The automatic multithreading added in Igor7 makes the CurveFit /NTHR flag obsolete. The automatic multithreading is more effective than the threading available in Igor6.

Automatic multithreading when doing ODR fits (see **Errors in Variables: Orthogonal Distance Regression** on page III-236) is most effective with fit functions having large numbers of fit coefficients and with user-defined fit functions that are expensive to compute. Surprisingly, it may be more effective with fewer input data points.

If you use the ThreadSafe keyword with your user-defined fit function and Igor's automatic multithreading is enabled for your fit, your function must not access waves or global variables. Therefore your function must not use WAVE, NVAR, or SVAR declarations.

Curve Fitting with Programmed Multithreading

The CurveFit, FuncFit, and FuncFitMD operations are threadsafe. Consequently another way to use multiple processors is to do more than one curve fit simultaneously, with each fit using a different processor. This requires threadsafe programming, as described under **ThreadSafe Functions and Multitasking** on page IV-329, and requires at least intermediate-level Igor programming skills.

The fitting function can be either a built-in fit function or a threadsafe user-defined fit function.

For an example, choose File→Example Experiments→Curve Fitting→MultipleFitsInThreads.

Constraints and ThreadSafe Functions

The usual way to specify constraints to a curve fit is via expressions in a text wave (see **Fitting with Constraints** on page III-227). As part of the process of parsing these expressions and getting ready to use them in a curve fit involves evaluating part of the expressions. That, in turn, requires sending them to Igor's command-line interpreter, in a process very similar to the way the **Execute** operation works. Unfortunately, this is not threadsafe.

Instead, you must use the method described in **Constraint Matrix and Vector** on page III-230. Unfortunately, it is hard to understand, inconvenient to set up, and easy to make mistakes. The best way to do it is to set up your constraint expressions using the normal text wave method (see **Constraint Expressions** on page III-228) and use the /C flag with a trial fit. Igor will generate the matrix and vector required.

In most cases, the basic expressions will not change from one fit to another, just the limits of the constraints will change. If that is the case, you can use the matrix provided by Igor, and alter the numbers in the vector to change the constraint limits.

User-Defined Fitting Functions

When you use the New Fit Function dialog to create a user-defined function, the dialog uses the information you enter to create code for your function in the Procedure window. Using the New Fit Function dialog is the easiest way to create a user-defined fitting function, but it is possible also to write the function directly in the Procedure window.

Certain kinds of complexities will *require* that you write the function yourself. It may be that the easiest way to create such a function is to create a skeleton of the function using the dialog, and then modify it by editing in the procedure window.

This section describes the format of user-defined fitting functions so that you can understand the output of the New Fit Function dialog, and so that you can write one yourself.

You can use a variety of formats for user-defined fit functions tailored to different situations, and involving varying degrees of complexity to write and use. The following section describes the simplest format, which is also the format created by the New Fit Function dialog.