
Subject: Reversed log axes in object graphics

Posted by [Randall Skelton](#) on Mon, 18 Oct 2004 17:10:05 GMT

[View Forum Message](#) <> [Reply to Message](#)

Firstly, thanks for reading. I realize with a subject line as above, many will be scared away...

I am trying to move one particular class of plotting routines (that I use very frequently) to object graphics and I've run into a problem. Essentially, I cannot plot a reversed, logarithmic axis. Below is some code which shows my problem. The equivalent direct graphics code as per David's tip (http://www.dfanning.com/graphics_tips/yadayada.html) is shown to illustrate what I expect the plot to look like. All contributions greatly appreciated!

Cheers,
Randall

```
=== log_test.pro ===
```

```
pro log_test
```

```
compile_opt DEFINT32
compile_opt STRICTARR
compile_opt STRICTARRSUBS
compile_opt LOGICAL_PREDICATE
```

```
p = [1013.00, 898.800, 795.000, 701.200, 616.600,
540.500, $
472.200, 411.100, 356.500, 308.000, 265.000,
227.000, $
194.000, 165.800, 141.700, 121.100, 103.500,
88.5000, $
75.6500, 64.6700, 55.2900, 47.2900, 40.4700,
34.6700, $
29.7200, 25.4900, 17.4300, 11.9700, 8.01000,
5.74600, $
4.15000, 2.87100, 2.06000, 1.49100, 1.09000,
0.797800, $
0.425000, 0.219000, 0.109000, 0.0522000, 0.0240000,
0.0105000, $
0.00446000, 0.00184000, 0.000760000, 0.000320000, 0.000145000,
7.10000e-05, $
4.01000e-05, 2.54000e-05 ]
```

```
t = [288.200, 281.700, 275.200, 268.700, 262.200, 255.700, 249.200, $
242.700, 236.200, 229.700, 223.300, 216.800, 216.700,
216.700, $
```

```
216.700, 216.700, 216.700, 216.700, 216.700, 216.700,
216.700, $
217.600, 218.600, 219.600, 220.600, 221.600, 224.000,
226.500, $
230.000, 236.500, 242.900, 250.400, 257.300, 264.200,
270.600, $
270.700, 260.800, 247.000, 233.300, 219.600, 208.400,
198.600, $
188.900, 186.900, 188.400, 195.100, 208.800, 240.000,
300.000, $
360.00 ]
```

```
z = [0.00000, 1.00000, 2.00000, 3.00000, 4.00000, 5.00000, 6.00000, $
7.00000, 8.00000, 9.00000, 10.0000, 11.0000, 12.0000,
13.0000, $
14.0000, 15.0000, 16.0000, 17.0000, 18.0000, 19.0000,
20.0000, $
21.0000, 22.0000, 23.0000, 24.0000, 25.0000, 27.5000,
30.0000, $
32.5000, 35.0000, 37.5000, 40.0000, 42.5000, 45.0000,
47.5000, $
50.0000, 55.0000, 60.0000, 65.0000, 70.0000, 75.0000,
80.0000, $
85.0000, 90.0000, 95.0000, 100.000, 105.000, 110.000,
115.000, $
120.000 ]
```

```
sp = size(p)
nlev = sp[1]
```

```
xtitle = 'Temperature (K)'
ytitle = 'Pressure (mbar)'
ytitle2 = 'Altitude (km)'
```

```
xrange = [min(t),max(t)]
yrange = [p[0], p[nlev-1]]
yrange2 = [min(z), max(z)]
```

```
;;; Plot in direct graphics (4 lines)
!x.margin=[10,8]
plot_io, ystyle=9, t, p, xtitle=xtitle, ytitle=ytitle,$
yrange=yrange, xrange=xrange, xstyle=1
```

```
idx = where ( z mod 10 eq 0 )
axis, yaxis=1, yticks=12, ytickv=p[idx], $
ytickname=string(z[idx],format='(i3)'), ytitle=ytitle2
```

```
;;; Plot in object graphics (more than 4 lines)
```

; Create view object.

```
thisView = OBJ_NEW('IDLgrView', Viewplane_Rect=[-1.0,-1.0,2,2], $  
ZClip=[1.15,-1.15] )
```

; Create model object.

```
thisModel = OBJ_NEW('IDLgrModel')  
thisView->Add, thisModel
```

; Create font object.

```
helvetica10pt = Obj_New('IDLgrFont', 'Helvetica', Size=10)
```

; Create helper objects. First, create title objects
; for the axes and plot. Color them green. Title objects
; will need to be located by hand for reversible axes.
; Be sure you DON'T add the title objects directly to
; the axis objects.

```
xTitleObj = Obj_New('IDLgrText', xtitle, $  
Font=helvetica10pt, /Enable_Formatting)  
yTitleObj = Obj_New('IDLgrText', ytitle, $  
Font=helvetica10pt, /Enable_Formatting)  
yTitleObj2 = Obj_New('IDLgrText', ytitle2, $  
Font=helvetica10pt, /Enable_Formatting)
```

; Create a plot

```
thisPlot = OBJ_NEW('IDLgrPlot', t, p, _Extra=extra)
```

; Get the data ranges of the plot.

```
thisPlot->GetProperty, XRange=xrange, YRange=yrange
```

```
ticklen = 0.02
```

; Axes are created after the plot so the range can be
; set correctly.

```
xAxis = Obj_New("IDLgrAxis", 0, Ticklen=ticklen, $  
Minor=4, Range=xrange, /Extend)  
xAxis->GetProperty, Ticktext=xAxisText  
xAxisText->SetProperty, Font=helvetica10pt
```

```
xAxis2 = Obj_New("IDLgrAxis", 0, Ticklen=ticklen, $  
Minor=4, Range=xrange, /Extend, TICKDIR=1, TEXTPOS=1, NOTEXT=1)
```

```
yAxis = Obj_New("IDLgrAxis", 1, Ticklen=ticklen, $
Minor=4, Range=reverse(yrange), /Log, /Extend)
yAxis->GetProperty, Ticktext=yAxisText
yAxisText->SetProperty, Font=helvetica10pt
```

```
yAxis2 = Obj_New("IDLgrAxis", 1, Ticklen=ticklen, $
Minor=4, Range=yrange2, /Extend, TICKDIR=1, TEXTPOS=1)
yAxis2->GetProperty, Ticktext=yAxisText2
yAxisText2->SetProperty, Font=helvetica10pt
```

```
; The axes may not use exact axis scaling, so the ranges may
; have changed from what they were originally set to. Get
; and update the range variables.
```

```
xAxis->GetProperty, CRange=xrange
xAxis2->GetProperty, CRange=xrange2
yAxis->GetProperty, CRange=yrange
yAxis2->GetProperty, CRange=yrange2
```

```
xs = Normalize(xrange, Position=[-0.5,0.5])
xs2 = Normalize(xrange2, Position=[-0.5,0.5])
ys = Normalize(yrange, Position=[-0.5,0.5])
ys2 = Normalize(yrange2, Position=[-0.5,0.5])
```

```
; Scale the axes and place them in the coordinate space.
; Note that not all values in the Location keyword are
; used. (I've put really large values into the positions
; that are not being used to demonstrate this.) For
; example, with the X axis only the Y and Z locations are used.
```

```
xAxis->SetProperty, Location=[9999.0, -0.5, -0.5], XCoord_Conv=xs
xAxis2->SetProperty, Location=[9999.0, 0.5, -0.5], XCoord_Conv=xs2
yAxis->SetProperty, Location=[-0.5, 9999.0, -0.5], YCoord_Conv=ys
yAxis2->SetProperty, Location=[0.5, 9999.0, -0.5], YCoord_Conv=ys2
```

```
; Scale the Plot.
```

```
thisPlot->SetProperty, XCoord_Conv=xs, YCoord_Conv=ys
```

```
; Add the plot and axes objects to the model.
```

```
thisModel->Add, thisPlot
thisModel->Add, xAxis
thisModel->Add, xAxis2
thisModel->Add, yAxis
thisModel->Add, yAxis2
```

; Add the title objects to the model.

```
thisModel->Add, xTitleObj  
thisModel->Add, yTitleObj  
thisModel->Add, yTitleObj2
```

; Get the window destination object, which is the value of
; an object draw widget. The view will be drawn in the window
; when the window is exposed.

```
thisWindow = obj_new('IDLgrWindow', DIMENSION=[15,15], UNITS=2, $  
GRAPHICS_TREE=thisView)  
thisWindow -> erase
```

; Once we have a window, find the size of the character box
surrounding the
; axis annotation, and calculate a location for the axis titles.
Note that the Y
; dimension of the X axis text box is always about 75% of what it
should be. This
; is the reason the X axis title always appears too close to the
axis compared
; to the Y and Z axis in the normal default placement. That is why
you see that
; number multiplied by 1.5 for the XTitleObj below. (The values
-0.5, 0.5, and 0
; are the endpoints and middle, respectively, of my axis in my
viewport rectangle.)

; To orient the text properly, you must supply the proper baseline
and up direction
; vectors to the Y and Z titles. The X title does not need this,
since the X "situation"
; is the default case. For example, read the Y title orientation
like this: draw the
; text parallel to the Y axis (Baseline=[0,1,0]), with the up
direction in the -X direction
; (UpDir=[-1,0,0]).

```
d = thisWindow->GetTextDimensions(xAxisText)  
xTitleObj->SetProperty, Location=[0, -0.5 - d[1]*2-ticklen, -0.5], $  
Alignment=0.5
```

```
d = thisWindow->GetTextDimensions(yAxisText)  
yTitleObj->SetProperty, Location=[-0.5 - d[0]-2*ticklen, 0, -0.5], $  
Baseline=[0,1,0], UpDir=[-1,0,0], Alignment=0.5
```

```
d = thisWindow->GetTextDimensions(yAxisText2)
```

```
yTitleObj2->SetProperty, Location=[0.5 + d[0]*2-ticklen, 0, -0.5], $  
Baseline=[0,1,0], UpDir=[-1,0,0], Alignment=0.5  
; Draw the window
```

```
thisWindow -> draw
```

```
end
```
