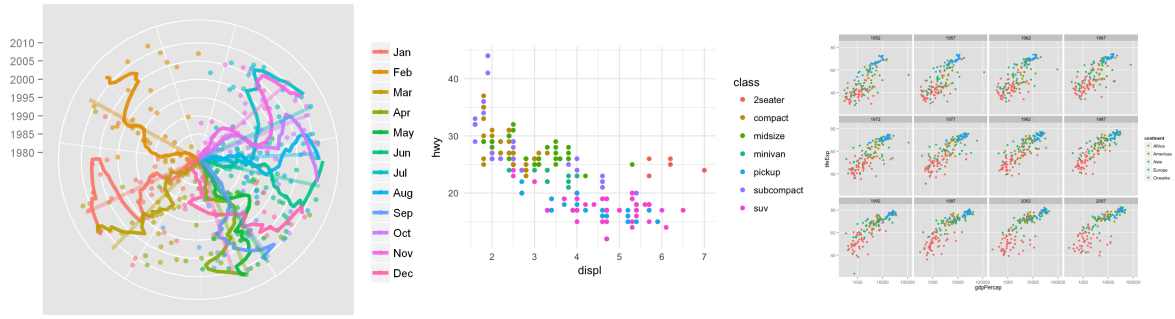
Titile: Internal ggplot2 logic introduction

Peijie Li
October 27, 2017

Introduction:

What do you think of ggplot? Do you think it is more organized and presented more elegant graphs compared to basic R 'plot' command? Do you think it is easy-use? The answers for me are:"Yes!" But why is like that? This post will introduce the internal logic in ggplot2 which will answer why the ggplot2 package can bring us those great traits. This first part of the post will compare ggplot and basic R 'plot' command line while the second part will focus on what is really going on when we use ggplot to make a graph. Showing the internal logic in ggplot2 in this order with some examples, I believe you can understand more about ggplot!



Motivation:

Many of the times, visualize data can give us more sense of what the data is trying to show us. Therefore, graphing command lines and the graphs that showing to us becomes really important. As soon as I learned ggplot2, I don't use 'plot' function in basic R anymore since ggplot can give us smoother and better visualized graphs. As a new beginner learning programming language, I was wondering what will cause the difference. Then after some research and interview, I realize that this is due to the internal logic difference. So I would like to try a bit to present my approach and conclusion of this question.

P.S. This topic is a very broad theory thing, I maynot be able to present them in a very professional way or in 100 percent accurate. So if there's anything I was wrong or you think there's anything I should add on it, please feel free to tell me and I will be very appreciate!

Background:

ggplot2 is a data visualization package for the statistical programming language R. Created by Hadley Wickham in 2005, ggplot2 is an implementation of Leland Wilkinson's Grammar of Graphics—a general scheme for data visualization which breaks up graphs into semantic components such as scales and layers. ggplot2 can serve as a replacement for the base graphics in R and contains a number of defaults for web and print display of common scales. Since 2005, ggplot2 has grown in use to become one of the most popular R packages. It is licensed under GNU GPL v2. Clik here to see the sourse

Content:

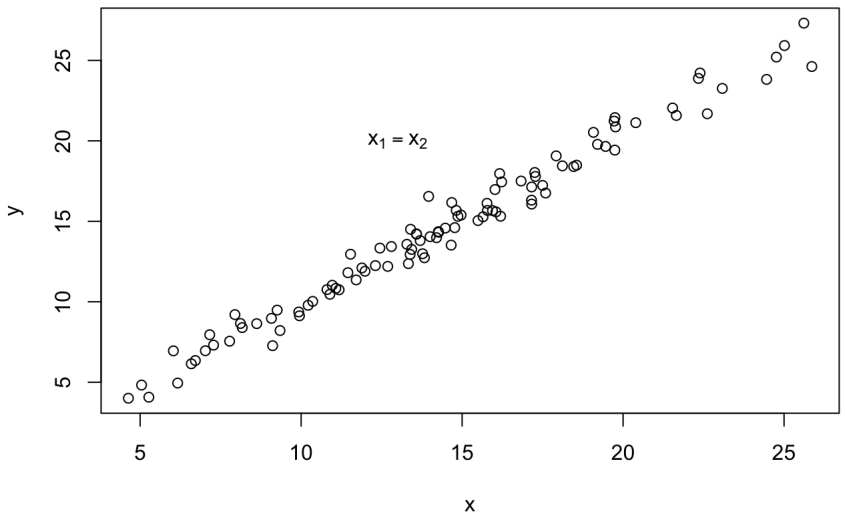
- Comparison between ggplot2 and basic R `plot` command line
- How does the ggplot2 make graphs

Part one-Comparison

> Whenever I used ggplot, I can always use `+` to add layout of the graph which can allow me make the graph step by step and decrese the risk of making mistakes. This particular **layout** graphing method is different from other plotting method. Each layout is a unique graph and ggplot just 'add' them all together. It allows the ggplot making graphs with combination of multiples layout pictures.
>
> So let's see some examples between basic R `plot` function and ggplot

Examples:

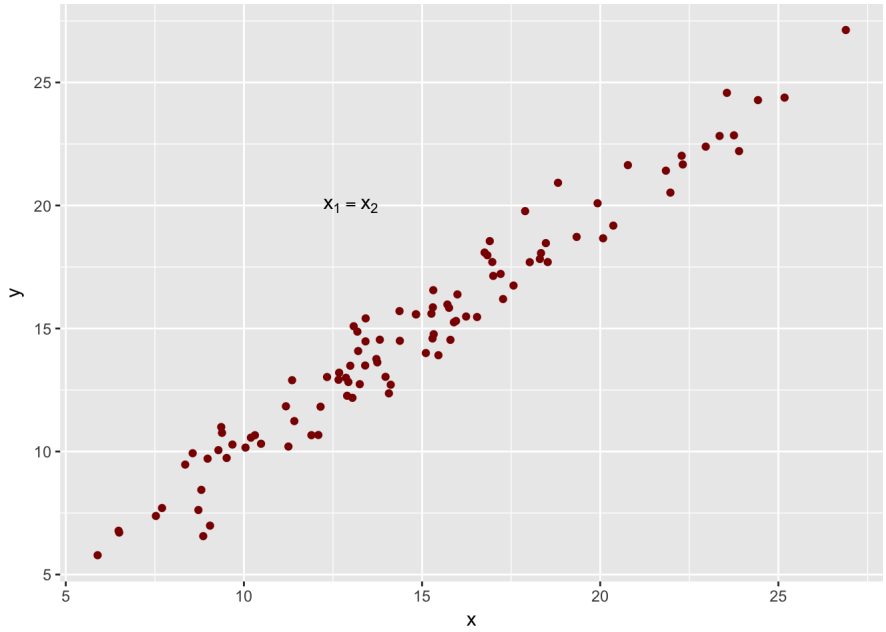
```
x <- rnorm(100,14,5)
y <- x + rnorm(100,0,1)
plot(x,y)
text(13,20, expression(x[1] == x[2]))
```



> There we can see the plotting started by `plot` command line and does not really have a particular stop sign. Moreover, the graph is changed directly with multiple command lines. Which also means, the plotting is "modified" by each command line after the `plot` function.
> However, ggplot "added" the layout command all together at first and then made the graphs. Which also means, every change you made at ggplot will cause re-graphing every time.

> Let's see why is that true.

```
library(ggplot2)
x <- rnorm(100,14,5)
y <- x + rnorm(100,0,1)
ggplot(data= NULL, aes(x = x, y = y)) +  #selecting data set and begin to make graphs
  geom_point(color = "darkred") +  #adding the
  annotate("text",x =13 , y = 20,parse = T,
           label = "x[1] == x[2]") #adding lables
```



> There we can see ggplot started graphing from `ggplot( )` function and stop at any command line that does not have "+" layers. So how do we know ggplot is combining all the "+" layers and making graph at last? Let's move to next part.

Part two-How does the ggplot2 make graphs

There are many things can be introduced in this topic. But for here I will only use 'aes()' to prove that ggplot satisfied the graphing method introduced above

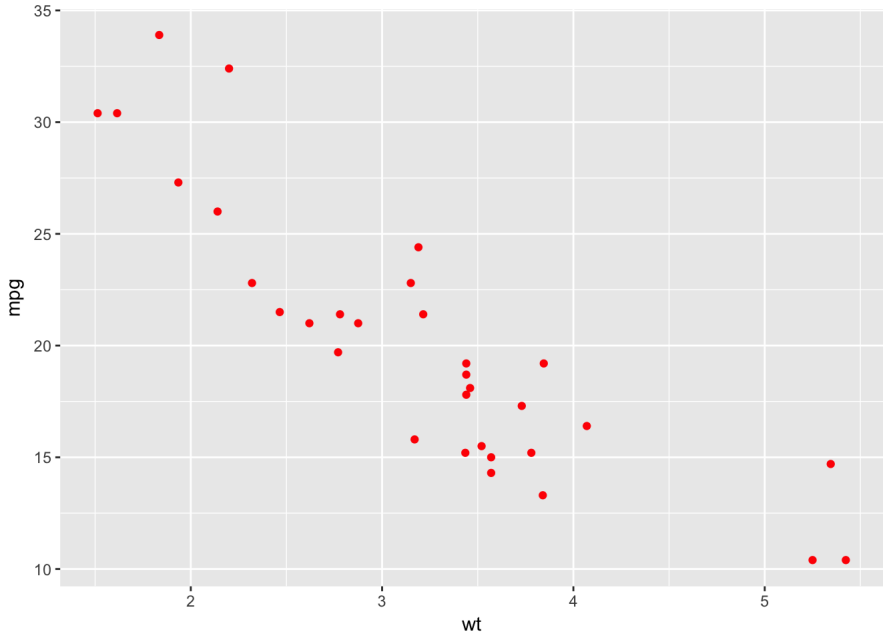
> aes is graphing the vecters in `aes( )` one by one. We will use the 'mtcars' data set in R stutio to approach the proof. Here is the 'mtcars' data set:

```
head(mtcars)
```

```
##                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

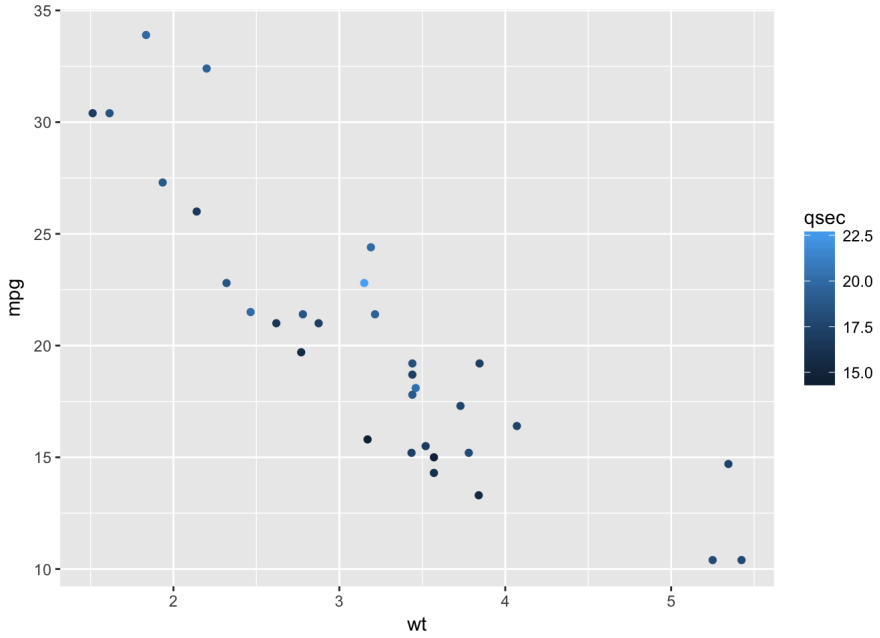
> The following method is what we learn from the lecture and lab. We just need to load the data and assign the x-axis and y-axis at ggplot (*code 1*), then plot the point and assign the color to the graphs(*code 2*) and there we go!

```
p <- ggplot(mtcars, aes(wt, mpg)) #code 1
p + geom_point(color = 'red') #code 2
```



> There is another way to color the graph with changing colors (source)

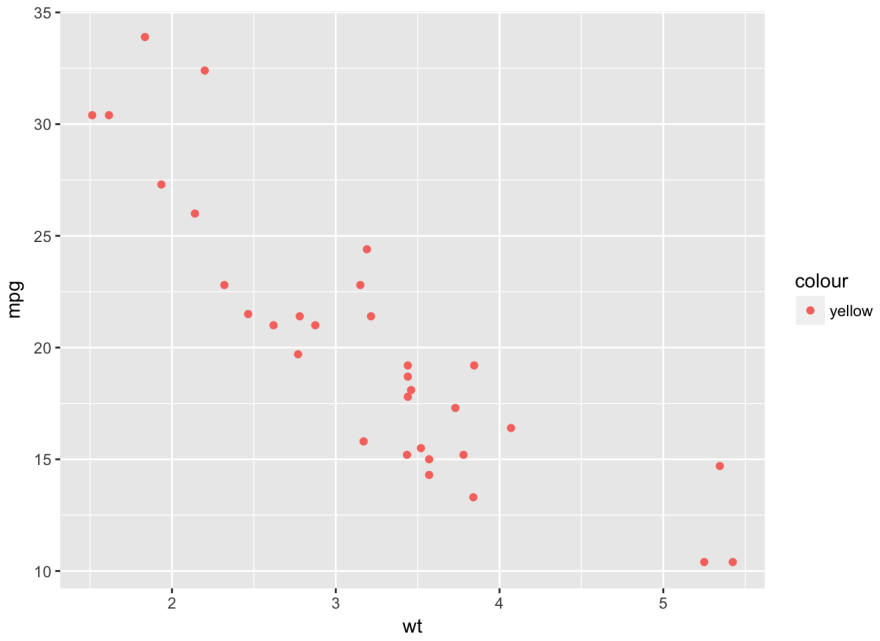
```
p <- ggplot(mtcars, aes(wt, mpg)) #code 1
p + geom_point(aes(color = qsec)) #code 2
```



> The second layout `geom_point` mapped the point and assigned color to the point according to vecter 'qsec'. The process behind this code is ggplot find point (wt, mpg) first and then using the 'qset' vector to color the point. This process will countinue untill the last plot has been plotted. That is why the color is different from the first graph.

Let's see an interesting example. What will happen if we set the color within the 'aes( )' function:

```
p <- ggplot(mtcars, aes(wt, mpg)) #code 1
p + geom_point(aes(color = 'yellow')) #code 2
```



It is clear that we set the color to yellow but the plot was in pink. Why is that? Here is the process ggplot proceed:

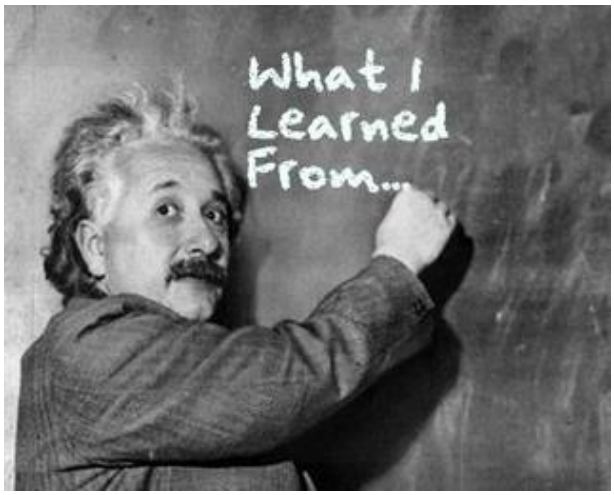
1. Loading the data set and setting the x-axis and y-axis(*code1*)
2. Ploting the plot(wt,mpg) with color changes according to vector "yellow". **WAIT!** There's no "yellow" vecter in data set! What should I do? (*code2*)
3. ggplot will then automatically regard "yellow" as a new vecter and change the color according to this new vector. In fact, this new vecter length is 1, so ggplot expanded the function as `factor(rep("green",nrow(mtcars)),levels = unique("green"))`
4. Now we can give some color to those points finally! color vecter is "yellow" with length 1 so the color should be the defult No.1 which is this pink. Second point, same with defult color pink. Third one……and we finally got that pink graph above.

Conclusion

I started at comparing the `plot` and ggplot graph. The point is to show ggplot manipulates all layout first then graphing the last, and the subpoint is ggplot has an advantage of having a clear starting point and stop command sign. Then in order to prove ggplot combined the layout command first and then graph the plot, I used `aes()` "color" part to present the approach. The first example is what we learn in class and the second is what I found at ggplot website, which changes the plot color according to the vector `qsec`. The reason ggplot can map out the color

in vector value order is because ggplot combined all layout first and map the point next, then repeat the process untill all the point has been mapped on the graph. The third example shows how does the `aes()` function work and can bring us some information which is “Take home message” No.3 in next part.

Take home message



1. ggplot manipulates each layout together first and then making the graphs. So for each modification, ggplot will do re-graphing again.
2. `aes()` can prove the graphing process of ggplot above
3. If you want the whole plot in only one color, use the `color = "color you want"` within the `geom_point`; If you want the plot has changing color according to some vector within data set, use `aes(color = "vecter in data set")` within `geom_point`

References:

- Wickham, Hadley (July 2010). “ggplot2: Elegant Graphics for Data Analysis”. Journal of Statistical Software. 35 (1)
- Wilkinson, Leland (June 2011). “ggplot2: Elegant Graphics for Data Analysis by WICKHAM, H”. Biometrics. 67 (2): 678–679.
- Wikipedia, ggplot2
- https://www.zhihu.com/question/24779017/answer/38750383
- http://ggplot2.tidyverse.org/reference/geom_point.html
- http://rmarkdown.rstudio.com/authoring_basics.html
- http://www.cookbook-r.com/Graphs/Colors_(ggplot2)/
- https://www.scribd.com/read/272069868/ggplot2-Essentials
- http://zevross.com/blog/2014/08/04/beautiful-plotting-in-r-a-ggplot2-cheatsheet-3/
- https://www.youtube.com/results?search_query=ggplot2