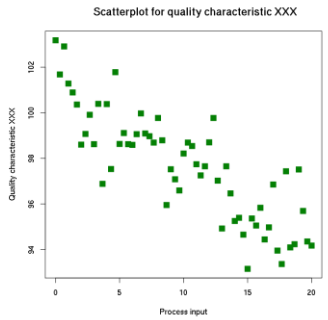
Dynamic Plot

Explorative Data Analysis

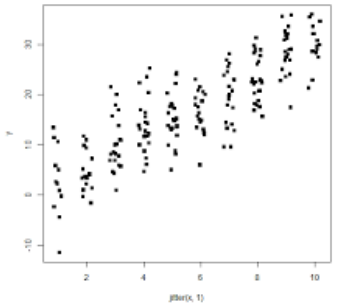
Neni Alya Firdausanti, M.Si.

Outline

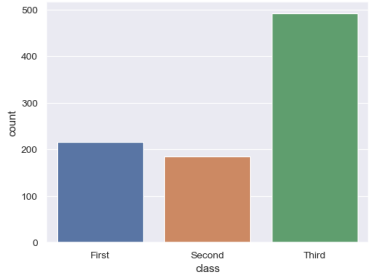
01. Scatter Plot



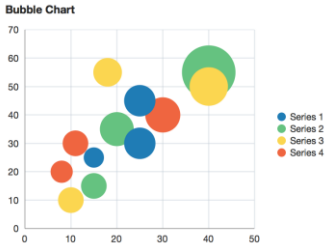
02. Jitter Plot



03. Count Plot



04. Bubble Plot

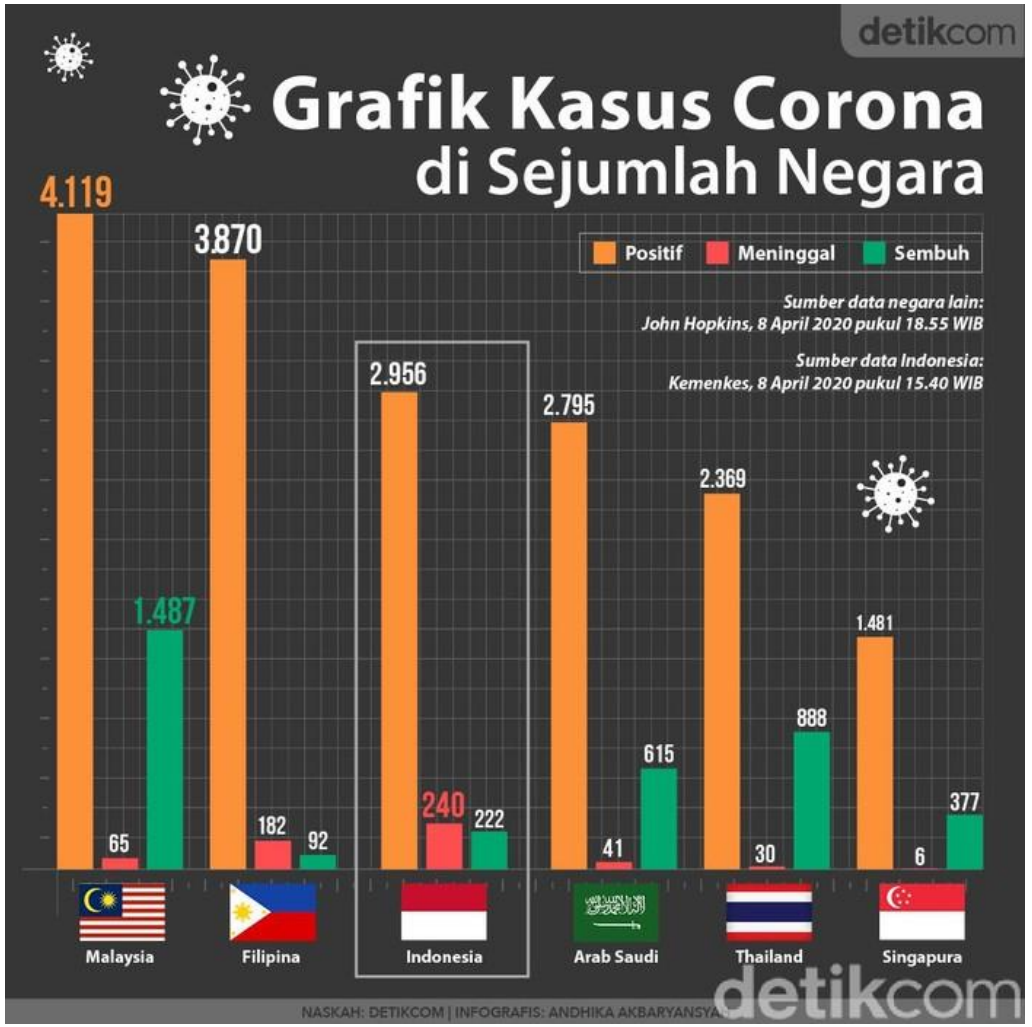


Packages:

- ❑ ggplot2
- ❑ RColorBrewer

Count Plot

A count plot can be thought of as a histogram across a categorical, instead of quantitative, variable. The basic Application Programming Interface (API) and options are identical to those for barplot, so you can compare counts across nested variables.

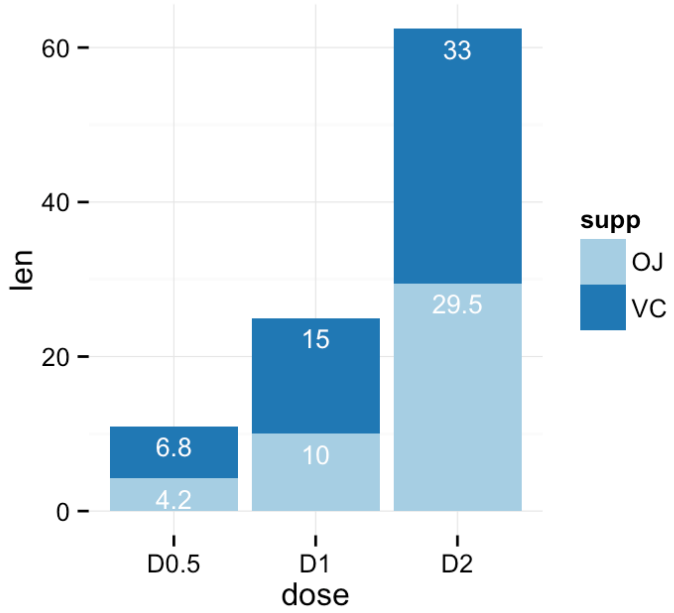


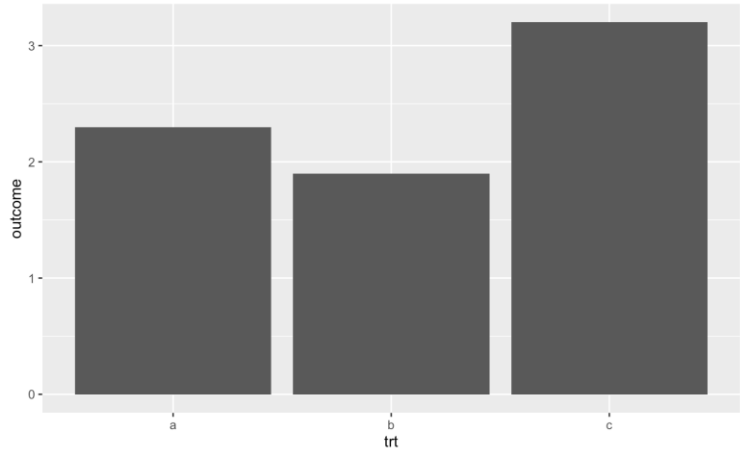
http://www.sthda.com/english/wiki/ggplot2-barplots-quick-start-guide-r-software-and-data-visualization

Count Plot

There are two types of bar charts:

1. geom_bar() makes the height of the bar proportional to the number of cases in each group (or if the weight aesthetic is supplied, the sum of the weights).
2. geom_col(): If you want the heights of the bars to represent values (e.g. means) in the data



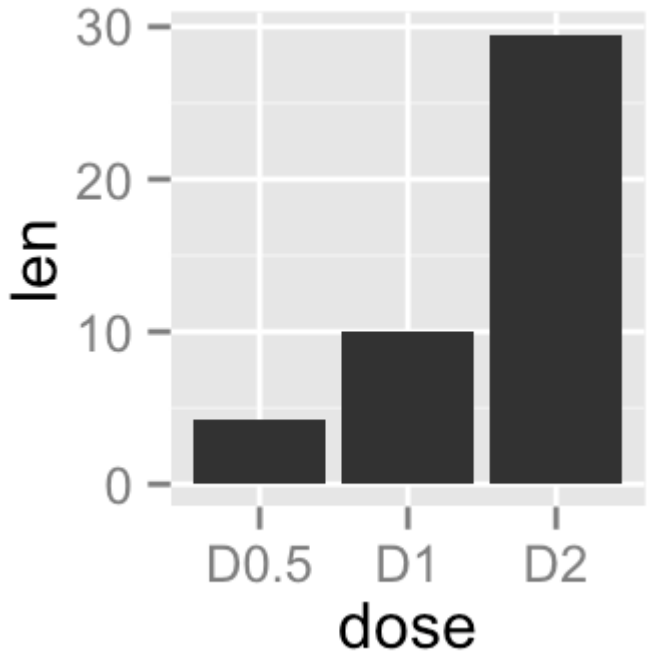


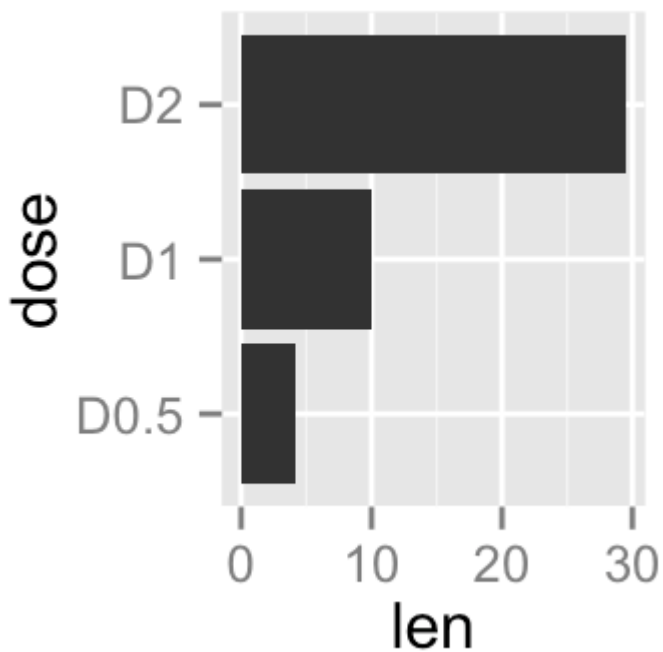
http://www.sthda.com/english/wiki/ggplot2-barplots-quick-start-guide-r-software-and-data-visualization

Syntax

Data derived from ToothGrowth data sets are used. ToothGrowth describes the effect of Vitamin C on Tooth growth in Guinea pigs.

```
# Change the width of bars
ggplot(data=df, aes(x=dose, y=len)) + geom_bar(stat="identity",
width=0.5)
# Change colors
ggplot(data=df, aes(x=dose, y=len)) + geom_bar(stat="identity",
color="blue", fill="white")
# Minimal theme + blue fill color
p<-ggplot(data=df, aes(x=dose, y=len)) + geom_bar(stat="identity",
fill="steelblue")+ theme_minimal()
p
```


30
20
10
0
len
D0.5
D1
D2
dose

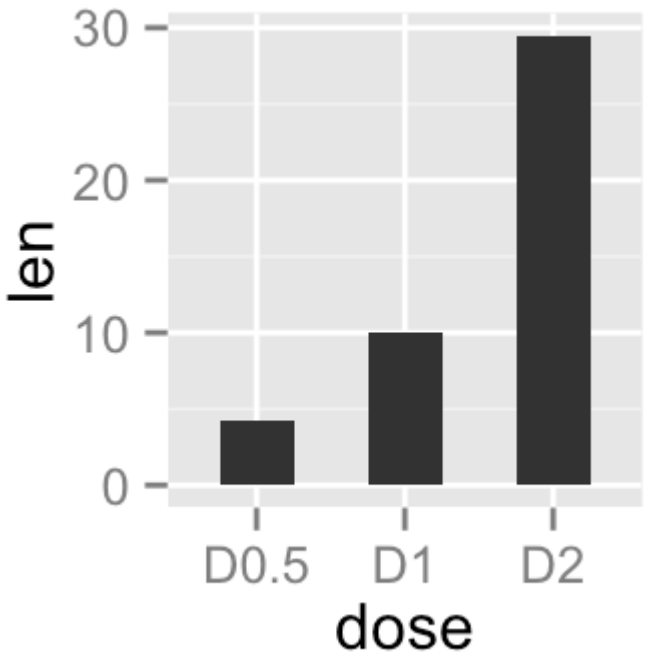

D2
D1
D0.5
dose
0
10
20
30
len

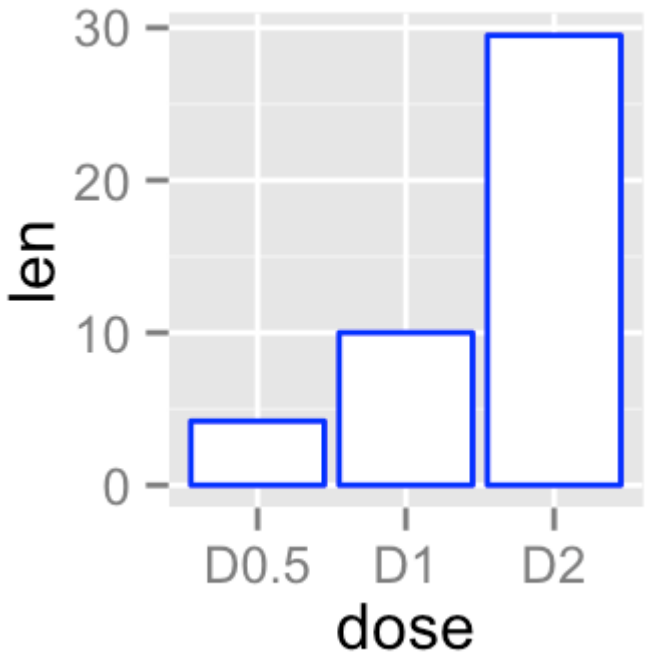
Syntax

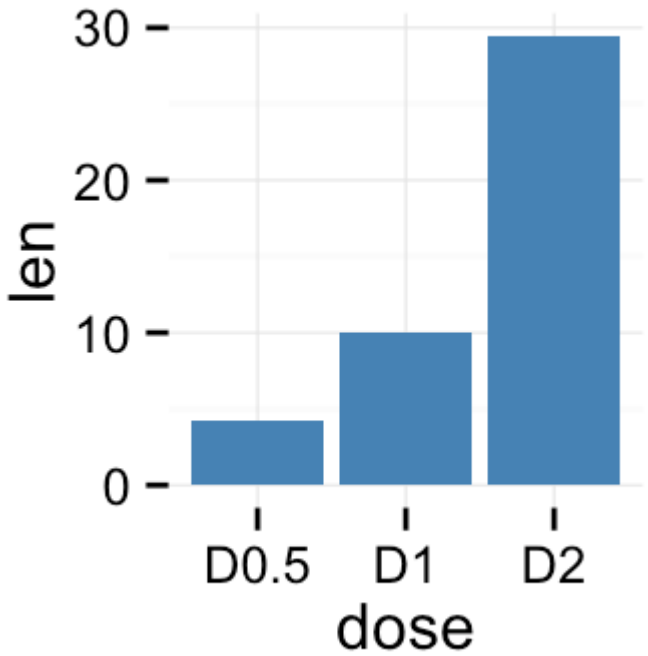
Change the width and the color of bars :

```
df <- data.frame(dose=c("D0.5", "D1", "D2"), len=c(4.2, 10, 29.5))
head(df)
```

```
library(ggplot2)
# Basic barplot
p<-ggplot(data=df, aes(x=dose, y=len))
+ geom_bar(stat="identity")
p
# Horizontal bar plot
p + coord_flip()
```


30
20
10
0
len
D0.5
D1
D2
dose

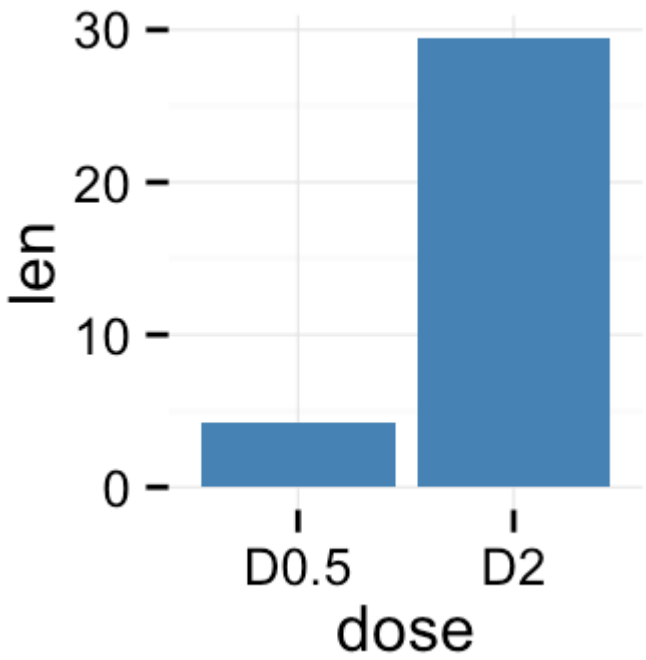

30
20
10
0
len
D0.5
D1
D2
dose


30
20
10
0
len
D0.5
D1
D2
dose

Syntax

Choose which items to display

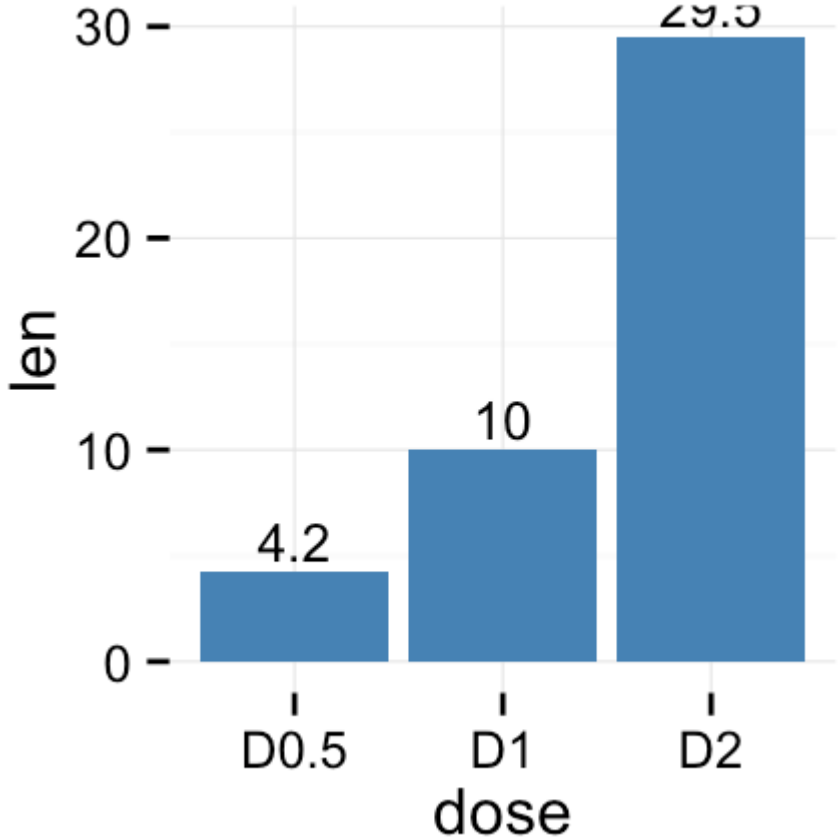
```
p + scale_x_discrete(limits=c("D0.5", "D2"))
```

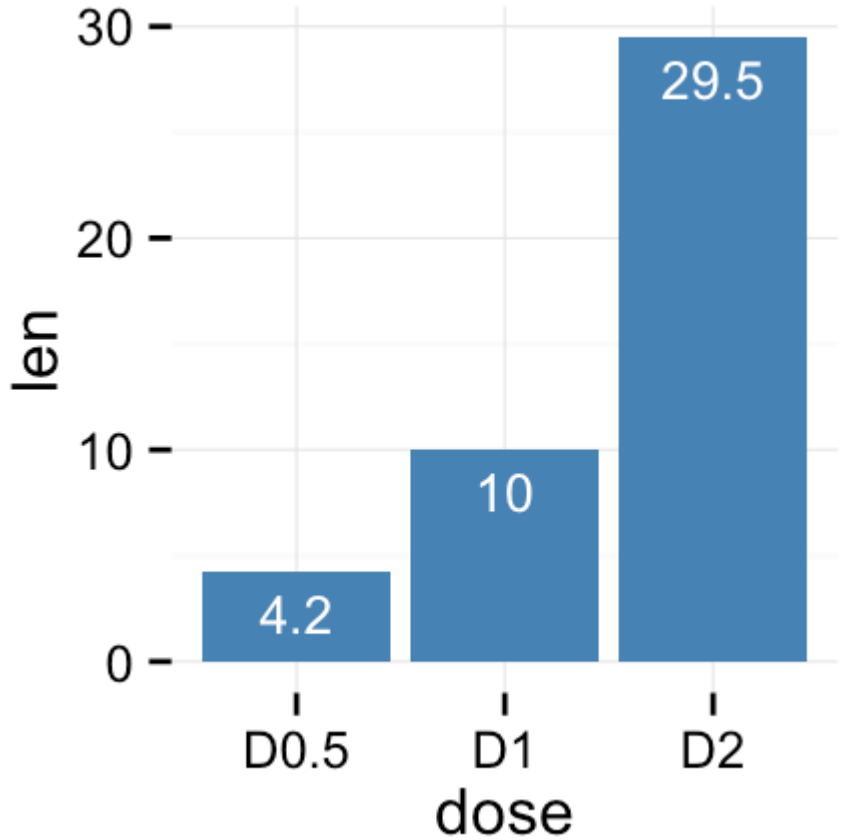


Syntax

Bar plot with labels

```
# Outside bars
ggplot(data=df, aes(x=dose, y=len)) +
 geom_bar(stat="identity", fill="steelblue")+
 geom_text(aes(label=len), vjust=-0.3, size=3.5)+
 theme_minimal()

# Inside bars
ggplot(data=df, aes(x=dose, y=len)) +
 geom_bar(stat="identity", fill="steelblue")+
 geom_text(aes(label=len), vjust=1.6, color="white", size=3.5)+
 theme_minimal()
```


30
20
10
0
len
4.2
10
29.5
D0.5
D1
D2
dose


30
20
10
0
len
4.2
10
29.5
D0.5
D1
D2
dose

Syntax

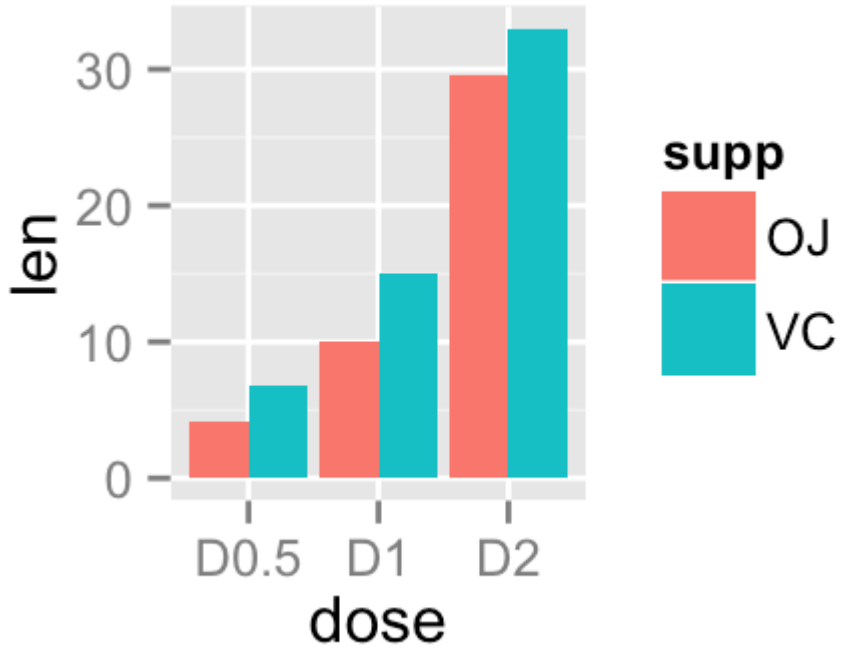
Data : ToothGrowth.

```
df2 <- data.frame(supp=rep(c("VC", "OJ"), each=3), dose=rep(c("D0.5",
"D1", "D2"),2), len=c(6.8, 15, 33, 4.2, 10, 29.5))
head(df2)
```

A stacked bar plot is created by default. You can use the function `position_dodge()` to change this. The bar plot fill color is controlled by the levels of dose : plot with labels

```
# Stacked barplot with multiple groups
ggplot(data=df2, aes(x=dose, y=len, fill=supp)) +
 geom_bar(stat="identity")
# Use position=position_dodge()
ggplot(data=df2, aes(x=dose, y=len, fill=supp)) +
 geom_bar(stat="identity", position=position_dodge())
```

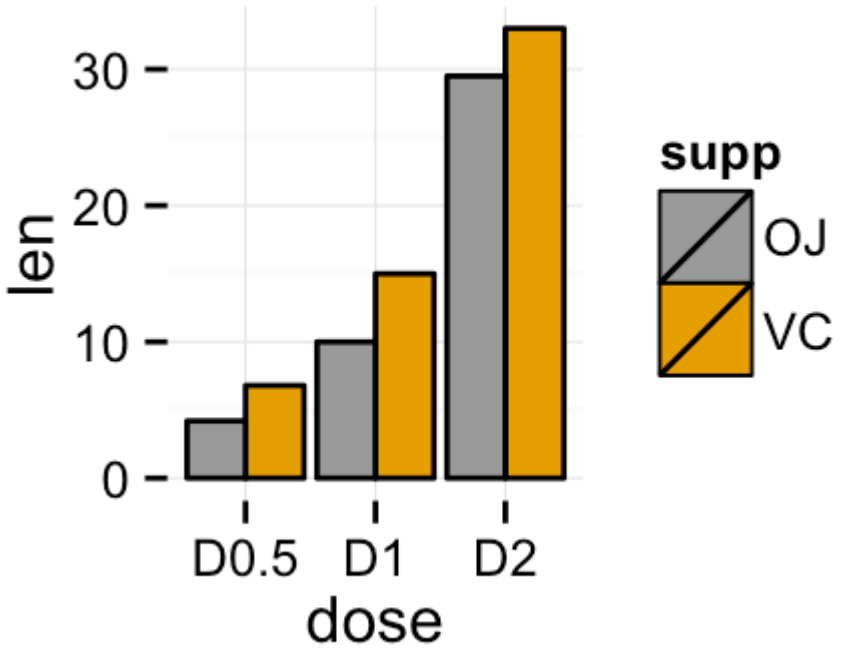

60
40
20
0
len
D0.5
D1
D2
dose
supp
OJ
VC

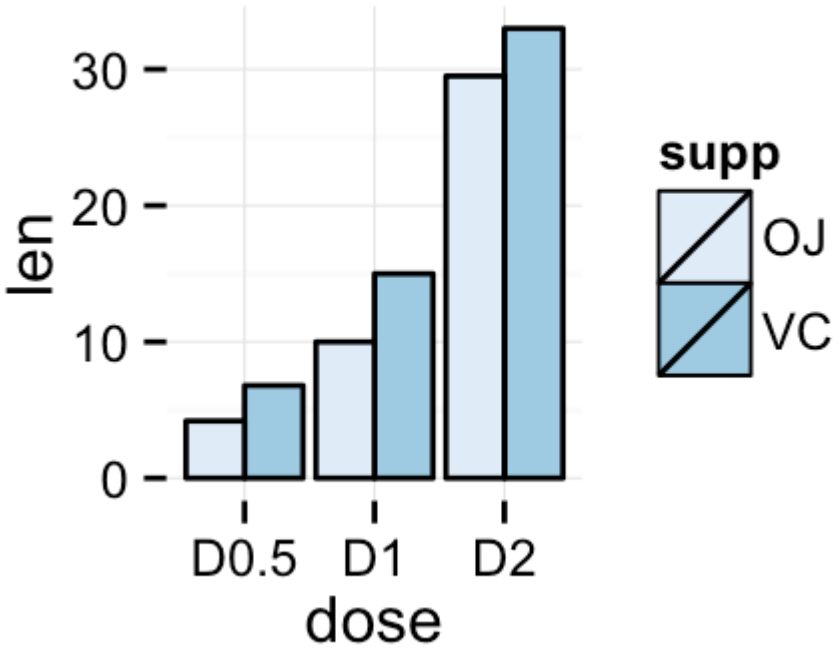

30
20
10
0
len
D0.5
D1
D2
dose
supp
OJ
VC

Syntax

Change the color manually :

```
# Change the colors manually
p <- ggplot(data=df2, aes(x=dose, y=len, fill=supp)) +
 geom_bar(stat="identity", color="black", position=position_dodge())+
 theme_minimal() # Use custom colors p +
 scale_fill_manual(values=c('#999999','#E69F00'))
# Use brewer color palettes
p + scale_fill_brewer(palette="Blues")
```

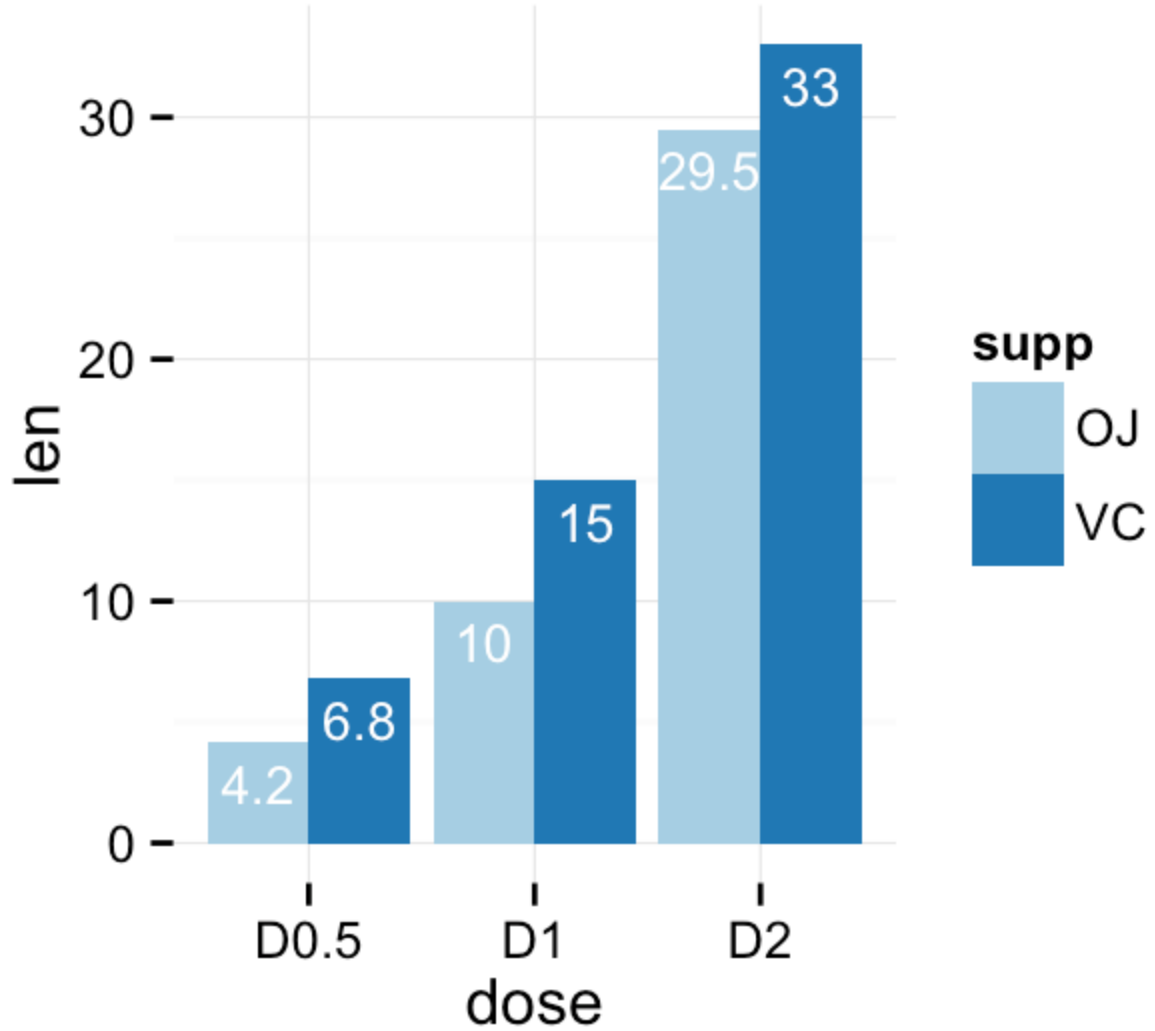

30
20
10
0
len
D0.5
D1
D2
dose
supp
OJ
VC


30
20
10
0
len
D0.5
D1
D2
dose
supp
OJ
VC

Syntax

Add labels to a dodged barplot :

```
ggplot(data=df2, aes(x=dose, y=len, fill=supp)) +
 geom_bar(stat="identity", position=position_dodge())+
 geom_text(aes(label=len), vjust=1.6, color="white",
           position = position_dodge(0.9), size=3.5)+
 scale_fill_brewer(palette="Paired")+
 theme_minimal()
```


30
20
10
0
len
4.2
6.8
10
15
29.5
33
D0.5
D1
D2
dose
supp
OJ
VC

Syntax

Add labels to a stacked barplot : 3 steps are required

1. Sort the data by dose and supp : the package plyr is used
2. Calculate the cumulative sum of the variable len for each dose
3. Create the plot

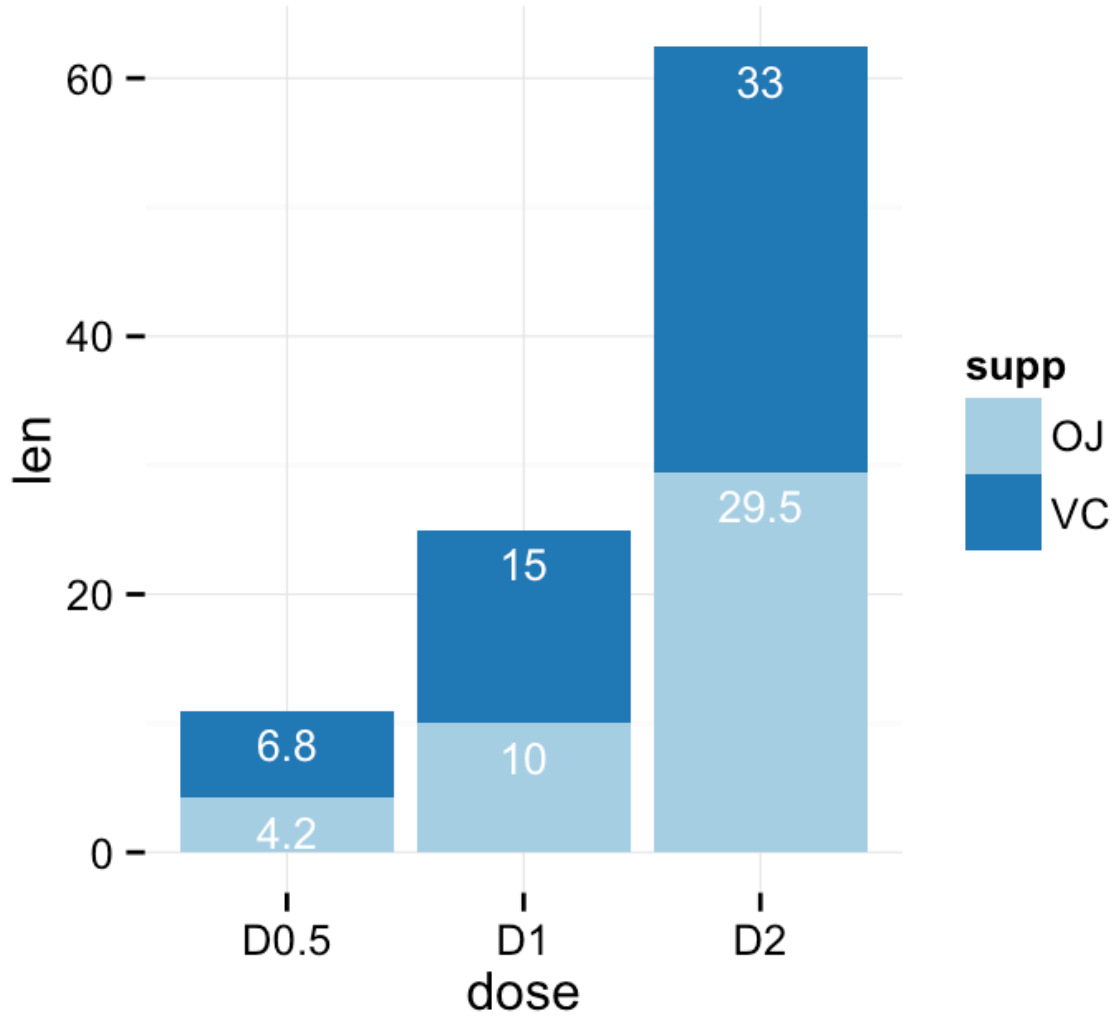
```
library(plyr)
# Sort by dose and supp
df_sorted <- arrange(df2, dose, supp)
head(df_sorted)

# Calculate the cumulative sum of len for each dose
df_cumsum <- ddply(df_sorted, "dose",
                   transform, label_ypos=cumsum(len))
head(df_cumsum)
```

Syntax

Add labels to a stacked barplot : 3 steps are required

1. Sort the data by dose and supp : the package plyr is used
2. Calculate the cumulative sum of the variable len for each dose
3. Create the plot

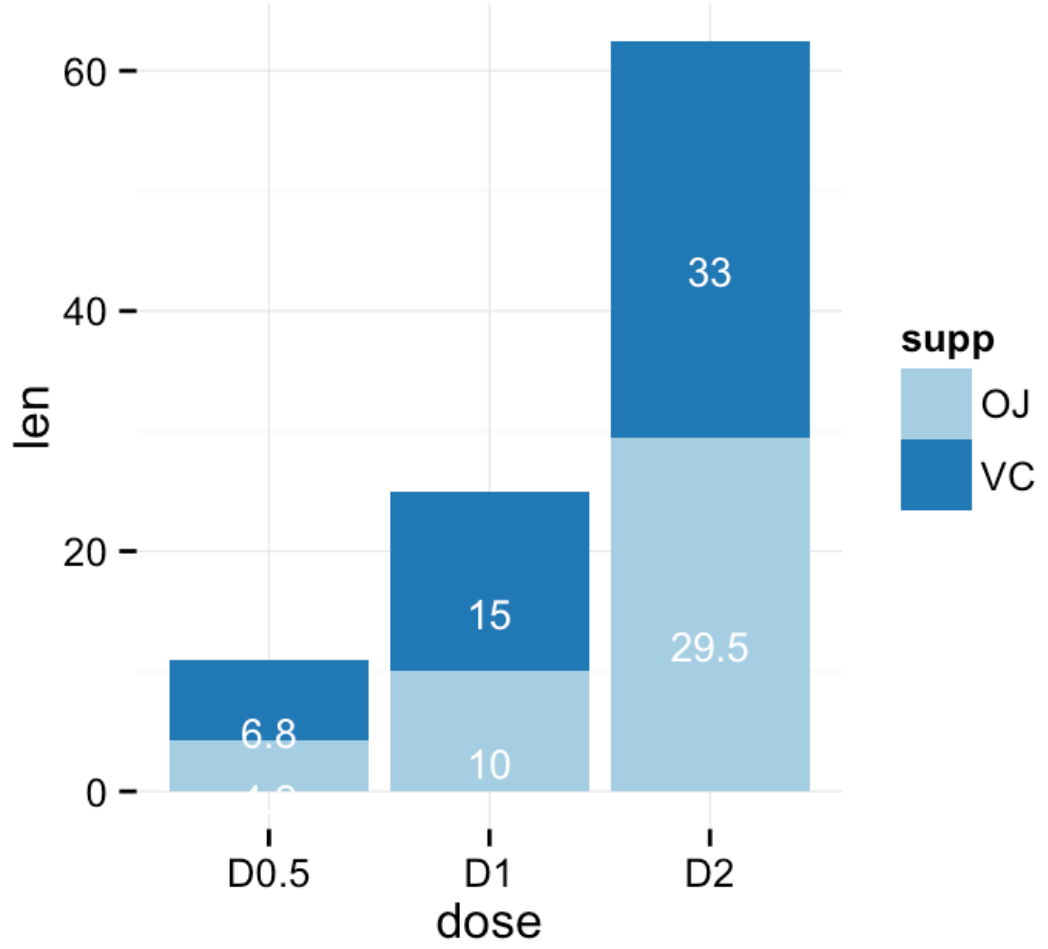
```
# Create the barplot
ggplot(data=df_cumsum, aes(x=dose, y=len, fill=supp)) +
 geom_bar(stat="identity")+
 geom_text(aes(y=label_ypos, label=len), vjust=1.6,
           color="white", size=3.5)+
 scale_fill_brewer(palette="Paired")+
 theme_minimal()
```


60
40
20
0
len
33
29.5
15
10
6.8
4.2
D0.5
D1
D2
dose
supp
OJ
VC

Syntax

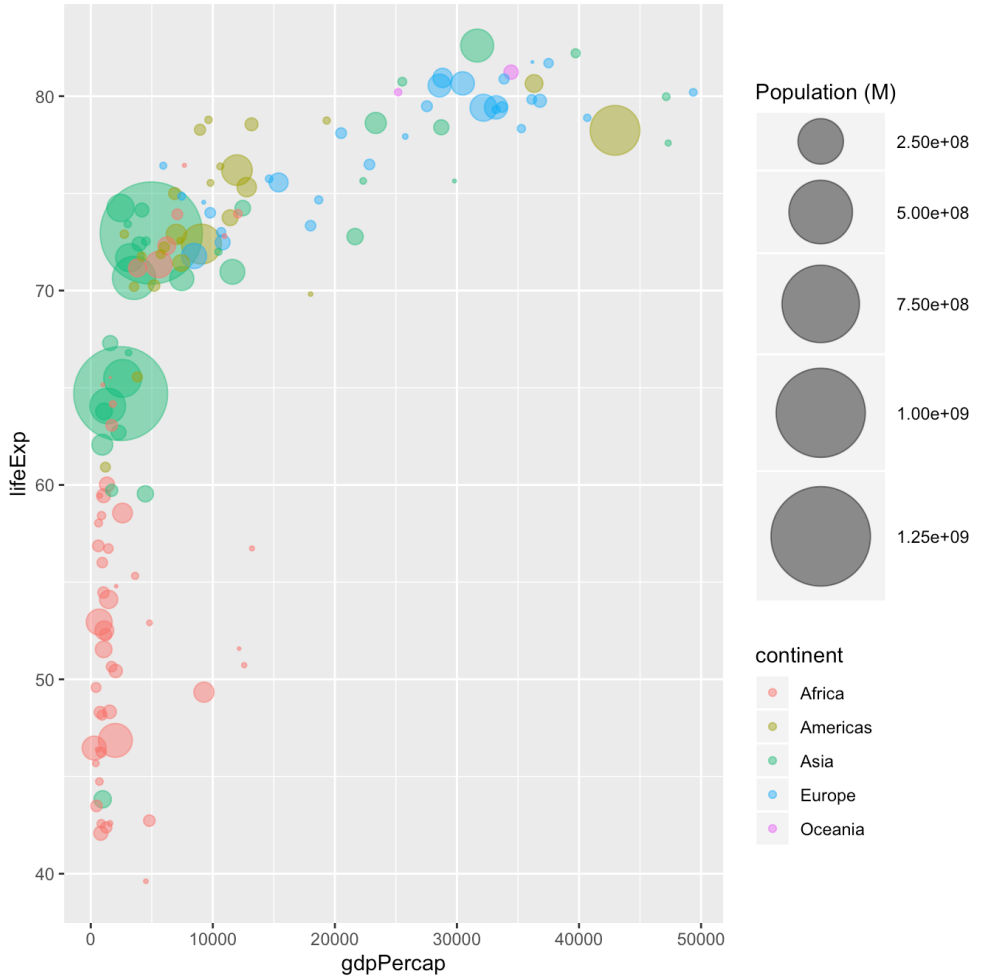
If you want to place the labels at the middle of bars, you have to modify the cumulative sum as follow :

```
df_cumsum <- ddply(df_sorted, "dose",
                   transform,
                    label_ypos=cumsum(len) - 0.5*len)
# Create the barplot
ggplot(data=df_cumsum, aes(x=dose, y=len, fill=supp)) +
 geom_bar(stat="identity")+
 geom_text(aes(y=label_ypos, label=len), vjust=1.6,
           color="white", size=3.5)+
 scale_fill_brewer(palette="Paired")+
 theme_minimal()
```


60
40
20
0
len
33
15
29.5
6.8
10
D0.5
D1
D2
dose
supp
OJ
VC

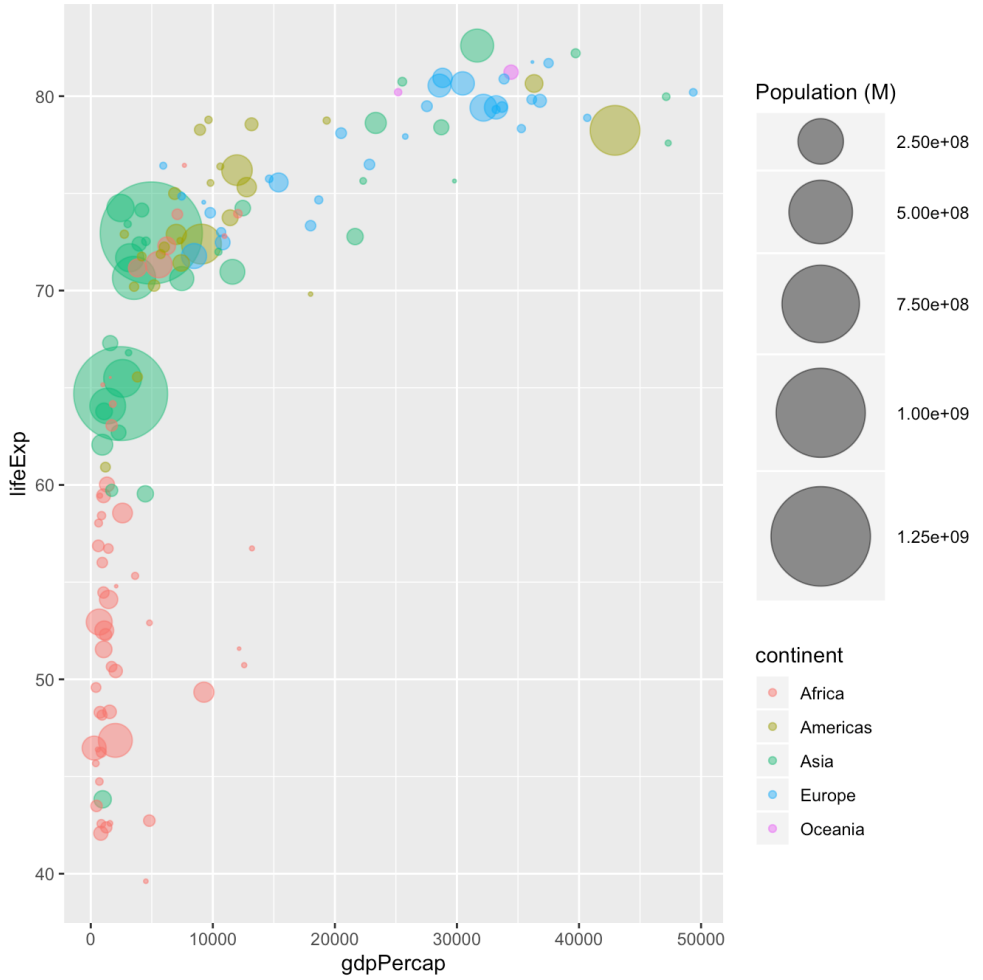
Bubble Plot

The relationship between life expectancy (y) and gdp per capita (x) of world countries is represented. The population of each country is represented through circle size and the continent of each country is used to control circle color.



Bubble Plot

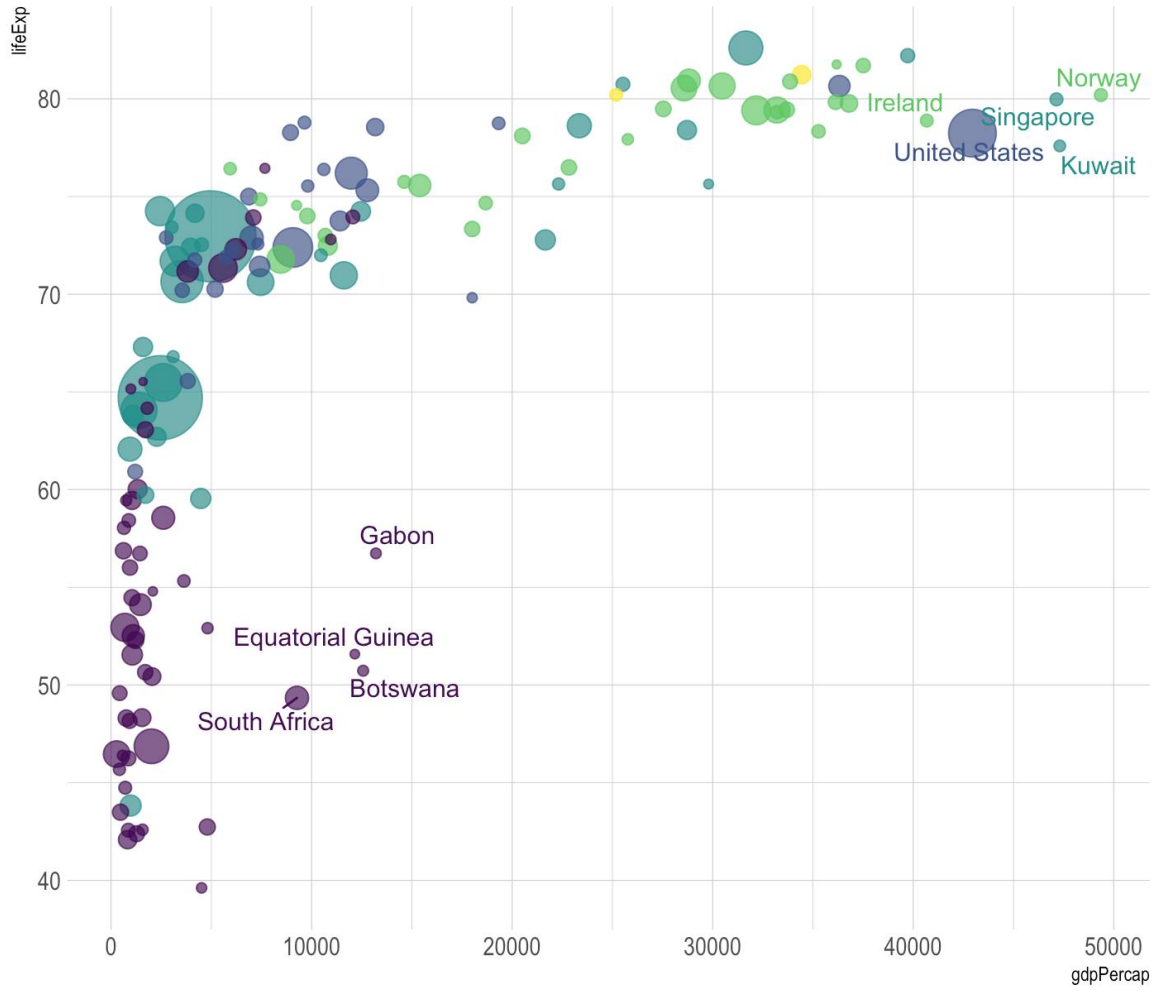
- Rich countries tend to live longer, with a threshold effect when gdp per capita reaches ~10,000.
- This relationship could have been detected using a classic scatterplot, but the bubble size allows to nuance this result with a third level of information: the country population.
- This last variable is much more difficult to interpret than the one on the X and Y axis. Indeed, area is hardly interpreted by the human eye.



Bubble Plot

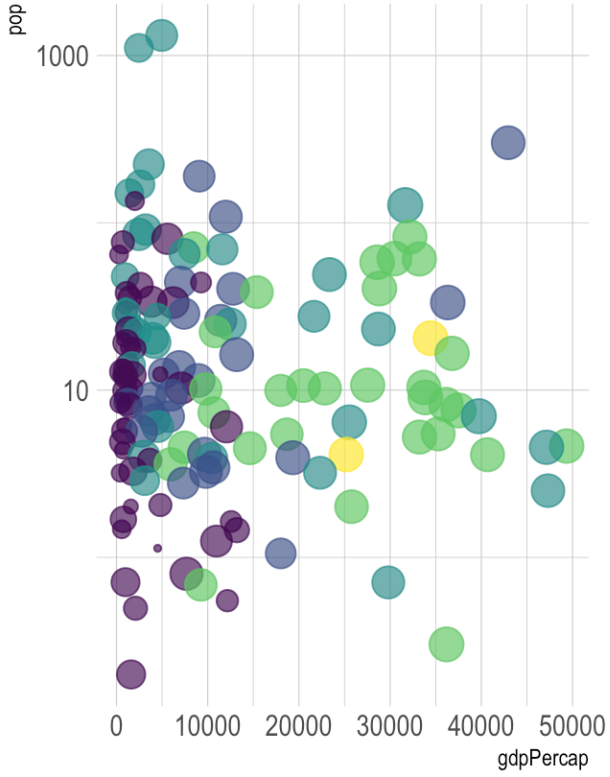
- The previous graphic is quite interesting since it allows to understand the relationship between gdp per capita and life expectancy.
- However it can be frustrating not to know what are the countries in the extreme part of the graphic, or what are the one out of the general trend.

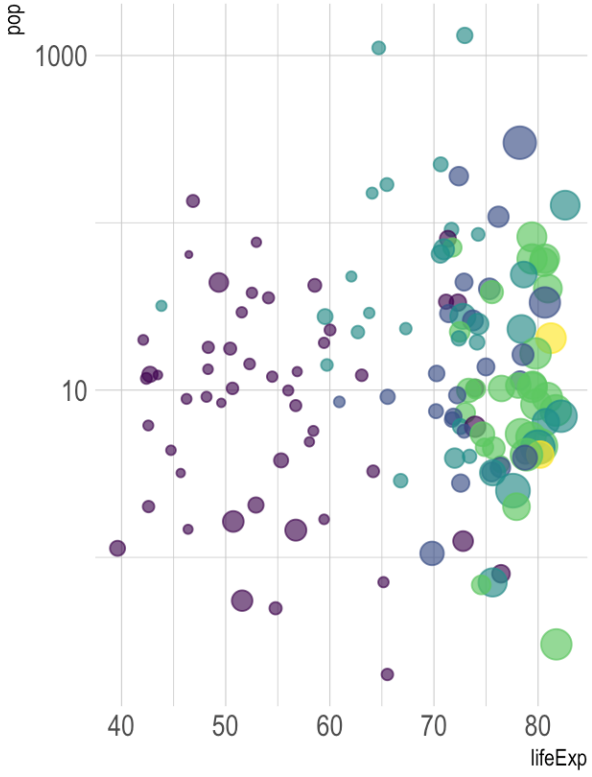
→ Annotating the graphic is a crucial step to make it insightful:



Common Mistake on Bubble Plot

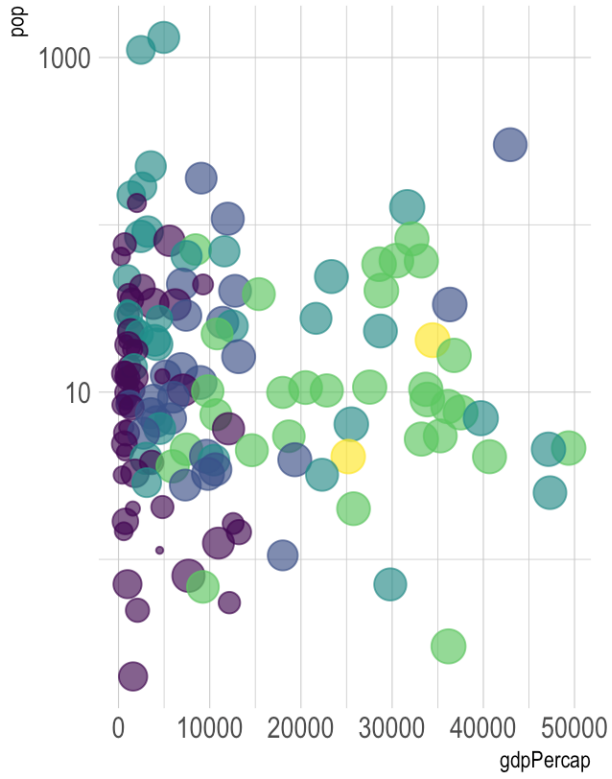
- The relationship between the variable of the X and Y axis is much more obvious than the relationship with the third variable.
- Prioritize the variables and be sure of what to show.

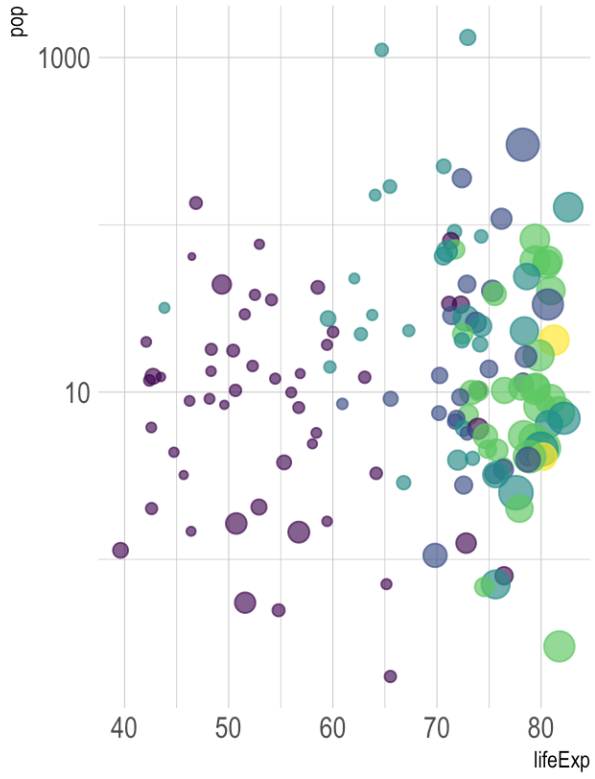




Common Mistake on Bubble Plot

- Use bubble area as metrics, not diameter.
- As for scatter, bubble plot suffers overplotting if sample size is too big.
- Show a legend for bubble size.



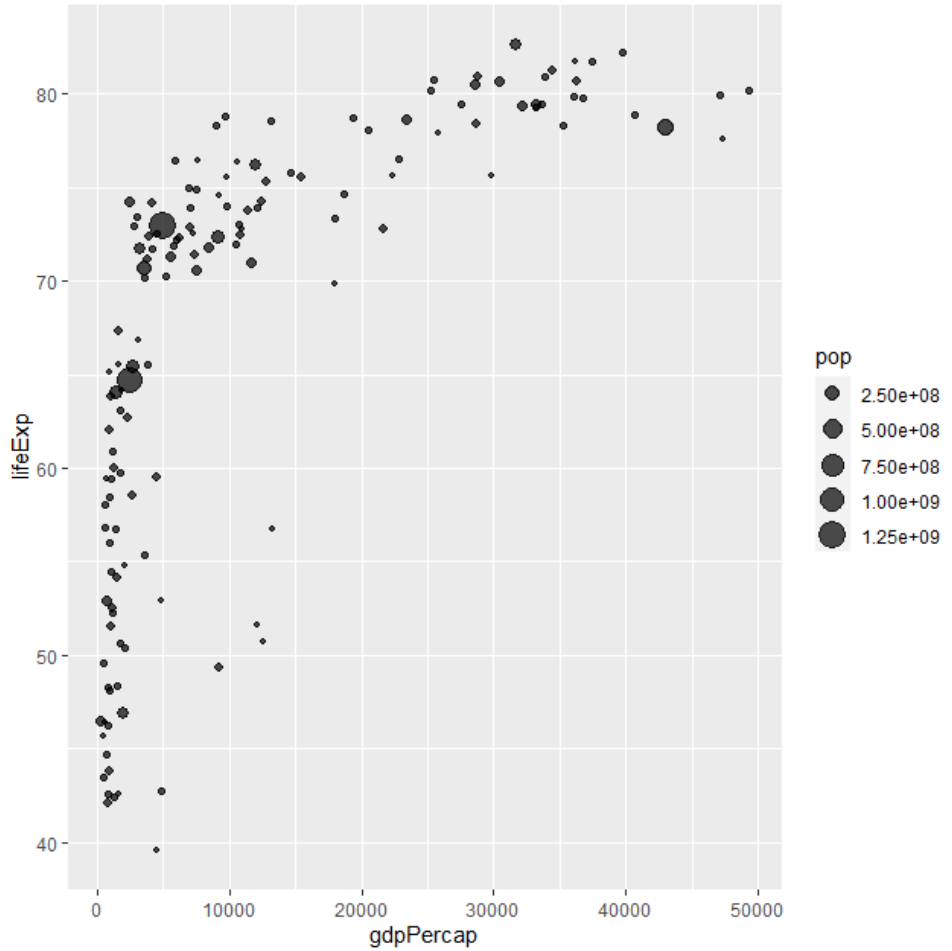


https://www.r-graph-gallery.com/320-the-basis-of-bubble-plot.html

Syntax

Data : Gapminder library

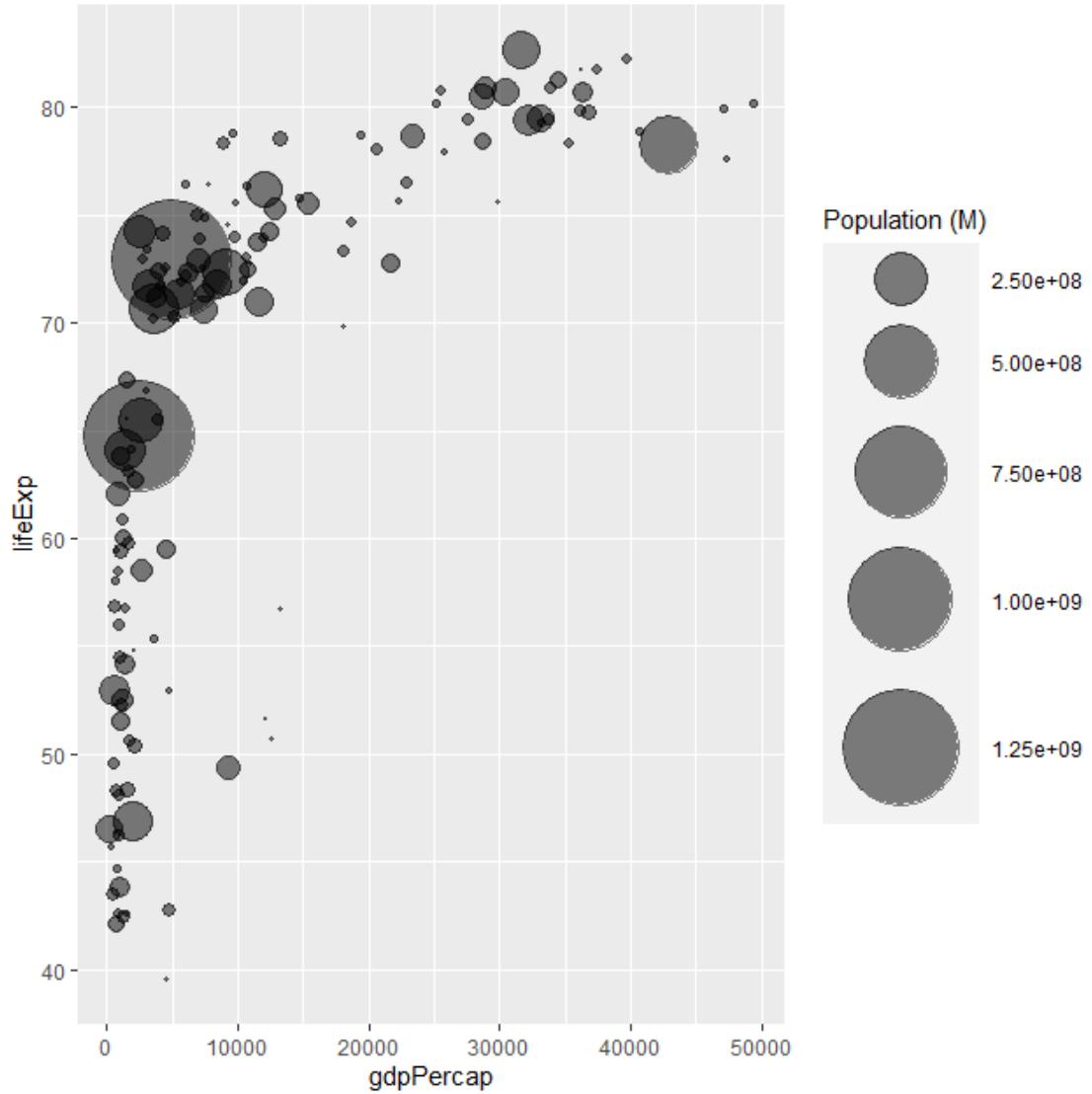
```
# Libraries
library(ggplot2)
library(dplyr)

# The dataset is provided in the gapminder library
library(gapminder)
data <- gapminder %>% filter(year=="2007") %>% dplyr::select(-year)

# Most basic bubble plot
ggplot(data, aes(x=gdpPercap, y=lifeExp, size = pop)) +
        geom_point(alpha=0.7)
```


80
70
60
50
40
lifeExp
0
10000
20000
30000
40000
50000
gdpPercap
pop
2.50e+08
5.00e+08
7.50e+08
1.00e+09
1.25e+09

Syntax

Control circle size with scale_size()

```
# Most basic bubble plot
data %>%
  arrange(desc(pop)) %>%
  mutate(country = factor(country, country)) %>%
  ggplot(aes(x=gdpPercap, y=lifeExp, size = pop)) +
              geom_point(alpha=0.5) +
              scale_size(range = c(.1, 24), name="Population (M)")
```


80
70
60
50
40
lifeExp
0
10000
20000
30000
40000
50000
gdpPercap
Population (M)
2.50e+08
5.00e+08
7.50e+08
1.00e+09
1.25e+09