Subject: Re: HELP: Levenberg-Marquardt method
Posted by agraps on Wed, 08 Mar 1995 01:04:40 GMT
View Forum Message <> Reply to Message

kletzing@unhedi2.unh.edu (Craig Kletzing) writes:

> Could you elaborate on the error in CURVEFIT? I've never looked over
> RSI's implementation very carefully.

> Craig Kletzing
> University of New Hampshire

Way back in '87, I dug into the guts of Bevington's math, then looked at Bevington's implementation and compared it to Numerical Recipes' implementation (called MRQMIN). I got different results for the test case I cooked up (see the test case below). After some inspection, I learned that IDL's CURVEFIT didn't make a final pass of calling the function when it had satisfied the chi-squared criteria, and MRQMIN did. So I added that into my version of CURVEFIT, and I got the same results that MRQMIN gave.

In addition, I occasionally need to use the output covariance matrix (instead of just the sigma's, which is the diagonal of the covariance matrix), in order to verify that a fit is good, so I added that capability to my CURVEFIT as well.

I sent Dave Stern at RSI a 9-track tape of my new CURVEFIT when I discovered this error (in '87), but it never appeared in any new release of the function, so I think I fell through the cracks there somehow.

This CURVEFIT topic came up last Fall when Mathew Larkum asked for a simple set of instructions on how to use CURVEFIT. I posted this exact same example and my version of CURVEFIT. He took my code and generalized it to work to call any function. Now with Mark Rivers' version, we can somehow combine the three???

DIGRESSING...
As you are probably aware, the Marquardt-Levenberg method is sort of a black art.

I've done a bit more work with CURVEFIT, fitting very complicated second-harmonic Voigt functions that are particularly sensitive to initial guesses. One of the tricks that I learned is that you can tweak the direction the algorithm goes in chi-square space with different "flambda's" for each parameter being fit. I played around with this flambda (not very rigorously), and sometimes it worked at giving me better fits, sometimes not. (Like I said,

it's a black art.)

I.e. words I set up and used flambda something like the
following:

Before entering CURVEFIT, I initialized it:

```
;percentage of lambda to increase/decrease
del_flambda = dblarr(nterms) ;# of params.
del_flambda = [30.d0,30.d0,30.d0,.30d0] ;Use different #s here for ea parameter
```

And then inside CURVEFIT I used:
flambda = flambda/del_flambda ;decrease flambda by the different factors

instead of:
flamda = flambda/100
(in the original CURVEFIT)

END DIGRESSING

Here is my test case described earlier with my covariance matrix
addition, and adding a final call to the function at the end of
the CURVEFIT algorithm. I *don't* have the del_flambda adjustment
in this one, but you can modify it easily.

Amara

```
 ------------------------------------------------------------ ------
PRO FUNCT, X,A,F,PDER
;This function is necessary to run IDL's CURVEFIT procedure
;A. Graps  10-29-87
;
F = A(0) + A(1) * X + A(2) * ALOG(X)
;
;Need partial derivatives
PDER = FLTARR(N_ELEMENTS(X),3)
PDER(*,0) = 1
PDER(0,1) = X
PDER(0,2) = ALOG(X)
RETURN
END
 ;************************************************************ ********
PRO CURVEFITTEST,XDATA,YDATA,FIT,A,SIGMAA,COVAR
;PURPOSE: To test IDL's CURVEFIT procedure.
;Amara Graps  11-3-87
;
XDATA=FLTARR(6)
YDATA=FLTARR(6)
```

```
;
;Input the values for XDATA and YDATA
FOR I=0,5 DO XDATA(I) = I+1
YDATA(0) = 2.6
YDATA(1) = 2.4
YDATA(2) = 3.0
YDATA(3) = 4.1
YDATA(4) = 5.4
YDATA(5) = 6.6
;
;Set the weights (Here, the weights = 1, i.e. no weighting)
W = REPLICATE(1.,6)
;
;Give the estimates for A
A = [.75,1.70,-2.45]
;
CURVEFIT,XDATA,YDATA,W,A,SIGMAA,FIT,COVAR
;
;Print results
PRINT, 'Calculated A= ',A
PRINT, 'Covariance Matrix= '
PRINT, COVAR
PRINT, 'Standard Deviation of Fit Parameters= '
PRINT, SIGMAA

;
END
 ;************************************************************ ***************

PRO CURVEFIT, X, Y, W, A, SIGMAA, YFIT, COVAR
;+
; NAME:
; CURVEFIT
; PURPOSE:
; Non-linear least squares fit to a function of an
; arbitrary number of parameters.
; Function may be any non-linear function where
; the partial derivatives are known or can be approximated.
; CATEGORY:
; E2 - Curve and Surface Fitting
; CALLING SEQUENCE:
; CURVEFIT,X,Y,W,A,SIGMAA,YFIT,COVAR
; INPUTS:
; X = Row vector of independent variables.
; Y = Row vector of dependent variable, same length as x.
; W = Row vector of weights, same length as x and y.
;  For no weighting
;  w(i) = 1., instrumental weighting w(i) =
```

```
;  1./y(i), etc.
; A = Vector of nterms length containing the initial estimate
;  for each parameter.  If A is double precision, calculations
;  are performed in double precision, otherwise in single prec.
;
; OUTPUTS:
; A = Vector of parameters containing fit.
; Function result = YFIT = Vector of calculated
;  values.
;       Covariance matrix= error of YFIT showing correlations
; OPTIONAL OUTPUT PARAMETERS:
; Sigmaa = Vector of standard deviations for parameters
;  A.
;
; COMMON BLOCKS:
; NONE.
; SIDE EFFECTS:
; The function to be fit must be defined and called FUNCT.
; For an example see FUNCT in the IDL User's Libaray.
; Call to FUNCT is:
; FUNCT,X,A,F,PDER
; where:
; X = Vector of NPOINT independent variables, input.
; A = Vector of NTERMS function parameters, input.
; F = Vector of NPOINT values of function, y(i) = funct(x(i)), output.
; PDER = Array, (NPOINT, NTERMS), of partial derivatives of funct.
;  PDER(I,J) = DErivative of function at ith point with
;  respect to jth parameter.  Optional output parameter.
;  PDER should not be calculated if parameter is not
;  supplied in call (Unless you want to waste some time).
; RESTRICTIONS:
; NONE.
; PROCEDURE:
; Copied from "CURFIT", least squares fit to a non-linear
; function, pages 237-239, Bevington, Data Reduction and Error
; Analysis for the Physical Sciences.
;
; "This method is the Gradient-expansion algorithm which
; compines the best features of the gradient search with
; the method of linearizing the fitting function."
;
; Iterations are perform until the chi square changes by
; only 0.1% or until 20 iterations have been performed.
;
; The initial guess of the parameter values should be
; as close to the actual values as possible or the solution
; may not converge.
;
```

```
; MODIFICATION HISTORY:
; Written, DMS, RSI, September, 1982.
; Modified, to output covariance matrix and make final function
; call, Amara Graps, August 1987
;
 ;---------------------------------------------------------- -------------
 ON_ERROR,2  ;RETURN TO CALLER IF ERROR
 A = 1.*A  ;MAKE PARAMS FLOATING
 NTERMS = N_ELEMENTS(A) ;# OF PARAMS.
 NFREE = (N_ELEMENTS(Y)<N_ELEMENTS(X))-NTERMS ;Degs of freedom
 IF NFREE LE 0 THEN STOP,'Curvefit - not enough points.'
 FLAMBDA = 0.001  ;Initial lambda
 DIAG = INDGEN(NTERMS)*(NTERMS+1) ;SUBSCRIPTS OF DIAGONAL ELEMENTS
;
 FOR ITER = 1,20 DO BEGIN ;Iteration loop
;
;  EVALUATE ALPHA AND BETA MATRICIES.
;
 FUNCT,X,A,YFIT,PDER ;COMPUTE FUNCTION AT A.
 BETA = (Y-YFIT)*W # PDER
 ALPHA = TRANSPOSE(PDER) # (W # (FLTARR(NTERMS)+1)*PDER)
;
 CHISQ1 = TOTAL(W*(Y-YFIT)^2)/NFREE ;PRESENT CHI SQUARED
;
; INVERT MODIFIED CURVATURE MATRIX TO FIND NEW PARAMETERS.
;
 REPEAT BEGIN
  C = SQRT(ALPHA(DIAG) # ALPHA(DIAG))
  ARRAY = ALPHA/C
  ARRAY(DIAG) = (1.+FLAMBDA)
  ARRAY = INVERT(ARRAY)
  B = A+ ARRAY/C # TRANSPOSE(BETA) ;NEW PARAMS
  FUNCT,X,B,YFIT ;EVALUATE FUNCTION
  CHISQR = TOTAL(W*(Y-YFIT)^2)/NFREE ;NEW CHISQR
  FLAMBDA = FLAMBDA*10. ;ASSUME FIT GOT WORSE
  ENDREP UNTIL CHISQR LE CHISQ1
;
       FLAMBDA = FLAMBDA/100. ;DECREASE FLAMBDA BY FACTOR OF 10
 A=B   ;SAVE NEW PARAMETER ESTIMATE.
 PRINT,'ITERATION =',ITER,' ,CHISQR =',CHISQR
 PRINT,A
 IF ((CHISQ1-CHISQR)/CHISQ1) LE .001 THEN GOTO,DONE ;Finished?
 ENDFOR   ;ITERATION LOOP
;
 PRINT,'CURVEFIT - Failed to converge'
;
DONE: FUNCT,X,A,YFIT,PDER
 ALPHA = TRANSPOSE(PDER) # (W # (FLTARR(NTERMS)+1)*PDER)
```

```
 COVAR = INVERT(ALPHA)
 SIGMAA = SQRT(COVAR(DIAG))  ;RETURN SIGMA'S

END
 ------------------------------------------------------------ ------

--

 ************************************************************ *************
```

Amara Graps email: agraps@netcom.com
Computational Physicist vita: finger agraps@sunshine.arc.nasa.gov
Intergalactic Reality bio: finger -lm agraps@netcom.com

```
 ************************************************************ *************
```

"Awaken the mind without fixing it anywhere." --Kungo Kyo