
Subject: Re: Text processing on plots
Posted by [thompson](#) on Sat, 15 Feb 1997 08:00:00 GMT
[View Forum Message](#) <> [Reply to Message](#)

Jeff Hicke <jhicke@al.noaa.gov> writes:

> Hello,

> I am interested in whether anyone has any IDL routines which allows one
> to overplot a variable length text string in the manner of a word
> processor. In other words, if I define a box (or width of a box), then
> the routine overplots the text within this box's width. If it is longer
> than one line, it will wrap to a new line. I'm looking at adding figure
> captions to IDL plots. Can anyone help?

I think this does what you want.

Bill Thompson

```
=====
=====
PRO WRITE_IN_BOX, X1, X2, Y1, Y2, TEXT, DATA=DATA, DEVICE=DEVICE, $
  NORMAL=NORMAL, MAXCHARSIZE=MAXCHARSIZE, COLOR=COLOR, $
  ALIGNMENT=ALIGNMENT
;+
; Project   : SOHO - CDS
;
; Name      : WRITE_IN_BOX
;
; Purpose   : Writes a text message within a box in a graphics window.
;
; Explanation : This procedure writes a short text message within a box-shaped
; area in a graphics window. The message may be split at word
; boundaries into several lines, and the character size and
; orientation may be adjusted for the text to fit within the box.
;
; Use       : WRITE_IN_BOX, X1, X2, Y1, Y2, TEXT
;
; Inputs    : X1, X2 = X coordinates of the box limits.
; Y1, Y2 = Y coordinates of the box limits.
; TEXT = ASCII text string containing the message.
;
; Opt. Inputs : None.
;
; Outputs   : None.
;
; Opt. Outputs: None.
;
```

```

; Keywords   : DATA   = If set, then the coordinates are in data units.
;             This is the default.
; DEVICE     = If set, then the coordinates are in device units.
; NORMAL     = If set, then the coordinates are in normalized
;             units.
; MAXCHARSIZE = The maximum character size to use in displaying
;             the message. If not passed, then determined from
;             !P.CHARSIZE.
; COLOR      = Color to use to display the text. The default is
;             !COLOR.
; ALIGNMENT  = Controls the alignment of the text in the box. A
;             value of 0 means to left justify the text, 0.5
;             means center it (default), and 1 stands for right
;             justification.
;
; Calls      : None.
;
; Common     : None.
;
; Restrictions: X2 must be greater than X1, and Y2 must be greater than Y1.
;
; Side effects: The appearance of the displayed message may not be optimal if
; any words are separated by multiple blanks, or by tab
; characters.
;
; Category   : Planning, Science.
;
; Prev. Hist. : None.
;
; Written    : William Thompson, GSFC, 7 July 1993.
;
; Modified   : Version 1, William Thompson, GSFC, 7 July 1993.
;             Version 2, William Thompson, GSFC, 24 September 1993.
;             Added ALIGNMENT keyword based on code provided by Jim
;             Pendleton, GRO/OSSE NU.
;             Version 3, William Thompson, GSFC, 21 September 1994
;             Added STRTRIM call.
;
; Version    : Version 3, 21 September 1994
;-
;
; ON_ERROR, 2
;
; Check the number of parameters.
;
IF N_PARAMS() NE 5 THEN MESSAGE, 'Syntax: X1, X2, Y1, Y2, TEXT'
IF (X1 GE X2) THEN MESSAGE, 'X2 must be greater than X1'
IF (Y1 GE Y2) THEN MESSAGE, 'Y2 must be greater than Y1'

```

```

;
; Select which alignment to use. The default is 0.5 (centered text).
;
IF N_ELEMENTS(ALIGNMENT) NE 1 THEN ALIGNMENT = 0.5
;
; Convert the input parameters to device coordinates.
;
IF KEYWORD_SET(NORMAL) THEN BEGIN
  DEV = CONVERT_COORD([X1,X2],[Y1,Y2],/NORMAL,/TO_DEVICE)
  XX1 = DEV(0,0)
  XX2 = DEV(0,1)
  YY1 = DEV(1,0)
  YY2 = DEV(1,1)
END ELSE IF KEYWORD_SET(DEVICE) THEN BEGIN
  XX1 = X1
  XX2 = X2
  YY1 = Y1
  YY2 = Y2
END ELSE BEGIN
  DEV = CONVERT_COORD([X1,X2],[Y1,Y2],/DATA,/TO_DEVICE)
  XX1 = DEV(0,0)
  XX2 = DEV(0,1)
  YY1 = DEV(1,0)
  YY2 = DEV(1,1)
ENDELSE
;
; Calculate the height and width of the box in characters.
;
WIDTH = (XX2 - XX1) / !D.X_CH_SIZE
HEIGHT = (YY2 - YY1) / !D.Y_CH_SIZE
;
; Decompose the message into words.
;
WORDS = STR_SEP(STRTRIM(STRCOMPRESS(TEXT),2),' ')
;
; Get the maximum character size.
;
IF N_ELEMENTS(MAXCHARSIZE) EQ 0 THEN BEGIN
  IF !P.CHARSIZE GT 0 THEN MAXCHARSIZE = !P.CHARSIZE ELSE $
    MAXCHARSIZE = 1
ENDIF
IF MAXCHARSIZE LE 0 THEN MESSAGE,'MAXCHARSIZE must be positive'
;
; Make two passes. During the first pass, try fitting the text in
; horizontally. During the second pass, try vertically.
;
BEST_SIZE = FLTARR(2)
FOR I_PASS = 0, 1 DO BEGIN

```

```

;
; Starting with the maximum character size, try to fit the text within the
; window. Format the text message into a series of lines, allowing each line
; to grow until it can't fit within the box any more.
;
;
CHARSIZE = MAXCHARSIZE
TRY_SIZE:
  LINES = STRARR(N_ELEMENTS(WORDS))
  I_LINE = 0
  FOR I_WORD = 0,N_ELEMENTS(WORDS)-1 DO BEGIN
    IF STRLEN(LINES(I_LINE)) EQ 0 THEN BEGIN
      LINES(I_LINE) = WORDS(I_WORD)
    END ELSE BEGIN
      TEST = LINES(I_LINE) + ' ' + WORDS(I_WORD)
      IF CHARSIZE*STRLEN(TEST) LE WIDTH THEN BEGIN
        LINES(I_LINE) = TEST
      END ELSE BEGIN
        I_LINE = I_LINE + 1
        LINES(I_LINE) = WORDS(I_WORD)
      ENDELSE
    ENDELSE
  ENDFOR
  LINES = LINES(0:I_LINE)
  N_LINES = N_ELEMENTS(LINES)
;
; Test whether or not the text message will fit in the box. If not, then
; decrease the character size by 30% and try again.
;
IF (CHARSIZE*MAX(STRLEN(LINES)) GT WIDTH) OR $
  (CHARSIZE*N_LINES GT HEIGHT) THEN BEGIN
  CHARSIZE = CHARSIZE * 0.7
  GOTO, TRY_SIZE
ENDIF
;
; Calculate how big the message can be and still fit within the box. Allow a
; little leeway for a margin.
;
CHARSIZE = (WIDTH / (MAX(STRLEN(LINES)) + 1.)) < $
  (HEIGHT / (N_LINES + 1.)) < MAXCHARSIZE
;
; Save the calculated character size, and the actual LINES array. Then try it
; rotated 90 degrees.
;
BEST_SIZE(I_PASS) = CHARSIZE
IF I_PASS EQ 0 THEN LINES_0 = LINES ELSE LINES_90 = LINES
WIDTH = (YY2 - YY1) / !D.X_CH_SIZE
HEIGHT = (XX2 - XX1) / !D.Y_CH_SIZE

```

```

ENDFOR
;
; Get the color to use in displaying the message.
;
IF N_ELEMENTS(COLOR) EQ 0 THEN COLOR = !COLOR
;
; Choose which orientation to display the text in. Give preference to the
; horizontal direction.
;
PREFERENCE = 0.3 * MAX(BEST_SIZE) / MAXCHARSIZE
IF BEST_SIZE(1) GT (1+PREFERENCE)*BEST_SIZE(0) THEN BEGIN
CHARSIZE = BEST_SIZE(1)
ORIENTATION = 90
LINES = LINES_90
END ELSE BEGIN
CHARSIZE = BEST_SIZE(0)
ORIENTATION = 0
LINES = LINES_0
ENDELSE
N_LINES = N_ELEMENTS(LINES)
;
; Display the message. Depending on the value of the ALIGNMENT keyword,
; either left justify (ALIGNMENT=0), center (ALIGNMENT=0.5) or right justify
; (ALIGNMENT=1) each line.
;
FOR I_LINE = 0,N_LINES-1 DO BEGIN
IF ORIENTATION EQ 90 THEN BEGIN
XX = (I_LINE-0.5*N_LINES+0.75)*CHARSIZE*!D.Y_CH_SIZE +$
0.5*(XX1 + XX2)
YY = (1-ALIGNMENT)*YY1 + ALIGNMENT*YY2
END ELSE BEGIN
XX = (1-ALIGNMENT)*XX1 + ALIGNMENT*XX2
YY = (0.5*N_LINES-I_LINE-0.75)*CHARSIZE*!D.Y_CH_SIZE +$
0.5 * (YY1 + YY2)
ENDELSE
XYOUTS, XX, YY, LINES(I_LINE), /DEVICE, CHARSIZE=CHARSIZE, $
ALIGNMENT=ALIGNMENT, COLOR=COLOR, FONT=-1, $
ORIENTATION=ORIENTATION
ENDFOR
;
RETURN
END

```
