Subject: Re: cgGallery with function graphics
Posted by Matthew Argall on Thu, 27 Feb 2014 15:46:22 GMT
View Forum Message <> Reply to Message

Woops! Here it is in its entirety (with the legend and coyote version)

==========
Copy + Paste
==========

```
; Set up variables for the plot. Normally, these values would be
; passed into the program as positional and keyword parameters.
; Create two random vectors.
data_1 = cgDemoData(17)
data_2 = cgDemoData(17)

; Calculate the data range, so you can create a plot with room at the top
; of the plot for the legend.
maxData = Max(data_1) > Max(data_2)
minData = Min(data_1) > Min(data_2)
dataRange = maxData - minData
yrange = [MinData, maxData + (dataRange*0.25)]

; Legend Properties
items = ['Experiment 1', 'Experiment 2']
psyms = [-15, -16]
colors = ['red7', 'blu7']

; Open a window and return its reference to the user.
aWindow = Window(WINDOW_TITLE="Line Plot with Legend")

;Create the Plot
ngPlot = Plot(data_1, /Current, Symbol='Circle', /Sym_Filled, Color='Red', $
            YRange=yrange, YStyle=1, XTitle='Time', YTitle='Signal', $
            Name='Experiment 1')
ngOPlot = Plot(data_2, Symbol='Square', /Sym_Filled, Color='Blue', $
             Overplot=ngPlot, Name='Experiment 2')

;Add the legend
ngLegend = Legend(/Auto_Text_Color, Position=[5, 120], Target=[ngPlot, ngOPlot], /Data, $
               Horizontal_Alignment='Left')
```

```
================
Coyote Version
================
PRO Line_Plot_With_Legend_FG, WINDOW=window
   ; Set up variables for the plot. Normally, these values would be
   ; passed into the program as positional and keyword parameters.
   ; Create two random vectors.
   data_1 = cgDemoData(17)
   data_2 = cgDemoData(17)

   ; Calculate the data range, so you can create a plot with room at the top
   ; of the plot for the legend.
   maxData = Max(data_1) > Max(data_2)
   minData = Min(data_1) > Min(data_2)
   dataRange = maxData - minData
   yrange = [MinData, maxData + (dataRange*0.25)]

   ; Legend Properties
   items = ['Experiment 1', 'Experiment 2']
   psyms = [-15, -16]
   colors = ['red7', 'blu7']

   ; Open a window and return its reference to the user.
   aWindow = Window(WINDOW_TITLE="Line Plot with Legend")

   ;Create the Plot
   ngPlot = Plot(data_1, /Current, Symbol='Circle', /Sym_Filled, Color='Red', $
            YRange=yrange, YStyle=1, XTitle='Time', YTitle='Signal', $
            Name='Experiment 1')
   ngOPlot = Plot(data_2, Symbol='Square', /Sym_Filled, Color='Blue', $
             Overplot=ngPlot, Name='Experiment 2')

   ;Add the legend
   ngLegend = Legend(/Auto_Text_Color, Position=[5, 120], Target=[ngPlot, ngOPlot], /Data, $
              Horizontal_Alignment='Left')

END  ;****************************************************** ******

; This main program shows how to call the program and produce
; various types of output.

  ; Display the plot in a  resizeable graphics window.
  Line_Plot_With_Legend_FG, Window=window

  ; Create a PostScript file. Linestyles are not preserved in IDL 8.2.3 due
  ; to a bug. Only encapsulated PostScript files can be created.
  window.save, 'ling_plot_with_legend.eps'
```

```
  ; Create a PNG file with a width of 600 pixels. Resolution of this
  ; PNG file is not very good.
  window.save, 'ling_plot_with_legend.png', WIDTH=600

  ; For better resolution PNG files, make the PNG full-size, then resize it
  ; with ImageMagick. Requires ImageMagick to be installed.
  window.save, 'ling_plot_with_legend_fullsize.png'
  Spawn, 'convert ling_plot_with_legend_fg_fullsize.png -resize 600
ling_plot_with_legend_fg_resized.png'

END
```