sunday, november 29. 2009

Plotting data with gnuplot in real-time

I needed to plot data from a program in real-time for a demo, that is, as the program generates the data, I wanted it to show in some nice diagram.

A friend pointed me to Visualize real-time data streams with Gnuplot by Thanassis Tsiodras, which is very nice: you specify the number of streams to plot and where to plot them, and feed it with lines like '0:1.23', '1:2.3', etc. defining the data points for the individual streams (here: first point on stream 0 is 1.23, first point on stream 1 is 2.3, etc.).

However, I needed to plot several streams in a single window, which that version lacked. So I extended that program with the ability to plot several streams into a single window (download).

For this, you specify both the number of streams you want to plot and the number of windows you want to show. Furthermore, you specify in which window each stream is plotted. I kept as much as possible of the original command line interface to ease the transition to this new version. The result looks like this:

Three data streams plotted in two windows.

You see the three data streams `sin`, `cos`, and `log` plotted in two windows. `sin` and `cos` are plotted in the first window, `log` is plotted in the second window. Also note that the data rate of `log` is different from the data rate of `sin` and `cos`. You could also plot all three data streams in a single window.

The input (on stdin or from a file) looks like this, as for the original version:

```
0:0.09983
1:0.99500
2:0.69314
0:0.19866
1:0.98006
0:0.29552
1:0.95533
2:1.79175
0:0.38941
1:0.92106
0:0.47942
1:0.87758
2:2.30258
0:0.56464
1:0.82533
0:0.64421
1:0.76484
2:2.63905
...
```

The number before the colon specifies to which stream the data point belongs, the number after the colon is the (next) data point. Note that in this example, the data points for stream 0 and 1 (`sin` and `cos` in the example above) come twice as fast as the data points for stream 2 (`log` in the example above).

The command line to generate these plots looks like this:

```
perl ./driveGnuPlotStreams.pl 3 2 \        # number of streams and windows
   50 50 \                                 # width of sliding window in each window
   -1 1 -2 6 \                             # min/max values for each window
   500x300+0+0   500x300+500+0 \           # geometry of each window
   'sin' 'cos' 'log' \                     # title of each stream
   0 0 1                                   # in which window to plot each stream
```

The first two numbers on the command line give the number of streams and number of windows to plot, respectively. The second line gives the number of data points to plot, that is, the size of the sliding window. The third line gives the min/max values for each window. The fourth line gives the geometry of the windows. The fifth line gives the titles of the streams. And the sixth and last line gives the window number in which each stream is to be plotted. Compared to the original program, I changed the position of the geometry specification and the titles, so that the options for the windows and the streams are each grouped together.

The command prints its status on stdout like this:

```
Will display 3 Streams in 2 windows...
Window 0 will use a window of 50 samples
Window 1 will use a window of 50 samples
Window 0 will use a range of [-1, 1]
Window 1 will use a range of [-2, 6]
Window 0 will use a geometry of '500x300+0+0'
Window 1 will use a geometry of '500x300+500+0'
Stream 0 will use a title of 'sin'
Stream 1 will use a title of 'cos'
Stream 2 will use a title of 'log'
Stream 0 will be plotted in window 0
Stream 1 will be plotted in window 0
Stream 2 will be plotted in window 1
```

It starts to plot the windows as soon as the data arrives.

The program expects its data on stdin or from a file/pipe given after all the command line options. For example, the plots above where generated like this:

```
(perl -e '
for (1..100) {
  $i=$_/10;
  print "0:", sin($i), "\n1:", cos($i), "\n";
  if ($_%2) {
    print "2:", log(2*$_), "\n"
  }
  system "sleep 0.1"
}' ; read) | \
perl ./driveGnuPlotStreams.pl 3 2 50 50 -1 1 -2 6 500x300+0+0 500x300+500+0 'sin' 'cos' 'log' 0 0 1
```

The result looks like this: