Gaston

A gnuplot library for Julia

v. 0.2

M. Bazdresch

April 10, 2012

Please note: Gaston is currently under development, and all functions and definitions are subject to change from one version to the next, as we figure out the best way to organize the code. Gaston has been tested on Linux (Ubuntu 10.04 and Arch), with gnuplot 4.6 and WxWindows. It only supports the wxt terminal.

1 Introduction

Gaston provides a way to plot scientific data using the Julia programming language. It accomplishes this by harnessing gnuplot, a versatile and time-tested plotting utility. Gaston also relies on gnuplot for interacting with plots (zooming and rotating a plot with the mouse, for instance).

The primary purpose of Gaston is to provide easy-to-use functions to quickly and conveniently plot the most common kinds of scientific and numeric data. It is concerned only with screen output (although we plan to support printing to a file in a future version), and provides only for the most common plot configurations. Tweaking a plot to produce a specific look or producing publication-quality graphics are outside its scope.

2 Installation

To use Gaston, follow this procedure:

1. Save the files `gaston.jl`, `gastonini.jl`, `gastonaux.jl` and `gastondemo.jl` somewhere convenient. Then, you may `cd` to that directory and start julia there, or do

   ```
   push(LOAD_PATH, "/path/to/gaston/jl")
   ```

Then, load the program with

```
load("gaston.jl")
```

2. To run a demo, do

```
load("gastondemo.jl")
demo()
```

This will create a series of figures that illustrate the current capabilities of the program. The same file may also serve as a guide on how to create different types of plots.

3 Plotting

In future versions, Gaston will offer plotting functions similar to Octave or Matlab's – what we call "high-level" functions. At the moment, it offers a "mid-level" set of functions that require that plots be created step-by-step. Before delving into that process, some definitions are in order.

3.1 Definitions

A **figure** is an independent window, which contains a set of axes, on which one or more **curves** are plotted. A figure may contain **labels** (for instance, on each coordinate axis), a **title**, and a **legend box** which identifies each curve. Gaston supports having any number of figures open at the same time; however, gnuplot requires that only one figure is able to offer mouse interactivity at a given time. Each figure is identified by a unique **handle**. Handles are natural numbers.

A curve is defined by a set of coordinates. Two-dimensional curves have **x** and **y** coordinates; in three dimensions, an additional **z** coordinate must be specified. A curve also has a plotting configuration (for instance, it may have a **linestyle**, a **linewidth**, a **linecolor**, etc), which define how the curve is to be plotted.

Please note that there is a single command to plot, which is `plot()`. According to the type of coordinates and plotstyle, it will figure out how to plot.

3.2 2-D plotting

Plotting proceeds in steps:

1. Create or select a figure with `figure(i)`, where `i` is a positive integer.
2. Add a curve (a set of coordinates plus a plot configuration), with `addcoords(x,y,conf)`. Here, `x` and `y` are vectors, and `conf` configures the plot and line styles, markers, legend, color, etc. Repeat this step for each curve you wish to include in the figure.

3. Add a configuration for the entire figure (axis), with `addconf(conf)`, where `conf` contains the figure configuration.

4. Issue the `plot()` command.

A curve configuration is created as follows:

1. Create a default configuration with `c = Curve_conf()`.

2. `c` is a structure, each of whose fields controls one aspect of the curve's configuration. These fields may be set individually. Available fields are: `legend`, `plotstyle`, `color`, `marker`, `linewidth`, and `pointsize`. For instance, to change a curve's color, do[1]

   ```
   c.color = "blue"
   ```

   See gnuplot's documentation to see the range of valid options.

A figure (axis) configuration is created as follows:

1. Create a default configuration with `a = Axes_conf()`

2. Just as in the case of a curve configuration, `a` is a structure whose fields may be modified. Available fields are: `title`, `xlabel`, `ylabel`, `zlabel`, `box`, `axis`.

Several rules apply:

- You can create as many figures, each with as many curves, as desired.
- Generally, if you don't provide some of the data, it will be inferred. For example, calling `addcoords` with a single vector `y` will assume the `x` coordinate is `1:length(y)` and set up the default plot configuration.
- If you call `addcoords` with matrix arguments, each column will be interpreted as a different plot.
- Calling `addcoords()` will create a new figure if none have been created yet.
- Calling `plot()` without an axis configuration will just use one by default.
- Gnuplot only provides mouse interaction support for the current figure. To use the mouse in a previously created figure `i`, just issue command `figure(i)`. This will also bring the figure to the front.

[1] Directly changing a structure's fields, instead of using setter functions, may be seen as non-idiomatic or inelegant. In this case we have decided to do the simplest thing that might possibly work.

3.2.1 2-D plotting styles

Besides simply plotting a set of `x` and `y` coordinates, Gaston supports other kinds of plots.

Error bars and lines. To add error bars or lines, just call `addcoords()` with one or two extra coordinates, and configure the plotstyle accordingly.

Histograms. To plot the histogram of a data vector `data` with `b` bins, first use the auxiliary function `(x,y) = histdata(data,b)` to create `x` and `y` coordinates that may be plotted with the `boxes` plotstyle.

3.3 3-D plotting

The same rules apply, except that `addcoords()` should be called as `addcoords(x,y,Z)`, where `Z` is a matrix whose element `j,k` corresponds to some function of `x[j],y[k]`.

For convenience, a function `Z = meshgrid(x,y,f)` is provided. Called with `x`, `y` coordinates and a function `f`, it will return a matrix that may be used to plot `f`.

3.4 Image plotting

Two image plotstyles are supported: `image` and `rbgimage`. At the moment Gaston requires that a figure contains only one image (and no other curves). This restriction will be removed in future versions.

Scalar image. To plot a matrix `Z` as a figure, use `addcoords([],[],Z)` with empty `x`, `y` coordinates and `Z` as third argument, and set the plotstyle to `image`. Element `(j,k)` of `Z` corresponds to element `(j,k)` in the plot, and its value determines the color at that point.

RGB image. To plot an RGB image, the matrix `Z` must be three dimensional. The first and second dimensions correspond to the value at each point in the image. The third dimension specifies red, green and blue values at each point. Finally, set the plotstyle to `rgbimage`.