fitting coefficients. You can either differentiate your function and write another function to provide derivatives, or you can use a numerical approximation. Igor uses a numerical approximation.

Confidence Bands and Nonlinear Functions

Strictly speaking, the discussion and equations above are only correct for functions that are linear in the fitting coefficients. In that case, the vector **a** is simply a vector of the basis functions. For a polynomial fit, that means 1, x, x^2, etc. When the fitting function is nonlinear, the equation results from an approximation that uses only the linear term of a Taylor expansion. Thus, the confidence bands and prediction bands are only approximate. It is impossible to say how bad the approximation is, as it depends heavily on the function.

Covariance Matrix

If you select the Create Covariance Matrix checkbox in the Output Options tab of the Curve Fitting dialog, it generates a covariance matrix for the curve fitting coefficients. This is available for all of the fits except the straight-line fit. Instead, a straight-line fit generates the special output variable V_rab giving the coefficient of correlation between the slope and Y intercept.

By default (if you are using the Curve Fitting dialog) it generates a matrix wave having N rows and columns, where N is the number of coefficients. The name of the wave is M_Covar. This wave can be used in matrix operations. If you are using the CurveFit, FuncFit or FuncFitMD operations from the command line or in a user procedure, use the /M=2 flag to generate a matrix wave.

Originally, curve fits created one 1D wave for each fit coefficient. The waves taken all together made up the covariance matrix. For compatibility with previous versions, the /M=1 flag still produces multiple 1D waves with names W_Covar*n*. Please don’t do this on purpose.

The diagonal elements of the matrix, M_Covar[i][i], are the variances for parameter i. The variance is the square of sigma, the standard deviation of the estimated error for that parameter.

The covariance matrix is described in detail in *Numerical Recipes in C*, edition 2, page 685 and section 15.5. Also see the discussion under **Weighting** on page III-199.

Correlation Matrix

Use the following commands to calculate a correlation matrix from the covariance matrix produced during a curve fit:

```
Duplicate M_Covar, CorMat  // You can use any name instead of CorMat
CorMat = M_Covar[p][q]/sqrt(M_Covar[p][p]*M_Covar[q][q])
```

A correlation matrix is a normalized form of the covariance matrix. Each element shows the correlation between two fit coefficients as a number between -1 and 1. The correlation between two coefficients is perfect if the corresponding element is 1, it is a perfect inverse correlation if the element is -1, and there is no correlation if it is 0.

Curve fits in which an element of the correlation matrix is very close to 1 or -1 may signal “identifiability” problems. That is, the fit doesn’t distinguish between two of the parameters very well, and so the fit isn’t very well constrained. Sometimes a fit can be rewritten with new parameters that are combinations of the old ones to get around this problem.

Identifiability Problems

In addition to off-diagonal elements of the correlation matrix that are near 1 or -1, symptoms of identifiability problems include fits that require a large number of iterations to converge, or fits in which the estimated coefficient errors (W_sigma wave) are unreasonably large.

The phrase "identifiability problems" describes a situation in which two or more of the fit coefficients trade off in a way that makes it nearly impossible to solve for the values of both at once. They are correlated in a way that if you adjust one coefficient, you can find a value of the other that makes a fit that is nearly as good.