Subject: Re: curve labeling program
Posted by f055 on Thu, 08 May 1997 07:00:00 GMT
View Forum Message <> Reply to Message

Mirko Vukovic wrote:
-David Fanning wrote:
->
-> Hi Mirko,
->
-> You write about a curve legend program:
->
-> > If there is not such an animal, I'll probably write it in a day or two.
-> > If interested in it let me know.
->
-> Good luck with this. This is one of those programs (like COLORBAR)
-> that you think will be simple but gets more and more complicated
-> the more you get into it. It is really hard, I think, to write a
-> *general* program.

Attached is legend.pro which I got with IDL 3.0 (I think) in the user_contrib subdirectory, contributed by Fred Knight. I've added an option for different line thicknesses and have played around with the character sizes. It's fairly general though and is what I (almost) always use for this sort of thing.

Cheers

Tim

..... Dr Tim Osborn . t.osborn@uea.ac.uk
.... ___/.. __ /.. /.. /. Senior Research Associate . phone:01603 592089
... /..... /. /.. /.. /.. Climatic Research Unit . fax: 01603 507784
.. /..... __/.. /.. /... School of Environmental Sciences.
. /..... /\ ... /.. /.... University of East Anglia .
____/.._/.._..____/..... Norwich NR4 7TJ .
..... UK .

```
;+
; Name:
; legend
; Purpose:
; This procedure makes a legend for a plot.  The legend can contain
; a mixture of symbols, linestyles, Hershey characters (vectorfont),
; and filled polygons (usersym).
; Examples:
; The call:
;  legend,['Plus sign','Asterisk','Period'],psym=[1,2,3]
; produces:
```

```
;  ----------------
;  |  |
;  |  + Plus sign |
;  |  * Asterisk |
;  |  . Period |
;  |  |
;  ----------------
; Each symbol is drawn with a plots command, so they look OK.
; Other examples are given in usage and in optional output keywords.
; Usage:
;       legend,items,thick=thick
; legend,items,linestyle=linestyle ; vertical legend at upper left
; legend,items,psym=psym    ; ditto except using symbols
; legend,items,psym=psym,/horizontal ; horizontal format
; legend,items,psym=psym,box=0  ; sans border
; legend,items,psym=psym,delimiter='=' ; embed an '=' betw psym & text
; legend,items,psym=psym,margin=2  ; 2-character margin
; legend,items,psym=psym,position=pos ; position of legend
; legend,items,psym=psym,number=2  ; plot two symbols, not one
; legend,items,/fill,psym=[8,8,8],colors=[10,20,30]; 3 filled squares
; Inputs:
; items = text for the items in the legend, a string array.
;  You can omit items if you don't want any text labels.
;  For example, items = ['diamond','asterisk','square'].
; Optional Inputs:
;       thick = array of line thicknesses
; linestyle = array of linestyle numbers  If linestyle(i) < 0, then omit
;  ith symbol or line to allow a multi-line entry.
; psym = array of plot symbol numbers.  If psym(i) is negative, then a
;  line connects pts for ith item.  If psym(i) = 9, then the
;  procedure usersym is called with vertices define in the
;  keyword usersym. psym(i)=8 uses currently defined usersym.
; N. B.: Choose either linestyle, psym, neither, or both.  If neither is
;  present, only the text is output.  If both linestyle and
;  psym parameters are present, they both have to have the
;  same number of elements, and normal plot behaviour occurs.
;  By default, if psym is positive, you get one point so there is
;  no connecting line.
; vectorfont = vector-drawn characters for the sym/line column, e.g.,
;  ['!9B!3','!9C!3','!9D!3'] produces an open square, a checkmark,
;  and a partial derivative, which might have accompanying items
;  ['BOX','CHECK','PARTIAL DERIVATIVE'].  If vectorfont(i) = '',
;  then plots is called to make a symbol or a line, but if
;  vectorfont(i) is a non-null string, then xyouts is called.
;  There is no check that !p.font is set properly, e.g., -1 for
;  X and 0 for PostScript.  This can produce an error, e.g., use
;  !20 with PostScript and !p.font=0, but allows use of Hershey
;  *AND* PostScript fonts together.
```

```
; Optional Keywords:
; /help = flag to print header
; /horizontal = flag to make the legend horizontal
; /vertical = flag to make the legend vertical (D=vertical)
; box = flag to include/omit box around the legend (D=include)
; delimiter = embedded character(s) between symbol and text (D=none)
; colors = array of colors for plot symbols/lines (D=!color)
; textcolors = array of colors for text (D=!color)
; margin = margin around text measured in characters and lines
; spacing = line spacing (D=bit more than character height)
; pspacing = psym spacing (D=3 characters)
; charsize = just like !p.charsize for plot labels
; position = normalized coordinates of the upper left of the legend
; number = number of plot symbols to plot or length of line (D=1)
; usersym = 2-D array of vertices, cf. usersym in IDL manual. (D=square)
; /fill = flag to fill the usersym
; Outputs:
; legend to current plot device
; Optional Output Keywords:
; corners = 4-element array, like !p.position, of the normalized
;   coords for the box (even if box=0): [llx,lly,urx,ury].
;   Useful for multi-column or multi-line legends, for example,
;   to make a 2-column legend, you might do the following:
;     c1_items = ['diamond','asterisk','square']
;     c1_psym = [4,2,6]
;     c2_items = ['solid','dashed','dotted']
;     c2_line = [0,2,1]
;     legend,c1_items,psym=c1_psym,corners=c1,box=0
;      legend,c2_items,line=c2_line,corners=c2,box=0,pos=[c1(2),c1( 3)]
;     c = [c1(0)<c2(0),c1(1)<c2(1),c1(2)>c2(2),c1(3)>c2(3)]
;      plots,[c(0),c(0),c(2),c(2),c(0)],[c(1),c(3),c(3),c(1),c(1)], /norm
;   Useful also to place the legend.  Here's an automatic way to place
;   the legend in the lower right corner.  The difficulty is that the
;   legend's width is unknown until it is plotted.  In this example,
;   the legend is plotted twice: the first time in the upper left, the
;   second time in the lower right.
;      legend,['1','22','333','4444'],linestyle=indgen(4),corners=c orners
;   ; BOGUS LEGEND---FIRST TIME TO REPORT CORNERS
;     xydims = [corners(2)-corners(0),corners(3)-corners(1)]
;   ; SAVE WIDTH AND HEIGHT
;      chdim=[!d.x_ch_size/float(!d.x_size),!d.y_ch_size/float(!d.y _size)]
;   ; DIMENSIONS OF ONE CHARACTER IN NORMALIZED COORDS
;     pos = [!x.window(1)-chdim(0)-xydims(0) $
;   ,!y.window(0)+chdim(1)+xydims(1)]
;   ; CALCULATE POSITION FOR LOWER RIGHT
;     plot,findgen(10) ; SIMPLE PLOT; YOU DO WHATEVER YOU WANT HERE.
;     legend,['1','22','333','4444'],linestyle=indgen(4),pos=pos
;   ; REDO THE LEGEND IN LOWER RIGHT CORNER
```

```
;   You can modify the pos calculation to place the legend where you
;   want.  For example to place it in the upper right:
;     pos = [!x.window(1)-chdim(0)-xydims(0),!y.window(1)-xydims(1)]
; Common blocks:
; none
; Procedure:
; If keyword help is set, call doc_library to print header.
; See notes in the code.
; Restrictions:
; Here are some things that aren't implemented.
; - It would be nice to allow data and device coords as well.
; - An orientation keyword would allow lines at angles in the legend.
; - An array of usersyms would be nice---simple change.
; - An order option to interchange symbols and text might be nice.
; - Somebody might like double boxes, e.g., with box = 2.
; - Another feature might be a continuous bar with ticks and text.
; - There are no guards to avoid writing outside the plot area.
; - There is no provision for multi-line text, e.g., '1st line!c2nd line'
;   Sensing !c would be easy, but !c isn't implemented for PostScript.
;   A better way might be to simply output the 2nd line as another item
;   but without any accompanying symbol or linestyle.  A flag to omit
;   the symbol and linestyle is linestyle(i) = -1.
; - There is no ability to make a title line containing any of titles
;   for the legend, for the symbols, or for the text.
; - It might be nice to force the legend to be placed at hardwired
;   locations in the plot, e.g., with keywords like /left/bottom for
;   lower left.  Allowing this requires knowing the width of the text
;   before it is printed, which is difficult.
; Side Effects:
; Modification history:
; write, 24-25 Aug 92, F K Knight (knight@ll.mit.edu)
; allow omission of items or omission of both psym and linestyle, add
;   corners keyword to facilitate multi-column legends, improve place-
;   ment of symbols and text, add guards for unequal size, 26 Aug 92, FKK
; add linestyle(i)=-1 to suppress a single symbol/line, 27 Aug 92, FKK
; add keyword vectorfont to allow characters in the sym/line column,
;   28 Aug 92, FKK
;       add thick for array of line thicknesses, 14/6/95, Tim Osborn
;-
pro  legend,help=help,items,linestyle=linestyle,psym=psym,vectorf ont=vectorfont $
   ,horizontal=horizontal,vertical=vertical,box=box,margin=marg in $
   ,delimiter=delimiter,spacing=spacing,charsize=charsize,pspac ing=pspacing $
   ,position=position,number=number,colors=colors,textcolors=te xtcolors $
  ,fill=fill,usersym=usersym,corners=corners,thick=thick
;
; =====>> HELP
;
on_error,2
```

```
if keyword_set(help) then begin & doc_library,'legend' & return & endif
;
; =====>> SET DEFAULTS FOR SYMBOLS, LINESTYLES, AND ITEMS.
;
ni = n_elements(items)
np = n_elements(psym)
nt = n_elements(thick)
nl = n_elements(linestyle)
nv = n_elements(vectorfont)
n = max([ni,np,nl,nv,nt])    ; NUMBER OF ENTRIES
strn = strtrim(n,2)     ; FOR ERROR MESSAGES
if n eq 0 then message,'No inputs!  For help, type legend,/help.'
if ni eq 0 then begin
  items = replicate('',n)    ; DEFAULT BLANK ARRAY
endif else begin
  szt = size(items)
  if (szt(szt(0)+1) ne 7) then message,'First parameter must be a string array.  For help, type
legend,/help.'
  if ni ne n then message,'Must have number of items equal to '+strn
endelse
symline = (np ne 0) or (nl ne 0) or (nt ne 0)  ; FLAG TO PLOT SYM/LINE
if (np ne 0) and (np ne n) then message,'Must have 0 or '+strn+' elements in psym array.'
if (nl ne 0) and (nl ne n) then message,'Must have 0 or '+strn+' elements in linestyle array.'
if (nt ne 0) and (nt ne n) then message,'Must have 0 or '+strn+' elements in thick array.'
if n_elements(linestyle) ne n then linestyle = intarr(n); D=SOLID
if n_elements(psym) ne n then psym = intarr(n)  ; D=SOLID
if n_elements(thick) ne n then thick = intarr(n)+1 ; D=SOLID
if n_elements(vectorfont) ne n then vectorfont = replicate('',n)
;
; =====>> CHOOSE VERTICAL OR HORIZONTAL ORIENTATION.
;
if n_elements(horizontal) eq 0 then begin   ; D=VERTICAL
  if n_elements(vertical) eq 0 then vertical = 1
endif else begin
  if n_elements(vertical) eq 0 then vertical = not horizontal
endelse
;
; =====>> SET DEFAULTS FOR OTHER OPTIONS.
;
if n_elements(box) eq 0 then box = 1
if n_elements(margin) eq 0 then margin = 0.3
if n_elements(delimiter) eq 0 then delimiter = ''
if n_elements(charsize) eq 0 then begin
  ;OLDcharsize=1
   ;OLDcharsize=!p.charsize-MIN([MAX([!p.multi(2)-1,0])*0.1,0.8 ])
  if !d.name ne 'PS' then begin
    charsize=1
  endif else begin
```

```
    charsize=148./float(!d.x_ch_size)
  endelse
endif
if charsize eq 0 then charsize = 1
if n_elements(spacing) eq 0 then spacing = 1.1
if n_elements(pspacing) eq 0 then pspacing = 3
if !d.name ne 'PS' then begin
  xspacing = !d.x_ch_size/float(!d.x_size) * (spacing > charsize)
  yspacing = !d.y_ch_size/float(!d.y_size) * (spacing > charsize)
endif else begin   ; patch for PS to account for multiplot scaling up font size
  xspacing = 148./float(!d.x_size) * (spacing > charsize)      ; 148,235 is 8point font size
  yspacing = 235./float(!d.y_size) * (spacing > charsize)
endelse
if !x.window(0) eq !x.window(1) then begin
  plot,/nodata,xstyle=4,ystyle=4,[0],/noerase
endif
; next line takes care of weirdness with small windows
pos =  [min(!x.window),min(!y.window),max(!x.window),max(!y.window) ]
if n_elements(position) eq 0 then position = [pos(0),pos(3)] + [xspacing,-yspacing]
if n_elements(number) eq 0 then number = 1
if n_elements(colors) eq 0 then colors = !color + intarr(n)
if n_elements(textcolors) eq 0 then textcolors = !color + intarr(n)
fill = keyword_set(fill)
if n_elements(usersym) eq 0 then usersym = 2*[[0,0],[0,1],[1,1],[1,0]]-1
;
; =====>> INITIALIZE POSITIONS
;
yoff = 0.25*yspacing    ; VERT. OFFSET FOR SYM/LINE.
maxx = 0     ; SAVED WIDTH FOR DRAWING BOX
x0 = position(0) + (margin)*xspacing  ; INITIAL X & Y POSITIONS
y0 = position(1) - (margin-0.5)*yspacing
y = y0      ; STARTING X & Y POSITIONS
x = x0
if vertical then begin    ; CALC OFFSET FOR TEXT START
  xt = 0     ; DEFAULT X VALUE
  for i = 0,n-1 do begin
    if psym(i) eq 0 then num = (number + 1) > 3 else num = number
    if psym(i) lt 0 then num = number > 2 ; TO SHOW CONNECTING LINE
    if psym(i) eq 0 then expand = 1 else expand = 2
    xt = (expand*pspacing*(num-1)*xspacing) > xt
    endfor
endif ; NOW xt IS AN X OFFSET TO ALIGN ALL TEXT ENTRIES.
;
; =====>> OUTPUT TEXT FOR LEGEND, ITEM BY ITEM.
; =====>> FOR EACH ITEM, PLACE SYM/LINE, THEN DELIMITER,
; =====>> THEN TEXT---UPDATING X & Y POSITIONS EACH TIME.
; =====>> THERE ARE A NUMBER OF EXCEPTIONS DONE WITH IF STATEMENTS.
;
```

```
for i = 0,n-1 do begin
  if vertical then x = x0 else y = y0  ; RESET EITHER X OR Y
  x = x + xspacing    ; UPDATE X & Y POSITIONS
  y = y - yspacing
  if (psym(i) eq 0) and (vectorfont(i) eq '') then num = (number + 1) > 3 else num = number
  if psym(i) lt 0 then num = number > 2  ; TO SHOW CONNECTING LINE
  if psym(i) eq 0 then expand = 1 else expand = 2
  xp = x + expand*pspacing*indgen(num)*xspacing
  if (psym(i) gt 0) and (num eq 1) and vertical then xp = x + xt/2.
  yp = y + intarr(num)
  if vectorfont(i) eq '' then yp = yp + yoff
  if psym(i) eq 0 then begin
    xp = [min(xp),max(xp)]   ; TO EXPOSE LINESTYLES
    yp = [min(yp),max(yp)]   ; DITTO
    endif
  if psym(i) eq 9 then usersym,usersym,fill=fill,color=colors(i)
  if vectorfont(i) ne '' then begin
    if (num eq 1) and vertical then xp = x + xt/2
    xyouts,xp,yp,vectorfont(i),width=width,color=colors(i) $
      ,size=charsize,align=0.5,/norm
  endif else begin
    if symline and (linestyle(i) ge 0) then plots,xp,yp,color=colors(i) $
       ,/normal,linestyle=linestyle(i),psym=psym(i),symsize=charsiz e,$
      thick=thick(i)
  endelse
  if vertical then x = x + xt else x = max(xp)
  if symline then x = x + xspacing
   xyouts,x,y,delimiter,width=width,/norm,color=textcolors(i),s ize=charsize
  x = x + width
  if width gt 0 then x = x + 0.5*xspacing
   xyouts,x,y,items(i),width=width,/norm,color=textcolors(i),si ze=charsize
  x = x + width
  if not vertical and (i lt (n-1)) then x = x+2*xspacing; ADD INTER-ITEM SPACE
  maxx = (x + xspacing*margin) > maxx  ; UPDATE MAXIMUM X
  endfor
;
; =====>> OUTPUT BORDER
;
x = position(0)
y = position(1)
if vertical then bottom = n else bottom = 1
ywidth = - (2*margin+bottom-0.5)*yspacing
corners = [x,y+ywidth,maxx,y]
if box then plots,[x,maxx,maxx,x,x],y + [0,0,ywidth,ywidth,0],/norm
return
end
```