$$y = f(x) = e^{(a+bx)} = ae^{bx}$$

Using the following algebra, we can linearize the equation:

$$\begin{aligned}\ln(y) &= \ln(ae^{bx}) \\ &= \ln(a) + \ln(e^{bx}) \\ &= \ln(a) + bx \\ &= A + bx \\ &= g(x) = A + bx\end{aligned}$$

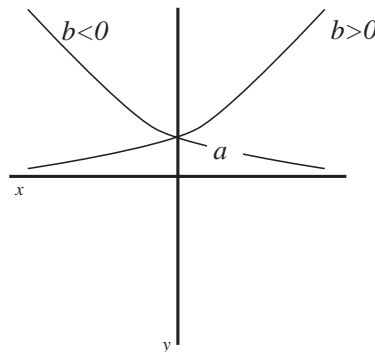
where:  $A = \ln(a)$  or  $e^A = a$

then:

$$\chi^2(a,b) = \sum_{i=1}^n \left( \frac{\ln(y_i) - g(x_i)}{\sigma_i} \right)^2 = \sum_{i=1}^n \left( \frac{\ln(y_i) - A - bx_i}{\sigma_i} \right)^2$$

$$\ln(y) = g(x) = A + Bx$$

As seen in the next figure, the trend for an exponential curve is toward 0 (zero) when  $x < 0$  and toward  $\infty$  when  $x > 0$ . Changing the sign of  $a$  and  $b$  flips the curve around both the X and Y axes. That means an exponential curve fit is defined for any set of data. However, this does not say how well this form will fit the data.



## Logarithmic Curve

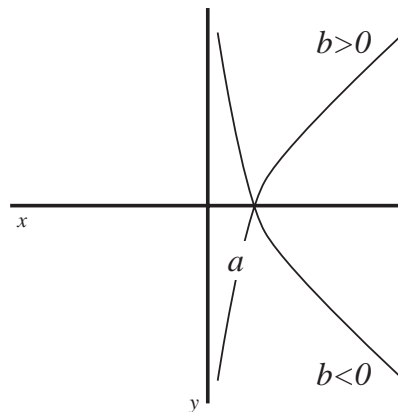
For a logarithmic curve fit, we use the following assumption about the model:

$$y = f(x) = a + b \ln(x)$$

Using a similar argument as for the power curve fit, we get the following form of the error:

$$\chi^2(a,b) = \sum_{i=1}^n \left( \frac{y_i - (a + b \ln(x_i))}{\sigma_i} \right)^2 = \sum_{i=1}^n \left( \frac{y_i - a - b \ln(x_i)}{\sigma_i} \right)^2$$

The form of the logarithmic curve fit is shown below. As seen in the figure, the curve tends toward  $\pm \infty$  as  $x$  approaches 0 (zero) and is undefined for  $x < 0$ .



Let us consider another case of interest in which a curve fit is known to pass through 0 (zero). Some processes, by definition, must be 0 at the start. By slightly modifying the curve fit we can obtain a curve which passes through 0. Consider the following linear case:

$$y = f(x) = a + bx$$

We can force this curve to pass through 0 by specifying  $a = 0$  regardless of the data being fit.

$$y = f(x) = 0 + bx$$

Using the same derivative of the merit function as above, we have the following:

$$0 = \frac{\partial \chi^2}{\partial a_1} = -2 \sum_{i=1}^n \frac{x_i(y_i - bx_i)}{\sigma_i^2} = -2 \sum_{i=1}^n \frac{x_i y_i - bx_i^2}{\sigma_i^2}$$

Solving for  $b$  we get the following:

$$b = \frac{S_{xy}}{S_{xx}} = \frac{\sum_{i=1}^n x_i y_i}{\sum_{i=1}^n x_i^2}$$

In conclusion, similar solutions may be obtained for other types of curves.

## Using Curve Fits

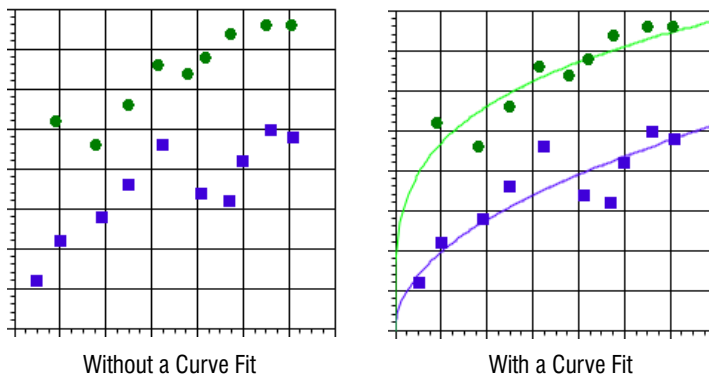


You can click the Curve Fits icon on the extended Command bar, or choose **Curve Fitting** from the Chart menu to draw a curve that is “fitted” to the plotted data points.

There are two classes of curve fits: “equations” and “splines.” Equations assume the selected model or curve fit type describes the data being fit. By minimizing the differences between equation and data, the model represents the best approximation of a curve based on all data points. Because the type of model may not accurately represent the data, the curve may not pass through any of the data points. By comparison, a spline fit calculates a series of curve fits internally which must pass through each data point.

Curves or curve fits can be applied to Line, Step, Scatter, Paired Scatter, XY Line, and Paired XY Line charts. You can draw curves for only one series, a selection of data series, or all the series.

DeltaGraph displays curves within the boundaries of the axes only. If you wish to see more of the curve you can extend the data ranges of your Value axes.



By their nature, the formulas used to derive curves in DeltaGraph are complex. A fairly detailed explanation for the equations has been included to help more statistically oriented users understand the methods being used. If you find the math confusing, you can skip over the more technical passages. If you want to learn more about curve fitting, most college-level statistics textbooks can help. You can also refer to *Numerical Recipes in C*, William H. Press, Brian P. Flannery, Saul A. Teukolsky, William T. Vetterling, Cambridge University Press.

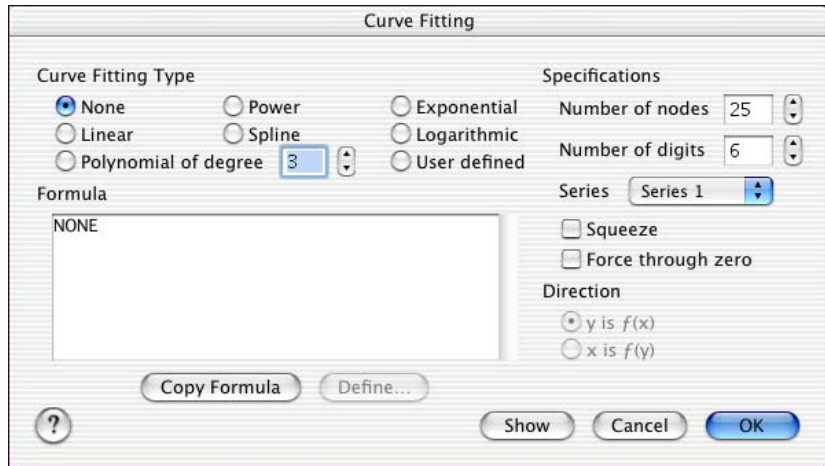
The procedure outlined below takes you step-by-step through the curve fitting process. When appropriate, the steps refer you to other sections of the chapter for additional information.

To fit a curve, do the following:

1. Select the chart. Curve Fits can only be performed on Line, Step, Scatter, Paired Scatter, XY Line, and Paired XY Line charts.



- Click the Curve Fits icon on the extended Command bar, or choose **Curve Fitting** from the Chart menu. A variation of the “Curve Fitting” dialog box appears.



- Click the “Series” pop-up menu in the “Specifications” section of the dialog box to select the series to which you wish to fit a curve.
- Choose the type of curve you want from the “Curve Fitting Type” section. This option determines the type of curve you want to apply to your data series. When you choose a linear, polynomial, power, exponential, or logarithmic curve type, the  $R^2$  or correlation coefficient value, along with the formula, is displayed in the “Formula” window. This value can be copied and pasted in a text block in your Chart page. The formula for a user-defined curve fit is displayed only after it is defined. Refer to the references below for additional information on the types of curve fits.
- Make changes to the “Specifications” and “Direction” sections as necessary. Refer to “Assigning Curve Fit Specifications and Direction” on page 13-18 for information on using these sections. To learn more about “Force Through Zero,” refer to “Force Through Zero Option” on page 13-20.
- Click **Show** to preview your changes without exiting the dialog box. This makes it easy to experiment with different effects. You can move the dialog box out of the way by dragging the Title bar.
- If you want to fit curves to more data series, select a different series (step 3) and repeat steps 4-6.

8. When you have the results you want, click **OK** to implement the changes and return to the Chart page. To exit the dialog box without changing the chart, click **Cancel**.

To learn more about...	Refer to...
Different types of curve fits	“Understanding the Types of Curve Fits” on page 13-15
User-defined curve fitting	“Creating a User-Defined Curve Fit” on page 13-29
Spline curves	“Force Through Zero Option” on page 13-20

## Understanding the Types of Curve Fits

Each of the “other” curve-fitting techniques (“Linear,” “Polynomial,” “Power,” “Exponential,” “Logarithmic,” and “User-defined”) uses the “linear least-squares method” to determine the “best fit” for the curve.  $x_i$  and  $y_i$  represent the data used to plot the chart. These curve fitting techniques are also displayed in the “Curve Fitting” dialog box.

### Choosing a Curve Fit

#### Linear

For the function  $f_x = ax + b$ ,  $a$  and  $b$  are calculated so that the sum of the squares of the errors given by  $(f(x_i) - y_i)^2$  is minimized, as follows:

$$f_x = a \cdot x + b$$

#### Polynomial

For the function  $f_x = a_0 + a_1 \cdot x + a_2 \cdot x^2 + \dots + a_n \cdot x^n$ , a multiple linear regression is performed using  $1, x_i, (x_i)^2, \dots, (x_i)^n$  as the values of the independent variables and  $y_i$  as the values of the dependent variable. The results are used for  $a_0, a_1, \dots, a_n$ .

You can enter a degree of the polynomial from 1 to 10. The default is 3. The degree of the polynomial is dependent on the number of data points plotted as follows:

- 2 points = First-degree polynomial (2 points is a Line)
- 3 points = Second-degree polynomial
- 4 points = Third-degree polynomial
- and so on...

It is not necessary to enter the exact polynomial degree for the plotted data. For example, if you have four points on your chart you can enter a degree of “5.” DeltaGraph automatically finds the appropriate degree to use. All of the “extra” terms in the polynomial higher than the “perfect fit” degree get coefficients of zero.

### Power

For the function  $g(x) = A \cdot x + B$ ,  $A$  and  $B$  are calculated so that the sum of the square of the errors given by  $\sum [g(\ln(x_i)) - \ln(y_i)]^2$  is minimized. Then  $b$  is calculated as  $b = e^B$ , and the formula is displayed as  $f(x) = bx^A$ .

### Spline

The formula is not displayed, because it is not convenient to display such a detailed formula in the “Curve Fitting” dialog box.

### Exponential

For the function  $g(x) = A + b \cdot x$ ,  $A$  and  $b$  are calculated so that the sum of the square of the errors given by  $\sum [g(x_i) - \ln(y_i)]^2$  is minimized. Then  $a$  is calculated as  $a = e^A$ , and the formula is displayed as  $f(x) = e^{(a+bx)} = ae^{bx}$ .

### Logarithmic

For the function  $f(x) = a \cdot \ln(x) + b$ ,  $a$  and  $b$  are calculated so that the sum of the square of the errors given by  $\sum [f(x_i) - y_i]^2$  is minimized.

### User-Defined

Allows you to create a curve-fit model function of your own. Click **Define** to define the parameters and establish your equation. For more information on creating a user-defined curve fit, refer to “Creating a User-Defined Curve Fit” on page 13-29.

## Understanding How Curve Fitting Affects Different Charts

In all the curve-fitting techniques, one or more model functions are chosen to represent the best fit to the data. Every function must get its data from somewhere—the *domain* of the function. The curve representing these functions will go from one end of the domain to the other.

Curve fitting on 2-D Scatter charts is different from curve fitting on a Line or Step chart. For Line and Step charts, the domain of the function is always the category numbers and all numbers in between them. For Scatter charts, the domain can be either the X axis or the Y axis. Here X and Y refer to the data components. The X component of data for the first series always comes from the first column of values selected. The Y component comes from the second column.

The X axis of a Scatter chart is horizontal unless “Switch Axes” is selected in the chart’s “Options” dialog box. To choose a component of the data to indicate the domain of the function, select **Y is F(x)** or **X is F(y)** from the “Direction” section of the “Curve Fitting” dialog box. “Y is F(x)” chooses the values on the  $x$  axis as the domain. “X is F(y)” chooses  $y$  values for the domain.

When a Scatter chart with a Spline curve is selected, the “Direction” buttons are dimmed. Two functions are calculated, one that returns  $x$  values and one that returns  $y$  values. The domain for each function is the category numbers.

In the options discussed below, the values  $x_i$  and  $y_i$  are determined by the chart type and the “Direction” selection. For a Line chart and a Step chart, the  $x_i$  are the category numbers starting at 1 for the first category. The  $y_i$  are the values in each of those categories. (The “Direction” selection does not matter.)

For a Scatter chart, there are two cases as follows:

- If “Y is F(x)” is selected,  $x_i$  are the  $x$  components and  $y_i$  are the  $y$  components.
- If “X is F(y)” is selected,  $x_i$  are the  $y$  components and  $y_i$  are the  $x$  components.

# Assigning Curve Fit Specifications and Direction

## Defining the Number of Nodes

DeltaGraph draws all curve fits as Bezier curves. A Bezier curve is defined by a list of nodes and control points. Nodes are points through which the curve must pass. The position of the nodes, the slopes through the nodes, and the curvature through the nodes, are the same for the curve in DeltaGraph and the model chosen. Between nodes, the position slopes and curvature are not necessarily as precise. Control points are points that affect the path of the curve between nodes.

“Number of Nodes,” in the “Specifications” section of the “Curve Fitting” dialog box, determines the number of Bezier nodes that the curve must pass through. For a Spline curve, this number must be the number of valid data points plotted, so the dialog box item is dimmed. For any other curve, you can enter any number greater than 1 and less than 251. The default is 25.

In general, more nodes means a more accurate representation of the curve. However, more than two nodes on a straight line is unnecessary, while a large number of nodes can use excessive memory.

Because DeltaGraph can curve fit data whose Value axis or axes are scaled by logarithm, the number of nodes for a linear fitted curve is not forced to two. If one Value axis is scaled by logarithm, a linear fitted curve is not a straight line. However, if one Value axis is scaled by logarithm, an exponential fitted curve is a straight line.

## Defining the Number of Digits

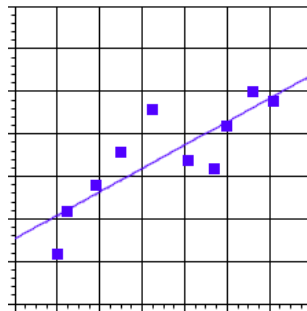
“Number of Digits” in the “Specifications” section of the “Curve Fitting” dialog box determines the number of digits that appear to the right of the decimal in the mantissa of each of the parameters that appear in the curve fit formula displayed in the “Formula” window. Enter a “0” to display *all* digits beyond the decimal. You can enter any number from 1 to 16. The default is 6.

## Selecting a Series

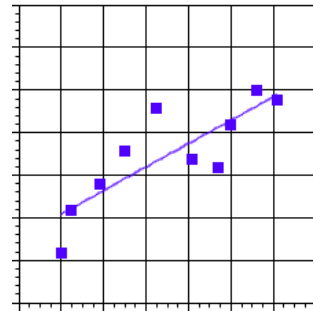
You can use a different type of curve fit on each series of data in the chart. To choose the different series, click the pop-up menu next to “Series” in the “Specifications” section of the “Curve Fitting” dialog box.

## Using the Squeeze Function

When “Squeeze” is off (default) in the “Specifications” section of the “Curve Fitting” dialog box, the curve fit is drawn over the entire axis of the domain of the curve fit function. When “Squeeze” is selected, the curve is drawn over an interval bounded by the smallest and largest domain values.



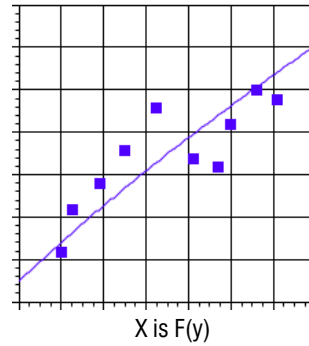
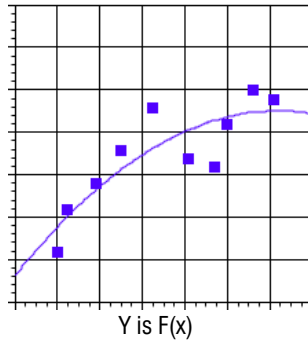
Squeeze Off, the line extends from axis to axis



Squeeze On, the line does not extend out to the axis

## Choosing the Axis Direction

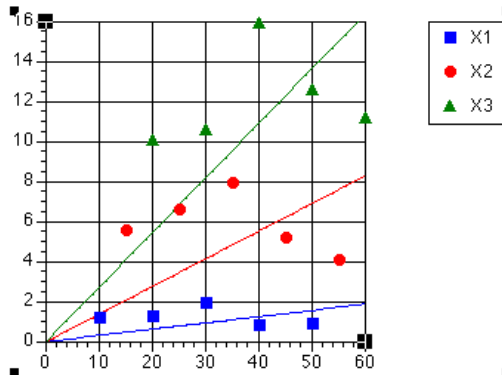
The “Direction” section of the “Curve Fitting” dialog box gives you two options: “Y is F(x)” chooses Y as the dependent variable, and “X is F(y)” chooses X as the dependent variable. The examples below use a second-degree polynomial curve fit.



## Force Through Zero Option

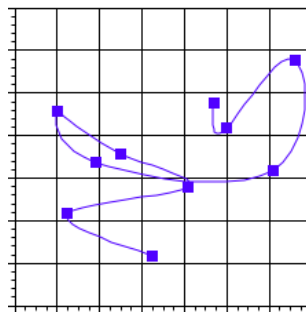
Curve fits typically fit the exact trend of data, meaning they begin part way up the Y axis (at the Y intercept). Sometimes, however, the data actually begin from zero (0) on the Y axis, and using “Force Through Zero” in the “Curve Fitting” dialog box allows you to force the curve through the point of origin.

This option is provided as a convenient means of starting a curve fit at the origin. It applies only to linear and polynomial curve fits. If you want to force a curve fit through some Y intercept, you can do so by creating a user-defined curve fit. A curve forced through the point of origin (zero) can be seen in the next example.

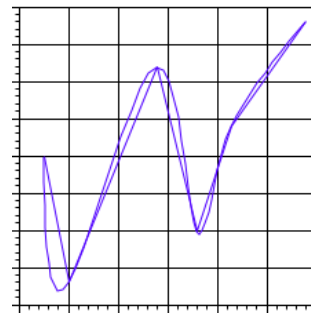


## Creating a Spline Curve

The Spline curve is drawn through all points on the chart. The effects of a Spline curve on a chart are determined by the type of chart plotted. Below are two different types of charts plotted with the same data. On the Scatter chart, the Spline curve connects each data point in the order it was selected from the Data view. An XY Line chart, on the other hand, connects the points from one end of the domain to the other no matter how the data were selected in the Data view.



Scatter Chart



XY Line Chart

## Applying Spline Curves to Line and Step Charts

Over each interval between categories, the curve follows the function

$f_x = a \cdot x^3 + b \cdot x^2 + c \cdot x + d$ , which is a third-degree polynomial. Each interval can have a completely different set  $a$ ,  $b$ ,  $c$ , and  $d$ .

The domain of  $f$  (the values that  $x$  can take) are the category numbers at the data points plotted and all real numbers in between. The range of  $f$  (the values computed by  $f$ ) are in the user scale for the Value axis.

The curve is smooth through the transitions from one third-degree polynomial to the next in the sense that the value of the function, its first and second derivatives at the right end of one interval, are equal to the corresponding values at the left end of the next interval to the right.

Because category numbers are always increasing from 1 to the total number of categories and also go from one end of the Category axis to the other, the curve does not cross over itself.

## Applying a Spline Curve to Scatter Charts

Here the Spline curve is defined by two functions. The  $x$  function is defined on the category number of the data values and all real numbers in between and takes on values in the user scale of the  $X$  axis, while the  $y$  function is defined on the category numbers and takes on values of the  $Y$  axis.

On each interval defined by adjacent category numbers, the curve follows the two functions as follows:

$$x_t = ax \cdot t^3 + bx \cdot t^2 + cx \cdot t + dx$$

$$y_t = ay \cdot t^3 + by \cdot t^2 + cy \cdot t + dy$$

As with the Line chart and Step chart, the  $ax$ ,  $bx$ ,  $cx$ , and  $dx$ , as well as the  $ay$ ,  $by$ ,  $cy$ , and  $dy$ , may be different for each interval. They are calculated so that the function values, and their first and second derivatives, match up on interval boundaries.

Because the  $x$  and  $y$  components of the data values can be listed in any arbitrary order, the  $x$  components are not necessarily increasing with category and neither are the  $y$  components. Therefore, the curve may cross over itself many times. If the desired curve should not cross itself, then the data should be sorted either by the  $x$  component or  $y$  component (either outside of DeltaGraph or with the “Sort” command on the Data menu).

## Understanding User-Defined Curve Fits

This feature allows you to define your own curve-fit function and provides more flexibility to model specifications.

User-defined curve fitting is a general way of picking a function to represent or describe your data. Without user-defined curve fitting, you are restricted to a very narrow range of functions. For instance, the “linear” curve fit allows you to pick functions of the following form:  $f(x) = a \cdot x + b$ .

DeltaGraph picks the best values of  $a$  and  $b$  so that when plotted the function lies “as close as possible” to the data points supplied by the user according to the “linear least-squares” method. But with linear scaling on both axes of a Scatter chart, this curve is not “curvy”—it is a straight line no matter what values are selected by

DeltaGraph for  $a$  and  $b$ . The other curve fit options may give a more “curvy” curve than “linear,” but each curve fit type has its own limited set of curve possibilities.

User-defined curve fits allow quite a bit more freedom in that you can define what the possibilities are. A “Linear” curve fit uses only addition and multiplication and the two parameters  $a$  and  $b$ . User-defined curve fits can use any of the operators, any of the functions (with a few restrictions on the arguments for some functions), and any number of parameters. Knowing what formula to type to get a particular family of shapes requires knowing analytical geometry. But you can use DeltaGraph as an experimental tool to see how different kinds of functions fit different kinds of data.

It is more likely, though, that you have a specific model in mind based on some knowledge of science, mathematics, engineering, economics, or statistics. Such models have a great deal of variety, yet many of them are expressible in the function language of user-defined curve fits. In order to make the translation from a given model in, say, a science text to the function syntax of user-defined curve fits, several features have been implemented.

## User-Defined Curve Fitting Restrictions

Variables can be named almost anything you choose. (Greek letters are not allowed.) Dependent variables don’t all have to be called “ $y$ ” or “ $f$ ,” independent variables don’t all have to be called “ $x$ ” or “ $t$ ,” and parameters don’t have to have names like “ $m1$ ” or “ $m2$ .”

Constants can be defined in order to simplify the formulas, and you can define as many as you like. The value of the constant can be a simple number or the result of evaluating an expression.

Even user-defined curve fit models have limits though. You may have only one independent variable. This represents the  $x$  component of data in a two-component data series (such as in a Scatter chart) or the category number in a one-component data series (such as in a Line chart). You can also have only one dependent variable. This represents the  $y$  component of data in a two-component series (such as in a Scatter chart) or the only component of data in a one-component data series (such as in a Line chart). Your models will not be able to use some commonly used functions that we do not supply; for example, complex functions, many statistical functions, linear transformations, tensor transformations, integration, and differentiation.

To display the “User-Defined Curve Fit” dialog box, do the following:



1. Select the chart to which you want to apply a user-defined curve fit.
2. Click the Curve Fits icon on the extended Command bar, or choose **Curve Fitting** from the Chart menu. The standard “Curve Fitting” dialog box is displayed.
3. Select **User Defined** in the “Curve Fitting Type” section of the dialog box.
4. Click **Define** at the bottom of the dialog box. The following dialog box appears.

### Data

This is the name given to the independent variable. You can call it “x,” “y,” “dollars,” or anything you like. However you spell it, this same spelling must be used whenever you refer to the independent variable in your model. This is just a name, not a formula or equation.

### Model

This is the equation that defines the possibilities in your curve fit model. It is always of the following form:

$$\langle \text{dependent variable name} \rangle = \langle \text{model formula} \rangle$$

$\langle \text{dependent variable name} \rangle$  follows the usual naming rules. The name is just for “show”—you never have to type it in any other box. It will be displayed in

the “Curve Fitting” dialog box’s “Formula” window and may also appear in the partial derivative expressions in the “User-Defined Curve Fit” dialog box’s “Output” window when “Show Partial” is selected.

<model formula> is the mathematical formula that details how the dependent variable of your model depends on the independent variable and the parameters. You can use any of the operators and functions that are valid in the “Formula Builder” dialog box. You must use at least one parameter in your formula. You must use the independent variable in your formula. You can use any constant you have defined in the “Constants” window.

### Constants

Each constant is a name and a formula. The constant name can be used in any other formula. It is shorthand for the result of evaluating the constant formula. Any number of constants can be defined (within limits of memory).

Each constant definition consists of a constant name and an equal sign followed by a constant expression. If you have more than one definition, they must each be separated by a list separator (; in the U.S. version). These are called “constants” because a constant’s formula may not depend on the independent variable or any of the parameters. Other than that restriction, a constant can have any formula.

The “Parameters” window can contain references to these constants in the formulas that set the initial values of the parameters. The model formula can also refer to any of these constants. These are entirely optional.

### Parameters

These are the quantities that give your model its flexibility. In your model formula, you place a parameter in the formula wherever you have an unknown quantity and you want DeltaGraph to figure out what the best value of that quantity should be.

Even though DeltaGraph will calculate a “best” value, you must supply an “initial” value. The technique used for determining best parameter values is an “iterative” algorithm (the Levenberg-Marquardt method, see *Numerical Recipes in C*) and as such requires a reasonable guess as to what the right answer is before it can “improve” on it to get the best answer.

Any name you use for a parameter, along with an initial value for that parameter, must be included in the “Parameters” window and must follow a certain format. List separators (; in the U.S. version) must be used to separate each parameter def-

initiation. Parameter names follow the usual rules for names. Initial values can be numbers or formulas referring to constants.

Please remember that any formulas used are for initial values only. DeltaGraph improves upon those values when you apply the curve fit. The final values of the parameters are displayed in the “Output” window of the “User-Defined Curve Fit” dialog box and in the “Formula” window of the “Curve Fitting” dialog box. If “Save All Iterations” is selected (default), then the best value so far of each parameter at each step of the iteration is displayed in the “Output” window when the curve fit is complete.

Since the iteration starts with your initial values for each of the parameters and improves on them with each step, the final values of the parameters may be very different for a different set of initial values.

## User-Defined Curve Fitting Rules

Every entry you make in the “Data,” “Model,” “Constants,” and “Parameters” fields in this dialog box starts out with a name followed by an equal sign (=). Each of these names is used to name the independent variable, the dependent variable or model, constants, or parameters that you are defining for your curve-fitting function. Refer to the “User-defined Curve Fit” dialog box illustration on page 13-24 and the figure on page 13-33 for a user-defined curve fit example. The names that you assign to these fields have the following restrictions:

- Names must be limited to 63 characters.
- An equal sign (=) must follow the name. (The “Data” field does not require an equal sign.)
- The name must be unique and cannot be used to name any other function in another field in the dialog box.
- The name can consist of a combination of letters and numbers or a single quote (') but cannot start with a number.
- The name cannot contain an underscore (\_) or spaces.

After the name and equal sign (=), the “Model,” “Constants,” and “Parameters” fields contain a definition for the name.

- Constants can be defined with expressions containing references to previously defined constants.
- Parameters can be initialized with expressions containing references to previously defined parameters *or* constants.
- Letters of the alphabet can be used in the expression to represent columns of data in the Data view. If you want to use letters in the expression to refer to something other than columns of data, you must define them in the “Constants” or “Parameters” window. Letters that refer to columns, or what the computer may refer to as columns, are automatically capitalized. Columns in the Data view are lettered from A through Z, then AA through IT.
- You can have multiple parameters or constants defined, however, be sure to separate definitions by a list separator (a semicolon is used in the U.S.).

## Using Formulas in a User-Defined Curve Fit

You can use function arguments from the “Formula Builder” dialog box in the user-defined curve fit. Formulas are appropriate in three places as follows:

- The constant expression in a constant definition.
- The initial value in a parameter definition.
- The model formula in the model definition.

Other formula restrictions are as follows:

### Model

You can use the following functions without restriction in the “Model” window: sin, asin, sinh, asinh, cos, acos, cosh, acosh, tan, atan, tanh, atanh, cot, acot, coth, acoth, csc, acsc, csch, acsch, sec, asec, sech, asech, Sqrt, Exp, Log, Ln. You can also use the formula for the nth root as described in “Formula Builder Functions” on page 4-32.

Any of the other functions can be used with the following restriction: neither the independent variable (“Data”) nor any parameter may appear in any of the function arguments. (This is to insure that the other function has zero-valued partial derivatives with respect to the independent variable and each parameter.) If you break the rule, an error message alerts you when you click **Calculate Now**.

### Constants and Parameters

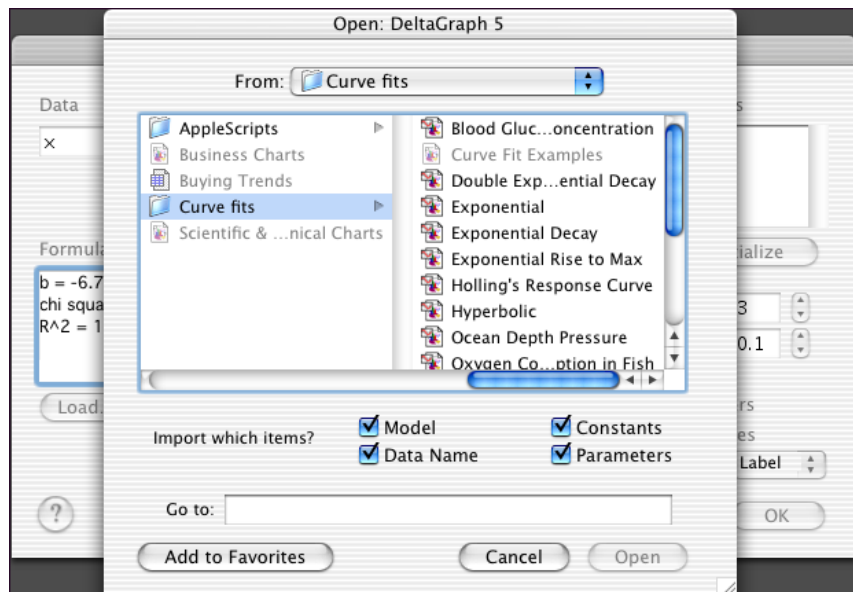
All of the functions can be used with no restrictions when defining values for constants and initial values for parameters.

To learn more about...	Refer to...
“Formula Builder” command	“Building Formulas” on page 4-27

## Loading and Saving a Curve Fit

After entering your curve fit information in the User-defined Curve Fit dialog box, click **Save**. You can name and save these formulas as you would a regular file.

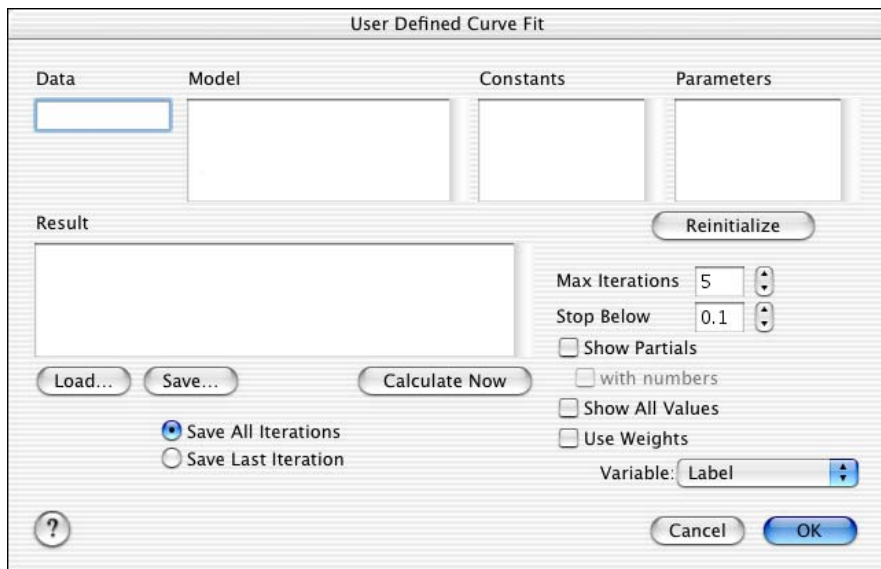
You can also load all or just a portion of a saved user-defined curve fit. From the User-defined Curve Fit dialog box, click **Load**. You can select whether or not you want the Model, Data Name, Constants, or Parameters section of the saved user-defined curve fit to be loaded in the “User-defined Curve Fit” dialog box. User-defined curve fits can only be opened from this dialog box.



## Creating a User-Defined Curve Fit

To create a user-defined curve fit, do the following:

1. Display the “User-defined Curve Fit” dialog box as previously described. (See page 13-24).



2. Enter the name you want to use to reference the independent variable in the text box next to “Data.”
3. In the window under “Constants,” enter the name of a constant, an equal sign (=), then an expression to initialize the constant. This is an optional entry. The “Constants” window does not require any entry if you do not want to use named constants.

You can have multiple constants, but you must separate them by a semicolon. The constants are used in the model expression (in the “Models” window) or in the parameter initializations (in the “Parameters” window).

You cannot use the independent variable, the dependent variable, the “Data” name, or a parameter name in the expression. Constants can be referenced and

used in the “Parameters” window, or you can enter references to a cell in the Data view.

To reference a cell, type the column letter followed by the cell number of the referenced data in square brackets. The example below references k1 as the constant name, D as the column letter, and 2 as the referenced cell in column D.

Example:  $k1=D[2]$

4. In the window under “Parameters,” enter the name of the parameter, an equal sign (=), then an expression to initialize the parameter.

You must have at least one parameter. If you have multiple parameters, separate them by a semicolon. Because your model must include at least one parameter, you must have at least one parameter name and initial value included in the “Parameters” window (otherwise DeltaGraph has nothing to calculate).

User-defined curve fitting uses an iterative algorithm. The initial value of each parameter is supplied here. Each iteration may improve the values of the parameters.

In the curve-fitting algorithm, the constant expressions are evaluated and stored in the order you list them. Then the initial values of the parameters are evaluated and stored in the order you list them. Note the order in which you list the constants and parameters, and take care not to refer to an undefined constant or uninitialized parameter.

You cannot use the independent variable or the dependent variable to define any of the initial values of the parameters.

5. In the window under “Model,” enter the name you want to use as the dependent variable, an equal sign (=), then the model expression.

You must have a model equation. The expression defines how the dependent variable in the model depends on the independent variable and the parameters. It should contain at least one reference to the independent variable (the name entered in the “Data” section) and at least one reference to a parameter.

6. Make changes to other options, as follows, in the dialog box as necessary.

**Reinitialize**

Changes the starting value for the next iteration based on the last value of the previous curve fit. If a curve fit has been calculated, clicking this button causes the best values of the parameters to be substituted for the initial values of the parameters. This change is reflected in the Parameters block. This is useful if the previous curve fit calculation has stopped because the maximum number of iterations was attained, but it is desirable to continue the curve fit algorithms for more iterations. Click **Reinitialize** followed by **Calculate Now** to continue the curve fitting algorithm. If a curve fit has not been calculated, then this button will be disabled.

**Max Iterations**

Determines the maximum number of calculations used in finding the best possible values for the parameters. Enter numbers from 1 to 32,767. The default is 5. Remember, the more iterations, the longer the computation takes. All iterations are displayed in the “Output” window and numbered 1:, 2:, etc. (see the figure on page 13-33). If you enter “0,” no iterations are calculated. The initial values of the parameters are used as the final values of the parameters

**Stop Below**

The algorithm stops if chi-squared is less than the entered value at the end of an iteration.

**Show Partial**

Shows partial derivatives of the model function with respect to the independent variable and parameters. This option does not affect the curve fit in any way; it is there for your own information.

**with numbers**

Controls how the parameters inside a partial derivative are displayed in the “Output” window. Show partials must be selected. If deselected, then the names of the parameters are displayed in the partial derivative formulas. If selected, then the best values of the parameters are displayed in place of names in the partial derivative formulas.

**Show all Values**

Determines how many iterations will be displayed in the “Output” window, good or bad, including chi-squared. When off, the “Output” box shows the best values of the parameters at any step shown. When on, the “Output” box shows the new parameters values used at each step shown, in addition to the best values.

**Use Weights**

Determines the value contributed to the total error of chi-squared using values from a column of data from the Data view. Default is 1.0. To explicitly define the weights, enter the weights in one column of the Data page. Select that column from the “Use Weights” pop-up menu. The number on the first row of the column is the weight for the first selected row in the data selection. Subsequent rows of the weight column correspond to subsequent rows of the data selection.

Specifying a weight much smaller than 1.0 for a particular data point magnifies the contribution of that data point to the total error and can make the total error much larger.

Specifying a weight much larger than 1.0 for a particular data point diminishes the contribution of the data value to the total error. It does not reduce the total error much unless all the data values are assigned weights greater than 1.0.

**Save All Iterations**

Puts the values of the parameters and the measure of fit after each step through the iteration in the “Output” window. Each iteration in the “Output” window is numbered 1:, 2:, etc. (see the figure on page 13-33).

**Save Last Iteration**

Puts the values of the parameters and the measure of fit after the last iteration only in the “Output” window.

7. Click **Calculate Now**. The iterations of the function appear in the “Output” window of the dialog box (see the figure on page 13-33).

The first line of the iteration contains the iteration number. This number has no significance other than to divide the iterations. Next comes the replacement values for the parameters for that iteration. These values change with every iteration and are used by the model function to obtain a best fit for the curve. A value for the square of the regression coefficient  $R^2$ , and chi-squared appear at the bottom of each iteration. Chi-squared is defined as follows:

$$\chi^2 = \sum \{[y_i - f(x_i)]/\sigma_i\}^2$$

where:

$y_i$  is the y component of the data

$x_i$  is the  $x$  component of the data

$f(x_i)$  is the value of the right-hand side of the model equation evaluated by substituting  $x_i$  for the independent variable

$\sigma_i$  is the weight associated with each data point

the sum,  $\Sigma$ , is taken over all the categories in a data selection

If “Show Partial Derivatives” was selected, the partial derivatives of the model appear at the end of the output statement. These derivatives are the partial derivatives (a concept from calculus) of the model function with respect to the independent variable and with respect to the parameters. You may find them useful, and the curve-fit algorithm needs to know them for the sake of precision. If you have  $n$  parameters, you will get  $n + 1$  partial derivatives. Following is an example of user-defined curve fit output.

```

1:
a = 1.618043E+0
b = -3.312051E+9
chi squared = 1.510217E+2
R^2 = 9.772041E-1

2:
a = 1.618043E+0
b = -3.312051E+9
chi squared = 1.510217E+2
R^2 = 9.772041E-1

3:
a = 1.618043E+0
b = -3.312051E+9
chi squared = 1.510217E+2
R^2 = 9.772041E-1

∂y/∂x = a*1/(x-b)
∂y/∂a = Ln(x-b)
∂y/∂b = a*-1/(x-b)

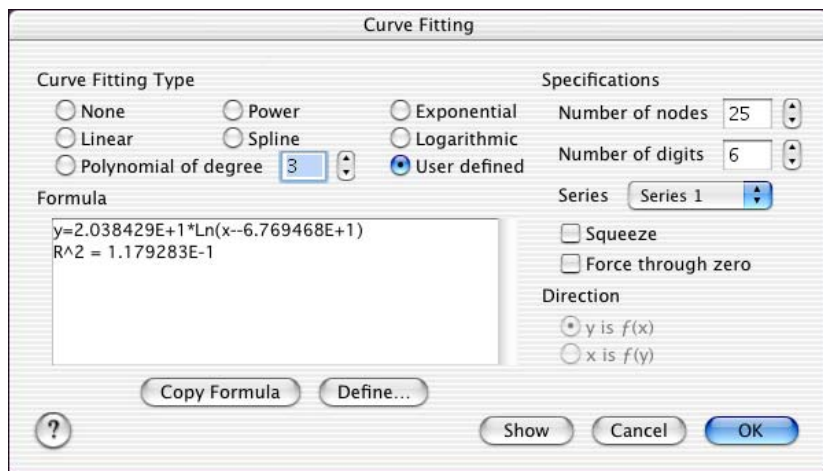
```

The Greek symbol “ $\partial$ ” is used in the standard way to specify the derivative of one variable (always the dependent variable) with respect to another variable (the independent variable or a parameter). To the right of the symbol for a derivative is an equal sign (=) followed by the formula for the specified derivative. That formula may have references to the independent variable and/or to the parameters, as dictated by the model formula you specify and the rules of calculus and algebraic simplification. Each partial derivative then looks like the following in the “Output” window:

$$\partial \langle \text{one variable} \rangle / \partial \langle \text{another variable} \rangle = \langle \text{formula} \rangle$$

8. Click **OK** to make the changes and return to the “Curve Fitting” dialog box. If you want to return to the “Curve Fitting” dialog box without specifying a user-defined curve fit, click **Cancel**.

The final model formula for the user-defined curve fit now appears in the “Formula” window of the “Curve Fitting” dialog box. The parameters in the formula have been replaced by numbers. These numbers are derived from the iteration with the best fit.



9. Click **Show** to preview your changes without exiting the “Curve Fitting” dialog box. “Show” makes it easy to experiment with different effects. You can move the dialog box out of the way by dragging the Title bar.

If the curve fit is not a good fit, it may require more iterations. Instead of using the same parameters and just requesting more iterations, replace the parameters in the “User-Defined Curve Fit” dialog box with the values in the “Formula” window. By replacing values, you can start from where you left off without recalculating any previous iterations.

10. To fit curves to more data series, select a different series by clicking the “Series” pop-up menu and repeat steps 2-9.
11. Click **OK** to implement the changes and return to the Chart view. To exit the dialog box without changing the chart, click **Cancel**.

## User-Defined Curve Fit Samples

The following table lists some example User-defined Curve Fit formulas and Models. The sample files are included with DeltaGraph and are located in the “Curve Fits” folder in the “Sample Files” folder. These formulas can be loaded in the “User-defined Curve Fit” dialog box in the “Curve Fitting” dialog box.

These samples can be used as starting points for creating your own formulas and models. The initial parameter values are listed and the formulas and models are displayed in mathematical terminology. For example:

$$f(x) = \frac{ax}{b+x} \text{ is the same as } y=(a*x)/(b+x)$$

### User-Defined Curve Fit Sample Formulas

Following are sample user-defined curve fit formulas included with DeltaGraph:

Curve Fit	Formula	Parameters
Double Exponential Decay	$f(x) = ae^{-bx} + ce^{-dx}$	$a, b, c, d$
Exponential	$f(x) = be^{x(-1a)}$	$a, b$
Exponential Decay	$f(x) = ae^{-bx}$	$a, b$
Exponential Rise to Max	$f(x) = a(1 - e^{-bx}) + c$	$a, b, c$
Hyperbolic	$f(x) = \frac{ax}{b+x}$	$a, b$
Parabola	$f(x) = x^2 - ax + b$	$a, b$
Power of Natural Log	$f(x) = \ln(x^9 + ax^5 + b)^3$	$a, b$
Reciprocal	$f(x) = \frac{1}{ax}$	$a$
Reciprocal of Square Root	$f(x) = \frac{1}{\sqrt{x^2 + a}}$	$a$
Sigmoid Logistic	$f(x) = \frac{a-d}{1 + \left(\frac{x}{c}\right)^b} + d$	$a, b, c, d$

Curve Fit	Formula	Parameters
Sine Wave	$f(x) = (a + b) \sin\left(\frac{2\pi x}{c} + d\right)$	$a, b, c, d$
Weighted 3rd Order Poly	$f(x) = a + bx + cx^2 + dx^3$	$a, b, c, d$

## User-Defined Curve Fit Sample Models

Following are the sample user-defined curve fit models included with DeltaGraph:

Curve Fit	Formula	Parameters
Blood Glucose Concentration (mg/dl)	$f(x) = a(1 - be^{-kx})$	$k = 0.049$ $b = 1$ constants: $a = 139.2$
Holling's Response Curve (prey eaten/minute)	$f(x) = \frac{ax}{b + x}$	$a = 0.5$ $b = 0.25$
Ocean Depth Pressure (N/M <sup>2</sup> )	$f(x) = a + bcx$	$a = 1.01 \cdot 10^5$ constants: $b = 1 \cdot 10^3$ $c = 9.8$
Oxygen Consumption in Fish (ml/hour)	$f(x) = ax^k$	$k = 0.76$ $a = 1$

## Achieving Better Curve Fits

This section provides you with valuable hints and tips for creating more accurate curve fits.

### Reinitializing Parameters

Reinitializing your initial parameters will sometimes increase their accuracy, making a much better fit. To reinitialize your parameters, do the following:

1. Input your model and starting parameter(s).
2. Increase the “Max Iterations” to a value appropriate to your model, computer speed, and computer math capabilities. This value can be as low as 1 for simple equations and as high as 10,000 for more complex ones. The models listed in the figure on page 13-33 were calculated using 20 to 50 iterations.
3. Click **Calculate Now**.
4. Click **Reinitialize**.
5. Repeat steps 3 and 4 until the parameters have minimal change or the “chi-squared” value is sufficiently small.
6. Click **OK**.

## Curve Fit Hints

Following are helpful hints and troubleshooting tips that may assist you in creating your curve fits:

- The graph on the screen matches the function, specified in the model, very well at the nodes. The nodes are evenly spaced along the curve in the x-axis direction if  $y + f(x)$  is selected for the curve fit. The placement of nodes can be controlled by the “Number of Nodes” option in the “Curve Fitting” dialog box. Sometimes a graph can look very different from the model because the number of nodes is too small.
- Avoid placing a node on an  $x$  value where the  $f(x)$  value is not defined. Remember some of the functions supplied in DeltaGraph have ranges of values where the function is not defined. For example:

$\sqrt{x}$  is undefined for  $x < 0$

$\log x$  is undefined for  $x \leq 0$

$\frac{1}{x}$  is undefined for  $x = 0$

$\tan x$  is undefined for  $x - \frac{\pi_i}{2} + k \cdot \pi_i$ , where  $k$  is any integer.

- Avoid selecting initial values of parameters that cause the evaluation of the model function at any of the data value's  $x$  coordinates to be undefined. This could cause a calculation error message to be displayed.
- Avoid selecting initial values of the parameters that cause the evaluation of the model function at any of the node's  $x$  coordinates to be undefined. This can cause the graph to be drawn "strangely" in the vicinity of such a node.
- Be sure the number of parameters initialized in the "Parameters" block matches the spelling and number of parameters used in the model functions.
- Compare the "chi-squared" error values displayed with the number in the "Stop below" value. Repeated "Calculate Now"/"Reinitialize" cycles are not necessary if the "Stop below" value adequately describes your requirement for terminating the iteration process.
- User-defined curve fits may not always return what is expected. In the process of iterating, the general method does not always converge to what you would get with the "same" formula chosen from the standard curve fit types. It may not converge to what "looks" like the best answer. It does not necessarily converge to the smallest possible "chi-squared" value. It does not necessarily converge at all. It may start out fine and then pick a value for a parameter that produces undefined results and not be able to improve on the solution from that point on.
- All user-defined curve fits depend on the data values supplied, the model, and the initial values of the parameters. Changing the data and initial values of parameters can make a very large difference in the final result.