Subject: GAUSS_FUNCT problem
Posted by Michael Wallace on Fri, 27 Feb 2004 17:11:15 GMT
View Forum Message <> Reply to Message

I have discovered something interesting in the RSI-provided gaussfit.pro. I hesitate to call this a "bug," but I don't know what else to call it. Hopefully, it's just something stupid I'm doing. I have copied the pertinent documentation and code below.

```
; CALLING SEQUENCE:
;   FUNCT,X,A,F,PDER
; INPUTS:
;   X = VALUES OF INDEPENDENT VARIABLE.
;   A = PARAMETERS OF EQUATION DESCRIBED BELOW.
; OUTPUTS:
;   F = VALUE OF FUNCTION AT EACH X(I).
;
; OPTIONAL OUTPUT PARAMETERS:
;   PDER = (N_ELEMENTS(X),6) ARRAY CONTAINING THE
;       PARTIAL DERIVATIVES.  P(I,J) = DERIVATIVE
;       AT ITH POINT W/RESPECT TO JTH PARAMETER.

; PROCEDURE:
;   F = A(0)*EXP(-Z^2/2) + A(3) + A(4)*X + A(5)*X^2
;   Z = (X-A(1))/A(2)
;   Elements beyond A(2) are optional.

PRO GAUSS_FUNCT,X,A,F,PDER
    COMPILE_OPT idl2, hidden
    ON_ERROR,2                      ;Return to caller if an error occurs
    n = n_elements(a)
    if a[2] ne 0.0 then begin
       Z = (X-A[1])/A[2]   ;GET Z
       EZ = EXP(-Z^2/2.)   ;GAUSSIAN PART
    endif else begin
       z = 100.
       ez = 0.0
    endelse

    case n of
       3:  F = A[0]*EZ
       4:  F = A[0]*EZ + A[3]
       5:  F = A[0]*EZ + A[3] + A[4]*X
       6:  F = A[0]*EZ + A[3] + A[4]*X + A[5]*X^2 ;FUNCTIONS.
    ENDCASE
```

The variable X is an array of values. In the case where A[2] is not equal to 0, Z and EZ are created as arrays. However, when A[2] is equal to 0, Z and EZ are created as scalars. Then the value of F is computed by using EZ. If EZ is an array, F is an array. If EZ is scalar, F is scalar. F should be the value of the function of each X[I]. F should be an array the same size as X, but when A[2] is equal to 0, F is just a single scalar value. The problem is that by the contract of procedure states that F will be an array. And GAUSSFIT itself expects an array, and this causes problems!!

Of course, the "fix" to this is to make Z an array of N elements with each element set to 100 and EZ an array of N elements with each element set to 0 in the case where A[2] is equal to 0. This ensures that F is always an array.

So, is this a real error that needs to get fixed or am I imagining things?

-Mike

Subject: Re: GAUSS_FUNCT problem
Posted by David Fanning on Mon, 01 Mar 2004 02:53:36 GMT
View Forum Message <> Reply to Message

Michael Wallace writes:

> It's really not the programming that bothers me. It's that the
> documentation doesn't match what the procedure does in all cases. Back
> in school, we'd get big points taken off if our inputs and outputs
> didn't match up with the documentation. So, I learned to be careful
> about how I document things! ;-)

Yeah, but I'm not sure you ever worked as a technical writer in a software company. Let's just say this isn't the most glorified job there is, and these people do remarkable work given the constraints of their job. I for one am willing to cut them some slack. Especially in a situation like this one.

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: GAUSS_FUNCT problem
Posted by Craig Markwardt on Mon, 01 Mar 2004 09:13:50 GMT
View Forum Message <> Reply to Message

Michael Wallace <mwallace.removethismunge@swri.edu.invalid> writes:

>> Well, I can half-heartedly defend the existing code. Note that if one
>> supplies 5 or 6 terms (linear or quadratic background) then GAUSS_FUNCT
>> properly returns an array when A[2] = 0. In the case of 3 terms you
>> are computing a function which only consists of a Gaussian with a sigma
>> width of 0, which probably indicates that you have made an earlier
>> mistake. So I don't begrudge GAUSS_FUNCT returning an anomalous result.
>
> I understand your point, however the documentation clearly states that
> an array will be returned in *all* cases. This particular case, no
> matter how improbable or illogical it may be, is an allowable input. My
> issue isn't so much about the behavior of the procedure, but rather that
> the documentation doesn't match what the procedure does in every case.

Michael, I understand where you are coming from, and you are right,
GAUSS_FUNCT is wrong. (*) I can see that in the course of fitting with
GAUSSFIT, it's quite possible that the "sigma" parameter might head
towards zero, but I'm at a loss as to why it would get stuck right at
zero... unless you set it there to begin with? "Doctor it hurts when
I do this." "Then don't do that."

(*) I think this is a dog food problem. Namely, that RSI is not
eating enough of its own dog food, so it isn't finding its own bugs.
[For non-nerds, "eating your own dogfood" = "using your own code"]

I can also gently direct you to MPFITPEAK, which is a plug in
replacement for GAUSSFIT, but doesn't have any of that ugly RSI code
in it, :-) and is built on the robust MPFIT fitting engine.

Craig

http://cow.physics.wisc.edu/~craigm/idl/idl.html (under fitting)