post02-josia-yuan

Josia Yuan

November 26, 2017

More Interesting Packages for Graphic Display in R

Introduction

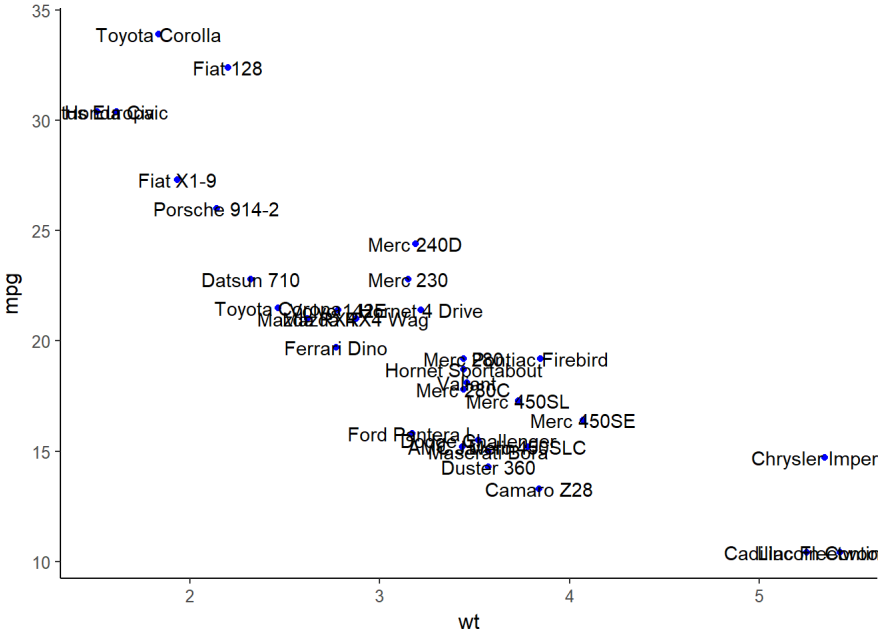
Throughout this semester, Stats133 has exposed students to some R pacages such as ggplot2, ggvis, and shinyapp for an introduction to data visualization, while each of the package has. As I practiced with those packages, not only did I begin to wonder if there were other arguments that could make a more customized visual outputs with those packages, but also whehter other packages could render even more interesting graphics. In a word, my motivation was to explore more R packages specific to data visualization and introduce some basics to my fellow classmates throught this post. This post will start off with brief description and demostration of some extention packages specific to `ggplot2`, followed by two other R packages that provide interesting graphics that have drwan my attention. The purpose of the post is to show how flexible and fun it could be to play around with various packages for data visualization, as well as introduce some of the basic element of specific packages.

Some Extentions Pacakges for ggplot2

ggrepel

No matter it was basic graphic output via `plot()`, or a prettier and more flexible version via `ggplot()`, one thing I always find frustrating is that many of the labels attached to the plotted points sometimes get overlapped with one another. Some introduce an abbreviations to those labels, which can either prove a simplicity or lack of sufficient information. A ggplot2 extention package `ggrepel` is used to literally repel the overlapping labels and provide a both infomative and clean graphic. The example below is a comparison between graphs with and without `ggrepel`.

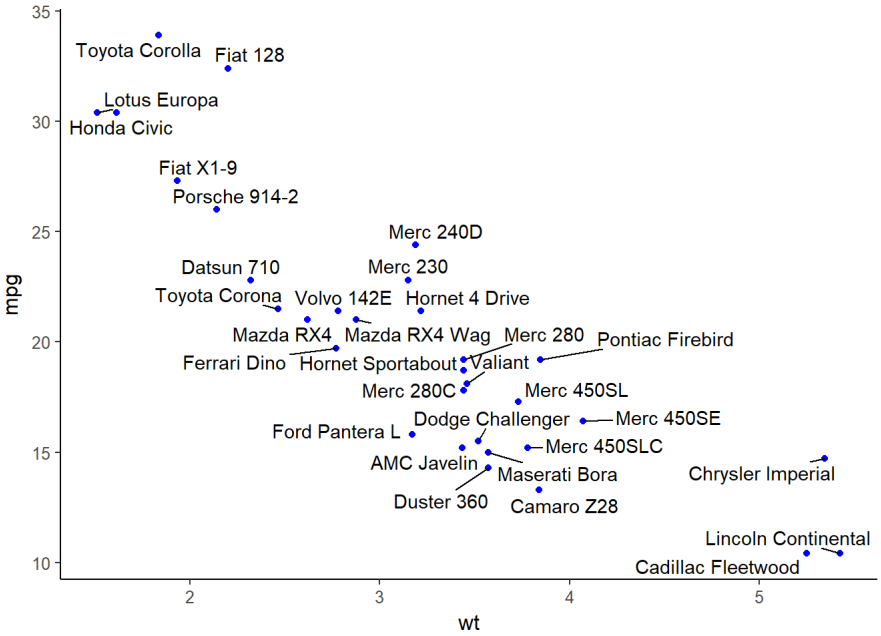
```
#Plotting with ggplot2
library(ggplot2)
ggplot(mtcars, aes(wt, mpg)) +
  geom_point(color = 'blue') +
  geom_text(aes(label = rownames(mtcars))) +
  theme_classic(base_size = 12)
```




```
#Plotting with ggrepel via ggplot2
library(ggrepel)
```

```
## Warning: package 'ggrepel' was built under R version 3.4.2
```

```
ggplot(mtcars, aes(wt, mpg)) +
  geom_point(color = 'blue') +
  geom_text_repel(aes(label = rownames(mtcars))) +
  theme_classic(base_size = 12)
```



ggalt

The documentation of this ggplot2 extention package states a concise description as the following:

> A compendium of 'geoms', 'coords', 'stats', scales and fonts for 'ggplot2', including splines, 1d and 2d densities, univariate average shifted histograms, a new map coordinate system based on the 'PROJ.4'-library and the 'StateFace' open source font 'ProPublica'.

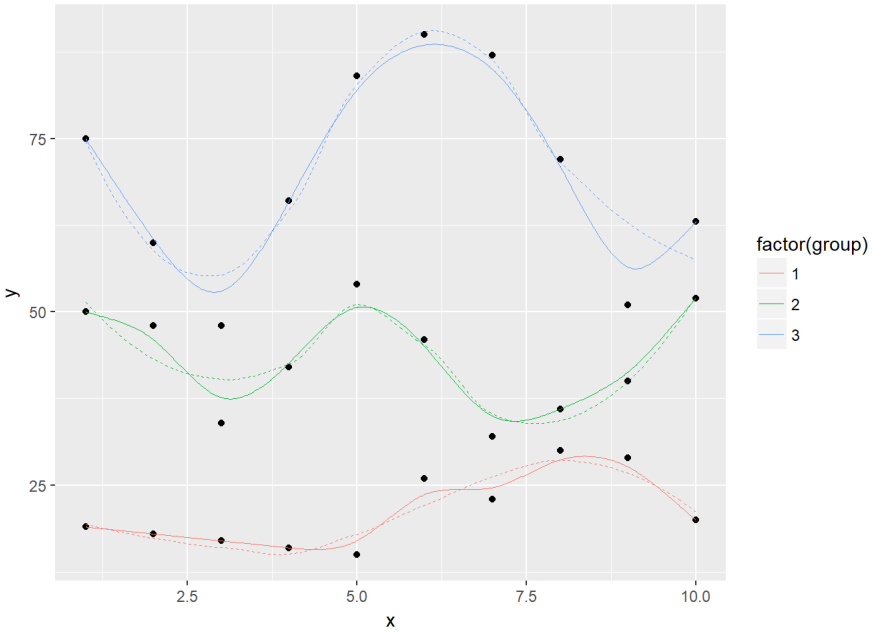
This was exactly what I was wondering about the notion of customization. Within the context of `ggplot2`, the package `ggalt` basically provides a lot more options and arguments for users to add more details to the rendered graphics. Among more than 20 more extra coordinate systems, geoms, statistical transformations, scales & fonts for `ggplot2`, I want to showcase in this post an argument for graphing the Smoothing spline. The three example graphs below demonstrate plots with `spline_shape` equal to 1, 0, and -1 respectively, rendering clear and pretty graphs that communicate a strong indication of statistical prediction.

```
#Creat a random set of dat
library(ggalt)
```

```
## Warning: package 'ggalt' was built under R version 3.4.2
```

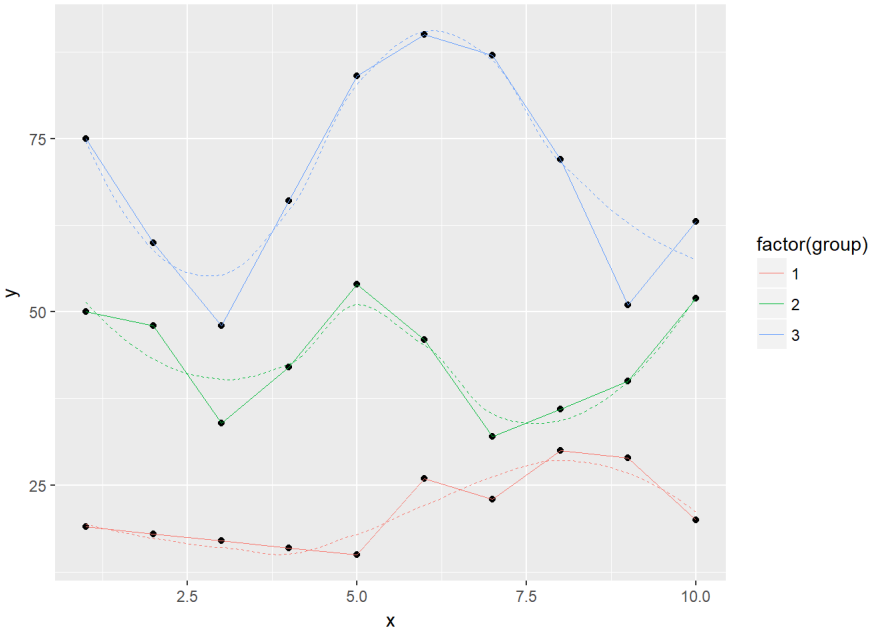
```
set.seed(1492)
dat <- data.frame(x=c(1:10, 1:10, 1:10),
                  y=c(sample(15:30, 10), 2*sample(15:30, 10), 3*sample(15:30, 10)),
                  group=factor(c(rep(1, 10), rep(2, 10), rep(3, 10)))
)

#Plot with xspline shape = 1
ggplot(dat, aes(x, y, group=group, color=factor(group))) +
  geom_point(color="black") +
  geom_smooth(se=FALSE, linetype="dashed", size=0.2) +
  geom_xspline(spline_shape=1, size=0.2)
```

```
## `geom_smooth()` using method = 'loess'
```



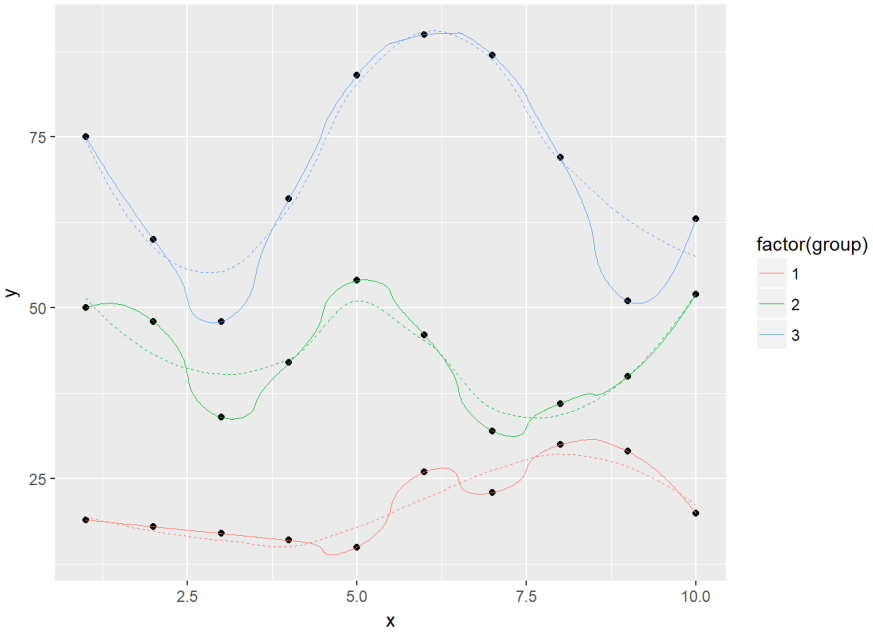

```
#Plot with xspline shape = 0
ggplot(dat, aes(x, y, group=group, color=factor(group))) +
  geom_point(color="black") +
  geom_smooth(se=FALSE, linetype="dashed", size=0.2) +
  geom_xspline(spline_shape=0, size=0.2)
```

```
## `geom_smooth()` using method = 'loess'
```




```
#Plot with xspline shape = -1
ggplot(dat, aes(x, y, group=group, color=factor(group))) +
  geom_point(color="black") +
  geom_smooth(se=FALSE, linetype="dashed", size=0.2) +
  geom_xspline(spline_shape=-1, size=0.2)
```

```
## `geom_smooth()` using method = 'loess'
```



ggthemes

I consider this package to be both fun and practical. When you are making graphs for any kind of report with R, it is not only necessary to take into account of the effectiveness of communiations, but also the fitness to the context and occasion. That is to say, when you are preparing a report to be included in The Economist Magazine, you don't want the style, or theme of the graph look like that from the fivethirtyeight.com. or The Wall Street Journal. The package `ggthemes` provdies additional geoms, scales and themes for ggplot from a large range of varieties. In the example below, I included a graph with different themes built in the packages for a quick showcase with reference to more examples here.

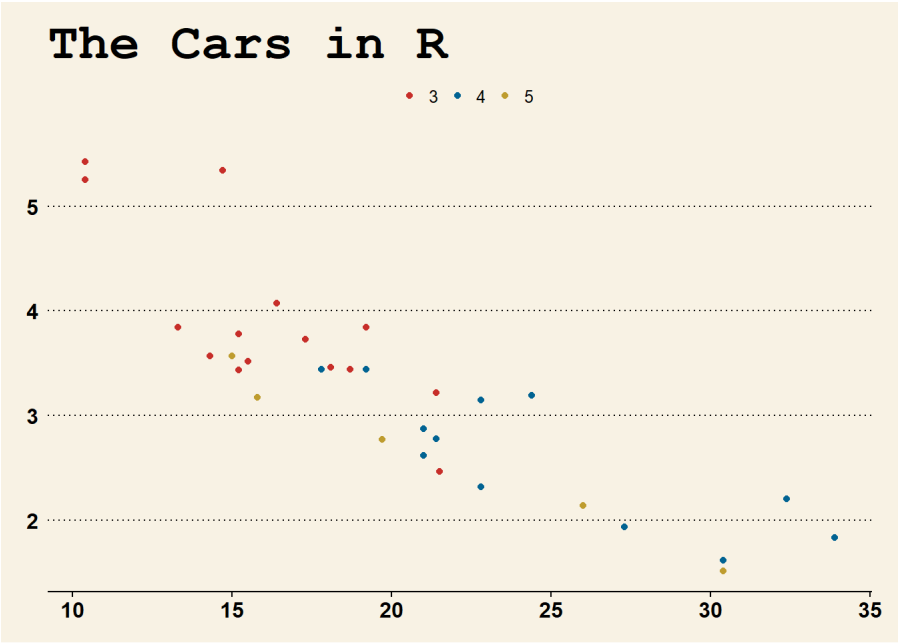
```
#Theme based on plots in The Wall Street Journal
library("ggthemes")
```

```
## Warning: package 'ggthemes' was built under R version 3.4.2
```

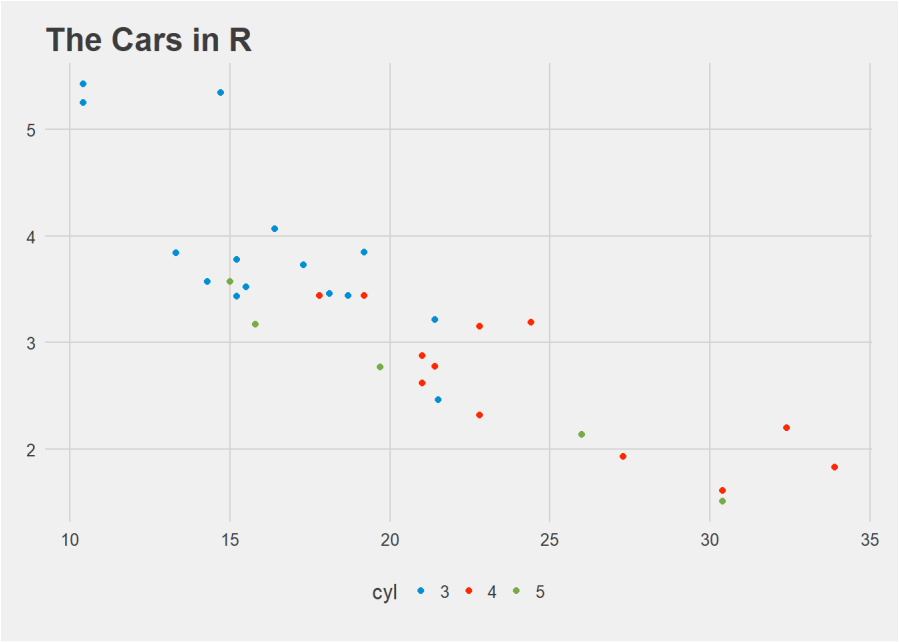
```
library("scales")
```

```
## Warning: package 'scales' was built under R version 3.4.2
```

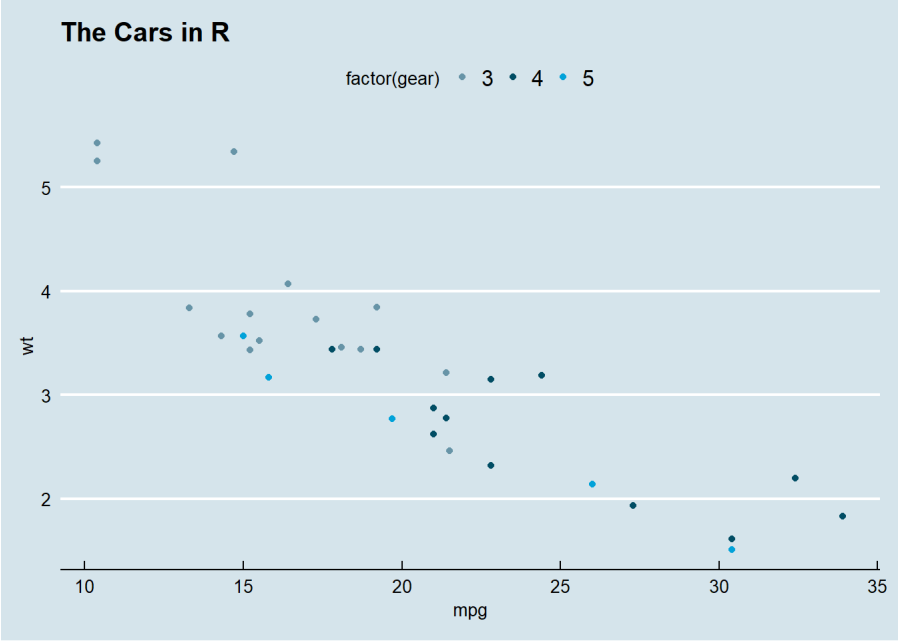
```
plot <- ggplot(mtcars, aes(x = mpg, y = wt, colour = factor(gear))) +
  geom_point() +
  ggtitle("The Cars in R")
plot + theme_wsj() + scale_colour_wsj("colors6", "")
```




```
#Theme based on the plots at fivethirtyeight.com.
plot +
  scale_color_fivethirtyeight("cyl") +
  theme_fivethirtyeight()
```




```
#Theme based on the style of plots in The Economist magazine.
plot + theme_economist() + scale_colour_economist()
```



And Some More!

There are many other extension packages such as `ggstance` for creating horizontal versions of ggplot2 geoms; `ggforce` to accelerate ggplot2; `ggomnet` that provides network visualizations in ggplot2; `gganimate` to create easy animations with ggplot2; `ggExtra` that gives marginal density plots or historgrames and many more!

Some Interesting Visualization Tools

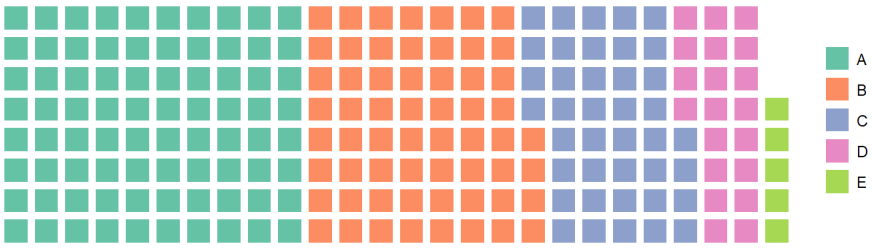
waffle

The package `waffle` is used to create waffle charts, that is, literally, square pie charts that look like breakfast waffles! The example below is just to show the most basic arguments to get a waffle chart going.

```
# A basic example of waffle
library(waffle)
```

```
## Warning: package 'waffle' was built under R version 3.4.2
```

```
example <- c(80, 60, 40, 20, 5)
waffle(example,rows = 8)
```



Something interesting associated with `waffle` is that it also takes argument `use_glyph=` that specifies a certain shape taken by those individual squares. To download the extrafont, you can go to the FontAwesome website to download a free packet of iconic font and CSS toolkit. A R-specific instruction can be found here Below is a picure of square changed to a person-like shape.

An example of waffle output using Extrafont

wordcloud

At the first glance of the output of this package, I personally just thought it would be something fun to play around with, because I think the cloud of words is an unconventional and eye-catching way of displaying data. When I do look into it, it was described as something as the following:

> A word cloud is a visual presentation or depiction of key words or frequent words from input textual data. Size of the word/text is linked to frequency of the word/text.

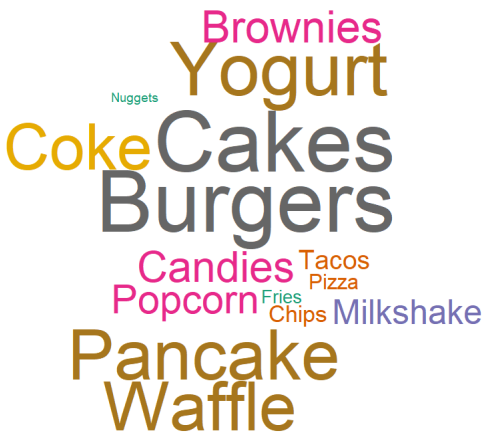
Therefore `wordcloud` does also have a strong weight on data analysis by visually displaying frequency of appearance of a certain word. The example below features some of my personal favorites of "not-so-healthy" food.

```
#Setting up for wordcloud
library(wordcloud)
```

```
## Warning: package 'wordcloud' was built under R version 3.4.2
```

```
## Loading required package: RColorBrewer
```

```
#Create a list of words (Random words concerning my work)
food <- c("Pizza","Burgers","Yogurt","Tacos","Cakes","Coke","Candies","Fries","Nuggets","Milkshake","Waffle","Pancake","Chips","Popcorn","Brownies")

#I give a frequency to each word of this list
fre <- sample(seq(0,1,0.005) , length(food) , replace=TRUE)

#The package will automatically make the wordcloud ! (I add a black background)
pal <-brewer.pal(8,"Dark2")
par(bg="white")
wordcloud(food , fre , col=pal)
```



Take-home Message

In a word, with extensive range of packages ready to be used and even created in R, there is really little limitations as to data manipulation and visualization. The several extension packages for `ggplot2` I have introduced in the post demonstrate the great flexibility of creating graphics via R, while the other two interesting packages I brought up demonstrate how users can really play around with the design and aesthetic aspects of data visualization. Hopefully after reading this post, the readers get to appreciate the potentials R has to offer in regard to data visualization, and even be motivated to have fun with some of their own experiments!

Reference

1. ggrepel
2. ggalt
3. Smoothing spline
4. ggthemes
5. waffle
6. Extra Fonts for waffle
7. wordcloud