2018

R and Data Visualization Workshop Using ggplot2

MARCH 15, 2018, MCGILL LIBRARY DATA LAB

BERENICA VEJVODA, NUMERIC DATA LIBRARIAN

Contents

What is ggplot2

Powerful graphics language for producing complex plots from data in a data frame (data frame=a two dimensional data structure in R for storing data tables)

Based on a Grammar of Graphics (as developed by Hadley Wickham) it allows one to concisely describe the components of a graphic.

Supports plotting both univariate and multivariate datasets (univariate omits y)

Currently cannot create 3D graphs

Learning Objectives

- To understand the logic of ggplot building blocks and layering and the basic structure of a plot
- Show how to produce a bar graph, a scatter plot and a line graph, including animated plotting using gganimate and plotly
- Cover some of the data cleaning functions necessary in order to get raw data into the right shape and format for the visualizations desired

Installing Packages

```
# install.packages("ggplot2", dependencies =TRUE)

# install.packages("devtools" ,dependencies = TRUE)

# install.packages("jtools", dependencies = TRUE)

# devtools::install_github("dgrtwo/gganimate") (pkg::name returns the value of the exported variable name in namespace pkg)

To find out what functions mean try using ?"::"

# install ImageMagick from http://www.imagemagick.org/script/download.php
# while installing change application path to C: (IMPORTANT)
# select - install legacy tools

# install.packages("plotly",dependencies = TRUE)
# Plotly for R allows for interactive plotting

# turn off the scientific notation

options(scipen = 999)
```

Importing Packages

library(ggplot2)

library(scales)

library(plotly)

library(gganimate)

Loading Data in Table Format

read in NHS dataset in table format and create a data frame from it

- df <- read.csv("C:\\RViz\\NHS.99M001X.E.2011.pumf.individuals_F1.csv", header = TRUE)

Creating a Subset

create a subset choosing the variables TOTINC, GROSRT and PR

- NHS_subset <- df[, c("PR","TOTINC","GROSRT")]

Removing Missing Values and Aggregating Data

remove missing values

aggregate TOTINC and GROSRT by PR

This code will show how to aggregate multiple variables simultaneously as well as remove the missing values for TOTINC and GROSRT and calculate the mean for TOTINC and GROSRT

- NHS_subset.agg <- aggregate(cbind(TOTINC,GROSRT)~PR,
  NHS_subset[!(NHS_subset$TOTINC==9999999 | NHS_subset$GROSRT==9999 |
  NHS_subset$GROSRT==8888),], mean)

create a new variable called ProvinceName with the names of the provinces (so they can be used as labels in our plot)

- NHS_subset.agg$ProvinceName <- c("Newfoundland & Labrador","Prince Edward Island","Nova Scotia","New Brunswick","Quebec","Ontario","Manitoba","Saskatchewn","Alberta","British Columbia","Northern Canada")

Creating a Bar Graph

plot a bar graph of PR versus TOTINC stacked with GROSRT

add **ggplot()** function to initialize a ggplot object
can invoke ggplot three ways: 1) ggplot(df, aes(x, y,)) 2) ggplot(df) 3) ggplot()
this example uses ggplot() which initializes a skeleton ggplot object which is fleshed out when layers are added. This third method is almost always followed by + to add components (layers) to a plot

add a **geom layer** (geoms are geometric objects which are the visual representation of observations. In other words they define the graph type.)

geom_bar is a bar graph representation. height of bars proportional to number of cases

add aesthetic mappings (**aes function**) which describe how variables in the data are mapped to visual properties of geoms

use **factor** when using discrete categories (vs continuous variables)

fill aesthetic = fill colour of object

data specifies a data frame

stat function is your statistical transformation

geom_bar uses stat_count by default so we will want to use **stat-="identity"** to leave data as is since we want to plot the calculated means and not have ggplot count the numbers of cases at each x position (i.e. add up the mean values)

geom_text is a label argument that draws text labels

label=paste0 – paste0 forces strong

size = how large objects appear (nsmall and size affect size of text

x axis is discrete

abbreviate built into label function (not necessarily ideal)

- ggplot() + geom_bar(aes(y = TOTINC, x = factor(ProvinceName), fill = GROSRT), data = NHS_subset.agg, stat = "identity") + geom_text(data = NHS_subset.agg, aes(x = factor(ProvinceName), y = TOTINC, label=paste0(format(round(GROSRT, 2), nsmall = 2))), size = 3.5) + scale_x_discrete(labels = abbreviate)

Flipping Coordinates

flipping coordinates (transpose x and y)

- ggplot() + geom_bar(aes(y = TOTINC, x = factor(ProvinceName), fill = GROSRT), data = NHS_subset.agg,stat = "identity", position = "stack") + geom_text(data = NHS_subset.agg,aes(x = factor(ProvinceName), y = TOTINC, label=paste0(format(round(GROSRT, 2), nsmall = 2))), size = 3.5) + scale_x_discrete(labels = abbreviate) + coord_flip()

Creating a Scatter Plot (Bubble Chart) and gganimate

select PR and HDGREE

- NHS_subset.degreePR <-df[!(df$HDGREE==88 | df$HDGREE==99), c("PR","HDGREE")]

aggregate HDGREE by PR

- NHS_subset.HdegreePRwise <- table(NHS_subset.degreePR)

select PR & HDGREE & TOTINC

- NHSsubset.PR_HDGREE_TOTINC <- (df[!(df$HDGREE==88 | df$HDGREE==99), c("PR","HDGREE","TOTINC")])

aggregate TOTINC on PR and HDGREE to find out average total income for each HDGREE in each province

- AGG.PR_HDGREE_TOTINC <- aggregate(TOTINC ~ PR + HDGREE, NHSsubset.PR_HDGREE_TOTINC, mean)

merge

scale_y_log10() – converts y axis to a logarithmic scale (power of 10) to make TOTINC values display better (condense the numbers)

- NHS_subset.HdegreePRTOTINC <- merge (NHS_subset.HdegreePRwise, AGG.PR_HDGREE_TOTINC, all.y=TRUE)

- g <-ggplot(NHS_subset.HdegreePRTOTINC, aes(x=HDGREE, y=TOTINC, size=Freq, frame=PR, color = HDGREE)) + geom_point() + scale_y_log10() + labs(x="Higher Education Degree", y="Total Income", title ="Total Income by Higher Education Degree by Province)")

interval = 1 shows frame for an interval of 1 second

- gganimate(g, interval = 1)

Creating a Scatter Plot with Jitter

scatter plot with jittering

not what happens when you consume too much coffee
jittering adds random noise to prevent over plotting

often an issue with large samples and conveniently rounded values for continuous variables (e.g. cases=88000, many people have same income range)

size of the circles is determined by the frequency count

- ggplot(NHS_subset.HdegreePRTOTINC, aes(x=HDGREE, y=TOTINC)) + geom_jitter(aes(col=PR, size=Freq)) + labs(x="Higher Education Degree", y="Total Income", title ="Province by Average Total Income by Higher Education Degree (Frequency wise)")

Adding a logarithmic scale to Y axis

log10() – converts y axis to logarithmic scale with base 10 (labels in powers of 10)

Further information on using logarithmic scales in charts and graphs

- ggplot(NHS_subset.HdegreePRTOTINC, aes(x=HDGREE, y=TOTINC)) + geom_jitter(aes(col=PR, size=Freq)) + labs(x="Higher Education Degree", y="Total Income", title ="Province Wise Total Income by Higher Education Degree (Frequency wise)") + scale_y_log10()

Creating Line Graphs

Lets use OECD open data and look at global GDP measures for Canada, the U.S. and Mexico

read the ForecastGDP dataset

- df_forecastgdp <- read.csv(file = "C:\\RViz\\RealGDP.csv", header = TRUE, strip.white = TRUE, stringsAsFactors = FALSE)

subset the annual data for Canada, Mexico & USA

- df_forecastgdp.subset <- subset(df_forecastgdp, subset = ((LOCATION == "MEX" & FREQUENCY =="A") | (LOCATION == "CAN" & FREQUENCY =="A") | (LOCATION == "USA" & FREQUENCY =="A")), select = c(LOCATION, TIME, Value))

replace the year with its abbvreviated form

gsuv = string replacement function

df_forecastgdp.subset$TIME <- gsub('^19',"'",df_forecastgdp.subset$TIME)
df_forecastgdp.subset$TIME <- gsub('^20',"'",df_forecastgdp.subset$TIME)

df_forecastgdp.subset$TIME <- factor(df_forecastgdp.subset$TIME, levels = df_forecastgdp.subset$TIME)

plot forecasted real GDP by country

- ggplot(data = df_forecastgdp.subset,aes(x = TIME, y=Value, group = LOCATION)) + geom_line(aes(color = LOCATION)) + geom_point(aes(color = LOCATION)) + theme(legend.position = "bottom")

read in GDP dataset for Millions U.S. Dollars

- df_gdp <- read.csv(file = "C:\\RViz\\GDP.csv",header = TRUE, strip.white = TRUE, stringsAsFactors = FALSE)

subset the data for Canada, Mexico & U.S. and select only “Million US Dollars”

- df_gdp.subset_MLNUSD <- subset(df_gdp, subset = ((ï..LOCATION == "MEX" & MEASURE =="MLN_USD") | (ï..LOCATION == "CAN" & MEASURE =="MLN_USD") | (ï..LOCATION == "USA" & MEASURE =="MLN_USD")), select = c(ï..LOCATION, TIME, Value))
- colnames(df_gdp.subset_MLNUSD) <- c("LOCATION","TIME","Value")

subset the data for Canada, Mexico & USA and select only the US Dollars/Capita

- df_gdp.subset_USDCAP <- subset(df_gdp, subset = ((ï..LOCATION == "MEX" & MEASURE =="USD_CAP") | (ï..LOCATION == "CAN" & MEASURE =="USD_CAP") | (ï..LOCATION == "USA" & MEASURE =="USD_CAP")), select = c(ï..LOCATION, TIME, Value))

- colnames(df_gdp.subset_USDCAP) <- c("LOCATION","TIME","Value")

plot GDP by country in MLNUSD

- ggplot(data = df_gdp.subset_MLNUSD, aes(x = TIME, y=Value, group = LOCATION)) + geom_line(aes(color = LOCATION)) + geom_point(aes(color = LOCATION)) + theme(legend.position = "bottom")

plot GDP by country in USDCAP

- ggplot(data = df_gdp.subset_USDCAP, aes(x = TIME, y=Value, group = LOCATION)) + geom_line(aes(color = LOCATION)) + geom_point(aes(color = LOCATION)) + theme(legend.position = "bottom")

use facets to show 3 different plots for Canada, Mexico & USA, respectively

transform the numbers in their abbreviated form (eg. 30000 as 30K)

- ggplot(data = df_gdp.subset_MLNUSD, aes(x = TIME, y=Value, group = LOCATION)) + geom_line(aes(color = LOCATION)) + geom_point(aes(color = LOCATION)) + theme(legend.position = "bottom") + facet_wrap(~ LOCATION, ncol =3) +

scale_y_continuous(name = "Million US Dollars", labels = unit_format(unit = 'M',scale = 1e-6)) + xlab("Years")

Adding Themes – There is One for APA

using APA theme to format the legend and labels to be consistent with the APA Style guide specifications for charts

- ggplot(data = df_gdp.subset_USDCAP,aes(x = TIME, y=Value, group = LOCATION)) + geom_line(aes(color = LOCATION)) + geom_point(aes(color = LOCATION)) + theme(legend.position = "bottom") + facet_wrap(~ LOCATION, ncol =3) + scale_y_continuous(name = "US Dollars/Capita", labels =unit_format(unit = 'K',scale = 1e-3)) + xlab("Years") +jtools::theme_apa(legend.pos = "bottom")

change the shape of the points to triangle & make the line graph thicker

- Newplot <- ggplot(data = df_gdp.subset_USDCAP, aes(x = TIME, y=Value, group = LOCATION)) + geom_line(aes(color = LOCATION),size =1.2) + geom_point(aes(color = LOCATION),shape =2) + theme(legend.position = "bottom") + facet_wrap(~ LOCATION, ncol =3) + scale_y_continuous(name = "US Dollars/Capita",labels =unit_format(unit = 'K',scale = 1e-3)) + xlab("Years") +jtools::theme_apa(legend.pos = "bottom")

Saving and Printing Plots

prints the plot in the plot window

- print(Newplot)

saves the plot in the desired format

- ggsave("C:\\myggplot.pdf", plot=Newplot)

for an interactive ggplot we can use the package plotly.

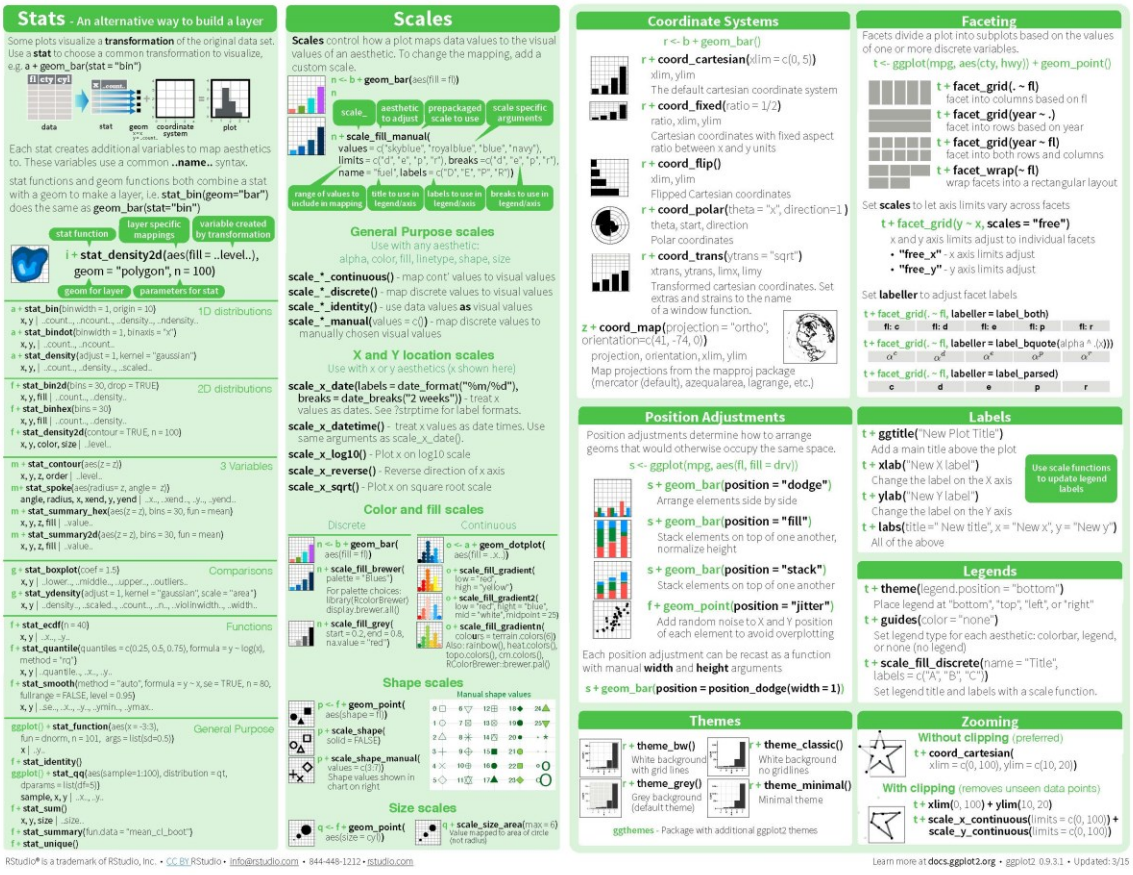
Plotly is Cool

we can zoom and see actual values on mouse-over and similar abilities using plotly

- ggplotly(newplot)

Further Resources

RStudio cheat sheet for ggplot2 https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf



[QuickR DataCamp]

[Data Visualization with ggplot2 (DataCamp)]

McGill Library eBooks by Hadley Wickham (ggplot2 creator)