Interfacing R with web technologies for interactive statistical graphics and computing with data

by

Carson Sievert

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Statistics

Program of Study Committee:

Heike Hofmann, Major Professor

Dianne Cook

Jarad Niemi

Ulrike Genschel

Grayson Calhoun

Iowa State University

Ames, Iowa

2016

TABLE OF CONTENTS

LIST OF TABLES **vi**

LIST OF FIGURES **viii**

ACKNOWLEDGEMENTS **xxiv**

CHAPTER 1. PROBLEM STATEMENT **1**

CHAPTER 2. OVERVIEW **3**

2.1 What makes a good statistical software interface? 3

2.1.1 Synergy between interfaces 3

2.1.2 Synergy among programming interfaces 8

2.2 Acquiring and wrangling web content in R 10

2.2.1 Interfaces for working with web content 10

2.2.2 Interfaces for acquiring data on the web 13

2.3 Interactive statistical web graphics 15

2.3.1 Why interactive graphics? 15

2.3.2 Web graphics 18

2.3.3 Translating R graphics to the web 20

2.3.4 Interfacing with interactive web graphics 22

2.3.5 Multiple MVC paradigms 23

2.3.6 Hybrid MVC for one interface 28
2.3.7 MVC for multiple interfaces 29

CHAPTER 3. TAMING PITCHF/X DATA WITH XML2R AND PITCHRX **33**

3.1 Introduction 34
3.1.1 What is PITCHf/x? 34
3.1.2 Why is PITCHf/x important? 34
3.1.3 PITCHf/x applications 35
3.1.4 Contributions of pitchRx and XML2R 35
3.2 Getting familiar with Gameday 36
3.3 Introducing XML2R 39
3.3.1 Constructing file names 39
3.3.2 Extracting observations 40
3.3.3 Renaming observations 42
3.3.4 Linking observations 44
3.3.5 Collapsing observations 45
3.4 Collecting Gameday data with pitchRx 46
3.5 Storing and querying Gameday data 48
3.6 Visualizing PITCHf/x 50
3.6.1 Strike-zone plots and umpire bias 50
3.6.2 2D animation 59
3.6.3 Interactive 3D graphics 62
3.7 Conclusion 64

CHAPTER 4. LDAVIS: A METHOD FOR VISUALIZING AND INTERPRETING TOPICS **66**

4.1 Introduction 67

4.2 Related Work 70
4.2.1 Topic Interpretation and Coherence 70
4.2.2 Topic Model Visualization Systems 72
4.3 Relevance of tokens to topics 73
4.3.1 Definition of Relevance 73
4.3.2 User Study 74
4.4 Our Visualization System 79
4.5 Discussion 83

CHAPTER 5. EXTENDING GGPLOT2'S GRAMMAR OF GRAPHICS IMPLEMENTATION FOR LINKED AND DYNAMIC GRAPHICS ON THE WEB 84

5.1 Introduction 85
5.2 Related Work 86
5.3 Extending the layered grammar of graphics 89
5.3.1 Direct Manipulation of Database Queries 90
5.3.2 Adding animation 92
5.3.3 World Bank Example 93
5.3.4 Implementation details 100
5.4 Exploring performance & scope with examples 104
5.5 Comparison study 105
5.5.1 The Grand Tour 105
5.5.2 World Bank Example 109
5.6 User feedback and observations 110
5.6.1 User perspective 110
5.6.2 Developer perspective 111
5.7 Limitations and future work 111

5.8 Conclusion 113

CHAPTER 6. INTERACTIVE DATA VISUALIZATION ON THE WEB USING R **114**

6.1 Introduction 114

6.2 Case Studies 116

6.2.1 Exploring pedestrian counts 116

6.2.2 Tracking disease outbreak 128

6.2.3 Exploring Australian election data 132

6.3 Conclusion 137

6.4 Acknowledgements 139

CHAPTER 7. PLOTLY FOR R **140**

7.1 Two approaches, one object 140

7.1.1 A case study of housing sales in Texas 141

7.1.2 Extending `ggplotly()` 153

7.2 The plotly cookbook 160

7.2.1 Scatter traces 160

7.2.2 Maps 182

7.2.3 Bars & histograms 185

7.2.4 Boxplots 191

7.2.5 2D frequencies 195

7.2.6 Other 3D plots 199

7.3 Arranging multiple views 200

7.3.1 Arranging htmlwidgets 201

7.3.2 Merging plotly objects 205

7.3.3 Navigating many views 213

7.4 Multiple linked views 216

7.4.1 Linking views with shiny 216

7.4.2 Linking views without shiny 220

7.5 Animating views 242

7.5.1 Key frame animations 242

7.5.2 Linking animated views 246

7.6 Advanced topics 251

7.6.1 Custom behavior via JavaScript 252

7.6.2 Translating custom ggplot2 geoms 253

7.6.3 Designing an htmlwidget interface 256

CHAPTER 8. IMPACT AND FUTURE WORK **259**

8.1 Impact 259

8.1.1 plotly 259

8.1.2 LDAvis 261

8.2 Future work 261

BIBLIOGRAPHY **263**

LIST OF TABLES

Table 3.1 Structure of PITCHf/x and related Gameday data sources accessible to ‘scrape()‘ 38

Table 5.1 Characteristics of 11 interactive visualizations designed with animint. The interactive version of these visualizations can be accessed via http://sugiyama-www.cs.titech.ac.jp/~toby/animint/. From left to right, we show the data set name, the lines of R code (LOC) including data processing but not including comments (80 characters max per line), the amount of time it takes to compile the visualization (seconds), the total size of the uncompressed TSV files in megabytes (MB), the total number of data points (rows), the median number of data points shown at once (onscreen), the number of data columns visualized (variables), the number of `clickSelects`/`showSelected` variables (interactive), the number of linked panels (plots), if the plot is animated, and the corresponding Figure number in this paper (Fig). 106

LIST OF FIGURES

Figure 2.1 A basic visual depiction of linked views in a standalone web page (A) versus a client-server model (B). In some cases (A), linked views can be resolved within a web browser, which generally leads to a better user experience. In other cases (B), updating views may require calls to a web server running special software. 9

Figure 2.2 A video demonstration of interactive and dynamic techniques for visualizing high-dimensional relationships in data using the R package **tourbrush**. You can view this movie online at https://vimeo.com/148050343. 15

Figure 2.3 Four different MVC paradigms. In all the scenarios, the graph acts as the controller, but the model (i.e., the data and logic which updates the view) exists in different places. In Scenario A, a mouse hover event manipulates the model within the underlying JavaScript library. In Scenario B, a window resizing manipulates the model within the HTMLwidget.resize() method, defined by the widget author. In Scenario C, a mouse hover event manipulates the model within the underlying JavaScript library *and* a model defined by both the user (in R) and the widget author (in JavaScript). In Scenario D, removing outliers from the raw data may require R code to be executed. 25

Figure 2.4 Linking views in plotly with shiny (scenario B) versus without shiny (scenario A). 30

Figure 2.5 The GGobi pipeline, as described by Lawrence (2002), in comparison to a centralized pipeline. The GGobi pipeline is shown in peach color while the centralized pipeline is in both yellow and blue to point out multiple interfaces can be linked in a centralized pipeline. 31

Figure 2.6 The MVC design for linking multiple htmlwidgets with crosstalk 32

Figure 3.1 Table relations between Gameday data accessible via `scrape()`. The direction of the arrows indicate a one to possibly many relationship. 38

Figure 3.2 Density of called strikes for right-handed batters and left-handed batters (from 2008 to 2013). 52

Figure 3.3 Density of called strikes minus density of balls for both right-handed batters and left-handed batters (from 2008 to 2013). The blue region indicates a higher frequency of called strikes and the red region indicates a higher frequency of balls. 54

Figure 3.4 Probability that a right-handed away pitcher receives a called strike (provided the umpire has to make a decision). Plots are faceted by the handedness of the batter. 57

Figure 3.5 Difference between home and away pitchers in the probability of a strike (provided the umpire has to make a decision). The blue regions indicate a higher probability of a strike for home pitchers and red regions indicate a higher probability of a strike for away pitchers. Plots are faceted by the handedness of both the pitcher and the batter. 58

Figure 3.6 The last frame of an animation of every four-seam and cutting fastballs thrown by NY Yankee pitchers Mariano Rivera and Phil Hughes during the 2011 season. The actual animation can be viewed at http://cpsievert.github.io/pitchRx/ani1. Pitches are faceted by pitcher and batting stance. For instance, the top left plot portrays pitches thrown by Rivera to left-handed batters. . . 61

Figure 3.7 The last frame of an animation of averaged four-seam and cutting fastballs thrown by NY Yankee pitchers Mariano Rivera and Phil Hughes during the 2011 season. The actual animation can be viewed at http://cpsievert.github.io/pitchRx/ani2. PITCHf/x parameters are averaged over pitch type, pitcher and batting stance. For instance, the bottom right plot portrays an average four-seam and average cutter thrown by Hughes to right-handed batters. 63

Figure 3.8 3D scatterplot of pitches from Rivera. Pitches are plotted every one-hundredth of a second. Cutting fastballs are shown in red and four-seam fastballs are shown in blue. The left hand plot takes a viewpoint of Rivera and the right hand plot takes a viewpoint near the umpire. Note these are static pictures of an interactive object. 64

Figure 4.1 The layout of LDAvis, with the global topic view on the left, and the token barcharts on the right. Linked selections allow users to reveal aspects of the topic-token relationships compactly. 69

Figure 4.2 Dotted lines separating the top-10 most relevant tokens for different values of λ, with the most relevant tokens for $\lambda = 2/3$ displayed and highlighted in green. 75

Figure 4.3 A plot of the proportion of correct responses in a user study vs. the value of λ used to compute the most relevant tokens for each topic. 78

Figure 4.4 The user has chosen to segment the topics into four clusters, and has selected the green cluster to populate the barchart with the most relevant tokens for that cluster. Then, the user hovered over the ninth bar from the top, 'file', to display the conditional distribution over topics for this token. 80

Figure 5.1 Linked database querying via direct manipulation using animint. A video demonstration can be viewed online at https://vimeo.com/160496419 92

Figure 5.2 A simple animation with smooth transitions and interactively altering transition durations. A video demonstration can be viewed online at https://vimeo.com/160505146 94

Figure 5.3 An interactive animation of World Bank demographic data of several countries, designed using `clickSelects` and `showSelected` keywords (top). Left: a multiple time series from 1960 to 2010 of life expectancy, with bold lines showing the selected countries and a vertical grey tallrect showing the selected year. Right: a scatterplot of life expectancy versus fertility rate of all countries. The legend and text elements show the current selection: year=1979, country= {United States, Vietnam}, and region={East Asia & Pacific, North America} 95

Figure 5.4 Animint provides a menu to update each selection variable. In this example, after typing 'th' the country menu shows the subset of matching countries. 97

Figure 5.5 A schematic explanation of compilation and rendering in the World Bank visualization. Top: the interactive animation is a list of 4 R objects: 2 ggplots and 2 option lists. Center: animint R code compiles data in ggplot geoms to a database of TSV files (→). It also compiles plot meta-data including ggplot aesthetics, animation time options, and transition duration options to a JSON meta-data file (→). Bottom: those data-dependent compiled files are combined with data-independent JavaScript and HTML files which render the interactive animation in a web browser (→). . . . 101

Figure 5.6 Interactive animation of tornadoes recorded from 1950 to 2012 in the United States. Left: map of the lower 48 United States with tornado paths in 1982. The text shows the selected year, and clicking the map changes the selected state, currently Texas. Right: time series of tornado counts in Texas. Clicking a bar changes the selected year, and the text shows selected state and the number of tornadoes recorded there in that year (119 tornadoes in Texas in 1982). 102

Figure 5.7 Visualization containing 6 linked, interactive, animated plots of Central American climate data. Top: for the selected time (December 1997), maps displaying the spatial distribution of two temperature variables, and a scatterplot of these two variables. The selected region is displayed with a black outline, and can be changed by clicking a rect on the map or a point on the scatterplot. Bottom: time series of the two temperature variables with the selected region shown in violet, and a scatterplot of all times for that region. The selected time can be changed by clicking a background tallrect on a time series or a point on the scatterplot. The selected region can be changed by clicking a line on a time series. 103

Figure 5.8 Linked selection in a grand tour with animint. A video demonstration can be viewed online at https://vimeo.com/160720834 . 107

Figure 5.9 Linked selection in a grand tour with ggvis and shiny. A video demonstration can be viewed online at https://vimeo.com/160825528 108

Figure 6.1 Missing values by station. 117

Figure 6.2 An interactive bar chart of the number of missing counts by station linked to a sampled time series of counts. See here for the corresponding video and here for the interactive figure. 118

Figure 6.3 Two frames from a grand tour of measures generated from seasonal, trend, and irregular time-series components. The first frame (the top row) displays the state of the tour roughly 16 seconds into the animation while the second frame (the bottom row) is at roughly 60 seconds. A given frame displays both a 2D projection (on the left) and the linear combination of variables used for the projection (on the right). In both frames, the Tin Alley-Swanson St (West) sensor is highlighted in red – a useful technique for tracking interesting or unusual point(s) throughout a tour. 121

Figure 6.4 Identifying and comparing unusual sensors (Tin Alley and Swanson St) using linked highlighting between a grand tour and a parallel coordinates plot. The second frame of Figure 6.3 helped to point out Bourke St as a somewhat unusual sensor with respect to trend. Highlighting that point and linking it to a parallel coordinate plot makes it easier to compare trend across sensors and compare the other measures among sensors of interest. 122

Figure 6.5 Linking views of seasonal trend decomposition summaries (first two rows) to the actual time series (last two rows). By linking raw counts and the hourly IQR, we can see that Tin Alley (in red) experiences relatively low traffic compared to Bourke St (in blue), and overall traffic (black). See here for the corresponding video and here for the interactive figure. 124

Figure 6.6 Seventeen time series features linked to a geographic map as well as raw counts. This static image was generated using a persistent brush to compare Tin Alley-Swanson St. (in red) to Waterfront City (in blue). In addition to being unusual in the feature space, these sensors are also on the outskirts of the city. The corresponding video and interactive figure (available here and here) also includes a grand tour and raw counts by day of the year. . . 126

Figure 6.7 Sensors with high first order autocorrelation (in red) versus sensors with low autocorrelation (in blue). See here for the corresponding video and here for the interactive figure. 127

Figure 6.8 Linking a dendrogram of hierarchical clustering results to multiple views of the raw data. See here for the corresponding video and here for the interactive figure. 128

Figure 6.9 Multiple views of the Zika outbreak data. On the left-hand side is a map of the reporting locations. On the right is the overall density of suspected/confirmed cases reported per week (on a log scale), and the overall weekly median over time. 129

Figure 6.10 A comparison of the overall cases (in black) to the cases conditional on the map bounds (in red). Zooming and panning the interactive map dynamically updates the density estimates and median number of incidents. 130

Figure 6.11 Zooming and panning to a region of the map that has a negative median of overall cases (Nicaragua). A video of the zooming and panning may be viewed here. 131

Figure 6.12 Cumulative confirmed (in red) and suspected (in blue) counts by location within 9 different countries. 132

Figure 6.13 Highlighting cumulative confirmed (in red) and suspected (in blue) counts by location within Colombia to verify re-classifications from confirmed to suspected. A video of the interactive highlighting may be viewed here. 133

Figure 6.14 Electorate demographics among the Liberal Party (in green), the Australian Labor Party (in orange), and other parties (in black). The vertical lines represent the mean value within each group. The interactive application used to generate this image may be accessed here and a video of the interactive highlighting may be viewed here. 134

Figure 6.15 Comparing voting outcomes and geographic location among the Liberal Party (in green), the Australian Labor Party (in orange), and other parties (in black). The bar chart in the upper-left hand panel shows the number of electorates won by each party. The upper-right hand panel shows the proportion of 1st preference votes for each party for given electorate. The lower-left hand panel shows the absolute difference in vote totals for each electorate. The lower-right hand panel show the locations of electorates. . . 136

Figure 6.16 Electorates that were determined by less than 10 percent of the total vote. These electorates tend to have voters that are younger, less religious, are less likely to own property, and lean towards the Labor party. 137

Figure 6.17 Electorates that experienced a close election as well as electorates with small populations (in orange). 138

Figure 7.1 Monthly median house price in the state of Texas. The top row displays the raw data (by city) and the bottom row shows 2D binning on the raw data. The binning is helpful for showing the overall trend, but hovering on the lines in the top row helps reveal more detailed information about each city. 143

Figure 7.2 Monthly median house price in Houston in comparison to other Texan cities. 146

Figure 7.3 Monthly median house price in Houston and San Antonio in comparison to other Texan cities. 149

Figure 7.4 First, second, and third quartile of median monthly house price in Texas. 151

Figure 7.5 Layering on a 4-year forecast from a exponential smoothing state space model. 152

Figure 7.6 Customizing the dragmode of an interactive ggplot2 graph. . . . 153

Figure 7.7 Adding a rangeslider to an interactive ggplot2 graph. 154

Figure 7.8 Using listviewer to inspect the JSON representation of a plotly object. 155

Figure 7.9 Using the `style()` function to modify hoverinfo attribute values of a plotly object created via `ggplotly()` (by default, `ggplotly()` displays hoverinfo for all traces). In this case, the hoverinfo for a fitted line and error bounds are hidden. 156

Figure 7.10 Leveraging data associated with a `geom_smooth()` layer to display additional information about the model fit. 158

Figure 7.11 Leveraging output from `StatBin` to add annotations to a stacked bar chart (created via `geom_bar()`) which makes it easier to compare bar heights. 159

Figure 7.12 Leveraging output from `StatDensity2d` to add annotations to contour levels. a stacked bar chart (created via `geom_bar()`) which makes it easier to compare bar heights. 161

Figure 7.13 Three versions of a basic scatterplot 163

Figure 7.14 Specifying symbol in a scatterplot 164

Figure 7.15 Mapping symbol to a factor 165

Figure 7.16 Variations on a numeric color mapping. 166

Figure 7.17 Three variations on a numeric color mapping 167

Figure 7.18 Three variations on a discrete color mapping 168

Figure 7.19 Controlling the size range via `sizes` (measured in pixels). 169

Figure 7.20 A 3D scatterplot 169

Figure 7.21 An interactive version of the generalized pairs plot made via the `ggpairs()` function from the **GGally** package 171

Figure 7.22 A coefficient plot 173

Figure 7.23 Median house sales with one trace per city. 174

Figure 7.24 Using `color` and/or `linetype` to differentiate groups of lines. . . 175

Figure 7.25 Various kernel density estimates. 176

Figure 7.26 Parallel coordinates plots of the Iris dataset. On the left is the raw measurements. In the middle, each variable is scaled to have mean of 0 and standard deviation of 1. On the right, each variable is scaled to have a minimum of 0 and a maximum of 1. 178

Figure 7.27 A path in 3D 178

Figure 7.28 A 3D line plot 179

Figure 7.29 A candlestick chart 180

Figure 7.30 Plotting fitted values and uncertainty bounds of a linear model via the **broom** package. 181

Figure 7.31 A map of Canada using the default cartesian coordinate system. 182

Figure 7.32 Three different ways to render a map. On the top left is plotly's default cartesian coordinate system, on the top right is plotly's custom geographic layout, and on the bottom is mapbox. 184

Figure 7.33 A map of U.S. population density using the `state.x77` data from the **datasets** package. 185

Figure 7.34 plotly.js's default binning algorithm versus R's `hist()` default . . 187

Figure 7.35 Number of diamonds by cut. 188

Figure 7.36 A trellis display of diamond price by diamond clarity. 189

Figure 7.37 A grouped bar chart 190

Figure 7.38 A stacked bar chart showing the proportion of diamond clarity within cut. 191

Figure 7.39 Using ggmosaic and ggplotly() to create advanced interactive visualizations of categorical data 192

Figure 7.40 Overall diamond price and price by cut. 193

Figure 7.41 Diamond prices by cut and clarity. 193

Figure 7.42 Diamond prices by cut and clarity, sorted by price median. . . . 195

Figure 7.43 Three different uses of `histogram2d()` 196

Figure 7.44 2D Density estimation via the `kde2d()` function 198

Figure 7.45 Displaying a correlation matrix with `add_heatmap()` and controling the scale limits with `colorbar()`. 199

Figure 7.46 A 3D surface of volcano height. 200

Figure 7.47 Printing multiple htmlwidget objects with `tagList()`. To render tag lists at the command line, wrap them in `browsable()` 202

Figure 7.48 Arranging multiple htmlwidgets with flexbox 203

Figure 7.49 Arranging multiple htmlwidgets with `fluidPage()` from the **shiny** package. 204

Figure 7.50 The most basic use of `subplot()` to merge multiple plotly objects into a single plotly object. 206

Figure 7.51 Five different economic variables on different y scales and a common x scale. Zoom and pan events in the x-direction are synchronized across plots. 207

Figure 7.52 Pre-populating y axis IDs. 208

Figure 7.53 A visual diagram of controling the `heights` of rows and `widths` of columns. 208

Figure 7.54 A joint density plot with synchronized axes. 209

Figure 7.55 Recursive subplots. 210

Figure 7.56 Multiple bar charts of US statistics by state in a subplot with a choropleth of population density 212

Figure 7.57 Arranging multiple faceted ggplot2 plots into a plotly subplot. . 213

Figure 7.58 Using plotly within a trelliscope 215

Figure 7.59 A video demonstration of plotly events in shiny. The video can be accessed here 217

Figure 7.60 A video demonstration of linked brushing in a shiny app. The video can be accessed here and the code to run the example is here 219

Figure 7.61 A video demonstration of clicking on a cell in a correlation matrix to view the corresponding scatterplot. The video can be accessed here and the code to run the example is here 220

Figure 7.62 Monthly median house sales by year and city. Each panel represents a city and panels are linked by year. A video demonstrating the graphical queries can be viewed here 222

Figure 7.63 Brushing a scatterplot matrix via the `ggpairs()` function in the **GGally** package. A video demonstrating the graphical queries can be viewed here 223

Figure 7.64 Highlighting lines with transient versus persistent selection. In the left hand panel, transient selection (the default); and in the right hand panel, persistent selection. The video may be accessed here 225

Figure 7.65 Linking views between plotly and leaflet to explore the relation between magnitude and geographic location of earthquakes around Fiji. The video may be accessed here 226

Figure 7.66 Selecting cities by indirect manipulation. The video may be accessed here 228

Figure 7.67 A bar chart of cities with one or more missing median house sales linked to a time series of those sales over time. The video may be accessed here 229

Figure 7.68 A diagram of the pipeline between the data and graphics. 230

Figure 7.69 Dynamically populating a boxplot reflecting brushed observations 233

Figure 7.70 Dynamically populating a bar chart reflecting brushed observations 234

Figure 7.71 Engine displacement versus highway miles per gallon by class of car. The linear model for each class, as well as the individual observations, can be selected by hovering over the line of fitted values. An individual observation can also be selected by hovering over the relevant point. 235

Figure 7.72 Using nested selections to highlight numerous diagnostics from different regions of the design matrix. 237

Figure 7.73 Clicking on a density estimate to highlight all the raw observations that went into that estimate. 238

Figure 7.74 A simple example of hierarchial selection 240

Figure 7.75 Leveraging hierarchical selection and persistent brushing to paint branches of a dendrogram. 241

Figure 7.76 Animation of the evolution in the relationship between GDP per capita and life expectancy in numerous countries. 243

Figure 7.77 Modifying animation defaults with `animation_opts()`, `animation_button()`, and `animation_slider()`. 244

Figure 7.78 Animation of GDP per capita versus life expectancy by continent. The ordering of the contintents goes from lowest average (across countries) life expectancy to highest. 245

Figure 7.79 Overlaying animated frames on top of a background of all possible frames. 246

Figure 7.80 Highlighting the relationship between GDP per capita and life expectancy in the Americas and tracking that relationship through several decades. 247

Figure 7.81 Comparing the evolution in the relationship between per capita GDP and life expectancy in countries with large populations (red) and small populations (blue). 248

Figure 7.82 Linking a dendrogram to a grand tour and map of the `USArrests` data to visualize a classification in 5 dimensions. 250

Figure 7.83 Using `onRender()` to register a JavaScript callback that opens a google search upon a 'plotly_click' event. 253

Figure 7.84 Converting GeomXspline from the **ggalt** package to plotly.js via `ggplotly()`. 256

Figure 8.1 CRAN downloads over the past 6 months from RStudio's anonymized CRAN mirror download logs. Shown are common packages for interactive web graphics. 260

Figure 8.2 A screenshot of the htmlwidgets gallery website. This website allows you to browse R packages built on the htmlwidgets framework and sort widgets by the number of GitHub stars. As of writing, plotly has 791 stars which is the most among all htmlwidgets. 260