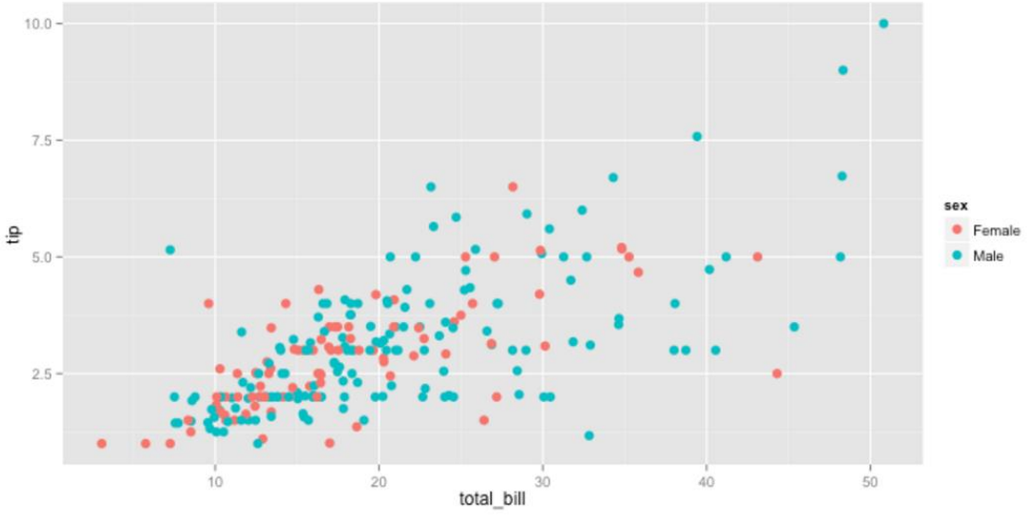
Advanced Analytics with R

Advanced Data Visualization

Three Main Components of Graph

- Primary data component design
  - Bars, dots, lines, etc.
- Secondary data component design
  - Text, grid lines, sticks, legends
- Non-data component design
  - Background

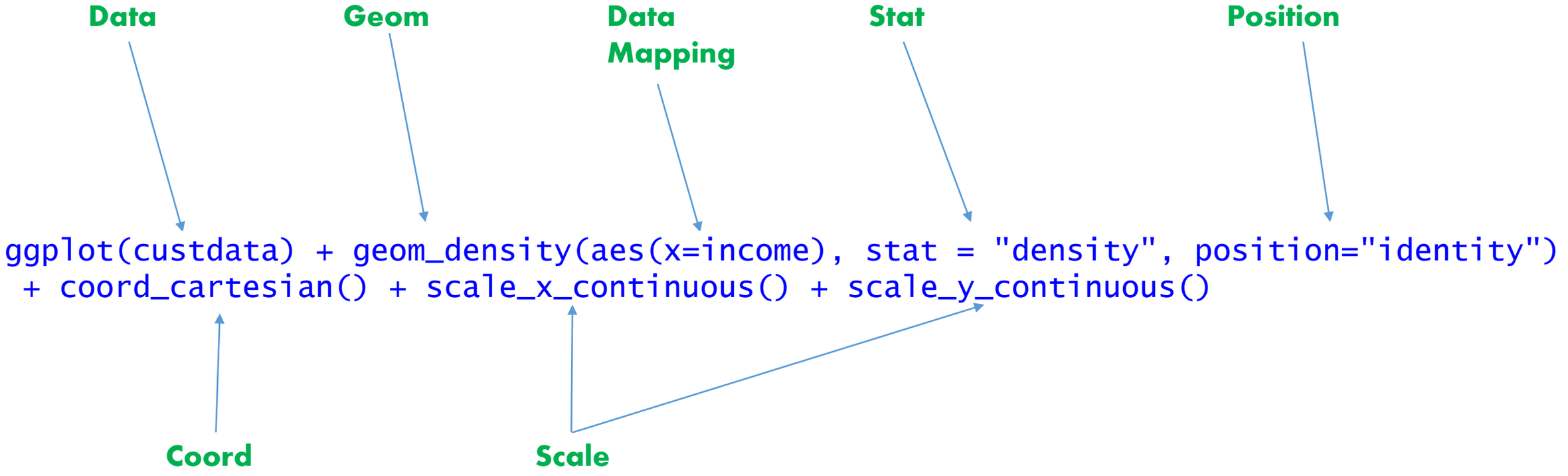


Components of ggplot2

- A default dataset with aesthetic mappings,
- One or more layers, each with a geometric object (“geom”), a statistical transformation (“stat”), and a dataset with aesthetic mappings (possibly defaulted),
- A scale for each aesthetic mapping (which can be automatically generated),
- A coordinate system, and
- A facet specification.

Grammar of Graphics

- Data source
- Data mapping
- Geom
- Coord
- Scale
- Facet ? Layer
- Stat
- Position


Data
Geom
Data Mapping
Stat
Position
ggplot(custdata) + geom_density(aes(x=income), stat = "density", position="identity")
+ coord_cartesian() + scale_x_continuous() + scale_y_continuous()
Coord
Scale

Common Geoms

- geom_Line
- geom_point
- geom_bar
- geom_density
- geom_histogram
- geom_smooth
- geom_abline
- geom_boxplot
- geom_hex

Common stat

- identity
- count
- sum
- unique
- spoke
- function
- summary

Geom specific stat

- bin
- contour
- abline
- Density
- Boxplot
- qq
- smooth
- quantile

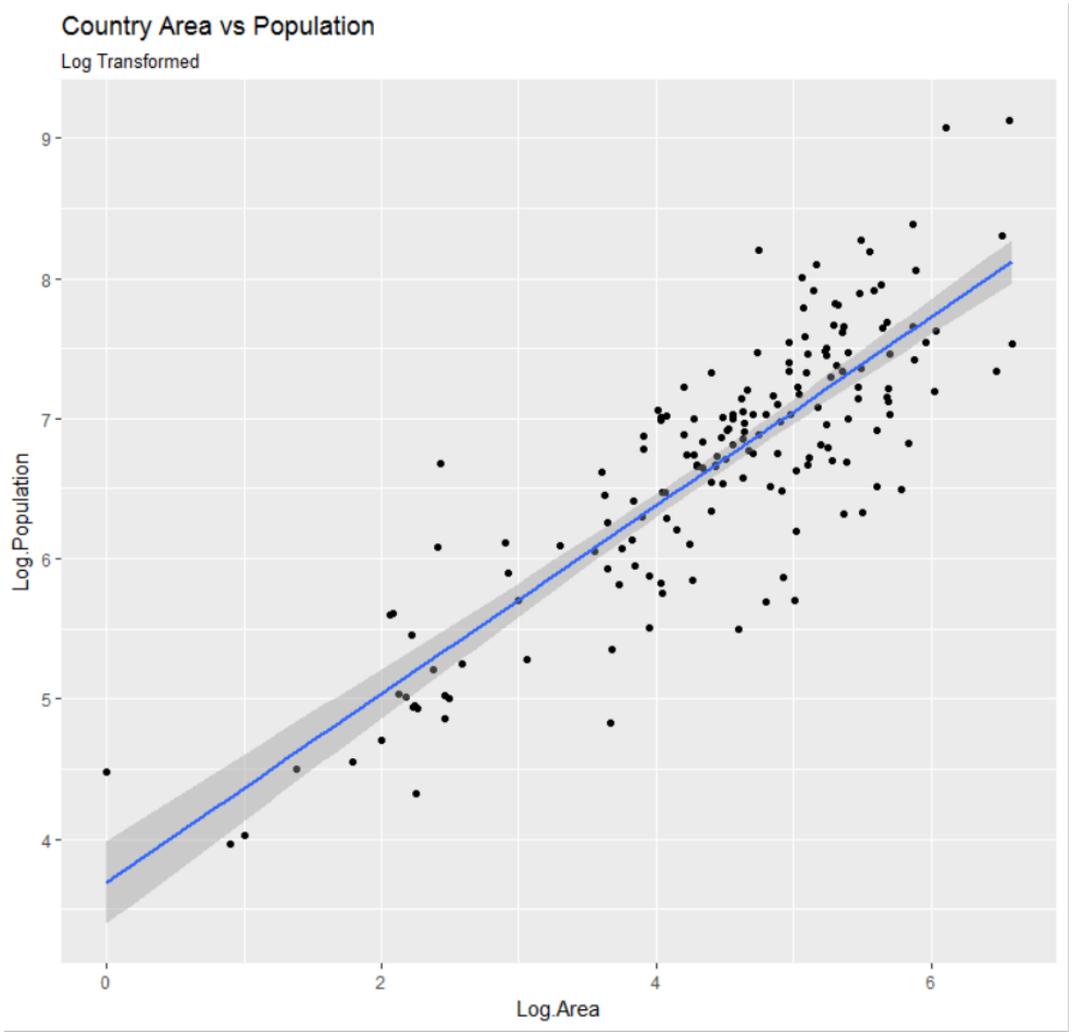
position

- identity
- jitter
- dodge
- stack
- fill

Chart Components

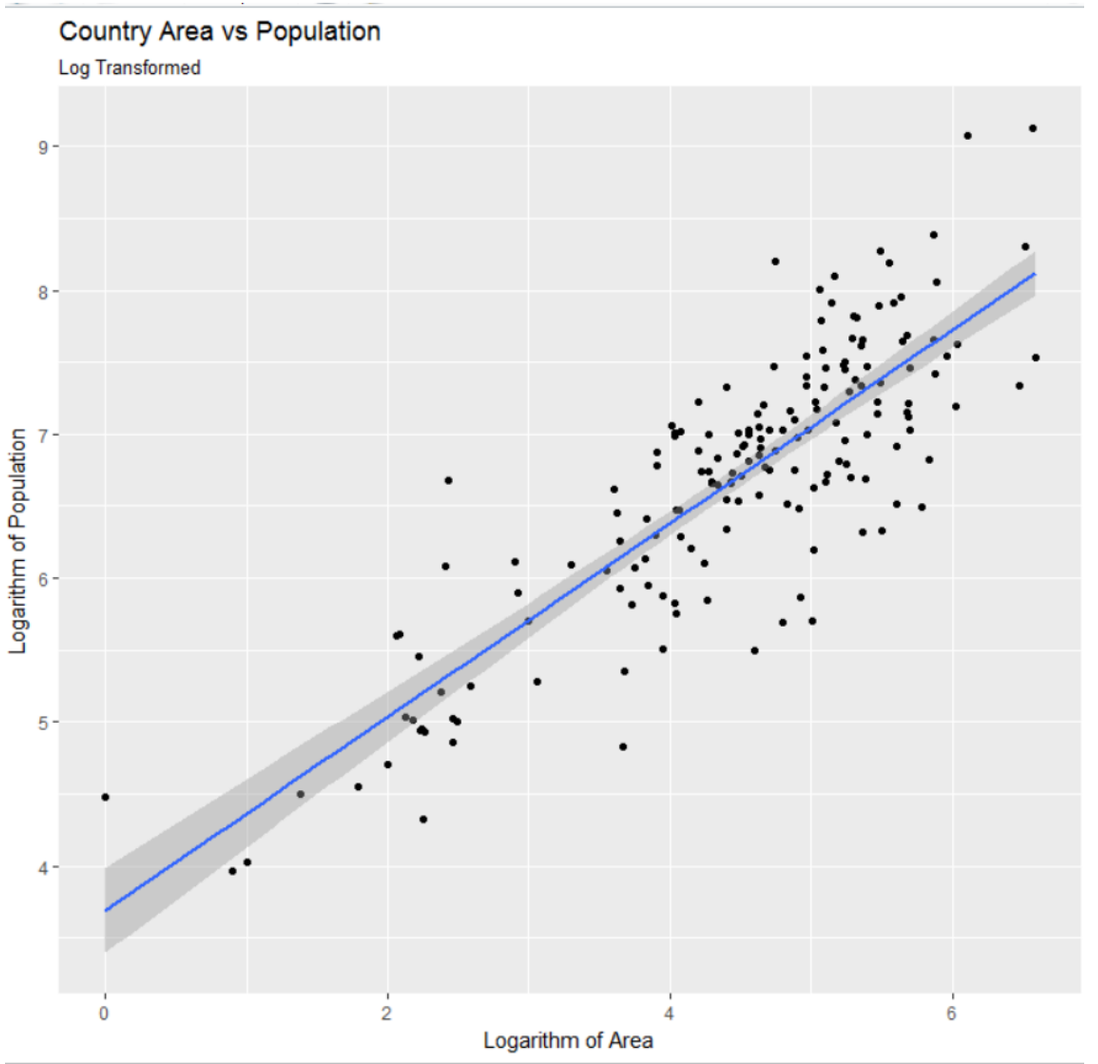
Title

```
g1 <- ggplot(df,aes(x=Log.Area,y=Log.Population)) + geom_point() + geom_smooth(method="lm")
g2 <- g1 + labs(title = "Country Area vs Population", subtitle = "Log Transformed")
```



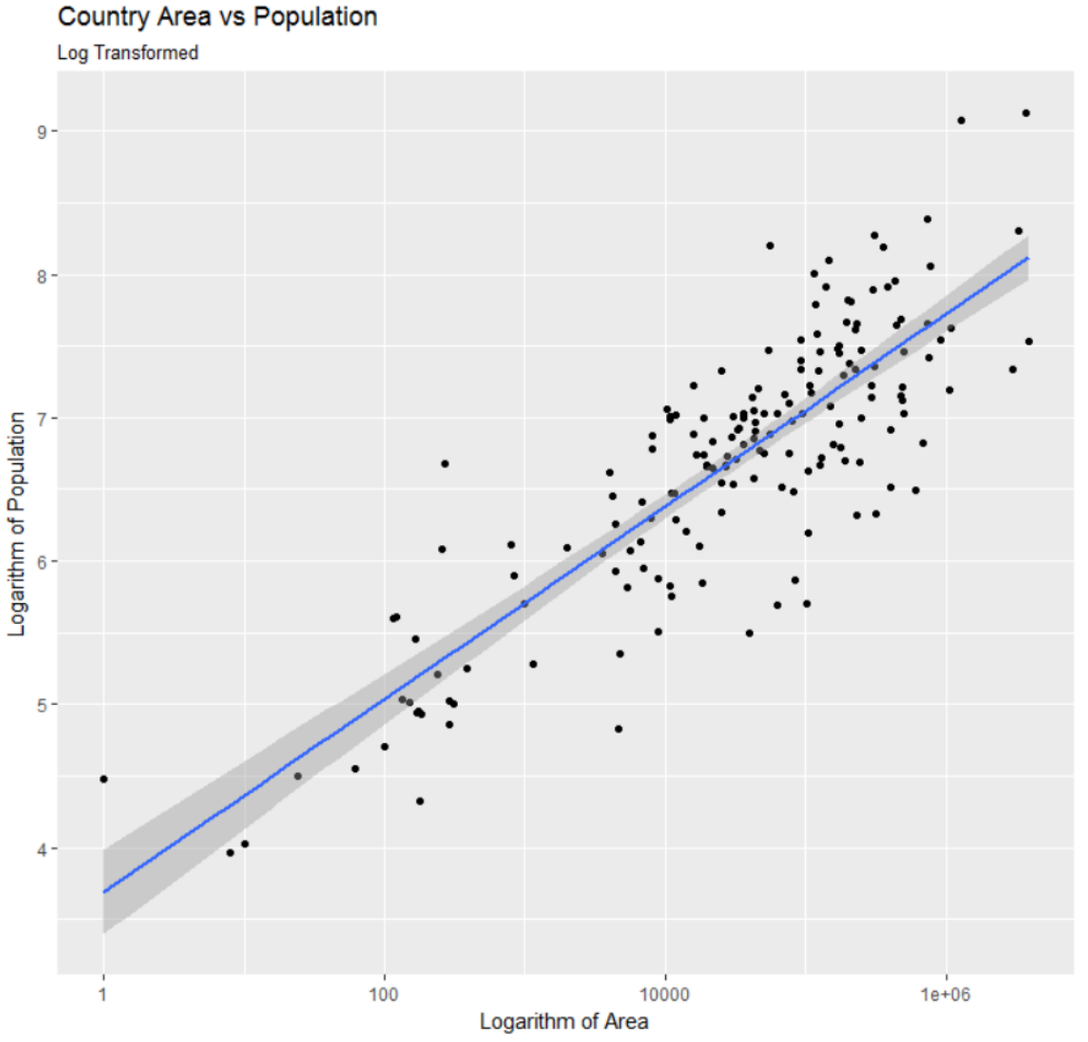
Axis

```
g3 <- g2 + labs(x="Logarithm of Area", y="Logarithm of Population")
```



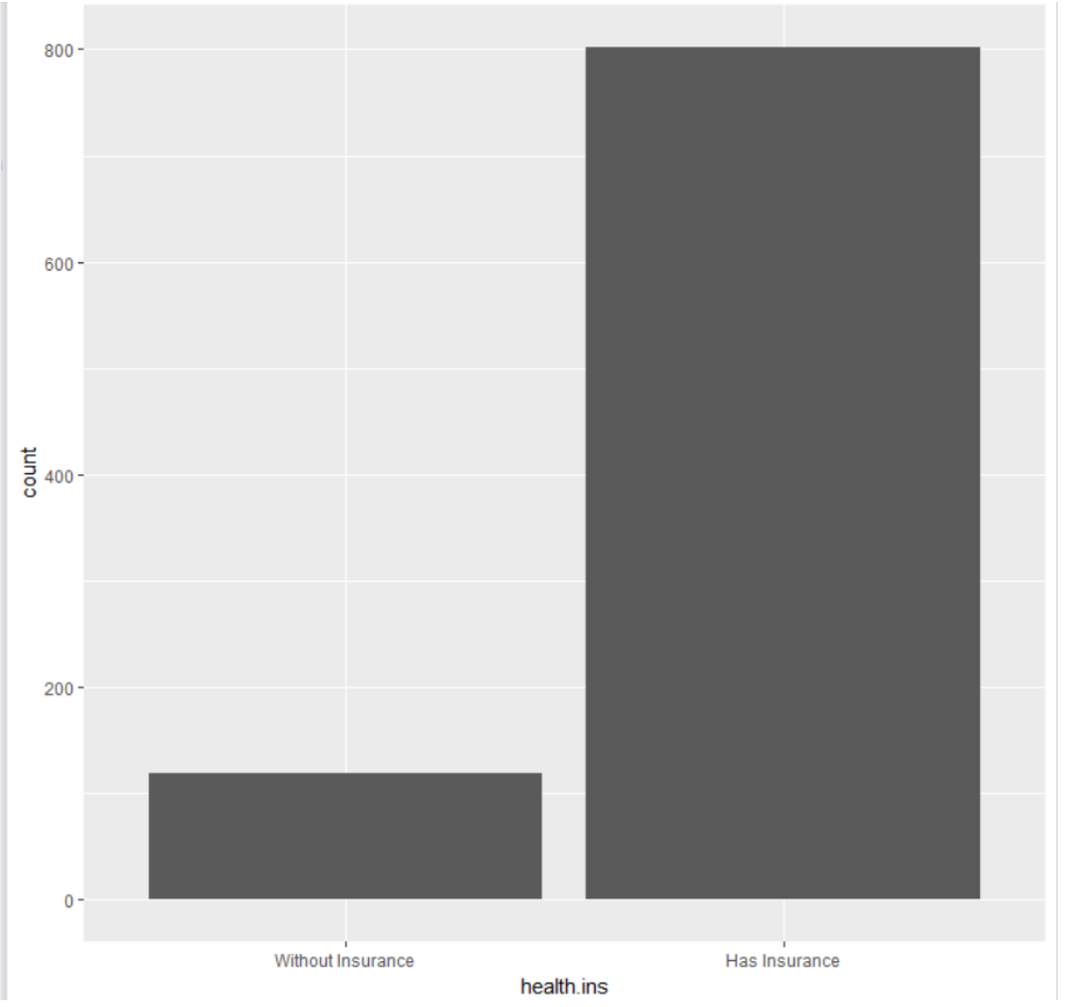
Change Labels with Function

```
g4 <- g3 + scale_x_continuous(labels = function(x) 10^x)
```



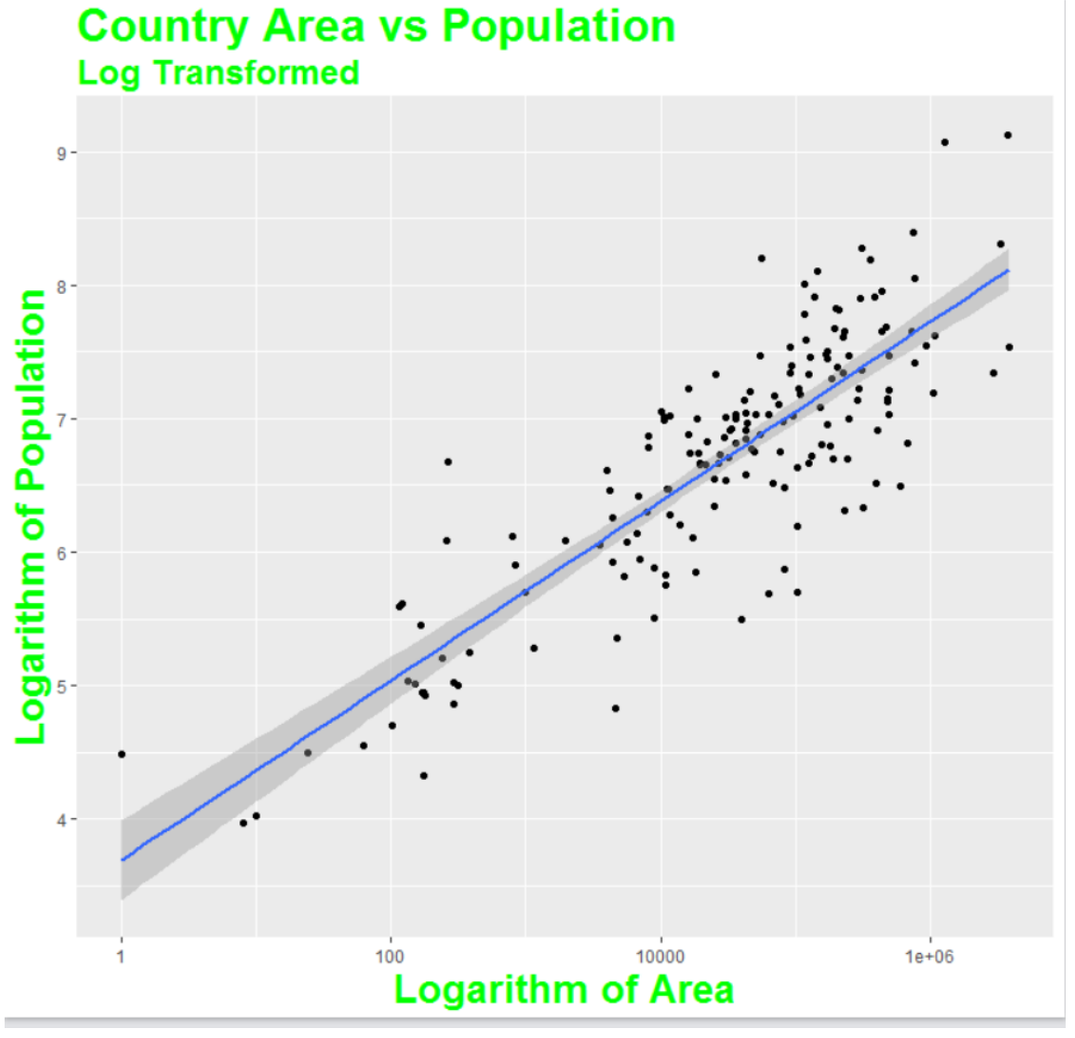
Another Example

```
ggplot(custdata) + geom_bar(aes(x=health.ins))
+scale_x_discrete(labels = function(x) ifelse(x, "Has Insurance", "Without Insurance"))
```



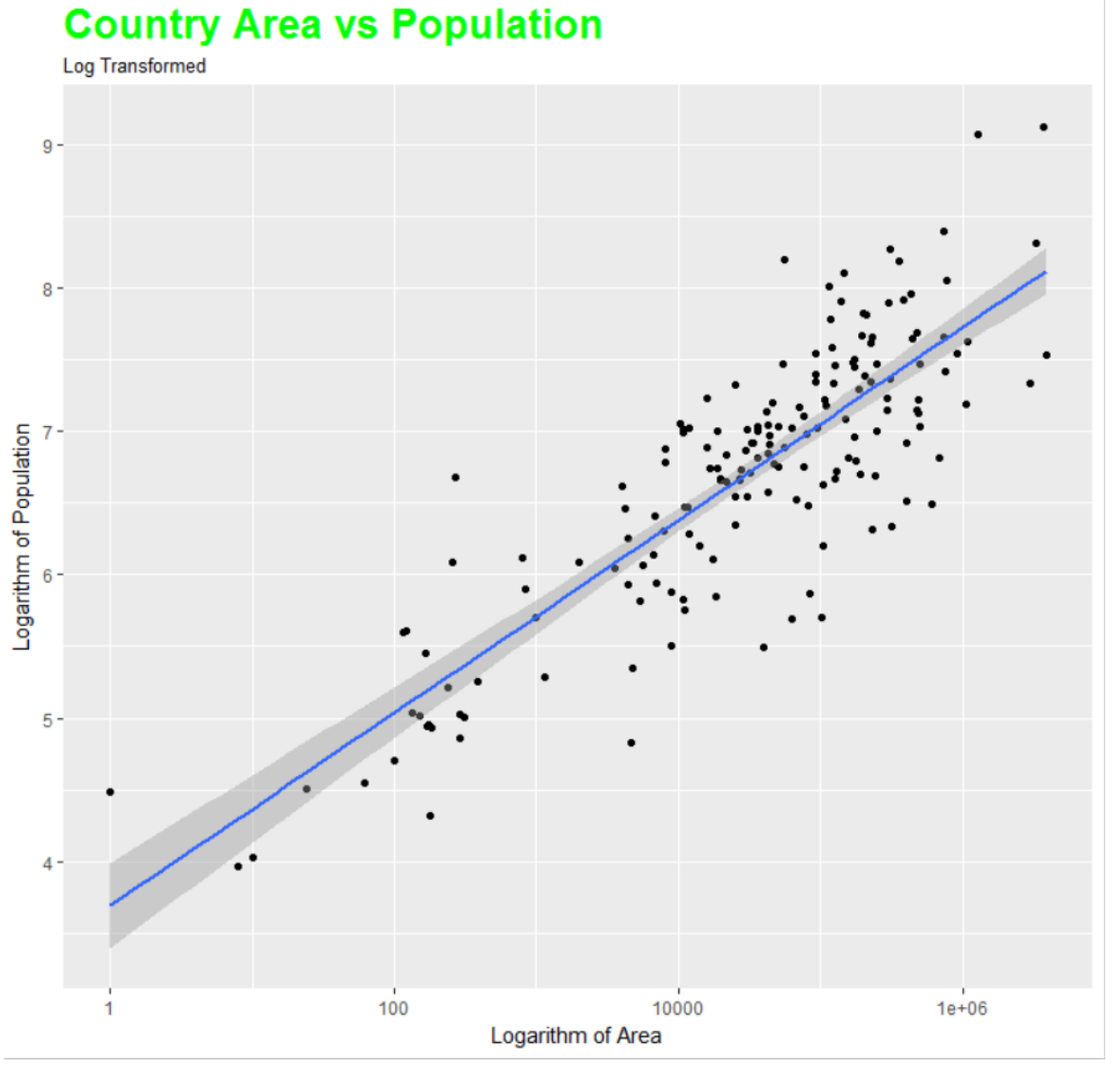
Title Theme

```
g5 <- g4 + theme(title=element_text(size=20,face="bold",color="green"))
```



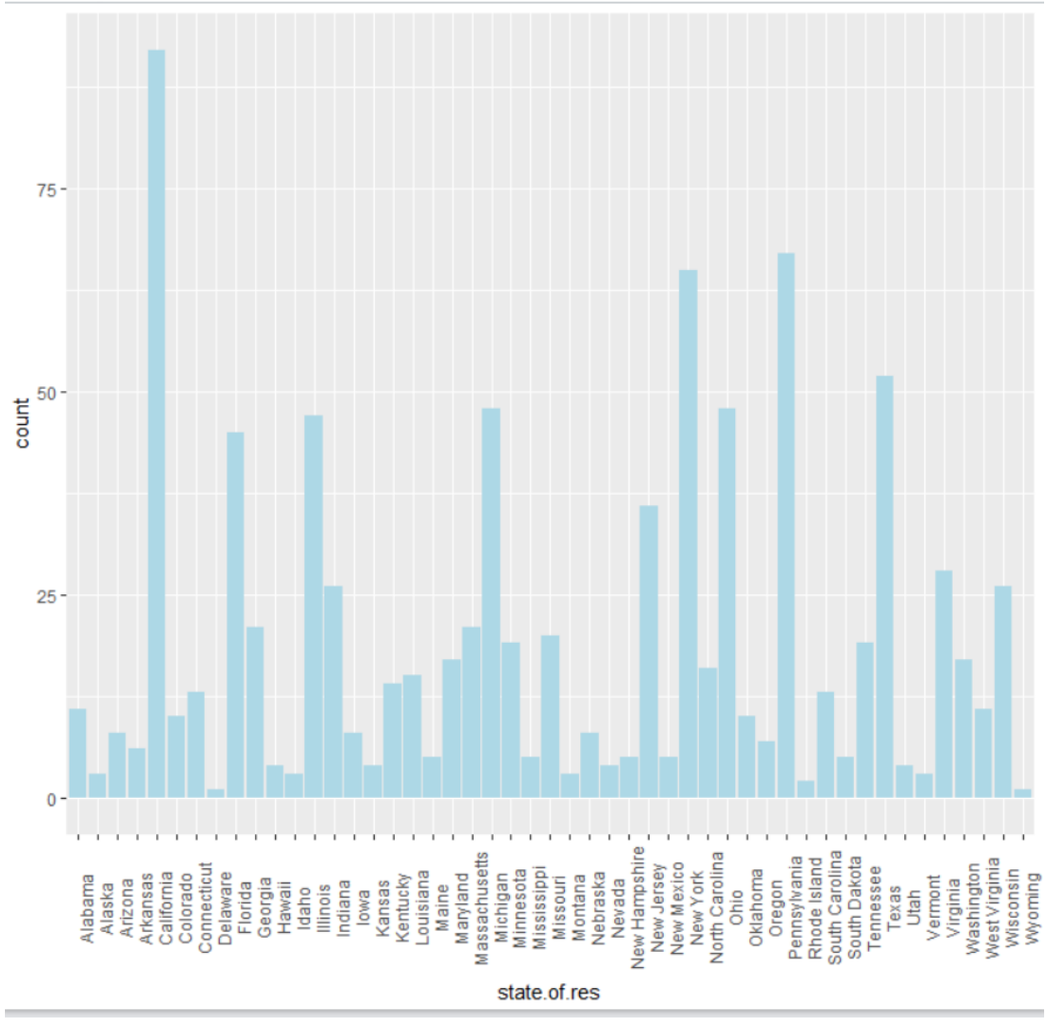
Change only main title

```
g5 <- g4 + theme(plot.title =element_text(size=20,face="bold",color="green"))
```



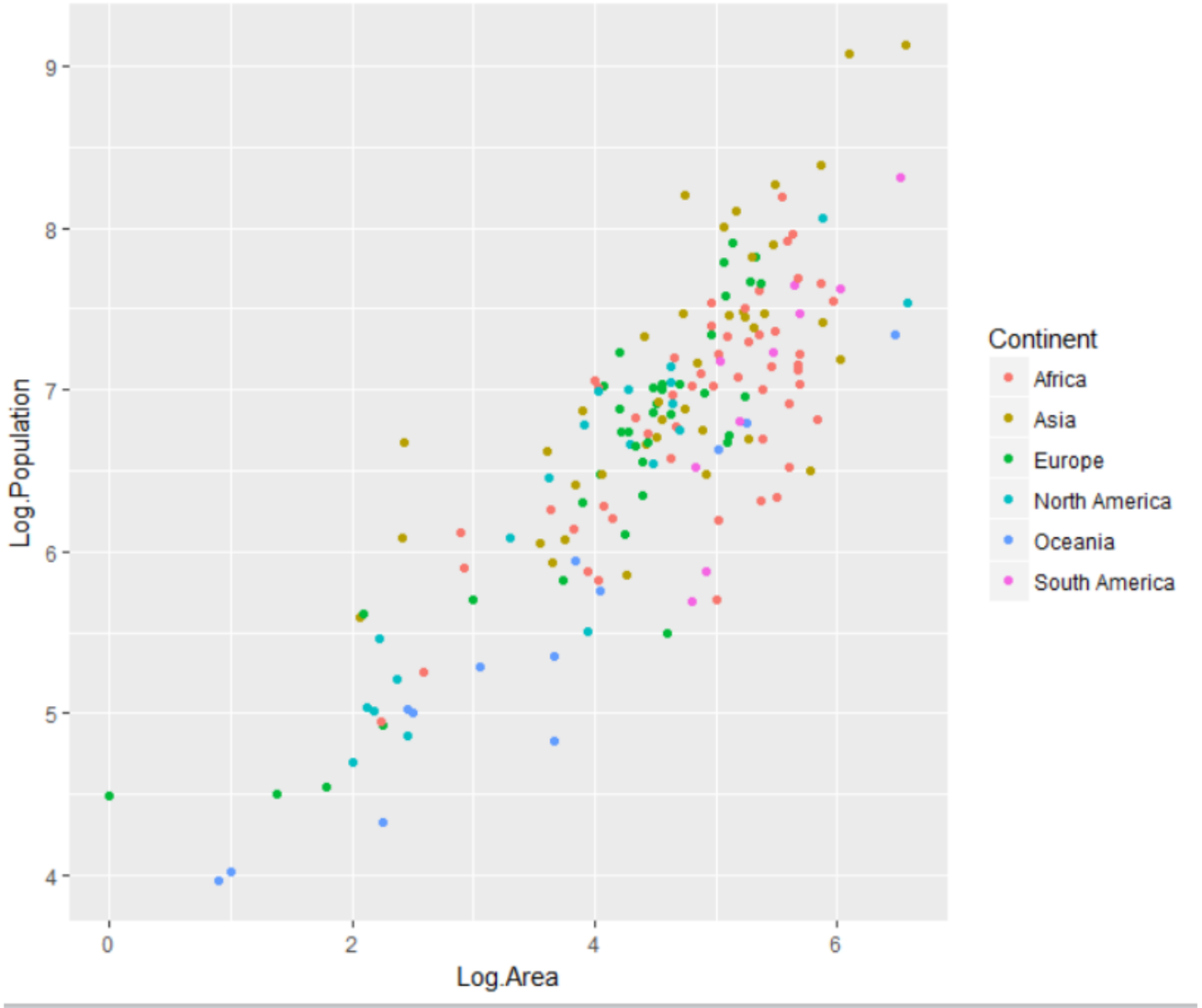
Tick text theme

```
ggplot(custdata) + geom_bar(aes(x=state.of.res), fill="lightblue") + theme(axis.text.x=element_text(angle=90))
```



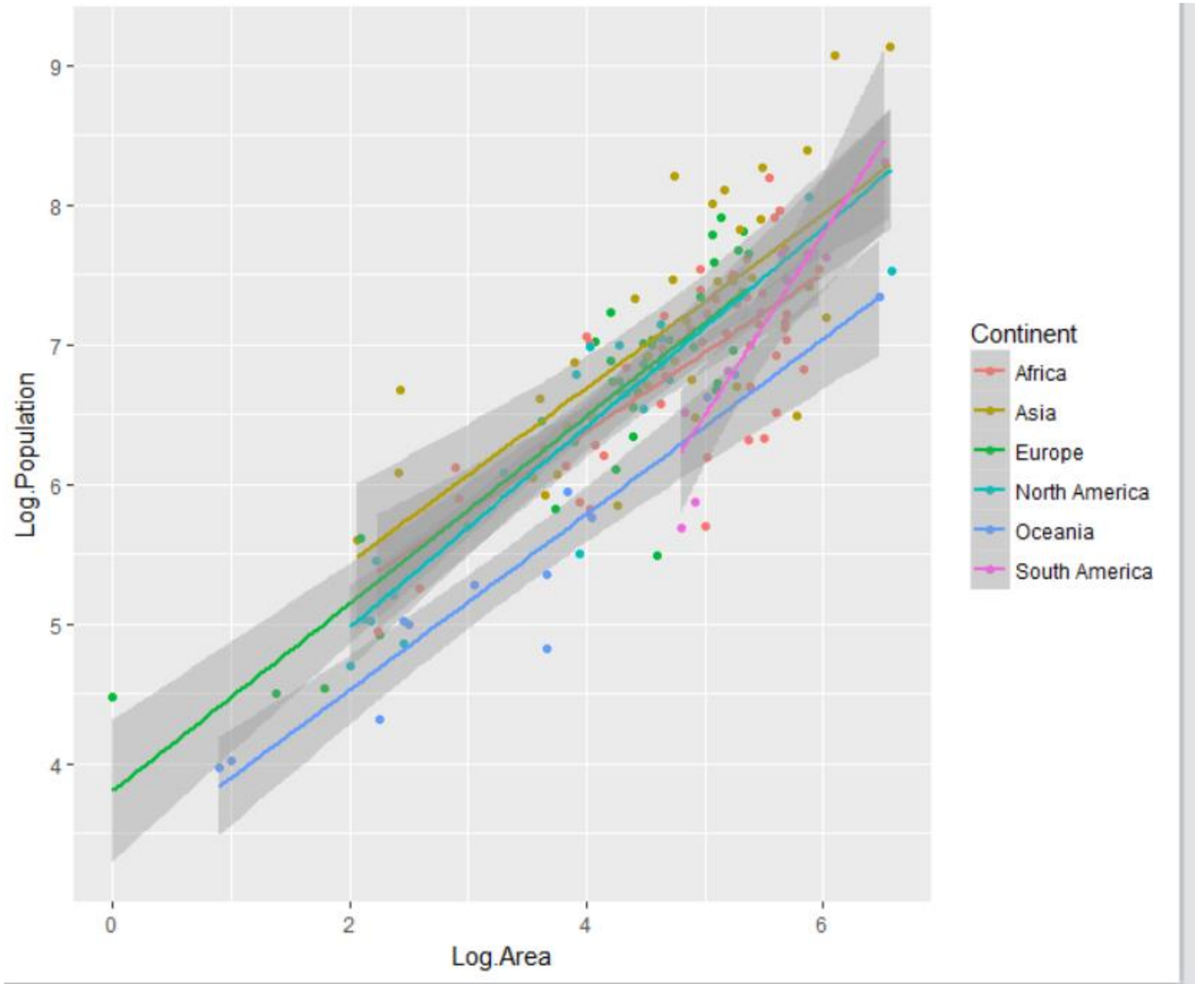
Colouring by factor

```
ggplot(df,aes(x=Log.Area,y=Log.Population, colour=Continent)) + geom_point()
```



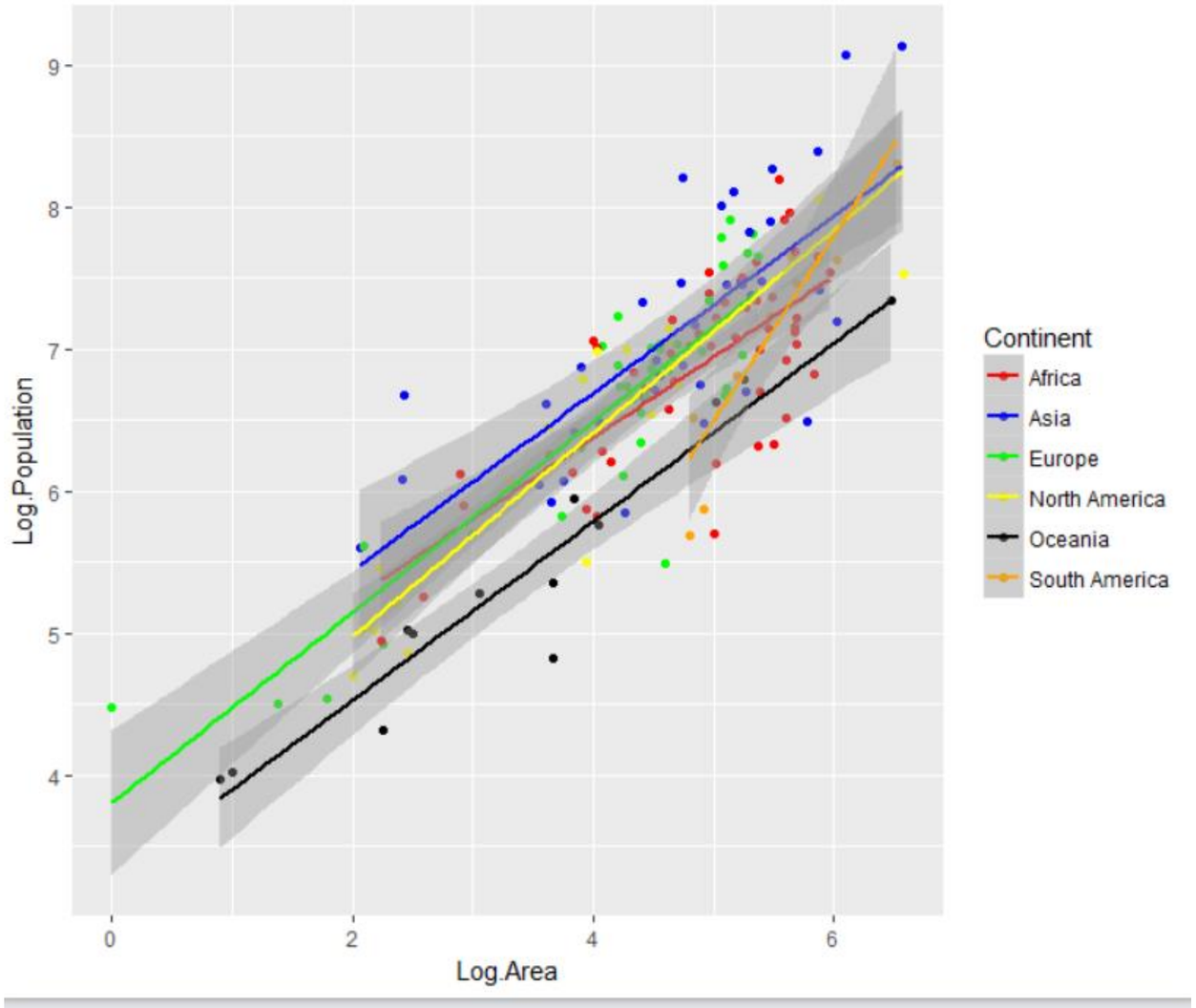
What can you tell?

```
ggplot(df,aes(x=Log.Area,y=Log.Population, colour=Continent)) + geom_point() + geom_smooth(method="lm")
```



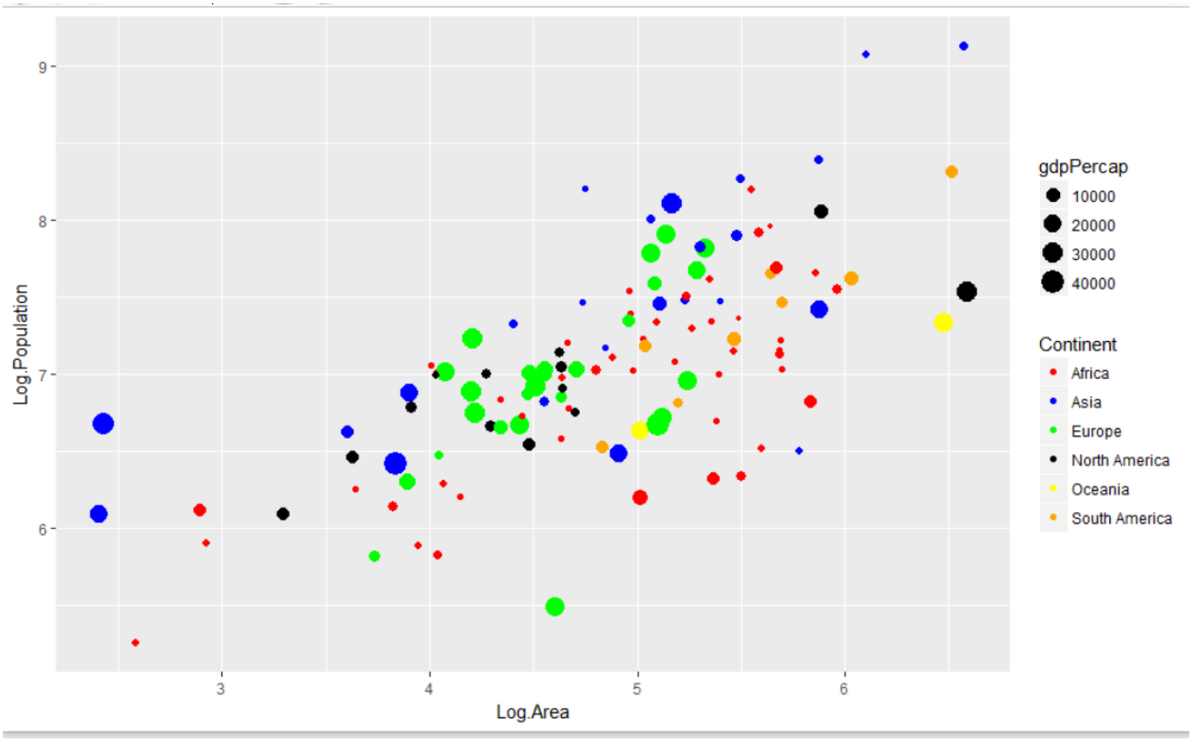
Set your own colours

```
ggplot(df,aes(x=Log.Area,y=Log.Population, colour=Continent)) + geom_point() + geom_smooth(method="lm")
+ scale_color_manual(values = c("red","blue","green","yellow","black","orange"))
```



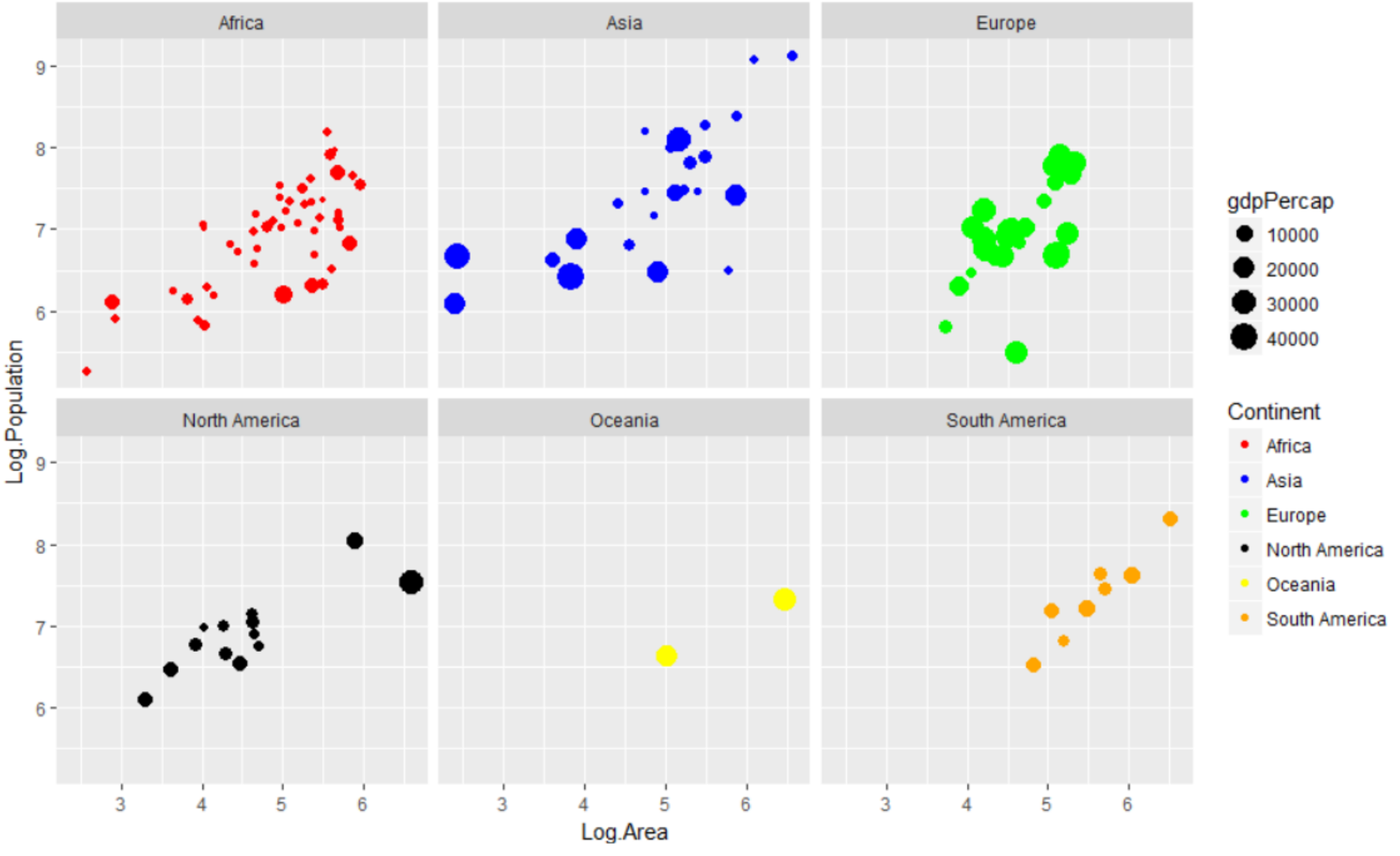
Sizing by factor

```
data<-read.csv("gapminder.csv") to country
data <- data[data$year=1997,]
df1 <- merge(df,data,by.x="Country",by.y="country")
ggplot(df1,aes(x=Log.Area,y=Log.Population, colour=Continent, cex=gdpPercap)) + geom_point()
+ scale_color_manual(values = c("red","blue","green","black","yellow","orange"))
```



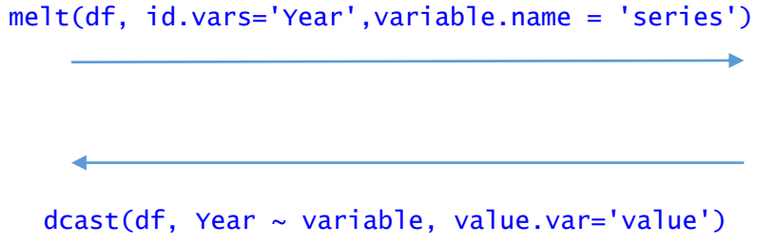
Breaking graph by factor

```
ggplot(df1,aes(x=Log.Area,y=Log.Population, colour=Continent, cex=gdpPercap)) + geom_point()
+ scale_color_manual(values = c("red","blue","green","black","yellow","orange")) + facet_wrap(~Continent)
```



Reshape2 package

	Year	Male	Female
1	1990	3236.524	3256.140
2	1991	2714.489	3012.146
3	1992	2997.675	3232.426
4	1993	2267.493	3066.795
5	1994	2889.264	2733.549
6	1995	3355.927	3205.007
7	1996	2855.110	2927.405
8	1997	3218.254	2358.021
9	1998	2515.704	2699.584
10	1999	2937.085	3049.776



	Year	variable	value
1	1990	Male	3236.524
2	1991	Male	2714.489
3	1992	Male	2997.675
4	1993	Male	2267.493
5	1994	Male	2889.264
6	1995	Male	3355.927
7	1996	Male	2855.110
8	1997	Male	3218.254
9	1998	Male	2515.704
10	1999	Male	2937.085
11	1990	Female	3256.140
12	1991	Female	3012.146
13	1992	Female	3232.426
14	1993	Female	3066.795
15	1994	Female	2733.549
16	1995	Female	3205.007
17	1996	Female	2927.405
18	1997	Female	2358.021
19	1998	Female	2699.584
20	1999	Female	3049.776

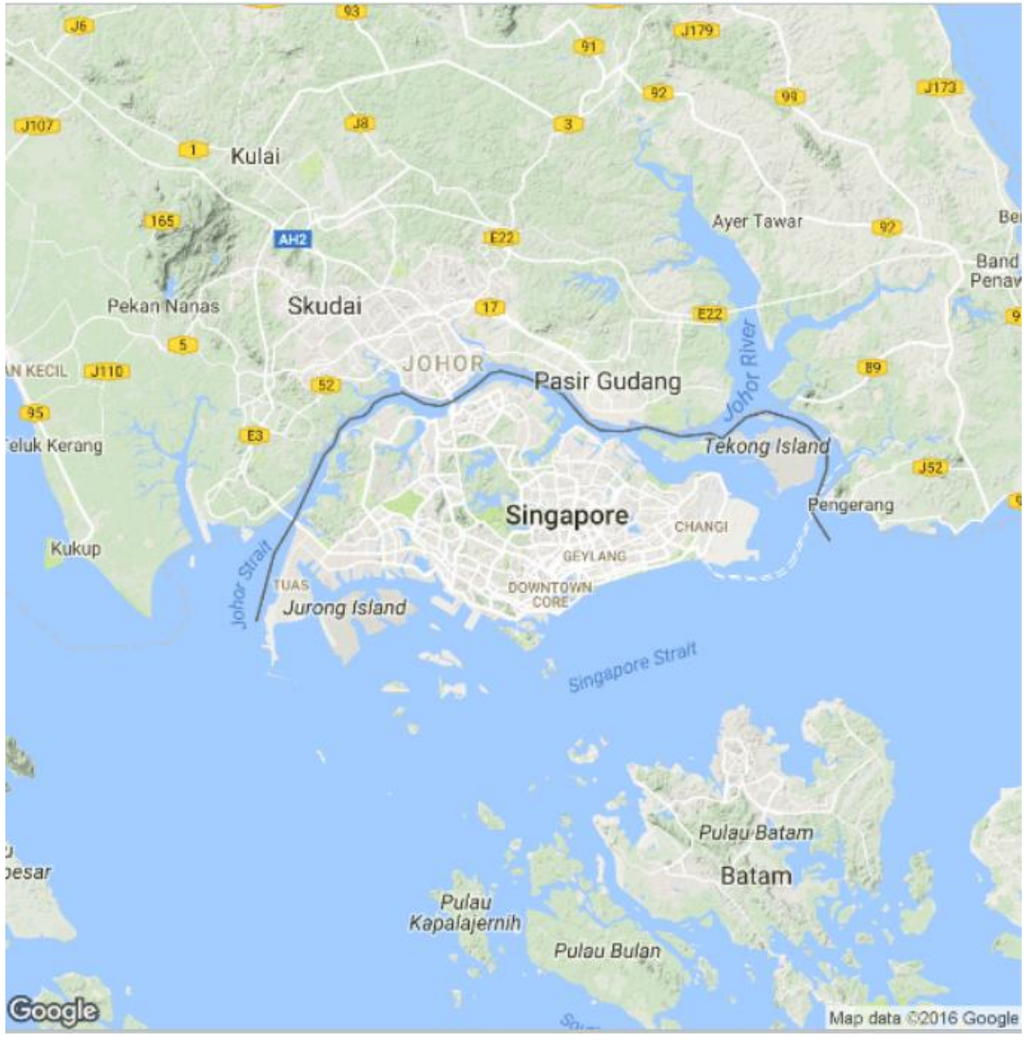
Visualizing Spatial Data

with ggmap

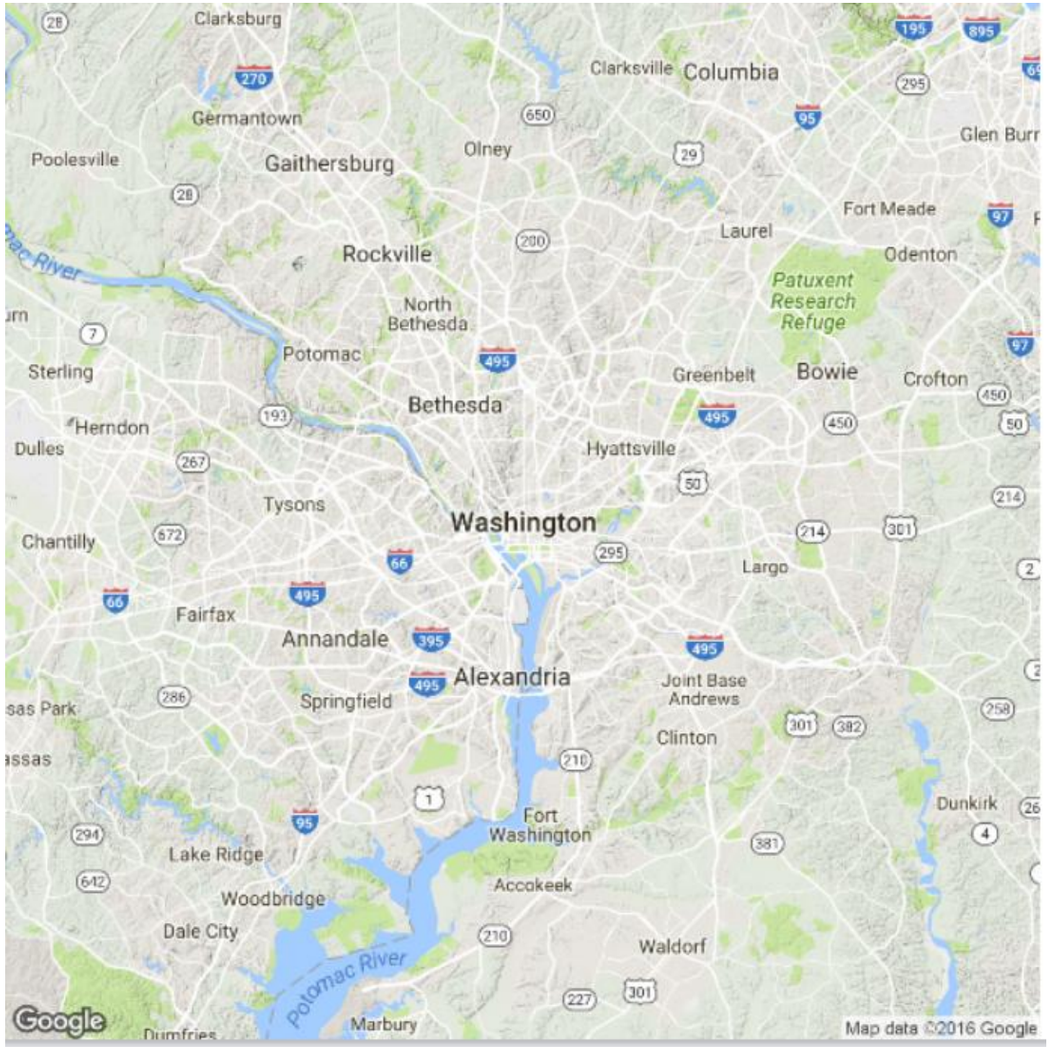
Introduction to ggmap

- “ggmap” is a package developed on top of *ggplot2* for visualizing spatial data.
- ggmap plots have all elements of ggplot2, but certain elements are fixed to map components.
  - The x aesthetic is fixed to longitude, the y aesthetic is fixed to latitude, and the coordinate system is fixed to the Mercator projection
- Other ggplot2 plots can overlay with ggmap plot.

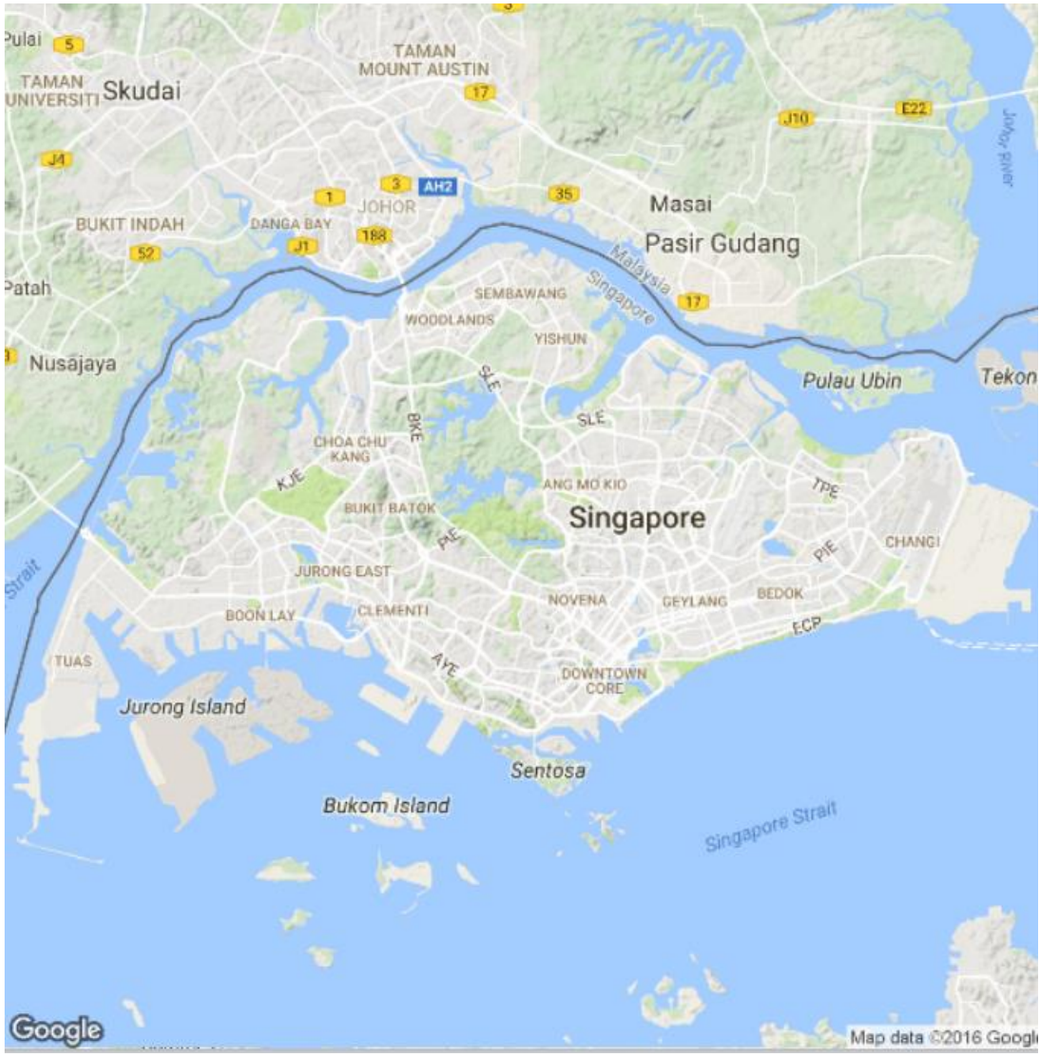
```
qmap(location="Singapore")
```




```
qmap(location="Washington")
```

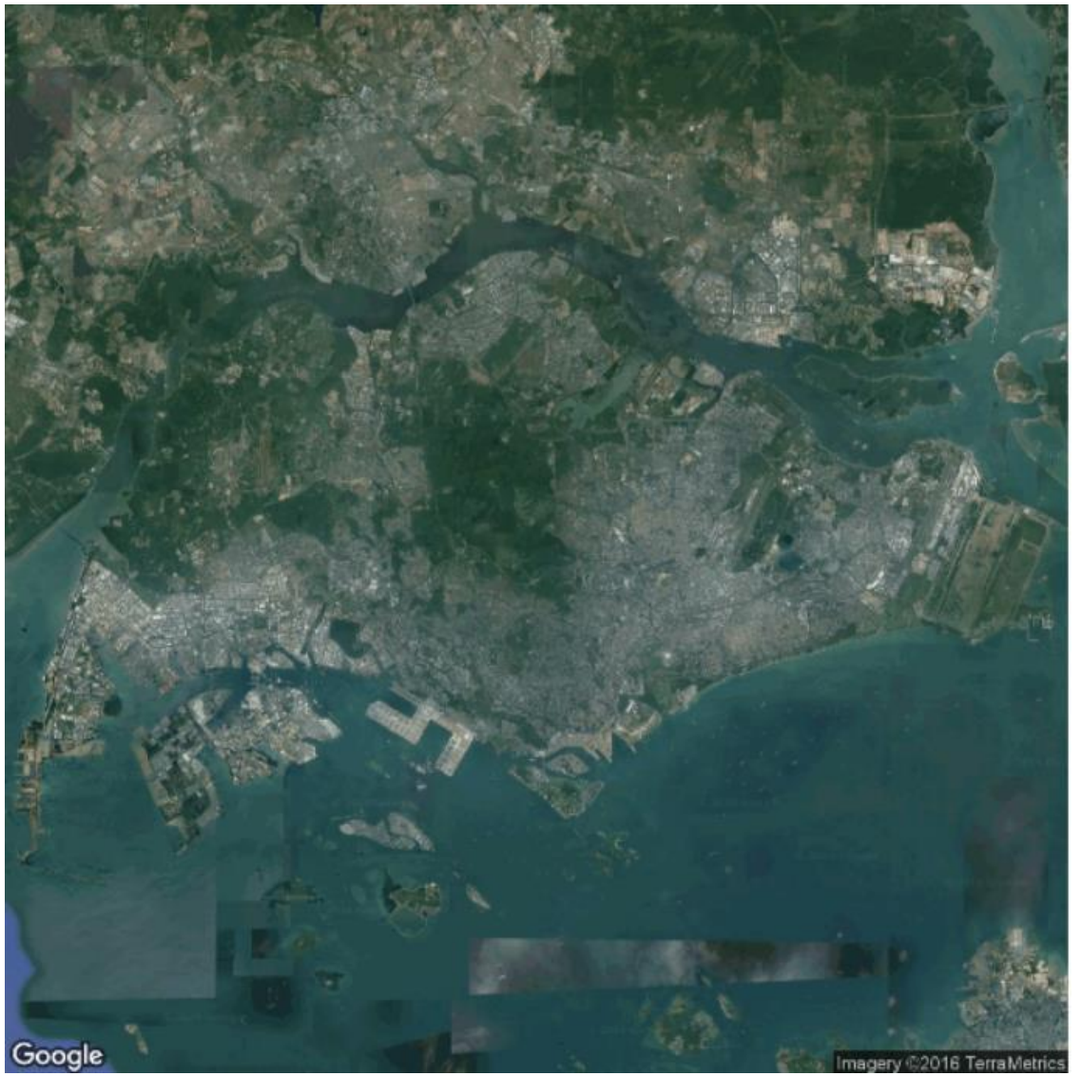


qmap(location="Singapore", zoom=11)



maptype

- satellite
- hybrid
- toner
- watercolor
- terrain-background
- toner-lite

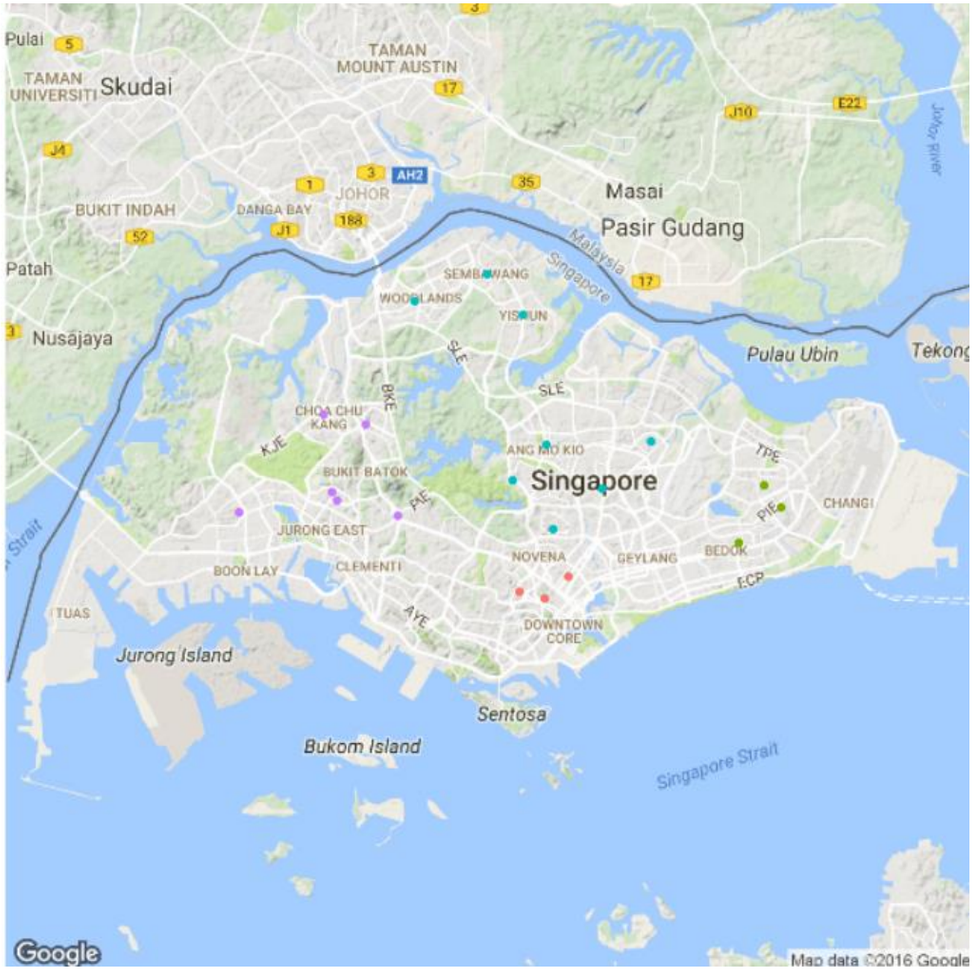


Now let’s make it interesting …

pizzahut.location <- read.csv("PizzaHut.csv",header = TRUE, colClasses = c("character","character","factor","character","numeric","numeric"))

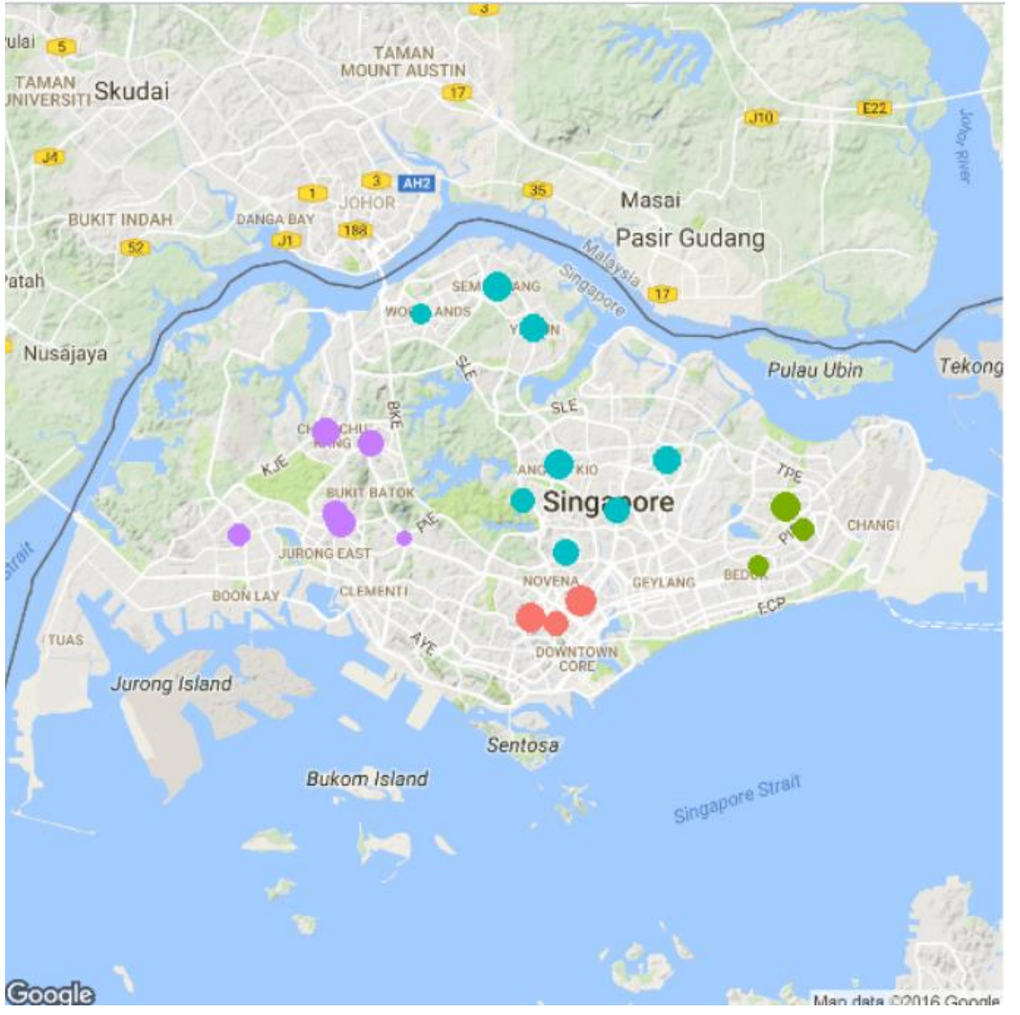
```
m1 <- qmap("Singapore", base_layer=ggplot(aes(x=lon, y = lat), data=pizzahut.location), zoom=11, scale=2)

m2<-m1 + geom_point(aes(color=Region))
```



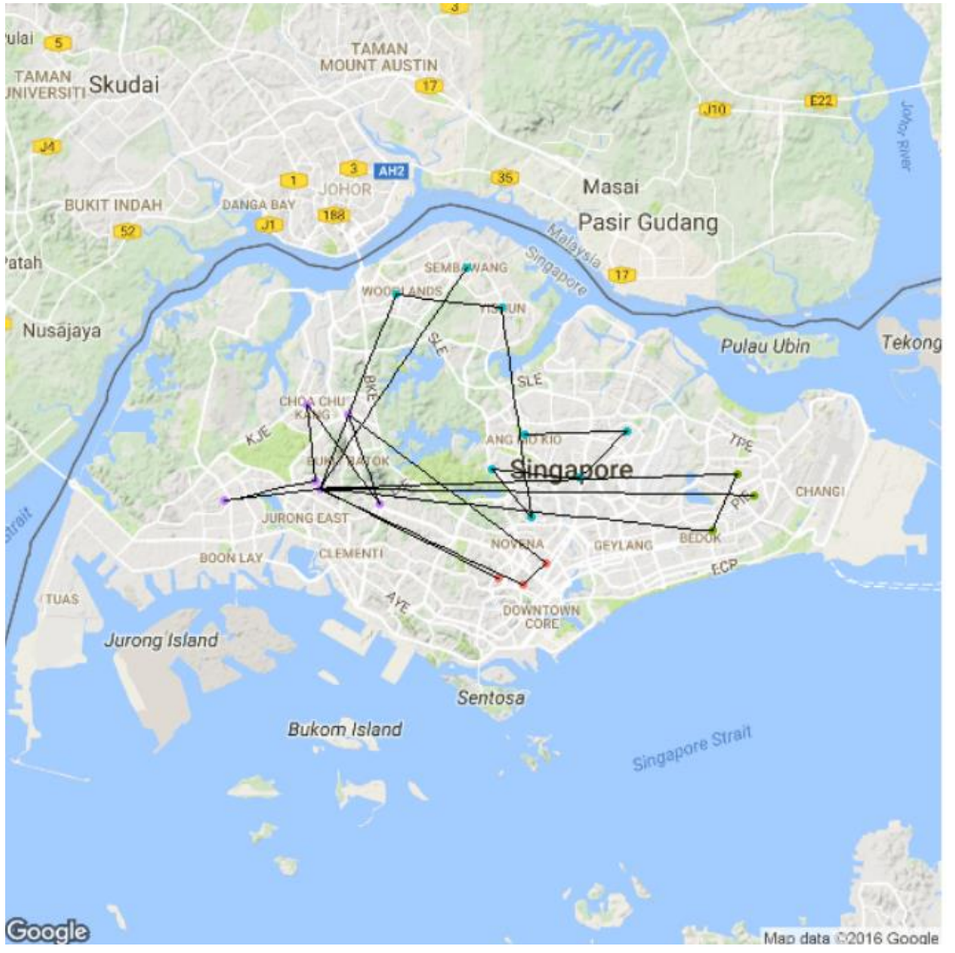
It is just ggplot2 …

```
pizzahut.location$Visits = round(rnorm(nrow(pizzahut.location),15000,5000))
m1 <- qmap("Singapore", base_layer=ggplot(aes(x=lon, y = lat), data=pizzahut.location), zoom=11, scale=2)
m3 <- m1 + geom_point(aes(color=Region, size=Visits))
```



More layer

```
m4 <- m1 + geom_point(aes(color=Region)) + geom_path()
```



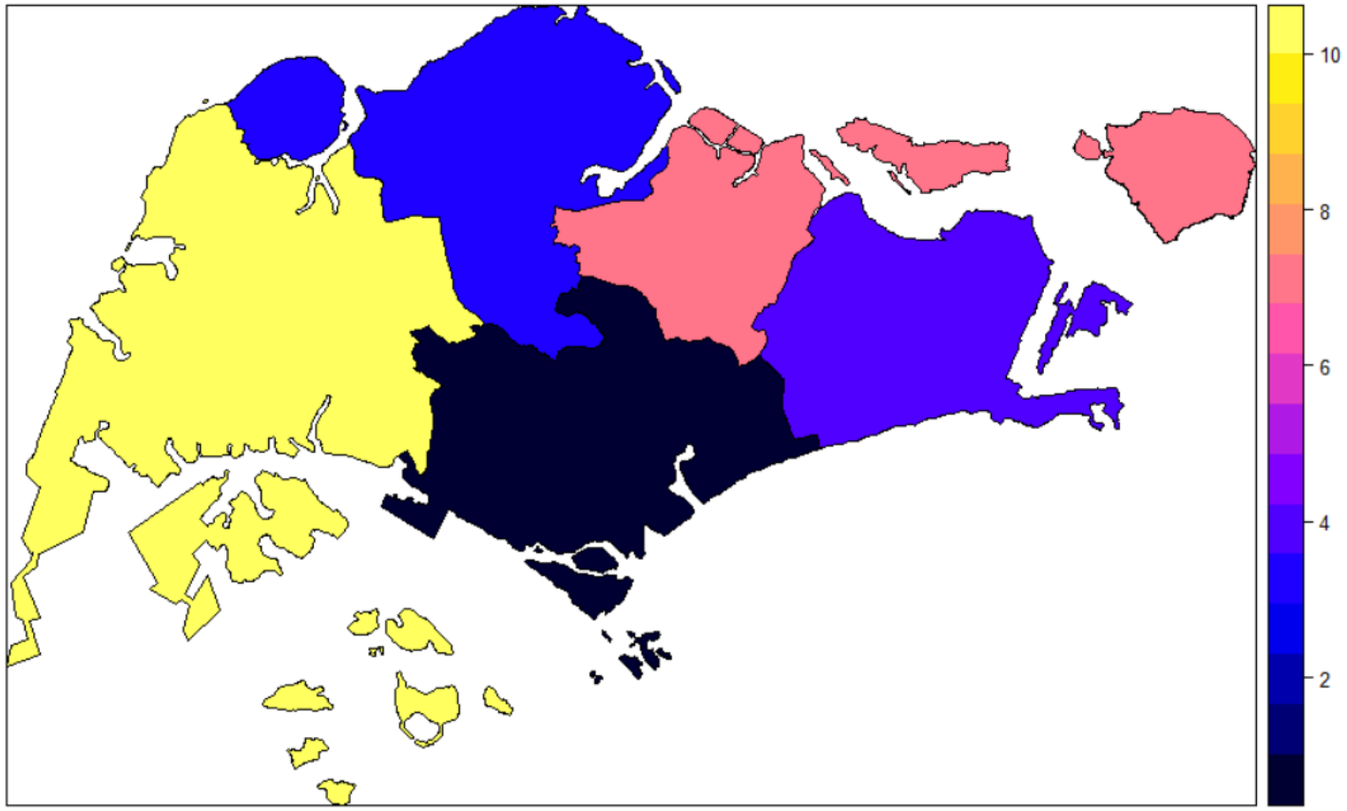
Map view by borders

```
install.packages("raster")

install.packages("rgdal")

library(raster)
library(rgdal)
library(XML)


SG<-getData('GADM', country='SG', level=1)
SG$value<-c(1,4,7,3,10)
spplot(SG,"value")
```



Interactive Plot with plotly

```
install.packages("plotly")

library(plotly)

set.seed(100)
d <- diamonds[sample(nrow(diamonds), 1000), ]
```

```
p <- ggplot(data = d, aes(x = carat, y = price)) +
  geom_point(aes(text = paste("Clarity:", clarity)), size = 1) +
  geom_smooth(aes(colour = cut, fill = cut)) + facet_wrap(~ cut)

(gg <- ggplotly(p))
```