Igor
Technical Notes

WaveMetrics Technical Support

#007: Multi-variate Curve Fitting

Written by: Staff May 1990

This Technical Note and accompanying experiment describe how to accomplish multi-variate curve fitting using iterative fits to user defined functions of the form: y=f(x1,x2,...xn).

Accompanying files:

'Multi-variate fit' -- sample experiment

Igor's curve fits work with equations of the form y= f(x). That is, just one independent and one dependant variable. With a little work it can be made to handle equations with multiple independent variables, i.e. y=f(x1,x2,...xn). The downside is that only the iterative fitting technique can be used and the user has to attend to a number of details. The upside is that non-linear as well as linear fits can be done.

The reason that it appears that Igor can not do multi-variate user function fitting is that the user function has to be of the form y=f(w,x) where w is a vector of parameters and x is the current x value. Thus it appears that only one x value can be passed to the function. The trick is to recognize that the fitting routine does not utilize the x values except to pass them to the user function. Thus, rather than being real x values they could be, for example, point numbers. The point numbers then could be used in the user function to index global waves containing the real x values.

Here, then, is the recommended procedure:

Step 1: Create 'generic' x-waves that will be referenced as globals in the function you are about to write. Try to resist the temptation of using your actual x data waves unless you are short on memory. The generic waves do not have to be of any particular length, number type or contain any data. If you are absolutely positive that you will never have to fit any other data sets then you can ignore this advice and use your actual data waves.

Step 2: Write your function referencing the generic x-waves using point access with the x value passed to the function as the point number.

Step 3: Use the Duplicate/O command to copy your real data into the generic waves. You may want to write a macro to do this if you will be fitting a number of data sets.

Step 4: Make sure you will be passing point numbers to the fitting function during the fit. Either ensure that the y data wave has point scaling or create an x wave containing point numbers.

Step 5: Create a parameter wave and fill with initial guesses.

Step 6: Create a destination wave and graph both (point number as x axis).

Step 7: Execute your function using the destination wave as the destination and using 'P' as the x value. (i.e. destw= foofunc(pwave,P)). Adjust your equation and/or parameter values until the destination wave matches the data wave reasonably well.

Step 8: Do the fit.

Here is an example using the above steps:

First we need to create some fake data and define the function we will be fitting. For this example we will be using just a simple linear combination:

```
Make xdata1=sin(p/20),xdata2=cos(p/15),xdata3=sqrt(p)
Make ydata= 1.2*xdata1 + 2.3*xdata2 + 0.07*xdata3 + gnoise(0.2)
```

Step 1: Generic x-waves:

```
Make x1,x2,x3
```

Step 2: Write function.

```
Function foofunc(w,p)
        wave w
        variable p

        variable t= w[0]*x1[p] + w[1]*x2[p]
        return t + w[2]*x3[p]
End
```

Step 3: Use Duplicate/O.

```
Duplicate/O xdata1,x1
Duplicate/O xdata2,x2
Duplicate/O xdata3,x3
```

Step 4: Ensure point scaling. Here we don't really have to do anything because ydata was created with point scaling. If it had not been, here is what we would have done:

```
Duplicate/O ydata,ydata_pnt; ydata_pnt= P
```

Step 5: Create a parameter wave and fill with initial guesses.

```
Make fooparam={1,2,0.1}
```

Step 6: Create a destination wave and graph both (point number as x axis).

```
Duplicate ydata,ydata_fit
Display ydata,ydata_fit
```

Step 7: Execute function.

```
ydata_fit= foofunc(fooparam,P)
```

Step 8: Do the fit.

```
•FuncFit foofunc fooparam ydata /D=ydata_fit
 Fit converged properly
 ydata_fit= foofunc(fooparam,x)
 fooparam={1.2226,2.3504,0.066544}
 V_chisq= 4.45568; V_npnts= 128; V_numNaNs= 0; V_numINFs= 0;
 W_sigma={0.0311,0.0311,0.00216}
```