Subject: Re: curvefit hang-ups ?????
Posted by edors on Tue, 03 Nov 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Hello Jens,

I am not sure if this is the right answer to your question but...here is a long-winded explaination of what I think is going wrong. I have noticed that sometimes the fitting function which curvefit uses evaluates to !values.f_nan, or !values.f_infinity. This throws a curveball to curvefit [pun intended :)]. I have modified curvefit to properly return to the caller under this circumstance. To implement this I have redefined calling procedure of both the fitting funciton and curvefit itself such that they return 0 or 1 on unsucessful or successful evaluation respectively. Here is an example fitting function (which is now an IDL function instead of a procedure).

```
FUNCTION TempFunctMaxwML,  x, a, f, pder

retval=1

ex1 = exp(-a(1)*x)
f = a(0)*ex1

dumb = where(finite(f) eq 0, num_bad)
IF ( num_bad GT 0 ) THEN BEGIN
   print, 'FitPassMaxwML: error in function evaluation!'
   retval=0
ENDIF ELSE BEGIN
   IF n_params() GE 4 THEN BEGIN
      pder = [[ex1], $
            [-a(0)*ex1*x]]
   ENDIF
ENDELSE

return, retval
END
```

I would love to share a copy of my modified version of curvefit with you, however, the original code was copyrighted by RSI and I am not sure if that puts restrictions on my modified code. In my personal opinion, their copyrighting of curvefit makes them look like big weenies because as they say in *their* documentation:
[begin quote]
; Copied from "CURFIT", least squares fit to a non-linear
; function, pages 237-239, Bevington, Data Reduction and Error
; Analysis for the Physical Sciences.
[end quote]

As it turns out, my modifications were made to an earlier version curvefit from IDL v3.x, so maybe it wouldn't be so helpful anyway. Here is a list of changes that I think should be made to curvefit.

1. Change all of their "call_procedure, function_name, z, a, yfit" lines to "IF(call_FUNCTION(Function_name, x, a, yfit) EQ 0) THEN return, 0".

2. Make yfit a keyword parameter (add it to the parameter list as yfit=yfit).

3. There is a line of code where the current value of chisqr is compared to a number which is considered to be a very small value of chisqr for the number of degrees of freedom in the problem at hand. Sometimes this can be a problem if you do not have a good absolute characterization of the errors (but do have a good sense of the relative errors). You may want to remove this line from curvefit if it gives you problems, I don't think that is ever much of a help, it does not provide a good measure of convergence.

4. Return the curvature matrix (hessian) via a keyword parameter to the caller. curvature = transpose(pder) # (w # (fltarr(nterms)+1)*pder) This matrix can help you extimate the errors in your fit parameters (in the limit that the chisqare surface at is approxamately linear around its minimum value). For a two dimensional fit, as in my example above you can obtain the variance of the fit parameters. This in turn, can be propigated to the errors in physical parameters such as density and temperature, if you were modeling a Maxwellian gas with this fit, for example. (See Press et al. "Numerical Recipies in C", chap. 15, especially section 15.6. Here is the joint variance at the 68% confidence level for a two parameter fit:

```
cm = curvature*2.       ;the hessian matrix
variance(0) = sqrt(cm(1,1)*2.30/(cm(0,0)*cm(1,1)-cm(0,1)^2))
variance(1) = sqrt(cm(0,0)*2.30/(cm(0,0)*cm(1,1)-cm(0,1)^2))
```

good luck,

Eric

```
--
 ============================================================ =====
| Eric E. Dors                  | SMTP: edors@lanl.gov        |
| Los Alamos National Laboratory | WWW: http://nis-www.lanl.gov |
 ============================================================ =====
```