Create Client-side Interactive Plots in R

Bingbing Yuan

May 30, 2019

Bioinformatics and Research Computing
Whitehead Institute

WHITEHEAD INSTITUTE

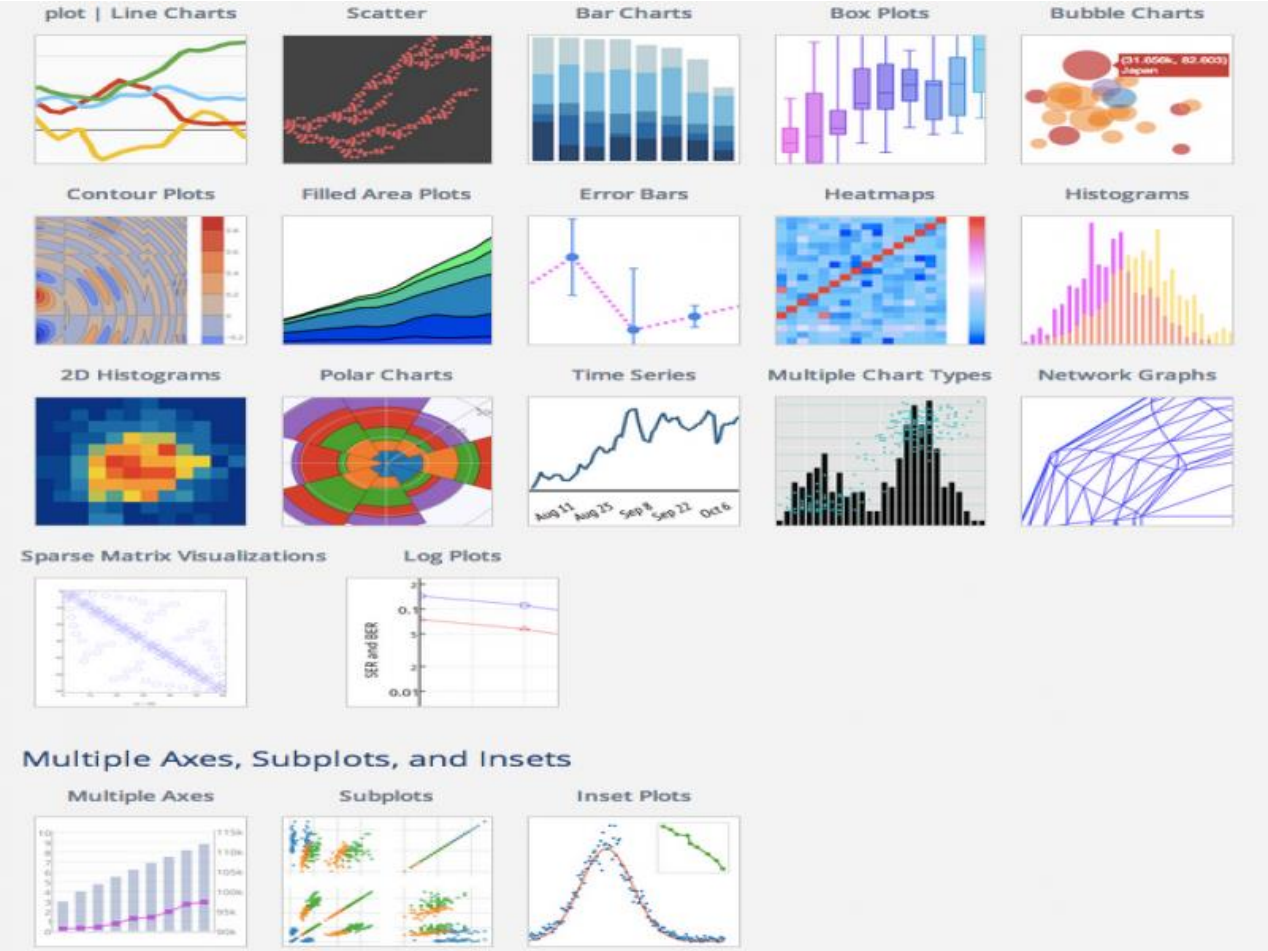
Why Client-side interactive plots?

- Easy for end users:
  - Just open a html file with browser
  - Don’t need to use R/RStudio
- Easy for program developers:
  - No need for creating multiple plots
- Example:
  - Mass spectrum data with three samples, no replicates:
    - 1 Control sample
    - 2 treated samples, under two different conditions

Behind demo:

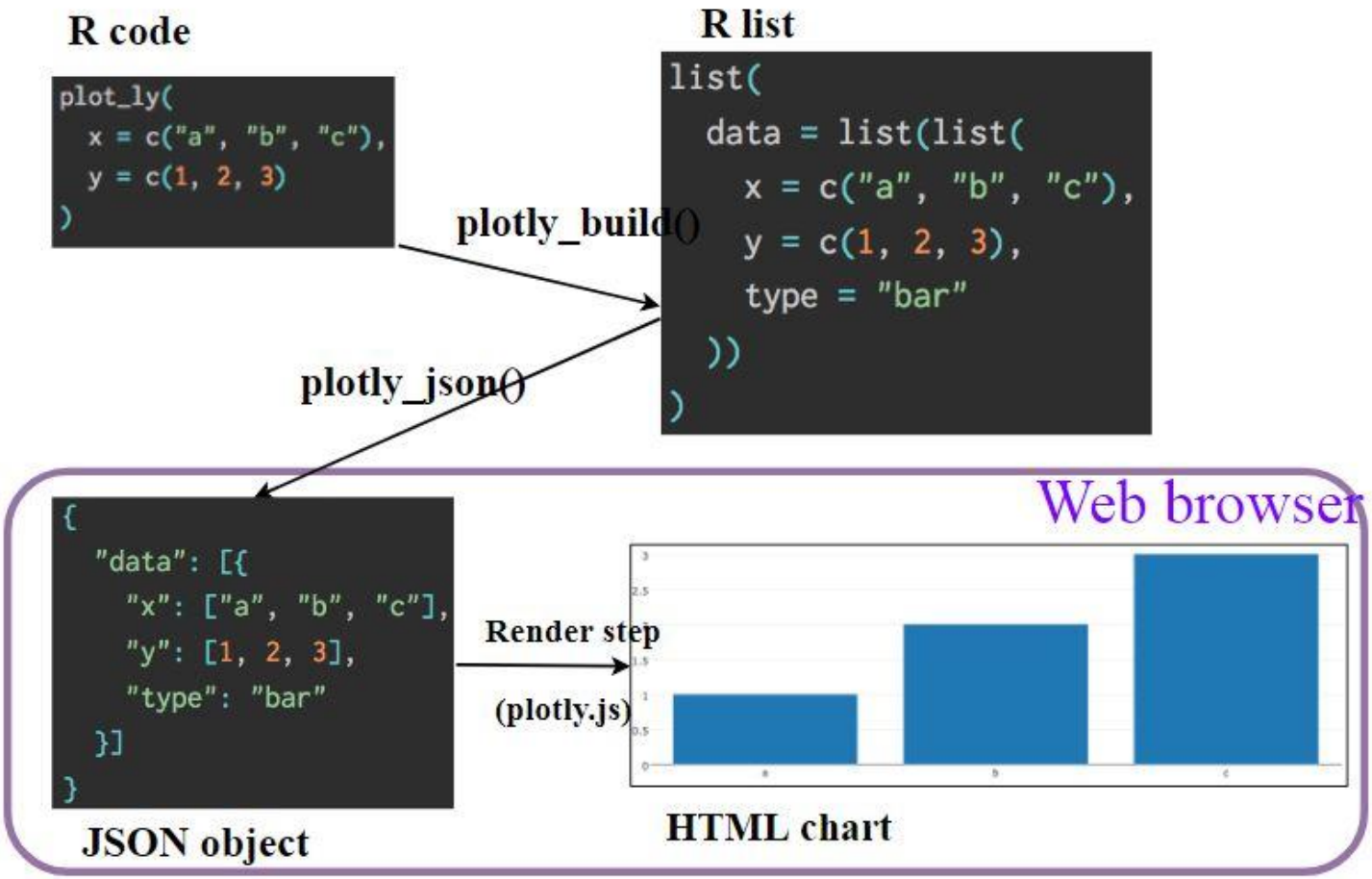
- Using html widgets to create figures
  - Plotly: interactive web graphics
  - DT: customizable data table library
- Linked figures/tables with brush
  - Crosstalk
- Layout
  - Flexdashboard
  - Subplot

Plotly



https://mandegar.info/?l=plotly+R+chart+attribute+reference

Behind Plotly



https://plotly-r.com/

plotly

- Thanks to plotly.js
- Ways to create plotly objects:
  - ggplotly()
    - help(ggplotly)
  - plot_ly()
    - help(plot_ly)

plotly

- Inspect JSON sent to plotly.js
  - plotly_json (p1)
- Inspect the data associated with a particular plot
  - plotly_data(p1)
- Get detail description with schema()

plot_ly

- plot_ly (data, color=I(“red”), stroke=I(“black”) type=“scatter”)
  - help(plot_ly)
  - add trace with add_*
    - add_histogram(), add_lines(), add_ribbons(),
    - help(add_trace)
- Add text(gene Symbol) to points
  - hoverinfo = "text“
- Scatter:
  - color, symbol could be variable/factor

ggplotly

- Control tooltip:
  - Add text:
    - ggplot(data, aes(x, y, text = row.names(data)))
  - Only text:
    - ggplotly(p, tooltip = "text")

Crosstalk

- Linked widgets:
  - Highlight/fading, selection/filtering
    - combine the filters by intersection
- Good widget candidate:
  - Data frame, data table: row-by-row bases
  - Not hierarchy/tree
- Compatible widgets:
  - Plotly: interactive graphing library
  - DT
    - DataTables provides filtering, pagination, sorting, etc.
  - SummaryWidget
    - display the count, sum or mean of one column of selected data

Crosstalk

- Create a SharedData from widgets:
  - sd <- SharedData$new(df)
  - plotly: sd <- highlight_key(df)
- Keys:
  - unique ID string for a row
  - Shared by Crosstalk widgets
  - Could be
    - default: row.names or ~ColumnName
  - filteredKeys returns the intersection of all lined widgets
- Multiple SharedData objects can form a group:
  - different SharedData instances must use identical keys (eg. data is subsetted)

widgets layout

- flexdashboard
  - Create within Rstudio
    - New R Markdown -> From Template -> Flex Dashboard
  - Easy to specify row and column-based layouts
  - Multiple pages
- subplot function:
  - Options:
    - nrows, width, heights,
    - Share axis: shareX, shareY

Layout with Flexdashboard

```
---
title: "Template with 3 Charts"
output:
  flexdashboard::flex_dashboard:
    orientation: columns
    vertical_layout: fill
---

```{r setup, include=FALSE}

library(flexdashboard)

```

Column {data-width=650}

Chart A

```{r}

```

Column {data-width=350}

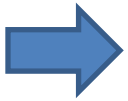
Chart B

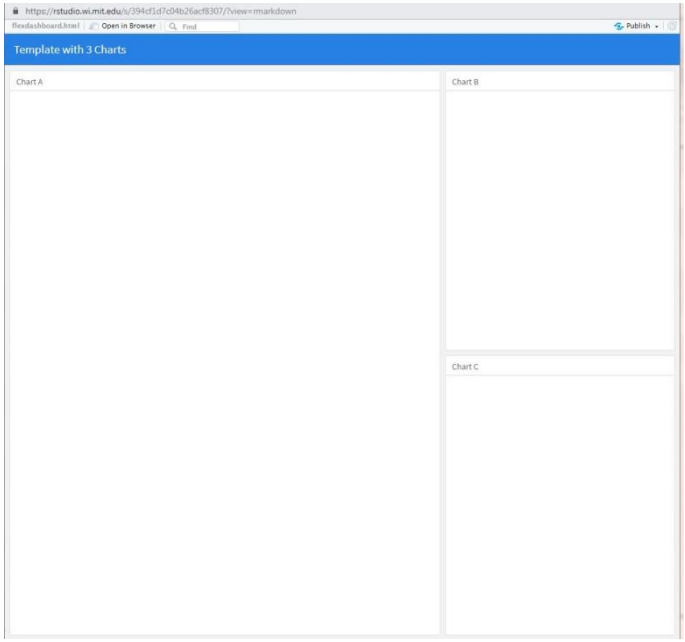
```{r}

```

Chart C

```{r}

```
```





limitation

- Data size
  - With toWebGL (Canvas instead of SVG)
- Complex interaction
  - Solution: shinny

summary

- Using html widgets to create figures
  - plotly: interactive web graphics
  - DT: customizable data table library
- Linked figures/tables with brush
  - Crosstalk
- Layout
  - Flexdashboard
  - Subplot

References:

- Plotly:
  - https://plotly-r.com
  - Book: https://plotly-r.com/plotly_book.pdf
  - Gallery of plotly examples: https://plot.ly/r/
  - Cheat sheet: https://images.plot.ly/plotly-documentation/images/r_cheat_sheet.pdf

- Crosstalk:
  - https://rstudio.github.io/crosstalk/
- Flexdashboard
  - https://rmarkdown.rstudio.com/flexdashboard/