RIGHT: an *HTML* canvas and *JavaScript*-based interactive data visualization package for linked graphics

ChungHa Sung[1], TaeJoon Song[2], Jae W. Lee[1], and Junghoon Lee[3*]

1. Sungkyunkwan University (SKKU), Suwon, Gyeonggi-Do, 440-746, South Korea
2. Samsung Electronics, Hwaseong-Si, Gyeonggi-Do, 143-130, South Korea
3. Merck Research Laboratories, Rahway, NJ, 08901, U.S.A.
*Contact author: jung_hoon_lee@merck.com

Keywords: interactive data visualization, linked graphics, *HTML* canvas, *JavaScript*

Interactive data visualization has received broad interest in the *R* community due to its obvious benefits over static visualization: more information can be delivered concisely and intuitively by user engagement. As a result, various *R* packages supporting single-layer, multi-layer, and linked graphics have been developed, including **rCharts**, **iPlots**, **cranvas**, **ggvis**, **animint** and **googleVis** [1]. R Interactive Graphics via HTml (**RIGHT**, https://code.google.com/p/r-interactive-graphics-via-html/) is an interactive data visualization package for linked graphics based on *HTML* canvas and *JavaScript*. It provides an *R* API similar to base graphics to easily construct various interactive plots, including scatter, line, bar, pie, and box-whisker plots.

This poster presents an overview of **RIGHT** and the *JavaScript* data structure that enables linked graphics. **RIGHT** is the first package that implements linked graphs using *HTML* canvas and *JavaScript*. Linked graphics help answer obvious questions a collection of plots tend to raise: how one point in the plot is related to another point in another plot. *HTML* canvas and *JavaScript* make it possible to deliver the visualization to various platforms, including mobile devices, since they are standard web technologies supported by most modern web browsers (albeit some remaining compatibility issues). This approach can also benefit from the improvement of *JavaScript* performance every generation, driven by various web applications with ever increasing complexity and sophistication.

References

[1] Toby Dylan Hocking, https://github.com/tdhock/interactive-tutorial (as of March 16, 2014).

RIGHT: an HTML canvas and JavaScript-based interactive data visualization package for linked graphics

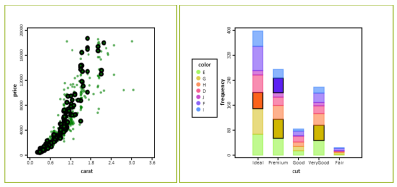
ChungHa Sung†, JongHyun Bae†, SangGi Hong†, TaeJoon Song†, Jae W. Lee†, Junghoon Lee ‡
†Sungkyunkwan University ‡Merck Research Laboratories
{sch8906, bnbbkr, bethetaedu, thepotter89, jaewlee, flammy}@gmail.com

Abstract

R Interactive Graphics via HTml (RIGHT) is the first package that implements linked graphs using HTML canvas and JavaScript and supports efficient linked graphics that can show obvious relationship between multiple plots using same data. Also, HTML canvas and JavaScript make it possible to deliver the visualization to various platforms, including mobile devices since they are standard web technologies.

Motivation

Why linked graphs?
Link between two graphs can provide users with deep insight when analyzing the data large set. Clicking or dragging nodes from one graph just shows the related nodes of another graph.



How does work?

<R command>

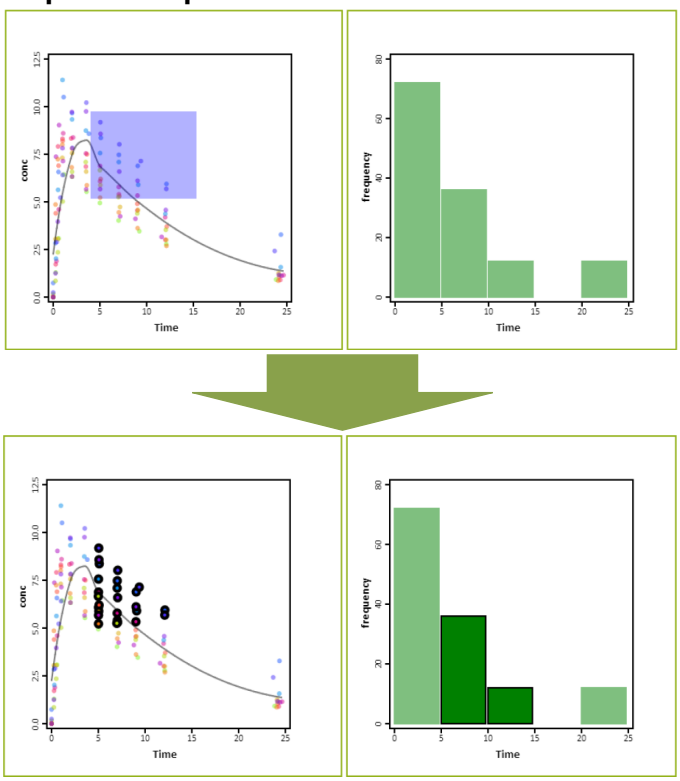
```
Obj <- RIGHT({
       plot(conc ~Time, Theoph, color = color)
       runServer.RIGHT(loessArray, {
            obj <- loess(conc ~ Time, data = Theoph)
            xRange <- range(Theoph$conc)
            simArray <- data.frame(conc = seq(xRange[1],
                                  xRange[2], length.out = 100))
            simArray$Time <- predict(obj, newdata = simArray)
            return(simArray)
       })
       lines(loessArray, Time, conc)
       hist(Time, Theoph)
}, Theoph)
```

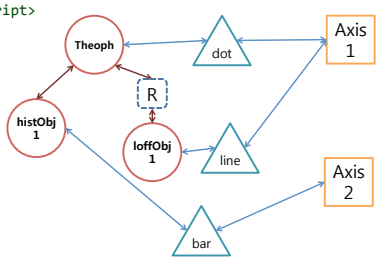
Code generation

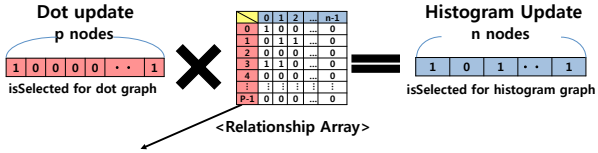
<JavaScript>

```
<script> var Theoph = createMainStructureE(rawArr1);</script>
<script>
var axis1 = new Axis(1, Theoph, 'Time', 'conc', {legend: 'Subject'});
var point1 = new Dot(axis1, Theoph, 'Time', 'conc', {});
var histObj1 = new ddply(Theoph, ['Time'], {});
var axis2 = new Axis(2, histObj1, 'Time', 'frequency', {});
var hist1 = new Bar(axis2, histObj1, 'Time', 'frequency', {});
Theoph.draw();
var AllAxisObjArr = [axis1, axis2];
eventTrigger(AllAxisObjArr);
</script>
<script>
// offload part.
var loessArray = createMainStructureE('Theoph');
var loffObj1 = new MakeLineObj(loessArray, 'Time', 'conc', {});
var loff1 = new Line(axis1, loffObj1, 'x1', 'x2', 'y1', 'y2', {});
$(function() {
          setTimeout(function() {
           window.Shiny.onInputChange('Theoph', Theoph.$isHidden);
          }, 1)
});
</script>
```

<Update Sequence>







	0	1	2	...	n-1
0	1	0	0	...	0
1	0	1	1	...	0
2	0	0	0	...	0
3	1	1	0	...	0
4	0	0	0	...	0
5	0	0	1	...	1
⋮	⋮	⋮	⋮		⋮
p-1	0	0	0	...	0

<Relationship Array>

<Event Processing Data Structures>
- isSelected (length of nodes): array for checking selected nodes of graph
- isHidden (length of nodes): array for checking hidden nodes of graph
- Relationship Array (p by n nodes)

Future Work

This project is still under active development:
- To provide server-offload using Shiny for users to help them draw graphs on low-computing devices, such as mobile.
- To develop an intuitive API to make it easy to draw graphs (e.g., ggplot2-based API)

This work was supported in part by the **Korean Ministry of Science, ICT & Future Planning** (KEIT-10047038) and by **Google Summer of Code** (GSoC) 2013 and 2014.