Subject: log axis
Posted by m218003 on Fri, 10 Sep 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Hi there,

until today I still wasn't completely satisfied about plotting logarithmic axis -- I often want to label the 2's and 5's as well as the 10's which is not THAT straightforward as it seems. I had already written a little function LOGLEVELS.PRO which will return "nice-looking" log levels depending on a Range input, and now I have added a routine named AXLABEL which puts these labels out at the right place -- please find these two attached.

Here is how it works:

```
 ; define dummy x and y
 X = findgen(101)/10.-4
 Y = X > .1

; get range of data
mind = MIN(Y,MAX=maxd)
Lblv = LOGLEVELS([mind,maxd])   ; <====== 1 =====
PLOT,X,Y,/YLOG,YTICKFORMAT='(A1)'
AXLABEL,Lblv,charsize=2.4 ; ,format='exponent'   ; <=== 2 ====
```

You could also relabel the x axis as follows:

```
 PLOT,X,Y,/YLOG,YTICKFORMAT='(A1)',XTICKFORMAT='(A1)',XTICK_G ET=xt
AXLABEL,Lblv,charsize=2.4 ; ,format='exponent'   ; <=== 2 ====
AXLABEL,xt,charsize=xc,format='(i12)',/XAxis     ; <=== 2 ====
```

The FORMAT keyword accepts a function name similar to XTICKFORMAT, so you can use e.g. Stein Vidar's EXPONENT function (which you can find in David Fanning's web universe).

Thought I'd like to share this with you,
best regards,
Martin

--
[[[[[[[[[
[[Martin Schultz Max-Planck-Institut fuer Meteorologie [[
[[Bundesstr. 55, 20146 Hamburg [[
[[phone: +49 40 41173-308 [[
[[fax: +49 40 441787 [[
[[martin.schultz@dkrz.de [[
[[[[[[[[[

```
;-------------------------------------------------------- --
; $Id$
;+
; NAME:
;       AXLABEL
;
; PURPOSE:
;       Put previously calculated axis labels onto the screen
;       at proper position. This routine was designed to work
;       together with LOGLEVELS to produce fancy log plots.
;       It involves several coordinate transformations in order
;       to be device independent and take into account the
;       character size. The user can specify a label format
;       and use 'external' formatting functions similar to
;       the [XYZ]TICKFORMAT keyword of PLOT.
;
; CATEGORY:
;       Plotting
;
; CALLING SEQUENCE:
;       AXLABEL,Value [,/XAxis] [,keywords]
;
; INPUTS:
;       VALUE -> A vector with the values to be labelled on the
;           axis.
;
; KEYWORD PARAMETERS:
;       /XAxis -> If set, the labels are placed on the X achis
;           rather than on the Y axis
;
;       /YAxis -> Place the labels on the Y axis (this is the default,
;           and this keyword is there for purely aesthetic reasons)
;
;       CHARSIZE -> The character size of the label
;
;       FORMAT -> An IDL format string (used as argument to the
;            STRING function) or the name of a function that returns
;            formatted labels. This function must accept three
;            arguments, the third of which is the current value
;            (see the online help to [XYZ]TICKFORMAT for more details).
;            AXLABEL always passes 0 to the first two arguments.
;
;       _EXTRA  keywords are passed on to XYOUTS (e.g. COLOR or
;            ORIENTATION). Note that the ALIGN keyword value is
;            determined automatically.
;
; OUTPUTS:
;       Axis labels without fuss.
```

```
;
; SUBROUTINES:
;       None.
;
; REQUIREMENTS:
;       A DATA coordinate system must be established by a previous
;       PLOT command.
;
; NOTES:
;       AXLABEL currently operates only on the left and bottom axes.
;
; EXAMPLE:
;
; MODIFICATION HISTORY:
;       mgs, 10 Sep 1999: VERSION 1.00
;
;-
; Copyright (C) 1999, Martin Schultz, Max-Planck-Institut f. Meteorologie
; This software is provided as is without any warranty
; whatsoever. It may be freely used, copied or distributed
; for non-commercial purposes. This copyright notice must be
; kept with any copy of this software. If this software shall
; be used commercially or sold as part of a larger package,
; please contact the author.
; Bugs and comments should be directed to martin.schultz@dkrz.de
; with subject "IDL routine axlabel"
 ;-------------------------------------------------------------- --


pro axlabel,value,Charsize=Charsize,XAxis=XAxis,YAxis=YAxis,   $
      Format=Format,_EXTRA=e

 ; Lblv = LOGLEVELS([mind,maxd])
 ; PLOT,X,Y,/YLOG,YTICKFORMAT='(A1)'

   ; Error catching
   if (N_Elements(VALUE) eq 0) then begin
      message,'Must supply a label value to AXLABEL!'
   endif

   ; Set default for CHARSIZE and FORMAT
   if (n_elements(CHARSIZE) EQ 0) then $
      CHARSIZE = 1.
   if (n_elements(FORMAT) EQ 0) then $
      FORMAT = '(f8.1)'

   if (keyword_set(XAxis)) then begin
```

```
      ; Get y position for label
      ; Subtract one character size
      PY = !Y.Window[0]
      PYOFF = CONVERT_COORD(1,!D.Y_CH_SIZE*CHARSIZE,/DEVICE,/TO_NORMAL)
      PY = PY - 1.05*PYOFF[1]
      PY = REPLICATE(PY,N_Elements(VALUE))

      ; Convert data values to normalized x coordinates
      PX =  CONVERT_COORD(VALUE,REPLICATE(!Y.CRange[0],N_Elements(VALUE) ), $
                  /DATA,/TO_NORMAL)
      PX = PX[0,*]

   endif else begin   ; Y axis label (default)

      ; Get x position for label
      PX = !X.Window[0] - 0.010
      PX = REPLICATE(PX,N_Elements(VALUE))

      ; Convert data values to normalized coordinates and
      ; subtract half the character size
      PYOFF = CONVERT_COORD(0,!D.Y_CH_SIZE*CHARSIZE,/DEVICE,/TO_NORMAL)
      PY =  CONVERT_COORD(REPLICATE(!X.CRANGE[0],N_Elements(VALUE)),VALU E,  $
                  /DATA,/TO_NORMAL)
      PY = PY[1,*]-0.5*PYOFF[1]
   endelse

   ; Format VALUE according to format string. If this string
   ; does not begin with '(', it is assumed that the user has passed
   ; a formatting function as for [XYZ]TICKFORMAT
   ; However, only the third (NUMBER) argument of this function is used
   if (STRPOS(FORMAT,'(') ne 0) then begin
      ValS = STRARR(N_Elements(VALUE))
      for j=0,N_Elements(VALUE)-1 do $
         ValS[j] = CALL_FUNCTION(FORMAT,0,0,VALUE[j])
   endif else $      ; apply format string directly
      ValS = STRING(VALUE,format=FORMAT)

   ValS = STRTRIM(ValS,2)

   XYOUTS,PX,PY,ValS,/NORMAL,align=1.-0.5*keyword_set(XAxis),  $
      charsize=CHARSIZE,_EXTRA=e

   return
end


 ;--------------------------------------------------------- --
; $Id: loglevels.pro,v 1.2 1999/03/24 04:30:05 mgs Exp $
```

```
;+
; NAME:
;       LOGLEVELS (function)
;
; PURPOSE:
;       Compute default values for logarithmic axis labeling
;       or contour levels. For a range from 1 to 100 these
;       would be 1., 2., 5., 10., 20., 50., 100.
;       If the range spans more than (usually) 3 decades, only
;       decadal values will be returned unless the /FINE keyword
;       is set.
;
; CATEGORY:
;       Tools
;
; CALLING SEQUENCE:
;       result = LOGLEVELS([range | MIN=min,MAX=max] [,/FINE] [,COARSE=dec])
;
; INPUTS:
;       RANGE -> A 2-element vector with the minimum and maximum
;          value to be returned. Only levels _within_ this range
;          will be returned. If RANGE contains only one element,
;          this is interpreted as MAX and MIN will be assumed as
;          3 decades smaller. RANGE superseeds the MIN and MAX
;          keywords. Note that RANGE must be positive definite
;          but can be given in descending order in which case
;          the labels will be reversed.
;
; KEYWORD PARAMETERS:
;       MIN, MAX -> alternative way of specifying a RANGE. If only
;          one keyword is given, the other one is computed as
;          3 decades smaller/larger than the given parameter.
;          RANGE superseeds MIN and MAX.
;
;       /FINE -> always return finer levels (1,2,5,...)
;
;       COARSE -> the maximum number of decades for which LOGLEVELS
;          shall return fine labels. Default is 3. (non-integer
;          values are possible).
;
; OUTPUTS:
;       A vector with "round" logarithmic values within the given
;       range. The original (or modified) RANGE will be returned
;       unchanged if RANGE does not span at least one label interval.
;       The result will always contain at least two elements.
;
;
; SUBROUTINES:
```

```
;       none
;
; REQUIREMENTS:
;       none
;
; NOTES:
;       If COARSE is lt 0, the nearest decades will be returned
;       instead. The result will always have at least two elements.
;       If COARSE forces decades, the result values may be out-of-
;       range if RANGE spans less than a decade.
;
;       Caution with type conversion from FLOAT to DOUBLE !!
;
; EXAMPLE:
;       range = [ min(data), max(data) ]
;       c_level = LOGLEVELS(range)
;       contour,...,c_level=c_level
;
;
; MODIFICATION HISTORY:
;       mgs, 17 Mar 1999: VERSION 1.00
;
;-
; Copyright (C) 1999, Martin Schultz, Harvard University
; This software is provided as is without any warranty
; whatsoever. It may be freely used, copied or distributed
; for non-commercial purposes. This copyright notice must be
; kept with any copy of this software. If this software shall
; be used commercially or sold as part of a larger package,
; please contact the author to arrange payment.
; Bugs and comments should be directed to mgs@io.harvard.edu
; with subject "IDL routine loglevels"
 ;-------------------------------------------------------------- --


function loglevels,range,min=mind,max=maxd,  $
          coarse=coarse,fine=fine


   if (n_elements(COARSE) eq 0) then COARSE = 3

   ; Make sure to have a valid range which is positive definite
   ; NOTE that range does not need to be sorted!
   if (n_elements(mind) gt 0) then begin
      mind = mind[0]
      if (n_elements(maxd) eq 0) then maxd = mind*1000.
   endif
   if (n_elements(maxd) gt 0) then begin
```

```
      maxd = maxd[0]
      if (n_elements(mind) eq 0) then mind = maxd*0.001
    endif
    ; still not defined, i.e. neither mind nor maxd given
    if (n_elements(mind) eq 0) then begin
      mind = 0.1
      maxd = 100.
    endif
    ; RANGE superseeds min and max
    if (n_elements(range) eq 0) then range = [ mind,maxd ]
    ; one element for RANGE is interpreted as MAX
    if (n_elements(range) eq 1) then range = [ range*0.001, range ]

    thisrange = double(range) > 1.D-100
    thisrange = thisrange(sort(thisrange))

    ; set lower range to 3 decades below upper range if it is zero
    if (thisrange[0] lt 1.D-36) then thisrange[0] = thisrange[1]/1000.


    ; get log of ranges and decadal log
    lrange = alog10(thisrange)
    if (lrange[0] lt 0.) then lrange[0] = lrange[0] - 1.0D-6
    if (lrange[1] lt 0.) then lrange[1] = lrange[1] - 1.0D-6
;   if (lrange[1] gt 0.) then lrange[1] = lrange[1] + 1.0D-6
    drange = fix(lrange)


    ; create label arrays to choose from
    ; currently 1.D-15 to 5.D+16
    ; ranges outside these limits always return only decades

    ; set mode according to following rules:
    ; - range outside limits -> return decades
    ; - coarse exceeded -> return decades
    ; - /fine set -> return 1,2,5,... for any number of decades
    ; - [automatic] -> return decades if more than COARSE decades
    ;                  otherwise 1,2,5,..

    mode = 0   ; return decades
    if (keyword_set(fine)) then mode = 1
    if ((lrange[1]-lrange[0]) le COARSE) then mode = 1
    if (thisrange[0] lt 1.D-15 OR thisrange[1] gt 5.D16) then mode = 0

    if (mode) then begin
      ; make overall array
      labels = [ 1.D-15, 2.D-15, 5.D-15, 1.D-14, 2.D-14, 5.D-14, $
                 1.D-13, 2.D-13, 5.D-13, 1.D-12, 2.D-12, 5.D-12, $
```

```
            1.D-11, 2.D-11, 5.D-11, 1.D-10, 2.D-10, 5.D-10, $
            1.D-09, 2.D-09, 5.D-09, 1.D-08, 2.D-08, 5.D-08, $
            1.D-07, 2.D-07, 5.D-07, 1.D-06, 2.D-06, 5.D-06, $
            1.D-05, 2.D-05, 5.D-05, 1.D-04, 2.D-04, 5.D-04, $
            1.D-03, 2.D-03, 5.D-03, 1.D-02, 2.D-02, 5.D-02, $
            1.D-01, 2.D-01, 5.D-01, 1.D+00, 2.D+00, 5.D+00 ]
    labels = [ labels, $
            1.D+01, 2.D+01, 5.D+01, 1.D+02, 2.D+02, 5.D+02, $
            1.D+03, 2.D+03, 5.D+03, 1.D+04, 2.D+04, 5.D+04, $
            1.D+05, 2.D+05, 5.D+05, 1.D+06, 2.D+06, 5.D+06, $
            1.D+07, 2.D+07, 5.D+07, 1.D+08, 2.D+08, 5.D+08, $
            1.D+09, 2.D+09, 5.D+09, 1.D+10, 2.D+10, 5.D+10, $
            1.D+11, 2.D+11, 5.D+11, 1.D+12, 2.D+12, 5.D+12, $
            1.D+13, 2.D+13, 5.D+13, 1.D+14, 2.D+14, 5.D+14, $
            1.D+15, 2.D+15, 5.D+15, 1.D+16, 2.D+16, 5.D+16 ]

     llabels = alog10(labels)
     ind = where(llabels ge lrange[0] AND llabels le lrange[1])

     ; if range values are too close, return original range
     if (ind[0] lt 0) then return,range

     ; return reversed labels if range[0] gt range[1]
     if (range[0] gt range[1]) then $
       return,reverse(labels[min(ind):max(ind)]) $
     else $
       return,labels[min(ind):max(ind)]

  endif else begin

    if (lrange[1] lt 0.) then drange[1] = drange[1] - 1

    exponent = indgen(drange[1]-drange[0]+1)+drange[0]
    if (n_elements(exponent) eq 1) then $
      exponent = [ exponent, exponent+1 ]

    if (range[0] gt range[1]) then $
      return,reverse(10.D0^exponent)  $
    else $
      return,10.D0^exponent
  endelse

end
```

File Attachments

1) axlabel.pro, downloaded 73 times
2) loglevels.pro, downloaded 89 times