PARIS
LODRON
UNIVERSITÄT
SALZBURG

plotly

Marvin Götze
David R. M. Graf

May 22nd, 2023

plotly

PARIS LODRON UNIVERSITÄT SALZBURG

https://clima.cbe.berkeley.edu/

plotly

- originally published as plotly.JS in JavaScript
- available for R, Julia, python, …
- 14M+ downloads of the R version from CRAN

plotly

Layered Grammar of Graphics

- theoretical foundation for many modern graphing tools, used in
  - `plotly`
  - `ggplot2`
- plot area is blank canvas, data and visualisations are added as
  - `traces` (plotly) or `geoms` (ggplot2)

Recommended further reading:

Hadley Wickham (2010) **A Layered Grammar of Graphics**, Journal of Computational and Graphical Statistics, 19:1, 3-28, DOI: 10.1198/jcgs.2009.07098

PARIS LODRON UNIVERSITÄT SALZBURG

Recap: `ggplot2`

`ggplot2` - basic commands

`ggplot()`	to create new plot
`aes()`	for aesthetic mapping
“+”	to concatenate commands
`<ggproto>`	objects (i.e. parts of plots) may be saved to variables

Basic plotting vocabulary

useful `geoms`:

- `geom_point`
- `geom_boxplot`
- `geom_histogram`
- `geom_abline/_hline/_vline`
- `geom_bar`

useful `geoms` for statistical visualization (`stats`):

- `stat_smooth`
- `stat_summary`

other useful functions:

- `facet_wrap` / `facet_grid`
- `theme` / `theme_bw` / `theme_[…]`

PARIS LODRON UNIVERSITÄT SALZBURG

Basics - `Plotly`

Basics

- plotly objects are created by:
  - direct initialization of a plotly object with `plot_ly()`
  - transforming a ggplot2 object into a plotly object with `ggplotly()`
- more options (not covered in this presentation):
  - plotly objects with `plot_geo()` or `plot_mapbox()`
  - plotly objects from list with `plotly_build()`
  - animations

Function: `plot_ly()`

- abstraction layer to plotly.js
- use the function depending on your programming style to:
  - directly add traces of a inferred or specified type or
  - create the blank plot area to subsequently add traces or
  - specify global arguments for any further functions on the plotly object
- attributes define how to

map variables (data) to visual properties (aesthetics) from a visual range

Function: `plot_ly()`

- most important attributes (some depend on trace type)
  - color → colors
  - symbol → symbols
  - linetype → linetypes
  - size → sizes
  - stroke, span should not be used to map variables
- in general:
  - numeric mapping generates one trace
  - discrete mapping generates multiple traces (one per factor/category)
  - mappings are avoided with function `I()`

Function: `plot_ly()`

- plotly objects and functions support:
  - the pipe operator `%>%`
  - data transformation with **dplyr**
- add new graphical layers with function `add_*()` like

`add_trace()`, `add_markers()`, `add_lines()`, `add_bars()`, `add_histogram()`

- change layout with function `layout()`

Scatter plots

Classical scatter:

- `add_markers()`

Line plots:

- `add_lines()` - connects according to x
- `add_paths()` - connects as rows in column

General:

- `add_trace(type = “scatter”, mode=”markers+lines”)`

Bar plots

Bar plots

- `add_bars()`
  - requires bar heights (like a y value)

Exercise 1

Statistical Plots in `plotly`

Boxplot-like plots:

- regular boxplot:
  - `add_boxplot/type = "box"`
- violin plot:
  - `type = "violin"`

histograms / heatmaps:

- `add_histogram()` - "normal" histogram, one value (`x/y`)
- `add_histogram2d()` - "2d+" histogram, automatic binning (`x, y`)
- `add_heatmap()` - needs z-value (`x, y, z`)

3D plots

3D scatter:

- same as “regular” scatter, just add z-dimension

surfaces:

- supply z-value, `add_surface()`

PARIS LODRON UNIVERSITÄT SALZBURG

Exercise 2

Function: `ggplotly()`

- convert ggplot2 objects to plotly objects
  - works also on ggplot2 extensions that return ggplot2 objects (no custom geoms) e.g. naniar, ggforce
- `ggplotly()` returns a plotly object
  - used in the same way as standard plotly objects
- most often you will modify:
  - layout: `hovertext`
  - add additional data/traces
  - style: modify/update data

Layout and Config - `Plotly`

Hover Text

default shows coordinate of trace under the cursor

may be customized using various methods:

- using the `text` argument, setting `hoverinfo = 'text'`
- using the `hovertemplate` argument, HTML tags and d3-syntax
  - supported tags: `<i>`, `<b>`, `<extra>`*
  - Github Link to d3 formatting syntax
- a combination of both `text` and `hovertemplate`

hovertemplate

may also be specified *for each trace* separately!

supported arguments for basic plots:

- `%{x}`
- `%{y}`
- `%{x/yother}` for multiple points
- `%{x/yaxis.title.text}` to access axis labels

you can also do formatting in the template:

- for numbers: `%{variable:d3-format}` options reference
- for dates: `%{variable|d3-time-format}` options reference

Function `layout()`

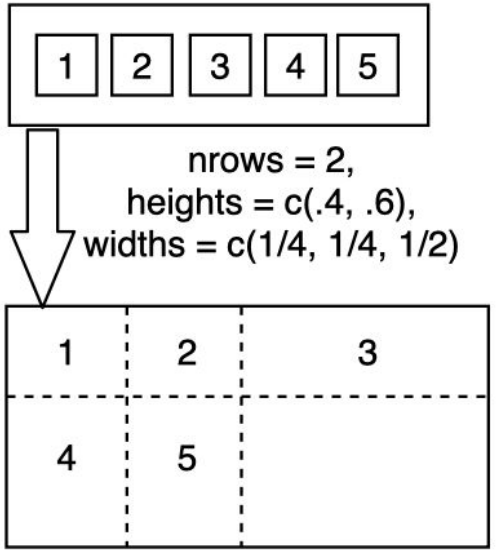
- function to customize the look of your plots
  - set titles for plots, legend and axes
    - `title = "title"`
    - `legend = list(title = list(text = "legend title"))`
    - `xaxis = list(title = "axis title")`
    - `yaxis = list(title = "axis title")`
  - define fonts or background colors
    - `plot_bgcolor = color`
    - `font = list(family = "Font Family", size = int, color = color)`
  - change size (must be passed to plotly from now on)
    - `autosize = T`
    - `width = 500`
    - `height = 500`
- careful with list() arguments or normal

Modebar configuration

- configure the modebar with `config()`

Function `subplot()`

- call on multiple plotly objects
  - `subplot(plotly1, plotly2, …)`
- or on a list of objects
  - `subplot(c[plotly1, plotly2,...])`
- merge multiple plotly objects into one interactive plot
- subplots either have their one independent axes or share the same
- subplots can be nested



https://plotly-r.com/images/proportions.svg

PARIS
LODRON
UNIVERSITÄT
SALZBURG

Exercise 3

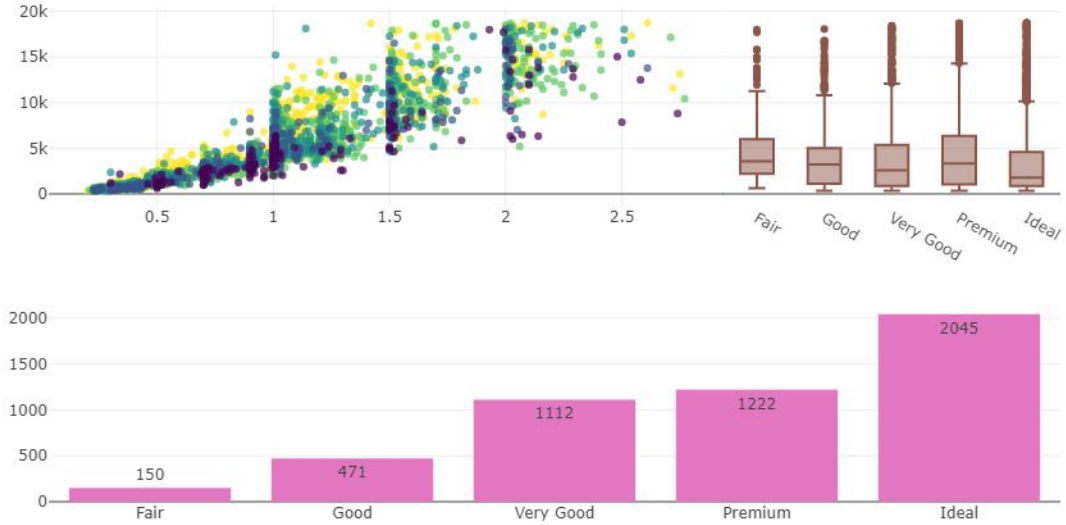
Exercise 3.2

recreate the plot you see on the right!

hint:

use plots plt1, bp1 and hp1!

subplots can be nested!



Advanced Interaction - `Plotly`

highlighting

- interactive highlighting of data based on graphical queries
- function `highlight_key()`
  - highlight data with the same property with respect to a selected data point ($selection_value)
  - imagine as a SQL query: SELECT * FROM data WHERE prop IN $selction_value
- function `highlight()`
  - define trigger events for turning graphical query on/off
  - possible triggers
    - `plotly_click, plotly_hover, plotly_doubleclick`

more highlighting

- further attributes of `highlight()` for defining more dynamic behaviour of highlighting
  - `selectize = TRUE` : dropdown widget
  - `persistent = TRUE` : retain previous selection
  - `dynamic = TRUE` : control highlight color

interactive elements

- simple slider for transforming the range of an axis
  - `rangeslider()`
- buttons and selectors can be used to customize layout even more (not covered in this presentation)

Graphical Queries - `crosstalk`

use `highlight_key()` to create a "`SharedData`" object

then proceed "as normal" to create plots, preferably in subplot - layout (or using "`DataTable`" for tabular display...)

by using `SharedData` objects, plotly can perform client-side linking and select from the graphic

`plotly` or `ggplot2` - when to use which?

PARIS LODRON UNIVERSITÄT SALZBURG

`plotly`

- dynamic
- interactive
- may be convoluted / worse help in R
- graphical queries with `crosstalk`

`ggplot2`

- great to produce static plots
- very consistent syntax
- great help and online resources
- many add-on packages (`ggridges`, `ggmosaic`, ...)

create plotly objects from ggplot objects using `ggplotly()` to get the best of both worlds!

Sources

- Plotly documentation. https://plotly.com/r/
- Plotly package - RDocumentation. https://www.rdocumentation.org/packages/plotly/versions/4.10.1
- Sievert, C. (2020). Interactive web-based data visualization with R, plotly and shiny. CRC Press, Taylor and Francis Group. https://plotly-r.com