Subject: Re: gaussfit
Posted by Robert Moss on Tue, 25 Jun 1996 07:00:00 GMT
View Forum Message <> Reply to Message

Frank Molster wrote:
>
> Hi everyone,
>
> I have encountered a small problem, I hope somebody can help me.
> I am not an experienced user but to me it seems that the keyword NTERMS
> in gaussfit doesn't work. I've tried several possibilities but nothing
> works. If I'm correct the next command should work, shouldn't it?
>
> IDL> yfit = gaussfit(x,y,A,nterms=4)
> % Compiled module: GAUSSFIT.
> % Procedure/function called with too many parameters: GAUSSFIT.
> % Execution halted at: $MAIN$
>
> It works correctly without the 'nterms=4' option.
> Is this a (known) bug or do I something wrong?
> Can anybody help me?

A quick look at the file idl_4/lib/gaussfit.pro shows that the function gaussfit does not take any keyword parameters. Looks like the documentation was updated, but the revised routine didnt make it into the distribution. I'm running IDL v4.0.1 by the way...

--
Robert M. Moss, Ph.D. - mossrm@texaco.com - FAX (713)954-6911
---------- -----
This does not necessarily reflect the opinions of Texaco Inc.

Subject: Re: gaussfit
Posted by Walid on Mon, 01 Jul 1996 07:00:00 GMT
View Forum Message <> Reply to Message

Frank Molster wrote:
>
> Hi everyone,
>
> I have encountered a small problem, I hope somebody can help me.
> I am not an experienced user but to me it seems that the keyword NTERMS
> in gaussfit doesn't work. I've tried several possibilities but nothing
> works. If I'm correct the next command should work, shouldn't it?
>
> IDL> yfit = gaussfit(x,y,A,nterms=4)

> % Compiled module: GAUSSFIT.
> % Procedure/function called with too many parameters: GAUSSFIT.
> % Execution halted at: $MAIN$
>
> It works correctly without the 'nterms=4' option.
> Is this a (known) bug or do I something wrong?
> Can anybody help me?
>
> Thanks a lot,
>
> Frank Molster
>
> --
> _____ _
> | |
> | Frank Molster % Internet: frankm@astro.uva.nl |
> | Astronomical Institute % Phone: (+31 20\020) 5257470 |
> | University of Amsterdam % FAX: (+31 20\020) 5257484 |
> | Kruislaan 403 % |
> | 1098 SJ Amsterdam % |
> |_____ __|

Hi,

I tried to respond earlier to this gaussfit message but something must be wrong with my news server because it never showed up! Anyway, Robert Moss is correct in stating that the new version of gaussfit was never shipped with the latest version (4.01) of IDL. I contacted RSI about this and they gave me the new version, which I am attaching. Also, if you are running IDL on a PC under windows, there is another bug which may rear its ugly head while running gaussfit, which calls curvefit, which calls NR_MACHAR(), a numerical recipes routine that returns machine specific information. This routine hangs the system. The (user-supplied) workaround is to compile a new routine, say, macharr(), and rename nr_machar() in the routine curvefit to macharr(). Macharr() was also given to me by RSI, and I am also attaching it. Hope this helps. Let me add that RSI informed me that this was still an "open" bug which is being given top priority.

Also, I should state that making the above changes worked MOST of the time, but when I tried using gauss2dfit, which calls gaussfit, and used the /TILT option, IDL hangs for my particular data set. If anyone knows how to work around this, please let me know. I will try it on a SUN station and see if it hangs there as well. If it does, I'll let RSI know, and hopefully they can do something about it.

Walid

atia@wam.umd.edu

```
; $Id: gaussfit.pro,v 1.3 1995/09/21 17:03:17 dave Exp $

PRO GAUSS_FUNCT,X,A,F,PDER
; NAME:
; GAUSS_FUNCT
;
; PURPOSE:
; EVALUATE THE SUM OF A GAUSSIAN AND A 2ND ORDER POLYNOMIAL
; AND OPTIONALLY RETURN THE VALUE OF IT'S PARTIAL DERIVATIVES.
; NORMALLY, THIS FUNCTION IS USED BY CURVEFIT TO FIT THE
; SUM OF A LINE AND A VARYING BACKGROUND TO ACTUAL DATA.
;
; CATEGORY:
; E2 - CURVE AND SURFACE FITTING.
; 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.
; COMMON BLOCKS:
; NONE.
; SIDE EFFECTS:
; NONE.
; RESTRICTIONS:
; NONE.
; 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.
; MODIFICATION HISTORY:
; WRITTEN, DMS, RSI, SEPT, 1982.
; Modified, DMS, Oct 1990.  Avoids divide by 0 if A(2) is 0.
; Added to Gauss_fit, when the variable function name to
;  Curve_fit was implemented.  DMS, Nov, 1990.
;
 n = n_elements(a)
 ON_ERROR,2                      ;Return to caller if an error occurs
 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

 IF N_PARAMS(0) LE 3 THEN RETURN ;NEED PARTIAL?
;
 PDER = FLTARR(N_ELEMENTS(X),n) ;YES, MAKE ARRAY.
 PDER(0,0) = EZ  ;COMPUTE PARTIALS
 if a(2) ne 0. then PDER(0,1) = A(0) * EZ * Z/A(2)
 PDER(0,2) = PDER(*,1) * Z
 if n gt 3 then PDER(*,3) = 1.
 if n gt 4 then PDER(0,4) = X
 if n gt 5 then PDER(0,5) = X^2
 RETURN
END



Function Gaussfit, x, y, a, NTERMS=nt
;+
; NAME:
; GAUSSFIT
;
; PURPOSE:
;  Fit the equation y=f(x) where:
;
;   F(x) = A0*EXP(-z^2/2) + A3 + A4*x + A5*x^2
;    and
;  z=(x-A1)/A2
;
; A0 = height of exp, A1 = center of exp, A2 = sigma (the width).
; A3 = constant term, A4 = linear term, A5 = quadratic term.
; Terms A3, A4, and A5 are optional.
;  The parameters A0, A1, A2, A3 are estimated and then CURVEFIT is
; called.
;
; CATEGORY:
; ?? - fitting
;
; CALLING SEQUENCE:
```

```
; Result = GAUSSFIT(X, Y [, A])
;
; INPUTS:
; X: The independent variable.  X must be a vector.
; Y: The dependent variable.  Y must have the same number of points
;  as X.
; KEYWORD INPUTS:
; NTERMS = Set NTERMS to 3 to compute the fit: F(x) = A0*EXP(-z^2/2).
;    Set it to 4 to fit:  F(x) = A0*EXP(-z^2/2) + A3
;    Set it to 5 to fit:  F(x) = A0*EXP(-z^2/2) + A3 + A4*x
;
; OUTPUTS:
; The fitted function is returned.
;
; OPTIONAL OUTPUT PARAMETERS:
; A: The coefficients of the fit.  A is a three to six
;  element vector as described under PURPOSE.
;
; COMMON BLOCKS:
; None.
;
; SIDE EFFECTS:
; None.
;
; RESTRICTIONS:
; The peak or minimum of the Gaussian must be the largest
; or smallest point in the Y vector.
;
; PROCEDURE:
; If the (MAX-AVG) of Y is larger than (AVG-MIN) then it is assumed
; that the line is an emission line, otherwise it is assumed there
; is an absorbtion line.  The estimated center is the MAX or MIN
; element.  The height is (MAX-AVG) or (AVG-MIN) respectively.
; The width is found by searching out from the extrema until
; a point is found less than the 1/e value.
;
; MODIFICATION HISTORY:
; DMS, RSI, Dec, 1983.
; DMS, RSI, Jun, 1995, Added NTERMS keyword.  Result is now float if
;    Y is not double.
;-
;
on_error,2                      ;Return to caller if an error occurs
if n_elements(nt) eq 0 then nt = 6
if nt lt 3 or nt gt 6 then $
   message,'NTERMS must have values from 3 to 6.'
n = n_elements(y)  ;# of points.
s = size(y)
```

```
c = poly_fit(x,y,1,yf)  ;Fit a straight line
yd = y - yf
if s(s(0)+1) ne 5 then begin ;If Y is not double, use float

    yd = float(yd) & c = float(c) & endif

ymax=max(yd) & xmax=x(!c) & imax=!c ;x,y and subscript of extrema
ymin=min(yd) & xmin=x(!c) & imin=!c

if abs(ymax) gt abs(ymin) then i0=imax else i0=imin ;emiss or absorp?
i0 = i0 > 1 < (n-2)  ;never take edges
dy=yd(i0)   ;diff between extreme and mean
del = dy/exp(1.)  ;1/e value
i=0
while ((i0+i+1) lt n) and $ ;guess at 1/2 width.
 ((i0-i) gt 0) and $
 (abs(yd(i0+i)) gt abs(del)) and $
 (abs(yd(i0-i)) gt abs(del)) do i=i+1
a = [yd(i0), x(i0), abs(x(i0)-x(i0+i))]
if nt gt 3 then a = [a, c(0)] ;estimates
if nt gt 4 then a = [a, c(1)]
if nt gt 5 then a = [a, 0.]
!c=0    ;reset cursor for plotting
return,curvefit(x,y,replicate(1.,n),a,sigmaa, $
  function_name = "GAUSS_FUNCT") ;call curvefit
end


function MACHARR, DOUBLE=double

    ; Values were obtained by executing Numerical Recipes
    ; example program XMACHAR (calls NR routine MACHAR).
    if keyword_set(double) then $
        return, {beta:2L, it:64L, irnd:3L, ngrd:1L, $
                machep:-52L, negep:-53L, iexp:11L, $
                minexp:-1022L, maxexp:1024L, $
                eps: 0.222044604925031D-015, $
                epsneg:0.111022302462516D-015, $
                xmin:0.222507385850720D-307, $
                xmax:0.179769313486232D+39 }   $
    else $
        return, {beta:2L, it:64L, irnd:3L, ngrd:1L, $
                machep:-23L, negep:-24L, iexp:8L, $
                minexp:-126L, maxexp:128L, $
                eps: 0.119209E-06, epsneg:0.596046E-07, $
                xmin:0.117549E-37,xmax:0.340282E+39 }
end
```

File Attachments
1) gaussfit.pro, downloaded 104 times
2) Macharr.pro, downloaded 86 times

Subject: Re: gaussfit
Posted by Karl Young on Tue, 02 Jul 1996 07:00:00 GMT
View Forum Message <> Reply to Message

Walid,

> ...if you are running IDL on a PC under windows, there is another bug which
> may rear its ugly head while running gaussfit, which calls curvefit,
> which calls NR_MACHAR(), a numerical recipes routine that returns
> machine specific information. This routine hangs the system. The
> (user-supplied) workaround is to compile a new routine, say, macharr(),
> and rename nr_machar() in the routine curvefit to macharr(). Macharr()
> was also given to me by RSI, and I am also attaching it. Hope this
> helps. Let me add that RSI informed me that this was still an "open" bug
> which is being given top priority...
>

Amazing; I was just going to report this problem again and ask for help, and here your message was. I had reported the problem of machar() hanging my NT machine previously but nobody responded. Since at the time I didn't really need machar() for anything crucial I just forgot about it; but now I need to use curevefit so it's come up again. Anyway just wanted to add my thanks for your posting this and register my hope that RSI fixes this in the next release.

-- Karl Young

Subject: Re: gaussfit
Posted by Frank Molster on Thu, 04 Jul 1996 07:00:00 GMT
View Forum Message <> Reply to Message

Walid wrote:
>
> Also, I should state that making the above changes worked MOST of the
> time, but when I tried using gauss2dfit, which calls gaussfit, and used
> the /TILT option, IDL hangs for my particular data set. If anyone knows
> how to work around this, please let me know. I will try it on a SUN
> station and see if it hangs there as well. If it does, I'll let RSI
> know, and hopefully they can do something about it.
>
> Walid
>

> atia@wam.umd.edu
>
Walid and Robert thank you very much.
I also got a message from CREASO (idl technical support for a couple of countries in Europe).
They confirmed your opinions.
They send me the updated version of gaussfit, which will be probably the same as Walid posted (I didn't checked it), but also an updated version
of GAUSS2DFIT. This might help you Walid.
I will attach it to this message.

Frank

==========begin GAUSS2DFIT.PRO==========

```
; $Id: gauss2dfit.pro,v 1.1 1995/07/06 22:53:34 dave Exp $
PRO GAUSS2_FUNCT, X, A, F, PDER
; NAME:
; GAUSS2_FUNCT
; PURPOSE:
; Evaluate function for gauss2fit.
; CALLING SEQUENCE:
; FUNCT,X,A,F,PDER
; INPUTS:
; X = values of independent variables, encoded as: [nx, ny, x, y]
; A = parameters of equation described below.
; OUTPUTS:
; F = value of function at each X(i,j), Y(i,j).
; Function is:
;  F(x,y) = A0 + A1*EXP(-U/2)
;  where: U= (yp/A2)^2 + (xp/A3)^2
;
;   If A has 7 elements a rotation of the ellipse is present and:
;  xp = (x-A4) * cos(A6) - (y-A5) * sin(A6)
;  yp = (x-A4) * sin(A6) + (y-A5) * cos(A6)
;   If A has 6 elements, A6 (theta) is 0, the major and minor axes
;   of the ellipse are parallel to the XY axes, and:
;  xp = (x-A4)   and   yp = (x-A5)
;
; Optional output parameters:
; PDER = (n_elements(z),6 or 7) array containing the
;  partial derivatives.  pder(i,j) = derivative
;  at ith point w/respect to jth parameter.
; PROCEDURE:
; Evaluate the function and then if requested, eval partials.
;
; MODIFICATION HISTORY:
```

```
; WRITTEN, DMS, RSI, June, 1995.
;

nx = long(x(0))  ;Retrieve X and Y vectors
ny = long(x(1))

tilt = n_elements(a) eq 7 ;TRUE if angle present.
if tilt then begin  ;Rotate?
    xp = (x(2:nx+1)-a(4)) # replicate(1.0, ny) ;Expand X values
    yp = replicate(1.0, nx) # (x(nx+2:*)-a(5)) ;expand Y values
    s = sin(A(6)) & c = cos(A(6))
    t =  xp * (c/a(2)) - yp * (s/a(2))
    yp = xp * (s/a(3)) + yp * (c/a(3))
    xp = temporary(t)
endif else begin
    xp = (x(2:nx+1)-a(4)) # replicate(1.0/a(2), ny) ;Expand X values
    yp = replicate(1.0/a(3), nx) # (x(nx+2:*)-a(5)) ;expand Y values
    s = 0.0 & c = 1.0
endelse

n = nx * ny
u = reform(exp(-0.5 * (xp^2 + yp^2)), n) ;Exp() term, Make it 1D
F = a(0) + a(1) * u

if n_params(0) le 3 then return ;need partial?  No.

PDER = FLTARR(n, n_elements(a)) ;YES, make partial array.
PDER(*,0) = 1.0   ;And fill.
pder(0,1) = u
u = a(1) * u   ;Common term for the rest of the partials
pder(0,2) = u * xp^2 / a(2)
pder(0,3) = u * yp^2 / a(3)
pder(0,4) = u * (c/a(2) * xp + s/a(3) * yp)
pder(0,5) = u * (-s/a(2) * xp + c/a(3) * yp)
if tilt then pder(0,6) = u * xp*yp*(a(2)/a(3)-a(3)/a(2))
END



Function Gauss2dfit, z, a, x, y, NEGATIVE = neg, TILT=tilt
;+
; NAME:
; GAUSS2DFIT
;
; PURPOSE:
;  Fit a 2 dimensional elliptical gaussian equation to rectilinearly
; gridded data.
;  Z = F(x,y) where:
```

```
;   F(x,y) = A0 + A1*EXP(-U/2)
;    And the elliptical function is:
;  U= (x'/a)^2 + (y'/b)^2
; The parameters of the ellipse U are:
;    Axis lengths are 2*a and 2*b, in the unrotated X and Y axes,
;  respectively.
;    Center is at (h,k).
;    Rotation of T radians from the X axis, in the CLOCKWISE direction.
;    The rotated coordinate system is defined as:
;  x' = (x-h) * cos(T) - (y-k) * sin(T)  <rotate by T about (h,k)>
;  y' = (x-h) * sin(T) + (y-k) * cos(T)
;
; The rotation is optional, and may be forced to 0, making the major/
; minor axes of the ellipse parallel to the X and Y axes.
;
; The coefficients of the function, are returned in a seven
; element vector:
; a(0) = A0 = constant term.
; a(1) = A1 = scale factor.
; a(2) = a = width of gaussian in X direction.
; a(3) = b = width of gaussian in Y direction.
; a(4) = h = center X location.
; a(5) = k = center Y location.
; a(6) = T = Theta the rotation of the ellipse from the X axis
;  in radians, counterclockwise.
;
;
; CATEGORY:
; curve / data fitting
;
; CALLING SEQUENCE:
; Result = GAUSS2DFIT(z, a [,x,y])
;
; INPUTS:
; Z = dependent variable in a 2D array dimensioned (Nx, Ny).  Gridding
;  must be rectilinear.
; X = optional Nx element vector containing X values of Z.  X(i) = X value
;  for Z(i,j).  If omitted, a regular grid in X is assumed,
;  and the X location of Z(i,j) = i.
; Y = optional Ny element vector containing Y values of Z.  Y(j) = Y value
;  for Z(i,j).  If omitted, a regular grid in Y is assumed,
;  and the Y location of Z(i,j) = j.
;
; Optional Keyword Parameters:
; NEGATIVE = if set, implies that the gaussian to be fitted
;  is a valley (such as an absorption line).
;  By default, a peak is fit.
; TILT = if set to  1, allow the orientation of the major/minor axes of
```

```
;  the ellipse to be unrestricted.  The default is that
;  the axes of the ellipse must be parallel to the X-Y axes.
;  In this case, A(6) is always returned as 0.
;
; OUTPUTS:
; The fitted function is returned.
; OUTPUT PARAMETERS:
; A: The coefficients of the fit.  A is a seven element vector as
;  described under PURPOSE.
;
; COMMON BLOCKS:
; None.
; SIDE EFFECTS:
; None.
; RESTRICTIONS:
; Timing:  Approximately 4 seconds for a 128 x 128 array, on a
;  Sun SPARC LX.  Time required is roughly proportional to the
;  number of elements in Z.
;
; PROCEDURE:
; The peak/valley is found by first smoothing Z and then finding the
; maximum or minimum respectively.  Then GAUSSFIT is applied to the row
; and column running through the peak/valley to estimate the parameters
; of the Gaussian in X and Y.  Finally, CURVEFIT is used to fit the 2D
; Gaussian to the data.
;
; Be sure that the 2D array to be fit contains the entire Peak/Valley
; out to at least 5 to 8 half-widths, or the curve-fitter may not
; converge.
;
; EXAMPLE:  This example creates a 2D gaussian, adds random noise
; and then applies GAUSS2DFIT:
; nx = 128  ;Size of array
; ny = 100
; ;**  Offs Scale X width Y width X cen Y cen  **
; ;**   A0  A1    a       b       h       k    **
; a = [ 5., 10., nx/6.,  ny/10., nx/2., .6*ny]  ;Input function parameters
; x = findgen(nx) # replicate(1.0, ny) ;Create X and Y arrays
; y = replicate(1.0, nx) # findgen(ny)
; u = ((x-a(4))/a(2))^2 + ((y-a(5))/a(3))^2  ;Create ellipse
; z = a(0) + a(1) * exp(-u/2)  ;to gaussian
; z = z + randomn(seed, nx, ny)  ;Add random noise, SD = 1
; yfit = gauss2dfit(z,b)   ;Fit the function, no rotation
; print,'Should be:',string(a,format='(6f10.4)')  ;Report results..
; print,'Is:      :',string(b(0:5),format='(6f10.4)')
;
; MODIFICATION HISTORY:
; DMS, RSI, June, 1995.
```

```
;-
;
on_error,2                      ;Return to caller if an error occurs
s = size(z)
if s(0) ne 2 then $
 message, 'Z must have two dimensions'
n = n_elements(z)
nx = s(1)
ny = s(2)
np = n_params()
if np lt 3 then x = findgen(nx)
if np lt 4 then y = findgen(ny)

if nx ne n_elements(x) then $
    message,'X array must have size equal to number of columns of Z'
if ny ne n_elements(y) then $
    message,'Y array must have size equal to number of rows of Z'

if keyword_set(neg) then q = MIN(SMOOTH(z,3), i) $
    ELSE q = MAX(SMOOTH(z,3), i) ;Dirty peak / valley finder
ix = i mod nx
iy = i / nx
x0 = x(ix)
y0 = y(iy)

xfit = gaussfit(x, z(*,iy), ax, NTERMS=4) ;Guess at params by taking slices
yfit = gaussfit(y, z(ix,*), ay, NTERMS=4)

; First guess, without XY term...
a = [ (ax(3) + ay(3))/2., $  ;Constant
 sqrt(abs(ax(0) * ay(0))), $ ;Exponential factor
 ax(2), ay(2), ax(1), ay(1)] ;Widths and centers

;  If there's a tilt, add the XY term = 0
if Keyword_set(tilt) then a = [a, 0.0]

;************* print,'1st guess:',string(a,format='(8f10.4)')
result = curvefit([nx, ny, x, y], reform(z, n, /OVERWRITE), $
  replicate(1.,n), a, itmax=50, $
  function_name = "GAUSS2_FUNCT", /NODERIVATIVE)

; If we didn't already have an XY term, add it = 0.0
if Keyword_set(tilt) eq 0 then a = [a, 0.0] $
else a(6) = a(6) mod !pi  ;Reduce angle argument
z= REFORM(z, nx, ny, /OVERWRITE) ;Restore dimensions
return, REFORM(result, nx, ny, /OVERWRITE)
end
```

==========end GAUSS2DFIT.PRO==========

File Attachments

1) idlproh, downloaded 94 times