Introducing (the new) CLNUPlot

James E. Prewett

December 20, 2009

- Motivation
- Why CLNUPlot?
- The original CLNUPlot
- The New CLNUPlot is different
  - The new CLNUPlot
- CLNUPlot “plots”
  - :PLOT and :SPLOT
  - Special printing rules for plot and splot arguments
  - :REPLOT and :REFRESH
  - :SETTINGS
  - :DEFINITIONS
  - :VERBATIM
  - Other special printing rules
- Request for Comments

Motivation

- Using gnuplot to plot scheduler simulation output for work
- Scheduler simulation output was being parsed by CL (duh!)
- Requirements seemed to be changing a lot
- gnuplot is a moving target (4.2 and 4.4 are different beasts!)
- Needed something that could write arbitrary gnuplot command files

Why CLNUPlot?

I sent in a request to the admins at common-lisp.net asking them to host a new gnuplot package named gnuCLot :)
Gary King responded that I should just take over the CLNUPlot package.
I didn't want to muck with the webpages, etc. currently at c-l.net until I had something remotely reasonable to unleash upon the world
You can still easily get Gary's version of CLNUPLot with asdf-install (by the CLNUPLot name anyway)!!! Hurry!

The original CLNUPlot

The following is Gary’s example from the (old) CLNUPlot webpage:

```
(clnuplot:write-plot
 (clnuplot:make-plot
  :lines-points
  ’((1 2) (2 2.5) (3 3.1) (4 3.4) (5 4.2))
  :pointsize 2.0
  :linewidth 3.0
  :filename "simple-example"
  :xlabel "Bin Number"
  :ylabel "Dance Partners"
  :x-coord #’first
  :y-coord #’second
  :title "Bin Number versus Dance Partners"
  :ymin 0.0)
 :pdf)
```

The New CLNUPlot is *different*

- Gary King's CLNUPlot created CLOS objects you could play with and introspect on, etc.
- I didn't have time to make his code do what I needed
- The new CLNUPlot is more of a "translation engine" that takes SEXPRs as input and writes gnuplot command file output to a stream (or file).

The new CLNUPlot

```
(gnuplot "/tmp/test1.gnu"
         '((:settings
            (:term :gif :size (:list 320 240))
            (:output "test1.gif")
            (:xlabel "Bin Number")
            (:ylabel "Dance Partners")
            (:title "Bin Number versus Dance Partners")
            (:yrange (:range 0.0 NIL)))
           (:plot
            (:data ((1 2) (2 2.5) (3 3.1) (4 3.4) (5 4.2))
             :linewidth 3.0
             :pointsize 2.0
             :with :linespoints))))
```

Previous example’s output

```
set term gif size 320, 240
set output "test1.gif"
set xlabel "Bin Number"
set ylabel "Dance Partners"
set title "Bin Number versus Dance Partners"
set yrange [0.0:]
plot ’-’ linewidth 3.0 pointsize 2.0 with linespoints \
   title ’-’
1 2
2 2.5
3 3.1
4 3.4
5 4.2
e
```

CLNUPlot “plots”

`GNUPLOT` method takes as its args a stream (or filename) to write to and &rest plots.
Each plot list consists of lists beginning with the following keywords:

- `:PLOT`
- `:SPLOT`
- `:REPLOT`
- `:REFRESH`
- `:SETTINGS`
- `:DEFINITIONS`
- `:VERBATIM`

:PLOT and :SPLOT

- `:PLOT` and `:SPLOT` lists describe what it is that is to be plotted and how (not global settings!).
- Following the `:PLOT` or `:SPLOT` keyword is a list of lists describing the datasets to be plotted preceeded by (optional) `:RANGE` lists.
- The dataset lists contain information about the data to be plotted and how. Each dataset may be plotted using a different style, etc.
- "special" keywords are:
  - :DATA
  - :TITLE
  - :NAME
  - :FUNCTION
  - :RANGE
  - :ITERATION

Special printing rules for plot and splot arguments:

- `:DATA` lists are saved and printed at the end of the gnuplot command file, separated by lines containing only the character 'e'.
- `:TITLE`s are printed as the word "title" followed by a ' delimited title.
- `:NAME`s are printed within single quotes[1]
- `:FUNCTION`s are printed verbatim.
- `:RANGE`s with two elements are printed inside square brackets, separated by a colon. Those with three elements are printed as the first element, followed by an '=' sign, then the last two are printed as before.
- `:ITERATION`s are not yet implemented!!!!

[1] This should be changed to be configurable!

:REPLOT and :REFRESH

:REPLOT and :REFRESH cause "replot" and "refresh" to be printed on a line by itself.
You cannot mix :REPLOT, :REFRESH, :PLOT, and :SPLOT.

:SETTINGS

Items (lists) in the settings list are printed as "set" followed by the rest of the items printed as with PRINC converted to lower case.

:DEFINITIONS

Items (lists) in the definitions list are printed as the CAR printed as if with PRINC, and "=" sign, then the CADR printed as if with PRINC.

:VERBATIM

:VERBATIM prints its contents verbatim.
Its to cover "everything else". :)

Other special printing rules

- Lists beginning with the keyword `:VERBATIM` are printed verbatim.
- Lists beginning with the keyword `:LIST` are printed as with the items inside separated by commas.
- Lists beginning with the keyword `:USING` are printed as the word "using" followed by the rest of the items in the list colon separated, with the whole thing curly-brace enclosed e.g. {1:2:(0)}
- `:RANGE`s with two elements are printed inside square brackets, separated by a colon. Those with three elements are printed as the first element, followed by an '=' sign, then the last two are printed as before.
- Inner lists (not starting with a special keyword) have their elements printed acccording to these rules.

Other special printing rules (cont.)

- Strings are printed as with `PRINC`
- Everything else is printed, as with `PRINC` with everything converted to lower-case.

PLEASE HELP!

- Should the `:DATA` sublist have another special keyword for specifying an output file?
- Hows it lookin?