Interactive Prototyping of Statistical Graphics with WeBIPP

Jimmy Oh
Department of Statistics
University of Auckland

1 Introduction

1.1 What is WeBIPP?

✓ Web-Based (no install required, just a browser)
✓ Interactive (Point-and-Click GUI)
✓ Build Statistical Graphics from scratch
  – 3 minutes to build a scatterplot
  – 5 minutes to build a dotchart
  – 10 minutes to build a population pyramid
  – 15 minutes to reproduce Minard's plot
✓ No coding required, but if you're comfortable with code...

1.2 Interactively write code

✓ Every action recorded as a function call
✓ Each step of building graphic fully reproducible
✓ Generalise a single graphic to other datasets
✓ Share your generalised plot as an addon
✓ Easy to add custom code to go beyond interface

2 R Interface

2.1 Limitations of WeBIPP

✗ Hard to manipulate data
✗ Hard to conduct formal analysis

2.2 Connect to R

✓ Integrated R terminal alongside WeBIPP
  – Conduct data manipulation and analysis in R
  – Send results to WeBIPP and visualise!
✗ Some restrictions due to security concerns
  – Will become more powerful and easier to use with further development

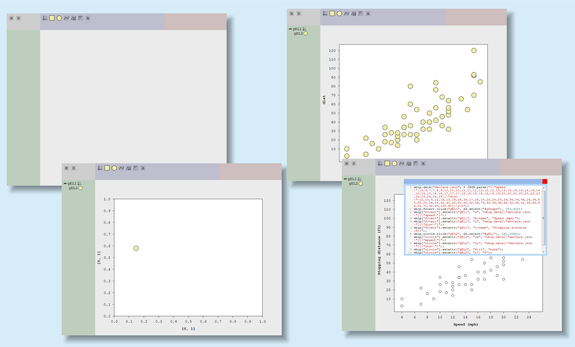
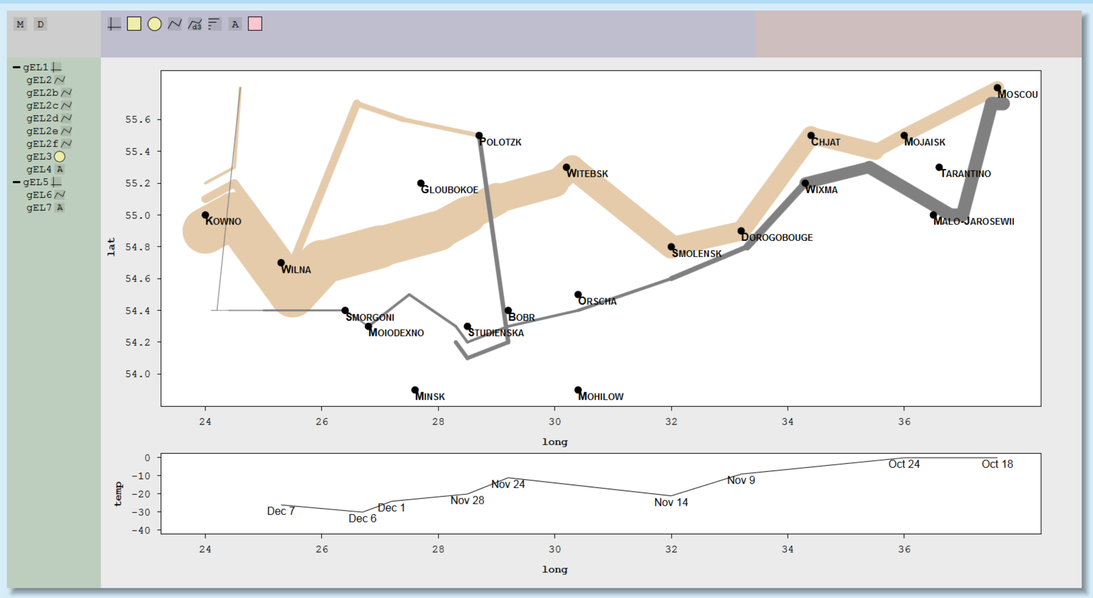
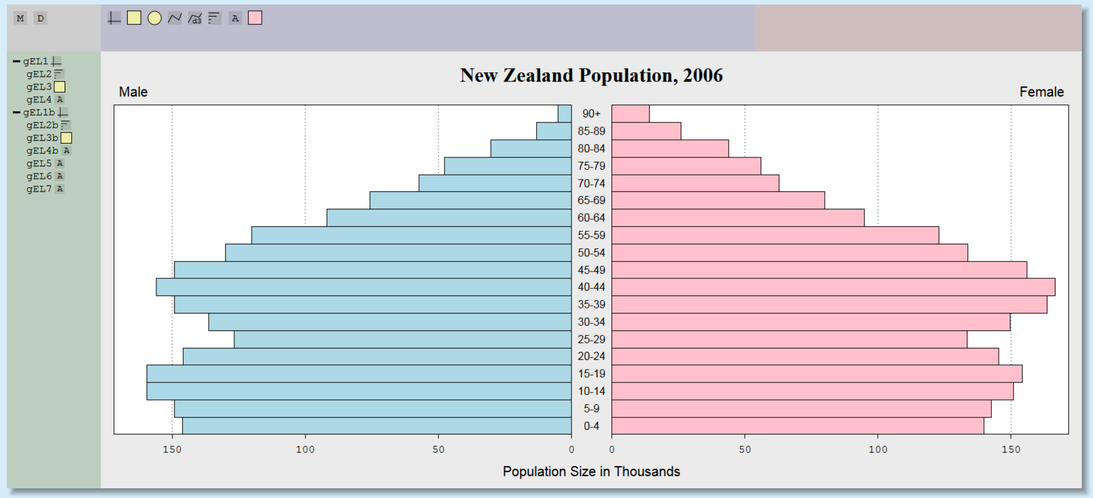
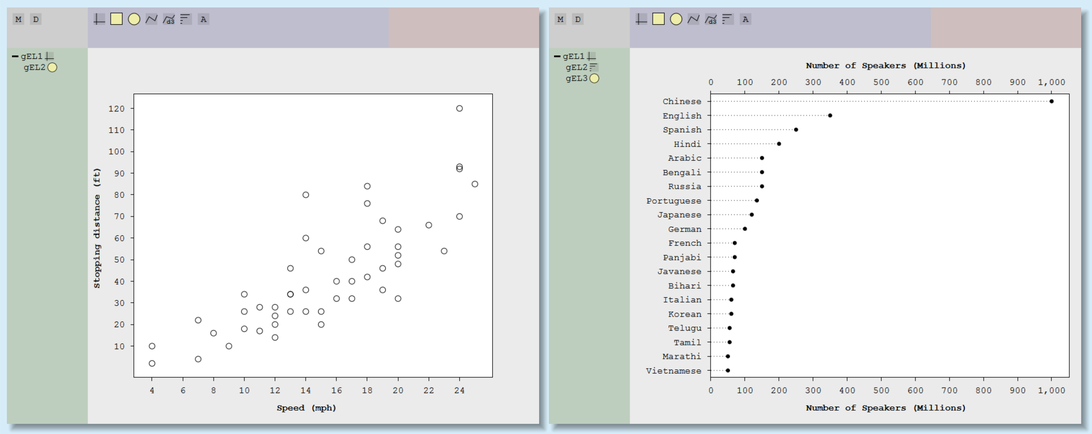


Figure 1: *Building a scatterplot*
Beginning with a blank page, by adding two objects (a frame and a circle), and then binding data to attributes of these objects (x and y), a scatterplot can be built from scratch rapidly.







3 Design Philosophy

3.1 Motivation

There are two broad methods of creating a statistical graphic:

1. Use a 'pre-packaged' solution, e.g. `plot` in R
   – Easy, but limited by what the pre-packaged solution can do
2. Code it yourself, e.g. using functions like `points` in R
   – Powerful, but requires time and effort

3.2 The Bridge Solution

WeBIPP is a bridge between the two methods:

✓ Can include 'pre-packaged' solutions as addons
✓ Access to low-level base objects via an interface
  – Typically using interface is much faster than writing code
  – But if writing code is faster, can still just write code!

Use of the WeBIPP interface quietly generates function calls in the background (e.g. to call `wbip.circle.setattr`). These WeBIPP functions make it easier to manipulate lower-level code. This is a similar concept to that used by the popular JavaScript Graphics library *D3.js*[1] (and WeBIPP itself uses D3.js to generate the final graphic), with a key advantage being that using WeBIPP does not limit the user in any way.
Anything possible at the lowest levels through direct manipulation of HTML, SVG, CSS and/or D3.js is still possible, it can only be made easier with the help of WeBIPP functions.

3.3 Everything is an addon

Another feature of WeBIPP's design is that almost everything is an addon, even 'core' features like the overall layout, buttons to add circles to the graphic, and the main menu items.
This means every aspect of WeBIPP must be deliberately designed for addons and modular capability in mind, and in turn, creating new addons for WeBIPP should be (relatively) painless.
This philosophy extends to the R interface, with a *Message* addon enabling WeBIPP to accept HTML messages. Though currently tuned to work with a specific R web interface, it could easily be extended to communicate with other software.

4 Future Work

Most of the development so far has been on laying down a strong foundation, to make it as easy as possible to create, share and include new addons. There is still some work to be done in this area.
Other work includes improvements to the user interface and documentation, to make WeBIPP more user friendly.
Keep posted on the latest developments:
`https://www.stat.auckland.ac.nz/~joh024/Research/WeBIPP/`

References

[1] Bostock, M., V. Ogievetsky, and J. Heer (2011). D3: Data-driven documents. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*.