
Subject: Re: Robust curve fitting
Posted by [address](#) on Thu, 06 Aug 1998 07:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

In article <ond8ag5et2.fsf@cow.physics.wisc.edu>,
craigmnet@astro.physics.wisc.edu wrote:

> Mark Elliott <mark@mail.mmrrcc.upenn.edu> writes:
>>
>> Do you or anyone reading this know if there are similar IDL (or
>> MINPACK) routines which perform Levenberg-Marquardt fitting to COMPLEX
>> functions?
>>
>
> I'm not an expert in the field, I just translated the program!
>
> I can tell you that MPFIT itself does not understand complex
> variables; they have to be either FLOAT or DOUBLE. I am not even sure
> what the least-squares problem means when you talk about complex
> numbers. If you want to minimize the Euclidean distance between data
> and model points on the complex plane, and if your data have
> independent errors in the real and imaginary components, then the
> solution should be easy.
>

Hi,

We routinely fit complex variables by putting the real part in the first part of an array and the imaginary part after that. We calculate the function and the derivatives the same way, real part first then the imaginary. See the added example which we use with a slightly adapted curvefit. You just consider the imaginary part as another set of data, since -indeed- in our case the errors are independent. Good luck.

```
*****  
;  
;+  
; CALLING SEQUENCE:  
; FUNCT, f_nr, X, FPAR, FUN, PDER  
; INPUTS:  
;   f_nr = function nr 1: single complex spectral line  
;   T   = VALUES OF INDEPENDENT VARIABLE. (time here)  
;   FPAR = PARAMETERS OF EQUATION DESCRIBED BELOW.  
;   X   = PARAMETERS (depends on function nr)  
; OUTPUTS:  
;   FUN = VALUE OF FUNCTION AT EACH X(I).  
;  
; OPTIONAL OUTPUT PARAMETERS:  
;   DFUN = (N_ELEMENTS(X), npar) ARRAY CONTAINING THE
```

```

; PARTIAL DERIVATIVES. DFUN(I,J) = DERIVATIVE
; AT ITH POINT W/RESPECT TO JTH PARAMETER.
; PROCEDURE:
; function nr = 2:
; FUN(0:cut-1) = sum(i) (A1i sin((w0+wi)t) + A2i cos((w0+wi)t)) exp(-bt)
; FUN(cut:* ) = sum(i) (A1i cos((w0+wi)t) - A2i sin((w0+wi)t)) exp(-bt)
;
; MODIFICATION HISTORY:
; WRITTEN, DMS, RSI, SEPT, 1982.
; Jan Willem van der Veen, jan 1998
;-
PRO FUNCT, f_nr,T,FPAR,FUN,DFUN

```

```

s = N_ELEMENTS(T)
NTERMS = N_ELEMENTS(FPAR)

```

```

cut = s/2

```

```

type = size(FPAR)
type = type(type(0)+1)
dpar = type eq 5

```

```

if dpar then begin
    FUN = dblarr(s)
    IF (N_PARAMS(0) GT 4) THEN DFUN = dblarr(s, nterms)
endif else begin
    FUN = fltarr(s)
    IF (N_PARAMS(0) GT 4) THEN DFUN = fltarr(s, nterms)
endelse

```

```

if (f_nr EQ 1) then begin
    A1 = FPAR(0)      ; amplitude cos 1
    A2 = FPAR(1)      ; amplitude sin 1
    B  = FPAR(2)      ; damping
    W  = FPAR(3)      ; frequency
    T2 = T(0:cut-1)

```

```

    BEXP = EXP(-B*T2)    ; decay
    FS   = SIN(W*T2)     ; sin part
    FC   = COS(W*T2)     ; cos part

```

```

    FUN(0:cut-1) = ( A1*FS+A2*FC )*BEXP
    FUN(cut:s-1) = ( A1*FC-A2*FS )*BEXP

```

```

    IF N_PARAMS(0) LE 4 THEN RETURN    ; need derivatives

```

```

    DFUN(0:cut-1,0) = FS*BEXP    ; d FUN / d A11
    DFUN(cut:s-1,0) = FC*BEXP

```

```
DFUN(0:cut-1,1) = FC*BEXP    ; d FUN / d A21
DFUN(cut:s-1,1) = -FS*BEXP
```

```
DFUN(0:cut-1,2) = FUN(0:cut-1)*(-T2)
DFUN(cut:s-1,2) = FUN(cut:s-1)*(-T2)
```

```
DFUN(0:cut-1,3) = ( A1*FC-A2*FS)*T2*BEXP
DFUN(cut:s-1,3) = (-A1*FS-A2*FC)*T2*BEXP
```

```
endif
RETURN
END
```

```
--
    \\\|/    email: prlkovsky at newsguy dot com
    >|- o|<   I was never sure about anything anyway...
-----oo0 U -Ooo-----
```
