Approaches to Web-based Interactive and Dynamic Statistical Graphics for Data Analysis

Carson Sievert

Abstract

In recent years, we've seen an explosion of software for creating web-based interactive graphics. These libraries are typically written entirely in `JavaScript`, the programming language of the web, and possess the wonderful quality of working in any modern web browser. Interfaces to these libraries in languages such as `R`, `python`, and `Julia` allow users without any `JavaScript` (or `HTML`) knowledge to create interactive graphics from environments with rich support for data analysis, data management, and statistical computation. These interfaces typically sacrifice flexibility for usability, so they are more restrictive than writing custom `JavaScript`; but if the abstraction fits the use case, they are incredibly useful and time saving. This paper examines such interfaces for the `R` language, and classifies their usages???

Introduction

- Why interactive?
- Why web-based?

Background

The history of interactive and dynamic statistical graphics is very broad, deep, and existed well before the advent of the World Wide Web. Statisticians tend to think of interactive and dynamic graphics as a tool for conducting initial data analysis (IDA) and/or exploratory data analysis (EDA), but as (**???**) points out, the potential applications are more broad:

> "The statistics community creates visualization systems within the context of data analysis, so the graphics are designed to support and enrich the statistical processes of data exploration, modeling, and inference."

In order for a visualization system to enrich the modeling process, it must be able to communicate with statistical software. In addition, a general purpose system needs to interface with software and/or provide utilities for manipulating data into a form suitable for visualization. As discussed in (Tierney 2005), this interface must be flexible and extensible. For example, `Lisp-Stat` provided statistical computation facilities via custom `LISP` bindings, as well as rich support for interactive graphics, but `LISP` lacks a packaging mechanism which allows users to extend it's functionality. Especially within academic circles, the ability to interface with the most current statistical methodology is of utmost importance.

One of the great success stories of the `R` language (R Core Team 2015) is the massive collection of user contributed packages which implement the latest breakthroughs in statistical computing and graphics. Before the Web browser became a viable platform for interactive graphics, a number of R packages provided interfaces to interactive graphics systems.

There are a large number of `R` packages that leverage this ecosystem by allowing users to design interactive visualizations from the R console ((Yihui Xie 2013); (Urbanek 2011); (Wickham et al. 2008); (T. D. Hocking, VanderPlas, and Sievert 2015); (Chang and Wickham 2015)). However, these packages differ dramatically in their choice for a graphics device as well as their reactive programming frameworks. These choices have significant consequences on the scope, quality, and usability of the software.

Interactive Web-Based Graphics

Thanks to the constant evolution and eventual adoption of `HTML5` as a Web standard, the modern Web browser provides a viable platform for building interactive statistical graphics systems. `HTML5` refers to a collection of technologies, each designed to perform a certain task. For example, `SVG` provides markup for drawing vector based graphics, `CSS` provides conventions for styling the Web page, and `JavaScript` provides event handling capabilities. `HTML5` technologies are publicly available, and benefit from thriving community of open source developers and volunteers. Perhaps the most important contribution is Data Driven Documents (`D3`), a `JavaScript` library which provides high-level semantics for binding data to Web elements (e.g., `SVG` elements) (Heer 2011)

Problems that web-based int. graphs still have to solve: * ability to link multiple windows * rendering many points (WebGL?) * How should the reactive model be designed? -> where should the central commander reside? The DOM (animint)? A dedicated R server (shiny)?

Figure converters

A growing trend is to convert static graphics into a format which supports interaction and animation, such as `SVG` (Murrell and Potter 2015) (Nolan and Lang 2012) (Riutta et. al. and Russell 2015) (T. D. Hocking, VanderPlas, and Sievert 2015) (Sievert et al.). A major benefit to this approach is that existing code/concepts can be used to create interactive versions of the same graphic. As discussed in detail later, we can also extend the grammar of *static* graphics to enable non-trivial interactive and dynamic features.

Wrapping `JSON` specifications

- plotly
- animint

Writing `JSON` specifications

- LDAvis

The space of possible visualization states is pre-computed to enable smooth transitions that preserve object constancy. If the states are not pre-computed, they need to be computed quickly on-the-fly so that users may make observations, draw generalizations and generate hypotheses [@2014-latency].

General Approaches

- ggvis
- vcd

Domain Specific Visualization Systems

- LDAvis
- summarytrees
- qtlcharts

Discussion

We've got a long way to go before we catch up to non-web-based interactive graphics systems

References

Chang, Winston, and Hadley Wickham. 2015. *Ggvis: Interactive Grammar of Graphics*. http://CRAN.R-project.org/package=ggvis.

Heer, Michael Bostock AND Vadim Ogievetsky AND Jeffrey. 2011. "D3: Data-Driven Documents." *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*. http://vis.stanford.edu/papers/d3.

Hocking, Toby Dylan, Susan VanderPlas, and Carson Sievert. 2015. *Animint: Interactive Animations*.

Murrell, Paul, and Simon Potter. 2015. *GridSVG: Export Grid Graphics as SVG*. http://CRAN.R-project.org/package=gridSVG.

Nolan, Deborah, and Duncan Temple Lang. 2012. "Interactive and Animated Scalable Vector Graphics and R Data Displays." *Journal of Statistical Software* 46 (1): 1–88. http://www.jstatsoft.org/v46/i01/.

R Core Team. 2015. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. http://www.R-project.org/.

Riutta et. al., Anders, and Kent Russell. 2015. *SvgPanZoom: R 'Htmlwidget' to Add Pan and Zoom to Almost Any R Graphic*. http://CRAN.R-project.org/package=svgPanZoom.

Sievert, Carson, Chris Parmer, Toby Hocking, Scott Chamberlain, Karthik Ram, Marianne Corvellec, and Pedro Despouy. *Plotly: Create Interactive Web-Based Graphs via Plotly's API*. https://github.com/ropensci/plotly.

Tierney, Luke. 2005. "Some Notes on the Past and Future of Lisp-Stat." *The Journal of Statistical Software* 13 (9). http://www.jstatsoft.org/v13/a09/paper.

Urbanek, Simon. 2011. *Acinonyx: IPlots EXtreme*. http://www.rforge.net/Acinonyx/.

Wickham, Hadley, Michael Lawrence, Duncan Temple Lang, and Deborah F Swayne. 2008. "An Introduction to Rggobi." *R-News* 8 (2): 3–7. http://CRAN.R-project.org/doc/Rnews/Rnews_2008-2.pdf.

Yihui Xie, Di Cook, Heike Hofmann. 2013. *Interactive Statistical Graphics Based on Qt*.