

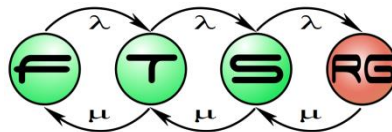
Visual analysis support in R

Visual Analysis of Measurement Data

László Gönczy

2019.10.24.

Budapest University of Technology and Economics
Fault Tolerant Systems Research Group



What is R?

- „Statistical programming language”
 - Supported by R Foundation
 - Extension of S
 - Statistics + OO principles
 - Datavis + advanced analytics (+...)
 - Interpreted language with bytecode compiler (!)

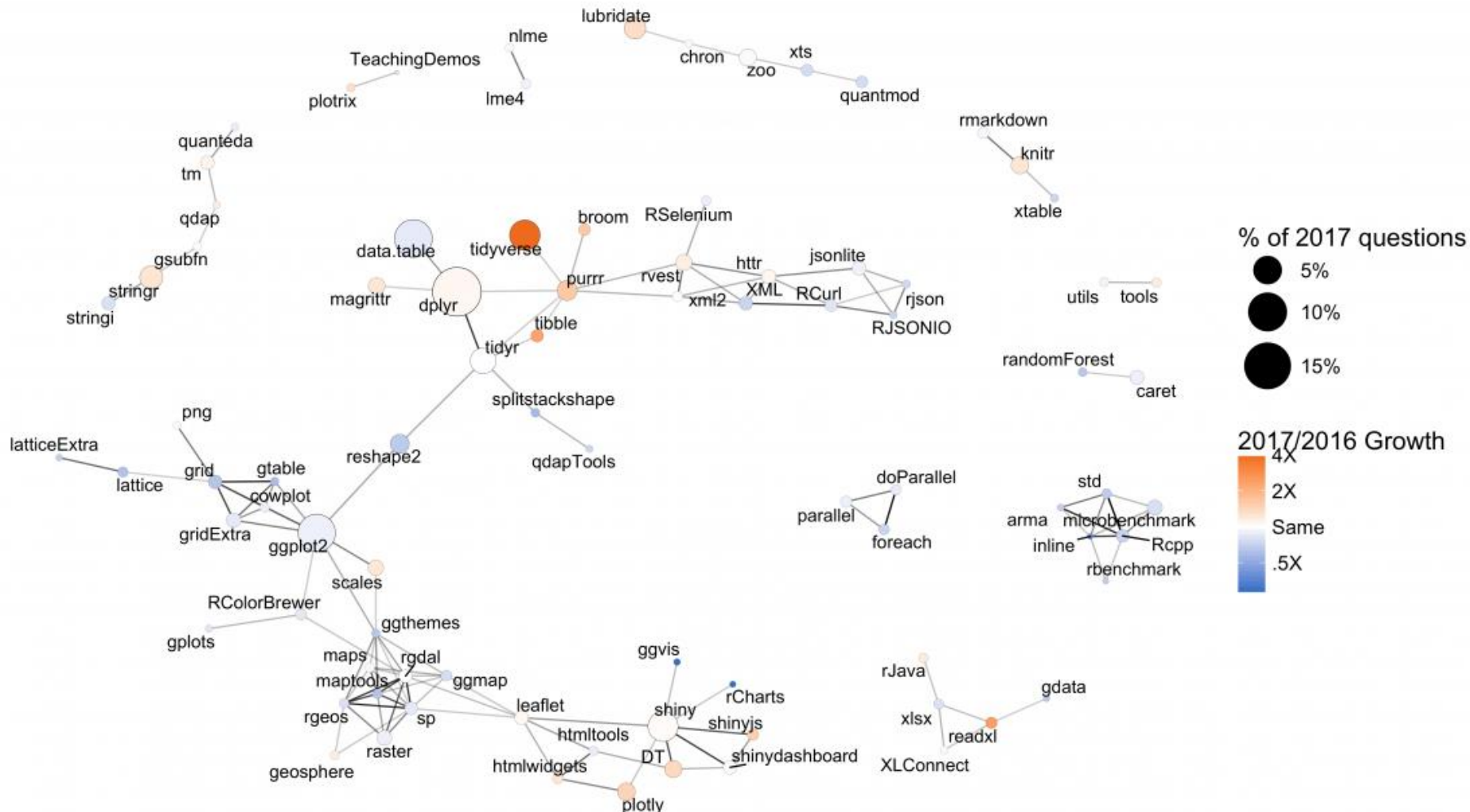
Why R?

- Large number of supporting packages
- Growing community
- CRAN: package repository
 - Maintenance + testing
- R support in many statistical environments
 - (... also R execution on top of SQL)
- Reference: e.g.
 - Graphical Data Analysis with R, Antony Unwin
<http://www.gradaanwr.net/>
 - <http://uc-r.github.io/gda>

R packages

Ecosystem of R packages

Correlations are based on packages often used in Stack Overflow answers on the same question.



<https://stackoverflow.blog/2017/10/10/impressive-growth-r/>

Basic plotting

- Basic plot types (histogram, boxplot, etc.) are supported in „plot”
 - Formatting can be difficult
 - Limited variety of plots
- E.g.
 - R Base Graphics: An Idiot's Guide
<http://rpubs.com/SusanEJohnston/7953>

Lattice

- Static graphs
 - `graph_type(formula, data=)`
 - Easy handling of numeric / factor variables
- References
 - <https://www.statmethods.net/advgraphs/trellis.html>
 - Deepanyan Sarkar: Lattice: Multivariate Data Visualization with R

GGPLOT2

- „Grammar of Graphics”
- References
 - Garrett Grolmund Hadley Wickham. R for Data Science <https://r4ds.had.co.nz/>
 - <https://ggplot2.tidyverse.org/reference/>
 - Winston Chang. R Graphics Cookbook: Practical Recipes for Visualizing Data (2nd edition in Nov 2018) <http://www.cookbook-r.com/Graphs/>

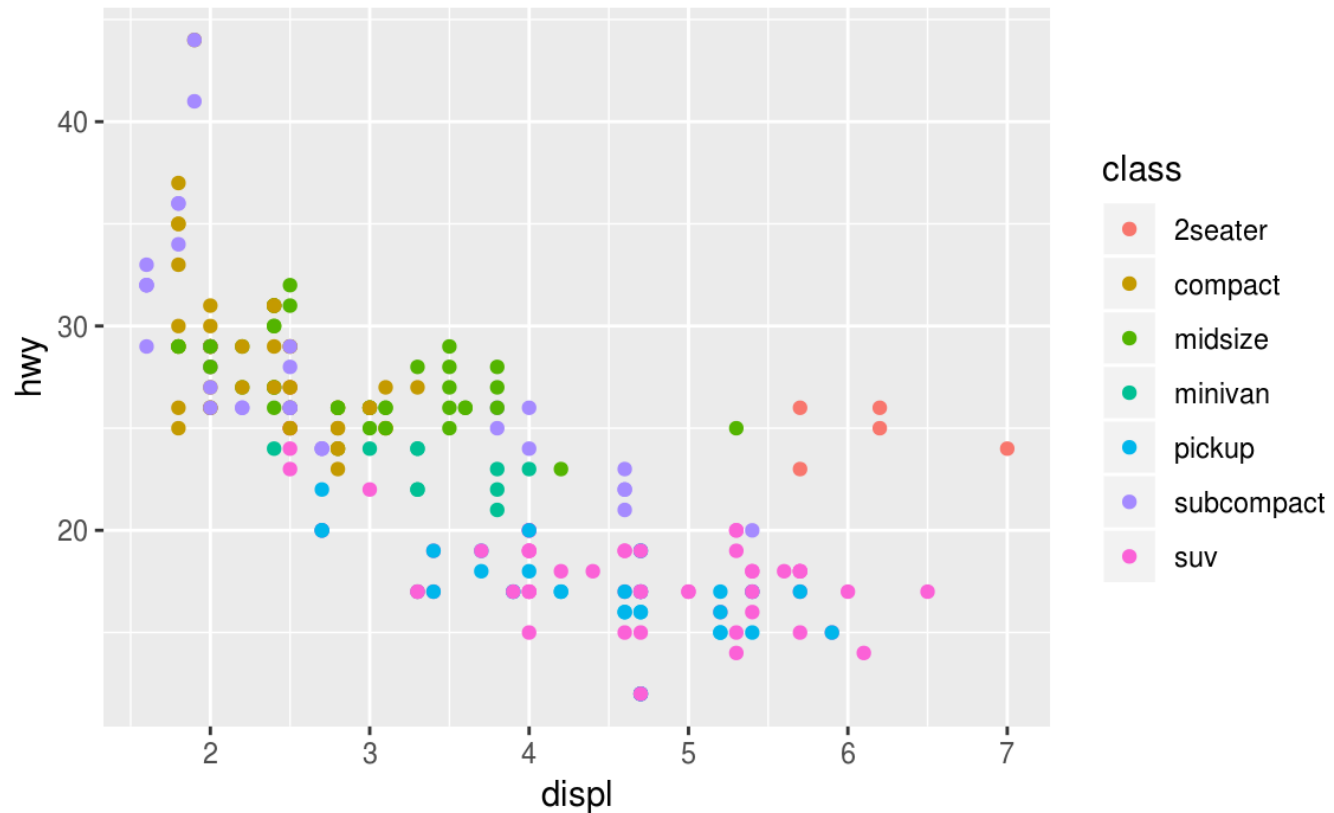
Ggplot2 synthax

■ General schema

```
ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(  
    mapping = aes(<MAPPINGS>),  
    stat = <STAT>,  
    position = <POSITION>  
  ) +  
  <COORDINATE_FUNCTION> +  
  <FACET_FUNCTION>
```


ggplot2 basic example

```
ggplot(data = mpg) +  
  geom_point(mapping = aes(x = displ, y = hwy, color = class))
```



<https://r4ds.had.co.nz/data-visualisation.html#aesthetic-mappings>

GGPLOT2 example

1. Begin with the **diamonds** data set

2. Compute counts for each cut value with **stat_count()**.

carat	cut	color	clarity	depth	table	price	x	y	z
0.23	Ideal	E	SI2	61.5	55	326	3.95	3.98	2.43
0.21	Premium	E	SI1	59.8	61	326	3.89	3.84	2.31
0.23	Good	E	VS1	56.9	65	327	4.05	4.07	2.31
0.29	Premium	I	VS2	62.4	58	334	4.20	4.23	2.63
0.31	Good	J	SI2	63.3	58	335	4.34	4.35	2.75
...

stat_count()

cut	count	prop
Fair	1610	1
Good	4906	1
Very Good	12082	1
Premium	13791	1
Ideal	21551	1

<https://r4ds.had.co.nz/data-visualisation.html#the-layered-grammar-of-graphics>

Ggplot2 example 2.

3. Represent each observation with a bar.
4. Map the **fill** of each bar to the **..count..** variable.

carat	cut	color	clarity	depth	table	price	x	y	z
0.23	Ideal	E	SI2	61.5	55	326	3.95	3.98	2.43
0.21	Premium	E	SI1	59.8	61	326	3.89	3.84	2.31
0.23	Good	E	VS1	56.9	65	327	4.05	4.07	2.31
0.29	Premium	I	VS2	62.4	58	334	4.20	4.23	2.63
0.31	Good	J	SI2	63.3	58	335	4.34	4.35	2.75
...

stat_count()

cut	count	prop
Fair	1610	1
Good	4906	1
Very Good	12082	1
Premium	13791	1
Ideal	21551	1

<https://r4ds.had.co.nz/data-visualisation.html#the-layered-grammar-of-graphics>

ggplot2 example

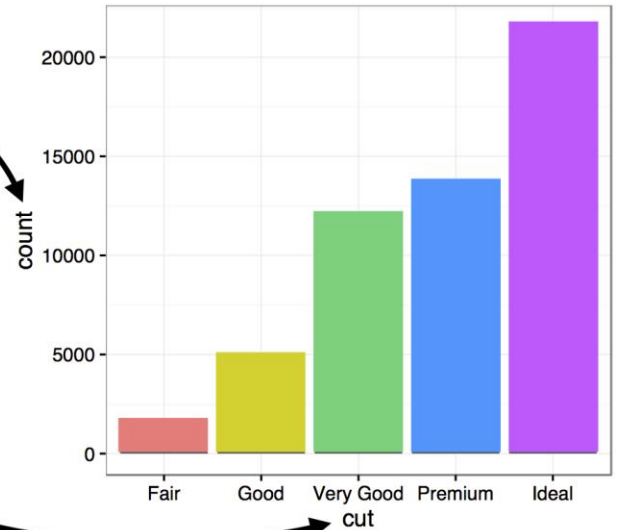
5. Place geoms in a cartesian coordinate system.

6. Map the y values to `..count..` and the x values to `cut`.

carat	cut	color	clarity	depth	table	price	x	y	z
0.23	Ideal	E	SI2	61.5	55	326	3.95	3.98	2.43
0.21	Premium	E	SI1	59.8	61	326	3.89	3.84	2.31
0.23	Good	E	VS1	56.9	65	327	4.05	4.07	2.31
0.29	Premium	I	VS2	62.4	58	334	4.20	4.23	2.63
0.31	Good	J	SI2	63.3	58	335	4.34	4.35	2.75
...

`stat_count()`

cut	count	prop
Fair	1610	1
Good	4906	1
Very Good	12082	1
Premium	13791	1
Ideal	21551	1



<https://r4ds.had.co.nz/data-visualisation.html#the-layered-grammar-of-graphics>

htmlwidgets

- JS → R
- Create widgets which can be embedded into Shiny/RMarkdown
- Linked brushing (color brushing) is supported
- <https://www.htmlwidgets.org/>

PLOT.LY

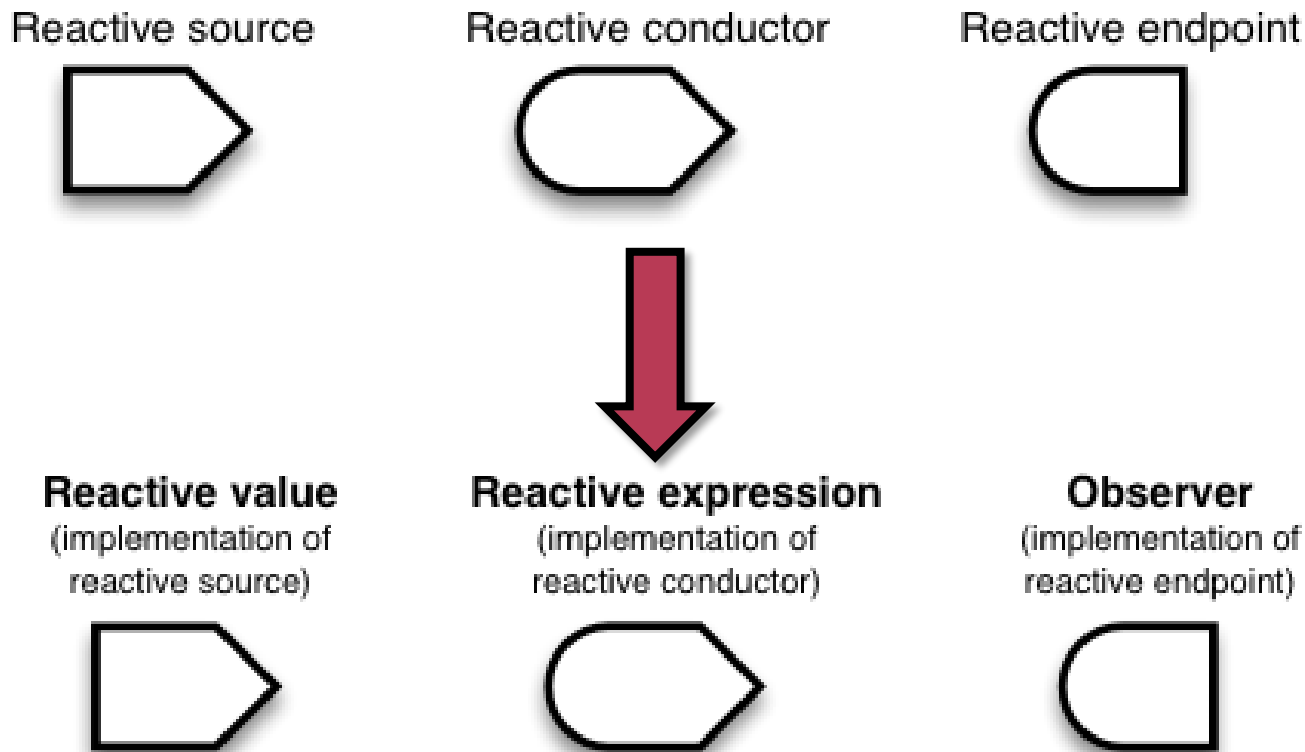
- JS frontend, includes a number of libraries (D3.js)
- Provides an API for statistical languages
 - E.g. Plotly for ggplot2
- See e.g.
 - Visual Data Analysis w/ R
 - https://rstudio-pubs-static.s3.amazonaws.com/187854_82b04e6b9c1a4d5f94424bfd56953870.html
- Dash: python dashboarding

EXAMPLE

Sample data in plotly

Shiny

- How to make R plot accessible on the web?
- „Reactive” programming paradigm



Basic elements

- ui.R: layout, control elements
- Server.R: „Reactive” logic
 - Function(input,output)

SHINY EXAMPLES

[https://shiny.rstudio.com/gallery/
shinyapps.io](https://shiny.rstudio.com/gallery/shinyapps.io)

Shiny vs JavaScript

- JavaScript can be embedded
 - Quick&dirty solution
 - Hard to debug, no IDE support

```
ui <- fluidPage(  
  tags$script(HTML(  
    "document.body.style.backgroundColor = 'skyblue';"  
  ))  
)
```

- Add JavaScript as a separate file (/www)
 - Application specific code

Shiny vs. JavaScript

- Use `htmlDependency`

- Like an R function

```
mypackageDependencies <- function() {  
  htmlDependency(name = "mypackage-assets", version = "0.1",  
    package = "mypackage",  
    src = "assets",  
    script = "js/myscript.js",  
    stylesheet = c("css/reset.css", "css/mystyles.css")  
  )  
}
```

- If an initialization code should be run only once on a single page
 - If JS code will be re-used in multiple packages

When (not) use Shiny

- Shiny is good for
 - Fast prototyping
 - Easy communication/sharing of data
 - Collaborative work
 - Dashboarding/Dynamic webpages
 - Incorporate advanced stats/ML
- Shiny is weak at
 - Creating standalone reports
 - Data upload/manipulation, ETL
 - General purpose EDA
 - Implementing complex control logic

Shiny support

- Shinyproxy
 - Deployment support
 - Integration of LDAP
 - Docker-based scalability
 - Configuration...
- Hosted: e.g. shinyapps (RStudio)

RMarkdown

- Create static/parametrized content
 - Notebooks
 - Websites
 - Reports
 - Embed shiny
- Reproducible research
- „Short” summary
 - <https://bookdown.org/yihui/rmarkdown/>

GGVIS

- Interactive ggplot-like plots
- Based on shiny infrastructure
- Performance? Support?
- <https://ggvis.rstudio.com/ggvis-basics.html>
- <https://ggvis.rstudio.com/cookbook.html>

Outlook: „drag n drop” ggplot2

- Esquisse
 - <https://github.com/dreamRs/esquisse>
- Easy to start
 - Simple, static plots
 - Aesthetics is easy to set up
 - Complex calculations/configurations not supported

iPlots

- Interactive visualization (based on Java)
- Most frequent plot types
- Linked selection, color brushing
- „iset”: view on dataframe
- <https://cran.r-project.org/web/packages/iplots/index.html>
- <http://rosuda.org/software/iPlots/>

IPLOTS DEMO

<https://www.r-exercises.com/2017/07/02/how-to-create-visualizations-with-iplots-package-in-r/>