Subject: Re: GAUSS_FUNCT problem
Posted by David Fanning on Fri, 27 Feb 2004 17:24:08 GMT
View Forum Message <> Reply to Message

Michael Wallace writes:

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

Are we voting!? I either recuse myself (I'm looking the word
up right now) or I assign my proxy to Craig. :-)

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 Wayne Landsman on Fri, 27 Feb 2004 18:37:10 GMT
View Forum Message <> Reply to Message

```
> 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
>
```

> 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.

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.

But yes, I agree that EZ should be set to an zero array of N elements in case A[2] = 0 (though it should probably be set to NaN wherever X = A[1]).

--Wayne Landsman

Subject: Re: GAUSS_FUNCT problem
Posted by Michael Wallace on Mon, 01 Mar 2004 02:23:22 GMT
View Forum Message <> Reply to Message

> 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.

The other problem is that this is a helper procedure for gaussfit and the gaussfit code always expects an array returned. Needless to say, gaussfit chokes on this, even though the inputs are valid according to the documentation.

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! ;-)

-Mike
