IE6600 Computation and Visualization for Analytics

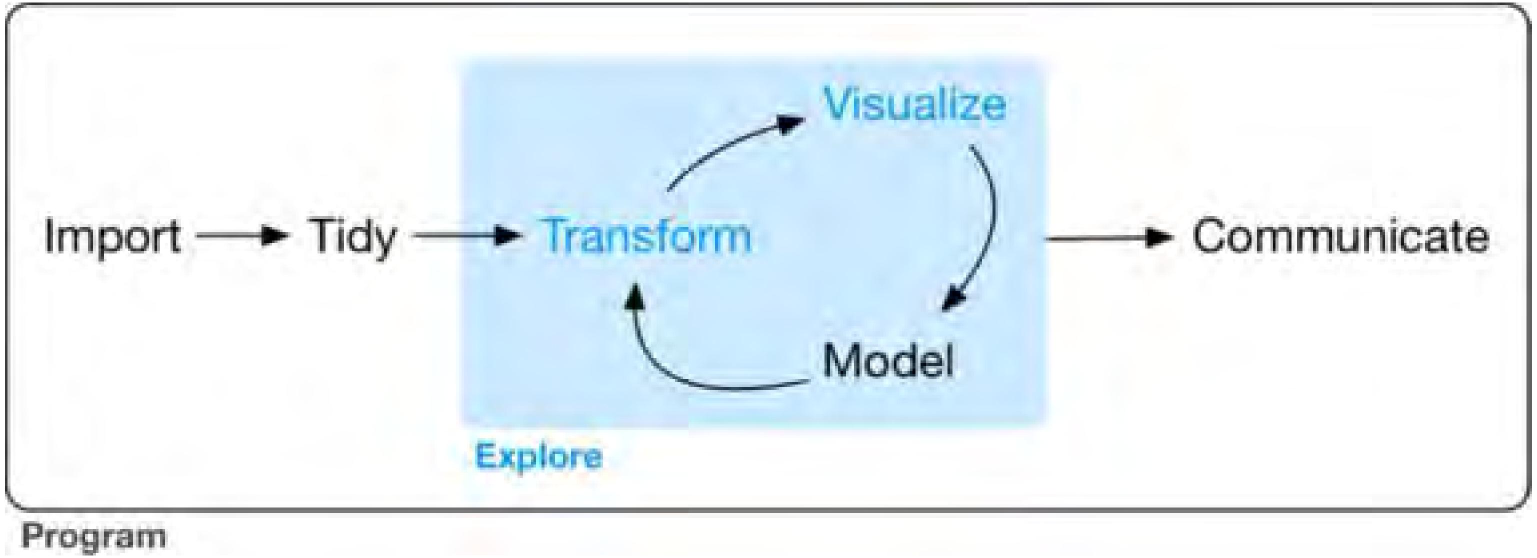
Application of Data Visualization

Zhenyuan Lu

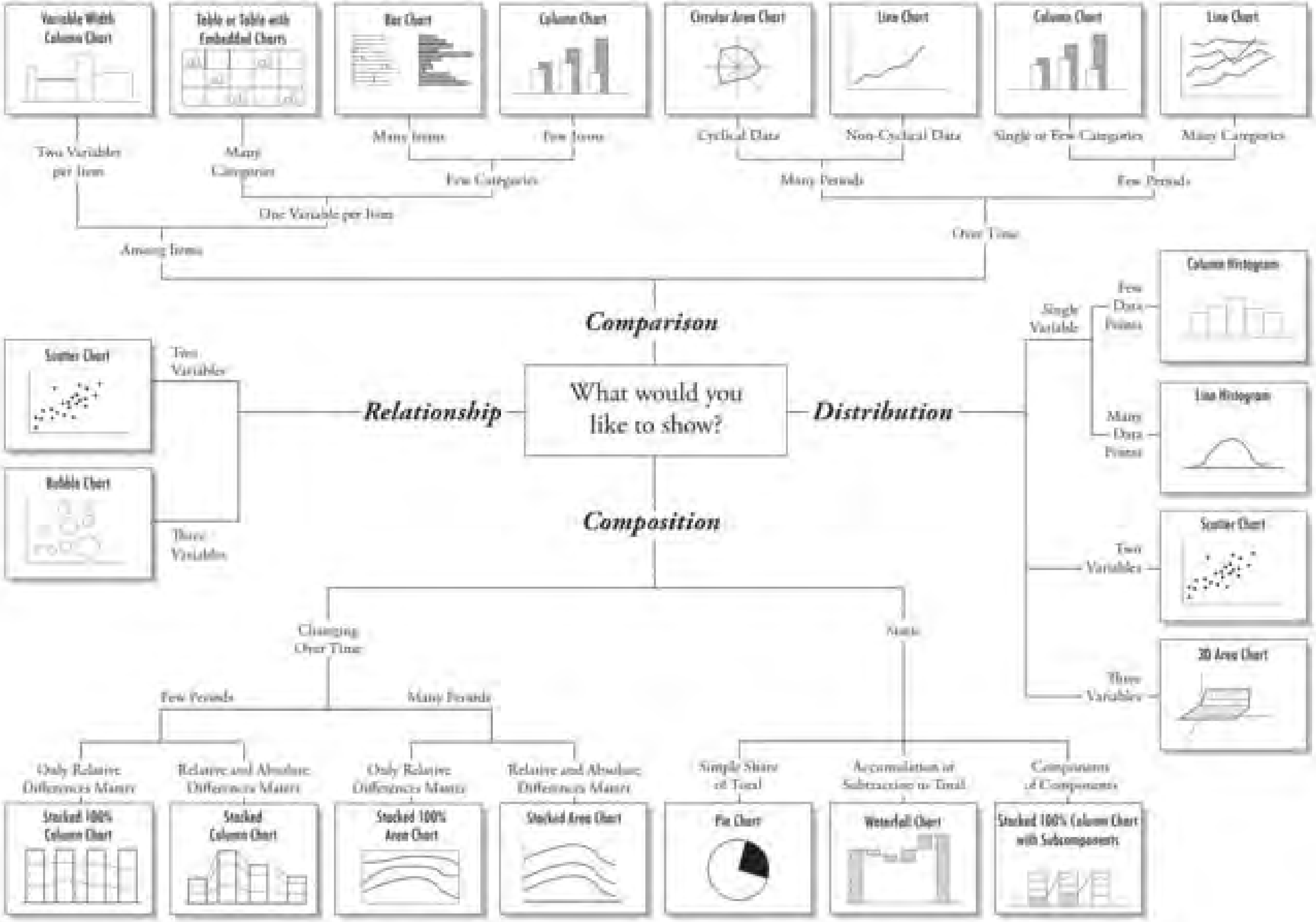
updated: 2022-07-11

Applicatin of Data Visualization

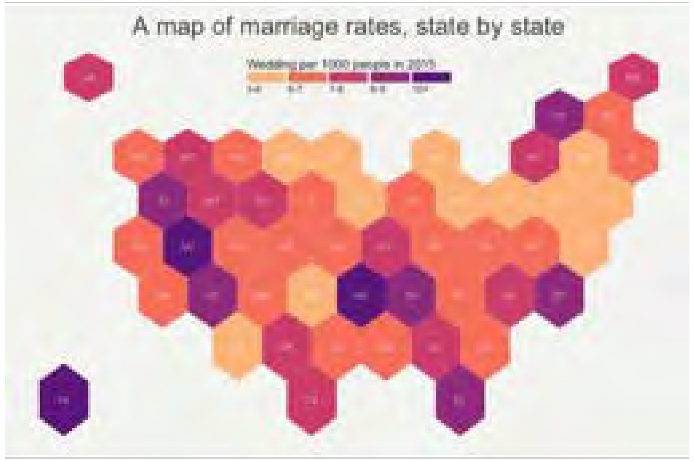
Goal



Wickham, Hadley, and Garrett Grolemund. R For Data Science. OReilly, 2017.


What would you like to show?
Comparison
Among Items
Two Variables per Item
Variable Width Column Chart
One Variable per Item
Many Categories
Table or Table with Embedded Charts
Few Categories
Many Items
Bar Chart
Few Items
Column Chart
Over Time
Many Periods
Cyclical Data
Circular Area Chart
Non-Cyclical Data
Line Chart
Few Periods
Single or Few Categories
Column Chart
Many Categories
Line Chart
Relationship
Two Variables
Scatter Chart
Three Variables
Bubble Chart
Distribution
Single Variable
Few Data Points
Column Histogram
Many Data Points
Line Histogram
Two Variables
Scatter Chart
Three Variables
3D Area Chart
Composition
Changing Over Time
Few Periods
Only Relative Differences Matter
Stacked 100% Column Chart
Relative and Absolute Differences Matter
Stacked Column Chart
Many Periods
Only Relative Differences Matter
Stacked 100% Area Chart
Relative and Absolute Differences Matter
Stacked Area Chart
Static
Simple Share of Total
Pie Chart
Accumulation or Subtraction to Total
Waterfall Chart
Components of Components
Stacked 100% Column Chart with Subcomponents

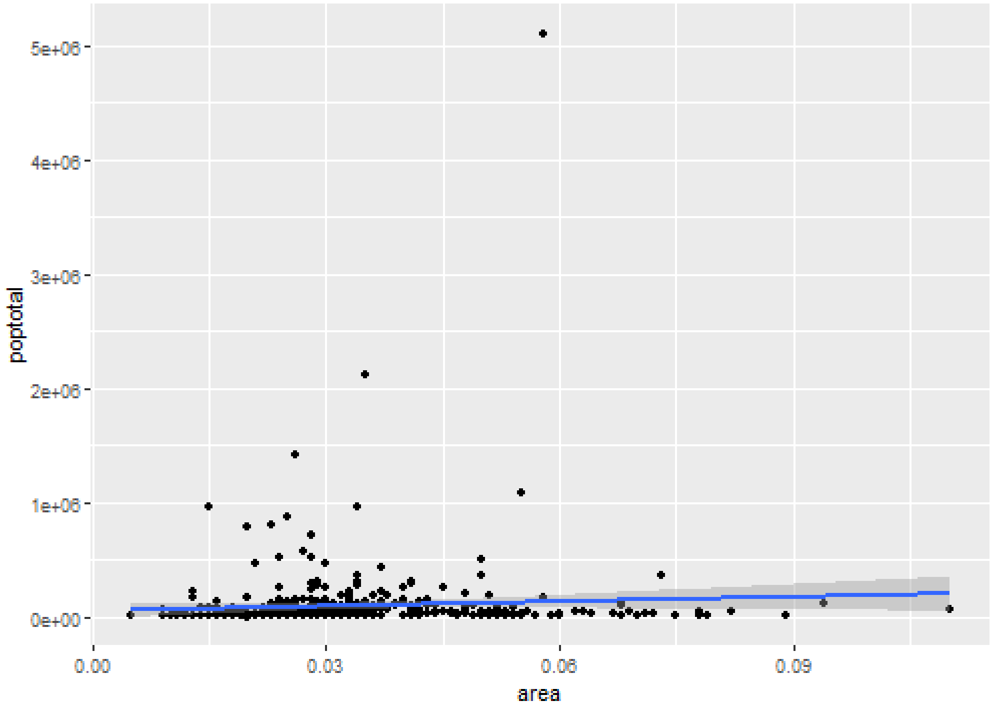
All the figures above are generated by ggplot2! Yes, included the middle one!


A map of marriage rates, state by state
Wedding per 1000 people in 2015

© 2022 Zhenyuan LU

ggplot2 - simple plot

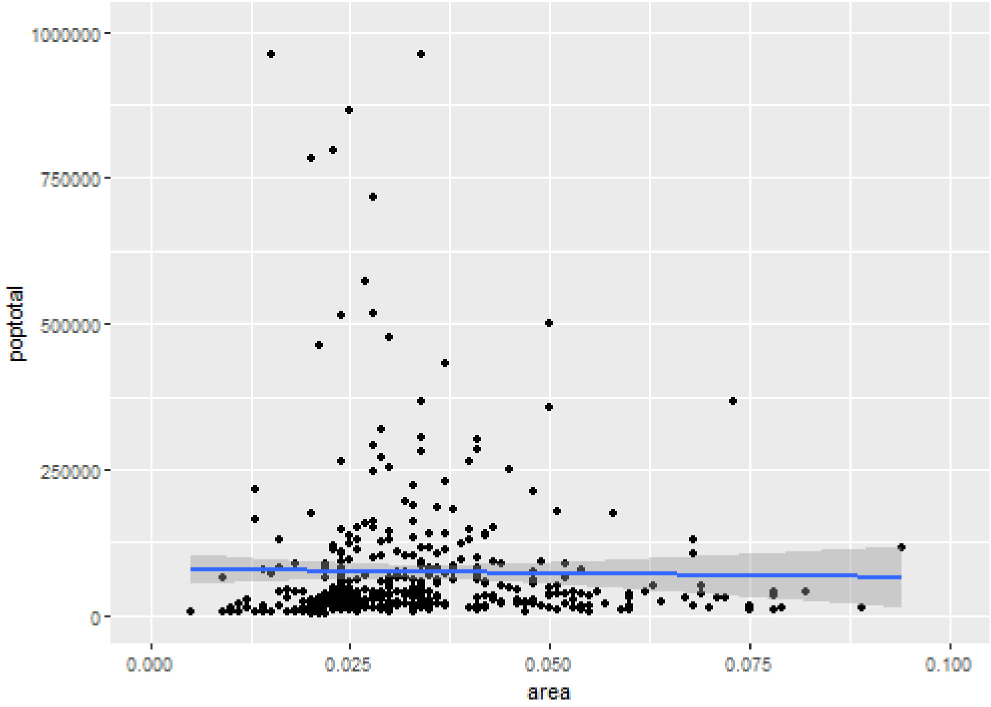
```
library(tidyverse)
# area and poptotal are columns in 'midwest’
midwest %>%
  ggplot(aes(x=area, y=poptotal)) +
  geom_point() + geom_smooth(method="lm")
```



Adjustment of X And Y Axis Limits

`Method 1`: deleting the points outside the range

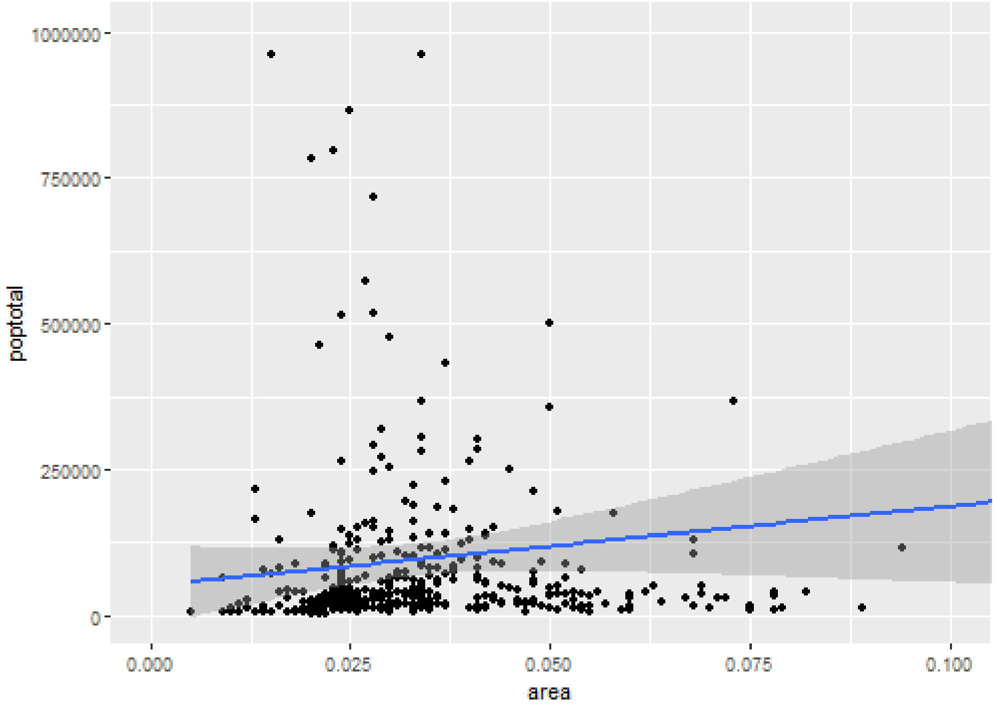
```
ggplot(midwest, aes(x = area, y = poptotal)) +
  geom_point() +
  geom_smooth(method = "lm") +
  xlim(c(0, 0.1)) + ylim(c(0, 1000000))
```



Adjustment of X And Y Axis Limits

`Method 2`: zooming in

```
ggplot(midwest, aes(x = area, y = poptotal)) +
  geom_point() +
  geom_smooth(method = "lm") +
  coord_cartesian(xlim = c(0, 0.1), ylim = c(0, 1000000))
```



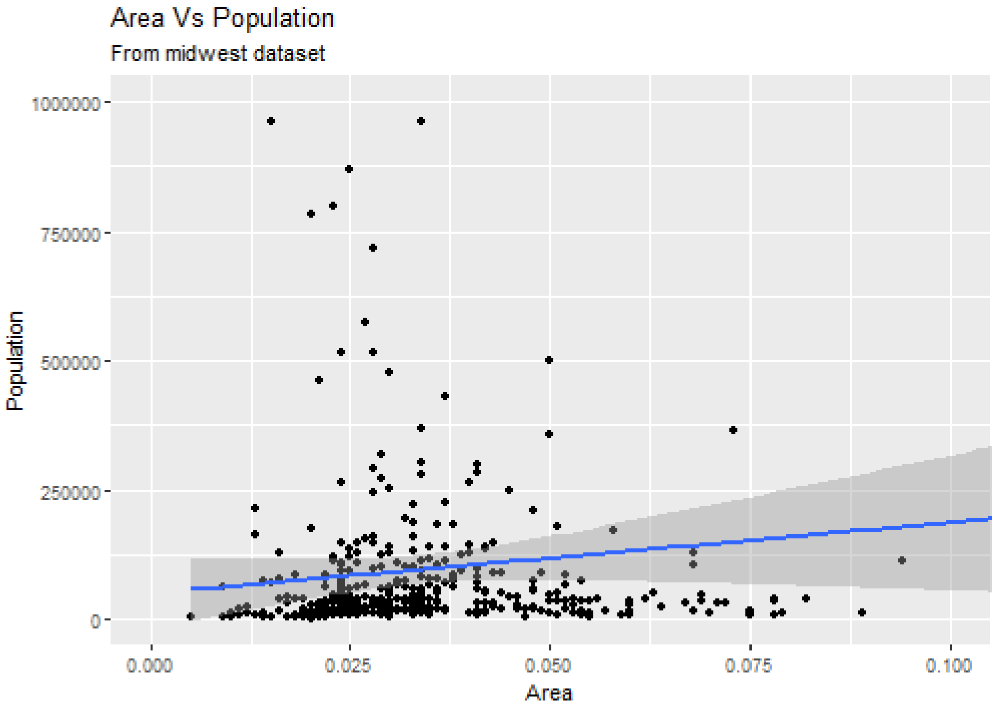
Titles and Axis Labels

Two different ways to change the titles and labels `Method 1`

```r
ggplot(midwest, aes(x = area, y = poptotal)) +
  geom_point() +
  geom_smooth(method = "lm") +
  coord_cartesian(xlim = c(0, 0.1), ylim = c(0, 1000000))+
  labs(title="Area Vs Population",
       subtitle="From midwest dataset",
       y="Population", x="Area", caption="Midwest Demographics")
```

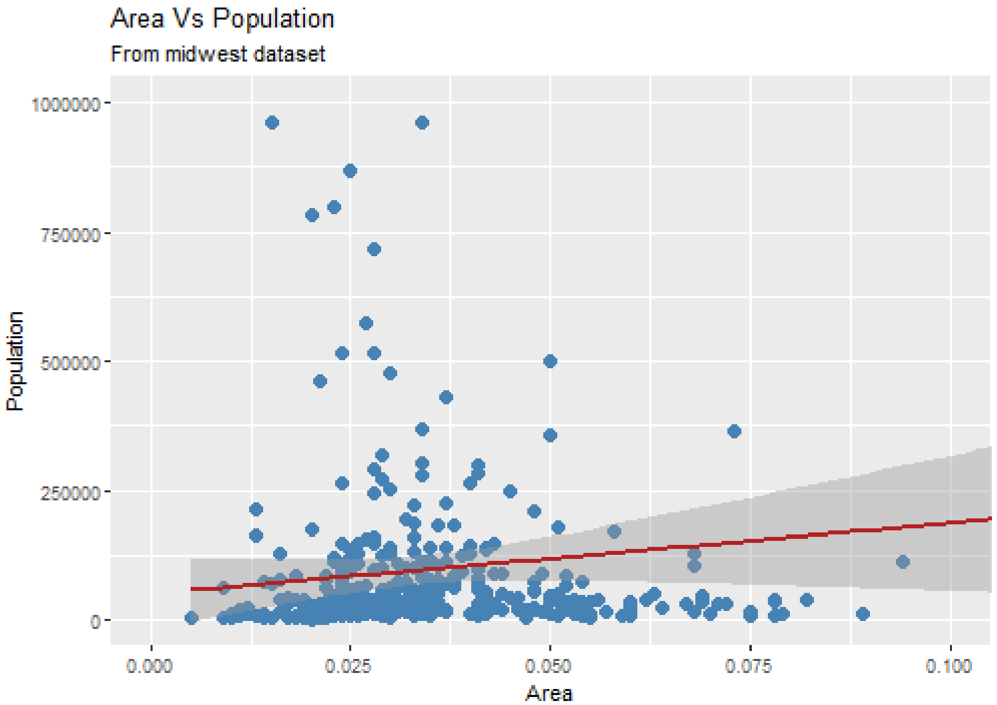
Method 2

```
ggplot(midwest, aes(x = area, y = poptotal)) +
  geom_point() +
  geom_smooth(method = "lm") +
  coord_cartesian(xlim = c(0, 0.1), ylim = c(0, 1000000))+
  ggtitle("Area Vs Population", subtitle = "From midwest dataset") +
  xlab("Area") + ylab("Population")
```



Color Changes

```
ggplot(midwest, aes(x = area, y = poptotal)) +
  geom_point(col="steelblue", size=3) +
  geom_smooth(method = "lm", col="firebrick") +
  coord_cartesian(xlim = c(0, 0.1), ylim = c(0, 1000000))+
  ggtitle("Area Vs Population", subtitle = "From midwest dataset") +
  xlab("Area") + ylab("Population")
```



Legend Removing

```
ggplot(midwest, aes(x = area, y = poptotal)) +
  geom_point(aes(color=state), size=3) +
  geom_smooth(method = "lm", col="firebrick") +
  coord_cartesian(xlim = c(0, 0.1), ylim = c(0, 1000000))+
  ggtitle("Area Vs Population", subtitle = "From midwest dataset") +
  xlab("Area") + ylab("Population") +
  theme(legend.position="None")
```

Color Palette Changing

```r
ggplot(midwest, aes(x = area, y = poptotal)) +
  geom_point(aes(color=state), size=3) +
  geom_smooth(method = "lm", col="firebrick") +
  coord_cartesian(xlim = c(0, 0.1), ylim = c(0, 1000000))+
  ggtitle("Area Vs Population", subtitle = "From midwest dataset") +
  xlab("Area") + ylab("Population") +
  theme(legend.position="None") +
  scale_colour_brewer(palette = "Set1")
```

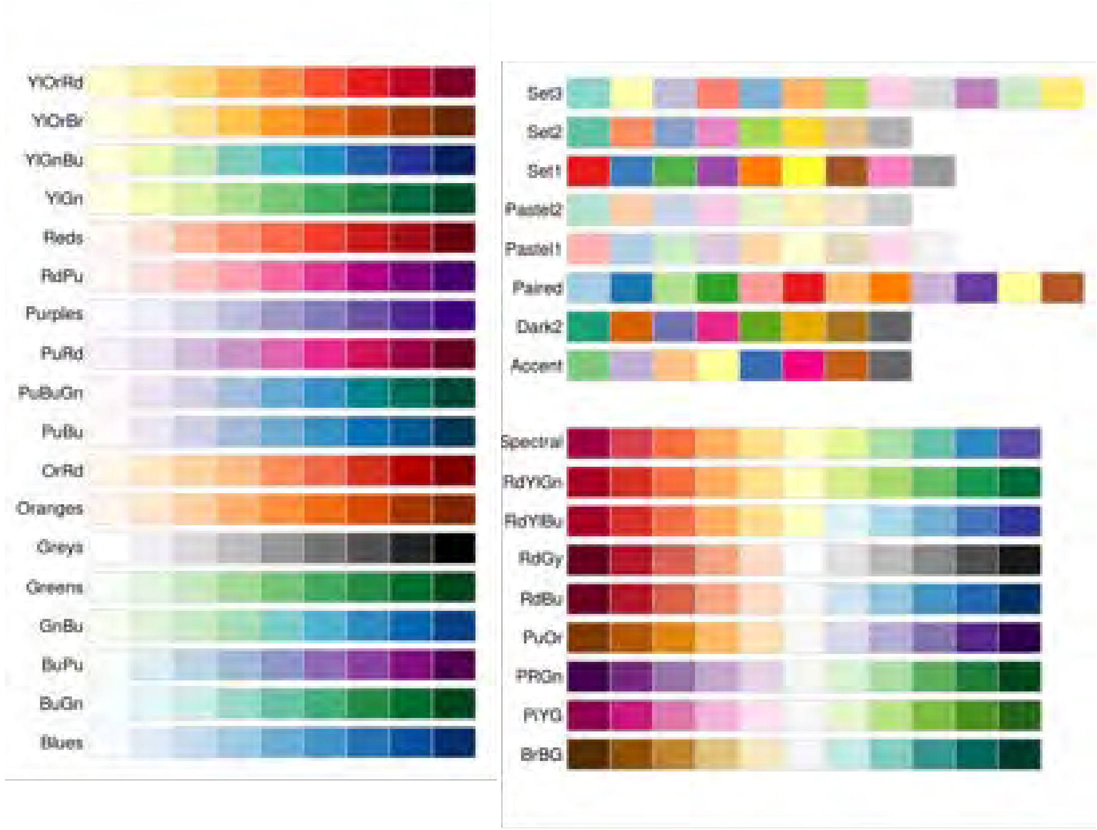
Show more color palette

```r
library(RColorBrewer)
head(brewer.pal.info, 10)
```

```
##          maxcolors category colorblind
## BrBG            11      div       TRUE
## PiYG            11      div       TRUE
## PRGn            11      div       TRUE
## PuOr            11      div       TRUE
## RdBu            11      div       TRUE
## RdGy            11      div      FALSE
## RdYlBu          11      div       TRUE
## RdYlGn          11      div      FALSE
## Spectral        11      div      FALSE
## Accent           8     qual      FALSE
```

```r
#Exp. scale_colour_brewer(palette = “BrBG")
```

More color palette



Change the Default Themes

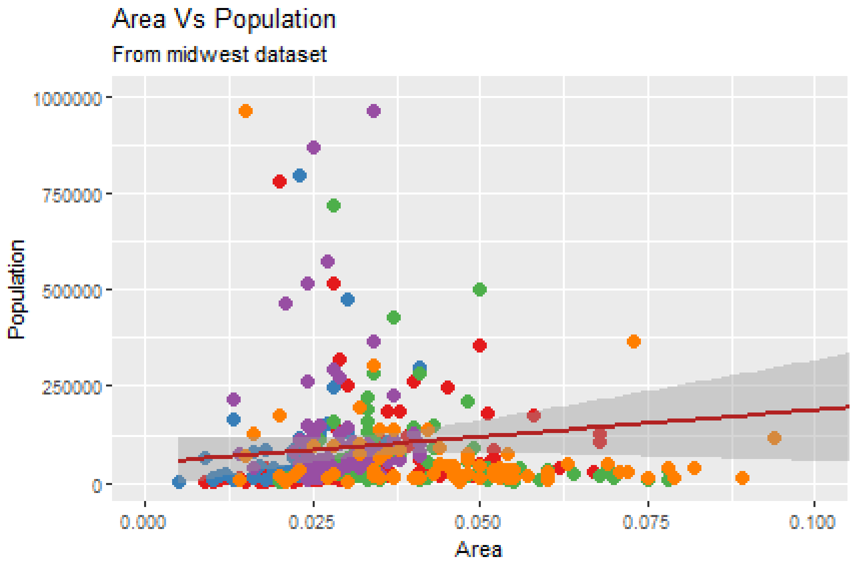
```r
ggplot(midwest, aes(x = area, y = poptotal)) +
  geom_point(aes(col=state), size=3) +
  geom_smooth(method = "lm", color="firebrick") +
  coord_cartesian(xlim = c(0, 0.1), ylim = c(0, 1000000))+
  ggtitle("Area Vs Population", subtitle = "From midwest dataset") +
  xlab("Area") + ylab("Population") +
  theme(legend.position="None") +
  scale_colour_brewer(palette = "Set1") +
  theme_bw() + labs(subtitle="BW Theme")
```

Change the Default Themes (cont'd)

```r
ggplot(midwest, aes(x = area, y = poptotal)) +
  geom_point(aes(col=state), size=3) +
  geom_smooth(method = "lm", color="firebrick") +
  coord_cartesian(xlim = c(0, 0.1), ylim = c(0, 1000000))+
  ggtitle("Area Vs Population", subtitle = "From midwest dataset") +
  xlab("Area") + ylab("Population") +
  theme(legend.position="None") +
  scale_colour_brewer(palette = "Set1") +
  theme_classic() + labs(subtitle="Classic Theme")
```

Default plot assigned

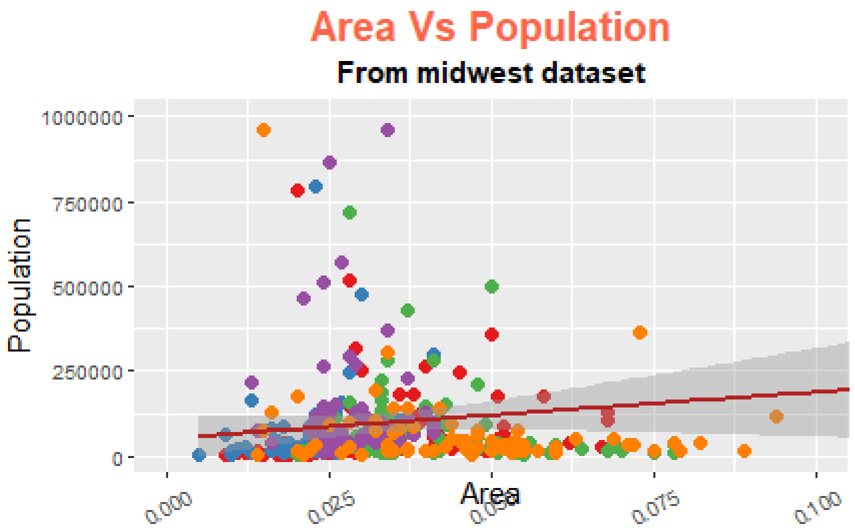
```r
gg <- ggplot(midwest, aes(x = area, y = poptotal)) +
  geom_point(aes(col = state), size = 3) +
  geom_smooth(method = "lm", color = "firebrick") +
  coord_cartesian(xlim = c(0, 0.1), ylim = c(0, 1000000)) +
  ggtitle("Area Vs Population", subtitle = "From midwest dataset") +
  xlab("Area") + ylab("Population") +
  theme(legend.position = "None") +
  scale_colour_brewer(palette = "Set1")
plot(gg)
```



Customizing Titles

```
#Customizing Titles
# title
gg + theme(plot.title=element_text(size=20, face="bold", family="American Typewriter", color="tomato", hjust=
# subtitle
plot.subtitle=element_text(size=15, family="American Typewriter", face="bold", hjust=0.5),
# caption
plot.caption=element_text(size=15),
# X axis title
axis.title.x=element_text(vjust=10, size=15),
# Y axis title
axis.title.y=element_text(size=15),
# X axis text
axis.text.x=element_text(size=10, angle = 30, vjust=.5),
# Y axis text
axis.text.y=element_text(size=10))
```

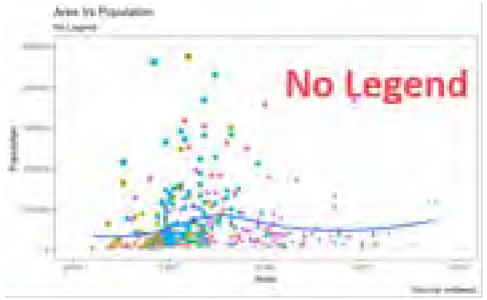
Customizing Titles (cont'd)



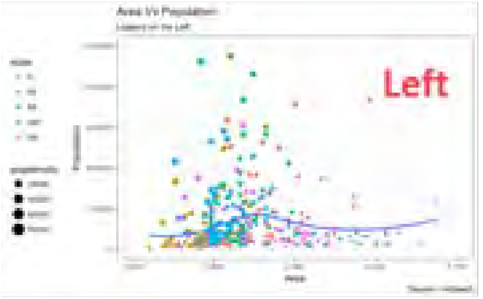
Legend Position

```
# No legend
gg + theme(legend.position="None") + labs(subtitle="No Legend")

# Legend to the left
#labs(subtitle="Legend on the Left")

# legend at the bottom and horizontal
#labs(subtitle="Legend at Bottom")

# legend at bottom-right, inside the plot
#labs(subtitle="Legend: Bottom-Right Inside the Plot")

# legend at top-left, inside the plot
#labs(subtitle="Legend: Top-Left Inside the Plot")
```

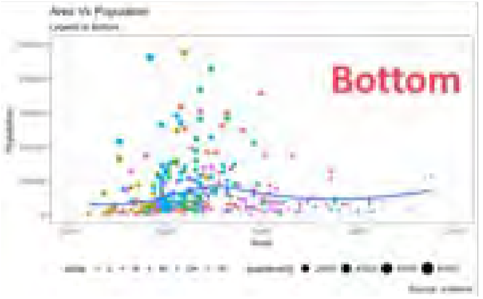
Legend Position (cont'd)



No Legend

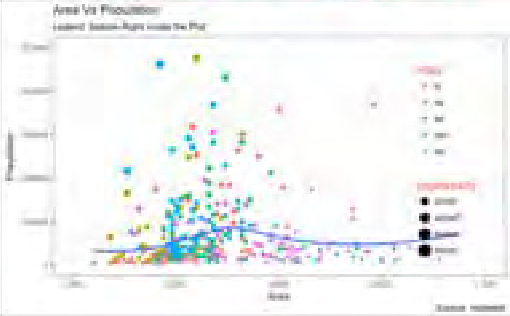


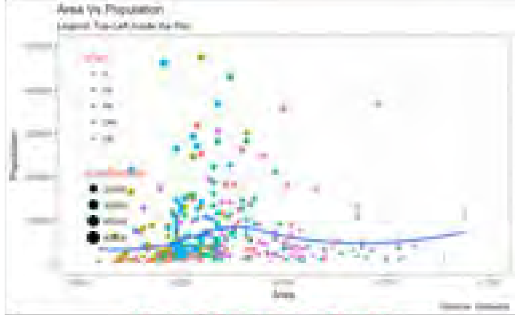
Left



Bottom

Inside, bottom right





Inside, top left

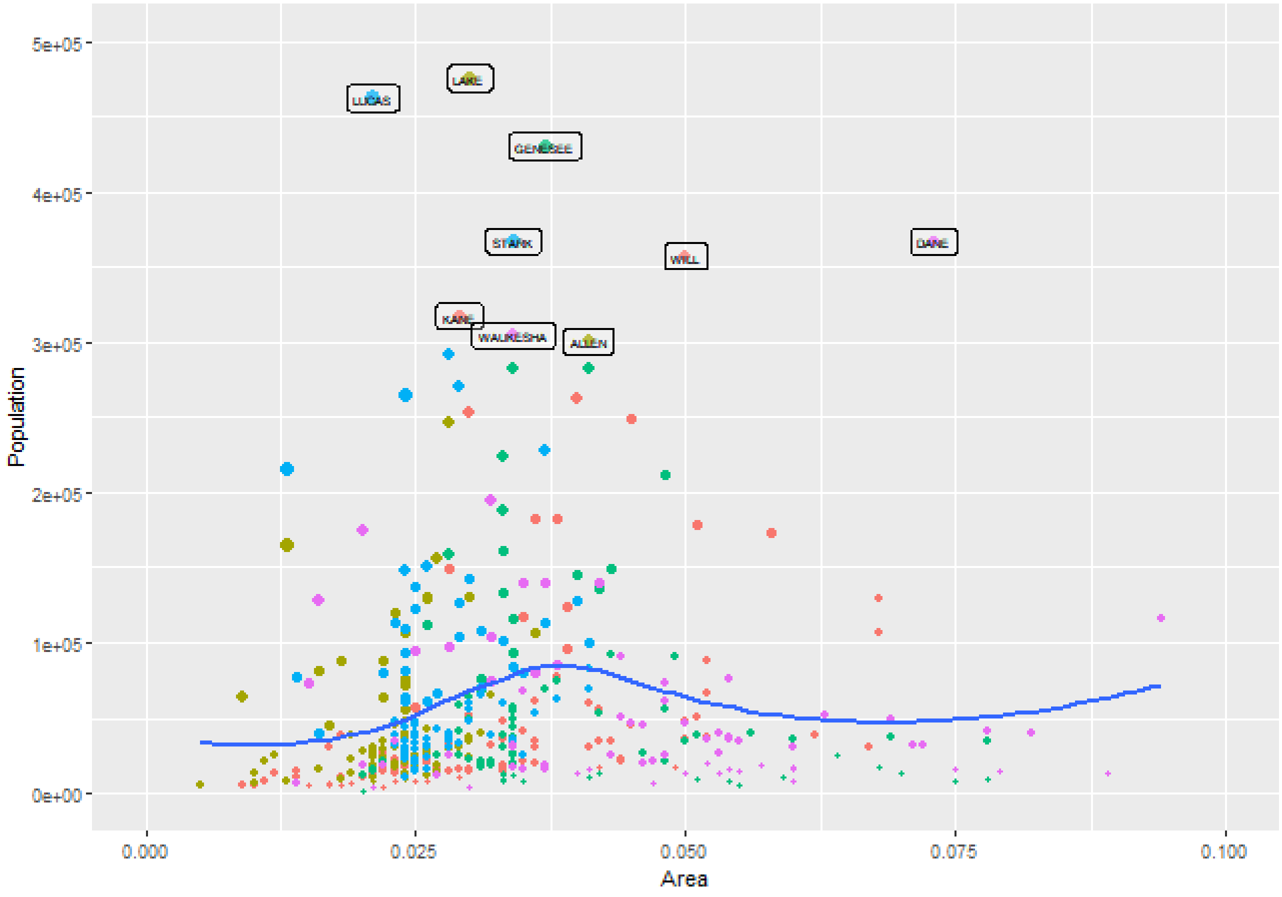
Label, Text, and Annotation

```r
# Filter required rows.
midwest_sub <- midwest %>% filter(poptotal > 300000) %>%
  mutate(large_county = ifelse(poptotal > 300000, county, ""))
# Base Plot
ggplot(midwest, aes(x = area, y = poptotal)) +
  geom_point(aes(col = state, size = popdensity)) +
  geom_smooth(method = "loess", se = F) + xlim(c(0, 0.1)) + ylim(c(0, 500000)) +
  labs( title = "Area Vs Population",
    y = "Population",
    x = "Area",
    caption = "Source: midwest"
  ) +
  # Plot text
  geom_text(data = midwest_sub, aes(label = large_county), size = 2) +
  labs(subtitle = "With ggplot2::geom_text") + theme(legend.position = "None") +
  # label
  geom_label(
    data = midwest_sub,
    aes(label = large_county),
    size = 2, alpha = 0.25
  ) + labs(subtitle = "With ggplot2::geom_label") + theme(legend.position = "None")
```

Label, Text, and Annotation

Area Vs Population

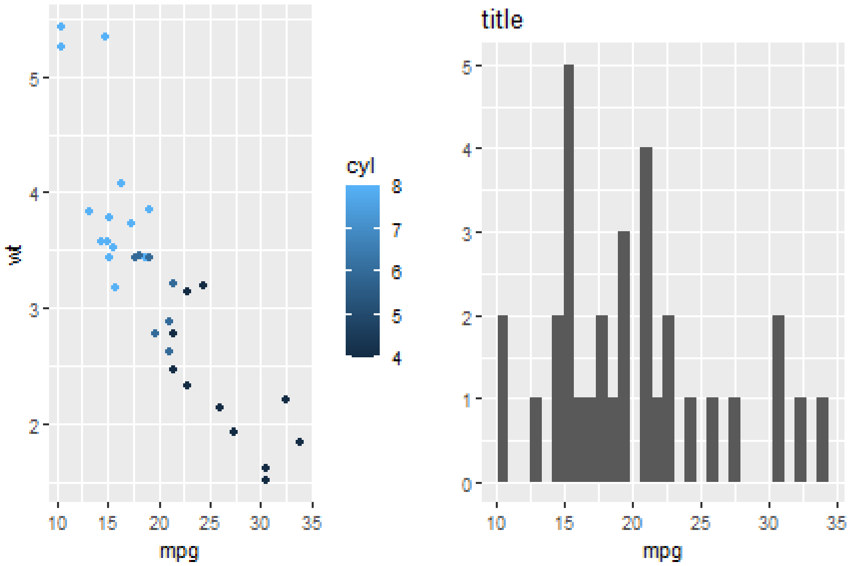
With ggplot2::geom_label



Source: midwest

Multiple plots

```
library(gridExtra)
p1 <- qplot(mpg, wt, data = mtcars, colour = cyl)
p2 <- qplot(mpg, data = mtcars) + ggtitle("title")
grid.arrange(p1, p2, nrow = 1)
```



Charts

1.Correlation

1. Scatterplot
2. Scatterplot With Encircling
3. Jitter Plot
4. Counts Chart
5. Bubble Plot
6. Correlogram

2.Deviation

1. Diverging Bars
2. Diverging Lollipop Chart
3. Diverging Dot Plot
4. Area Chart

3.Ranking

1. Ordered Bar Chart
2. Lollipop Chart
3. Dot Plot
4. Slope Chart
5. Dumbbell Plot

4.Distribution

1. Histogram
2. Density Plot
3. Box Plot
4. Dot + Box Plot
5. Tufte Boxplot
6. Violin Plot
7. Population Pyramid

5.Change

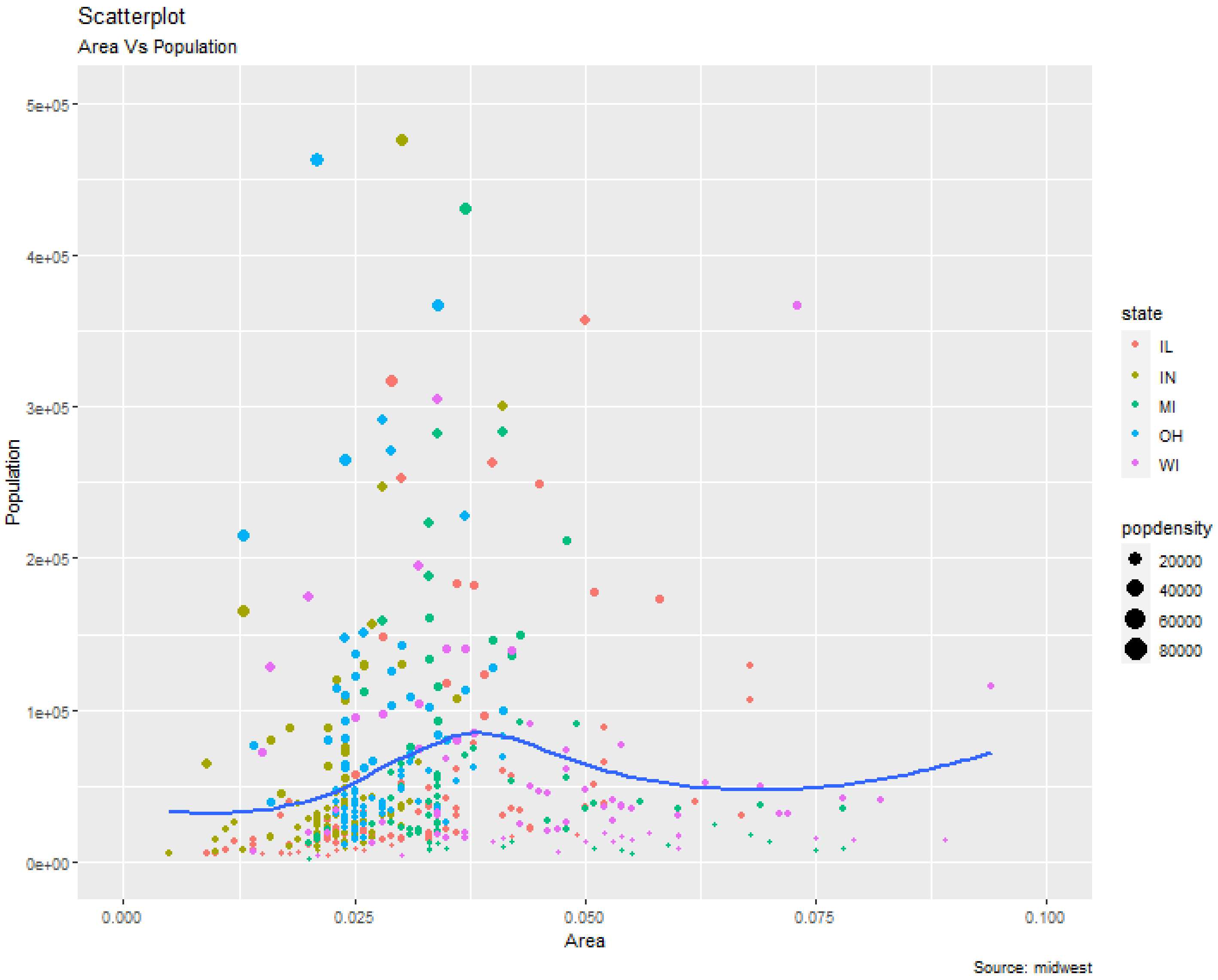
1. Time Series Plots
2. Stacked Area Chart
3. Calendar Heat Map
4. Seasonal Plot
5. Heat Map

6.Groups

1. Dendrogram

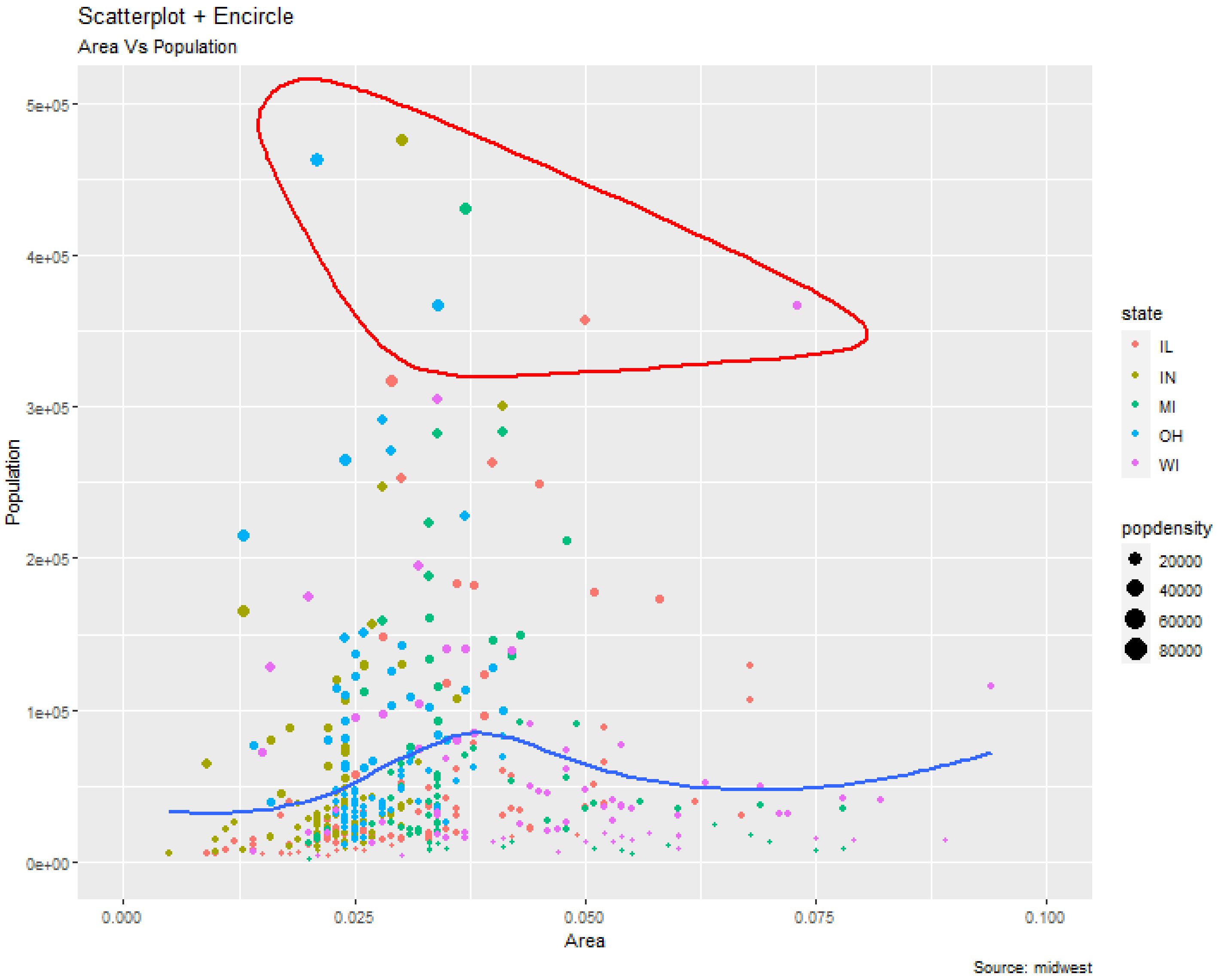
Correlation, scatterplot

```r
ggplot(midwest, aes(x=area, y=poptotal)) +
  geom_point(aes(col=state, size=popdensity)) +
  geom_smooth(method="loess", se=F) +
  xlim(c(0, 0.1)) +
  ylim(c(0, 500000)) +
  labs(subtitle="Area Vs Population",
       y="Population",
       x="Area",
       title="Scatterplot",
       caption = "Source: midwest")
```


Scatterplot
Area Vs Population
Population
5e+05
4e+05
3e+05
2e+05
1e+05
0e+00
0.000
0.025
0.050
0.075
0.100
Area
Source: midwest
state
IL
IN
MI
OH
WI
popdensity
20000
40000
60000
80000

Correlation, scatterplot with encircling

```r
library(ggalt)
midwest_select <- midwest[midwest$poptotal > 350000 &
                            midwest$poptotal <= 500000 &
                            midwest$area > 0.01 &
                            midwest$area < 0.1,]
# Plot
ggplot(midwest, aes(x = area, y = poptotal)) +
  geom_point(aes(col = state, size = popdensity)) +   # draw points
  geom_smooth(method = "loess", se = F) +
  xlim(c(0, 0.1)) +
  ylim(c(0, 500000)) +   # draw smoothing line
  geom_encircle(
    aes(x = area, y = poptotal),
    data = midwest_select,
    color = "red",
    size = 2,
    expand = 0.08
  ) +   # encircle
  labs( subtitle = "Area Vs Population",
    y = "Population",
    x = "Area",
    title = "Scatterplot + Encircle",
    caption = "Source: midwest"
  )
```

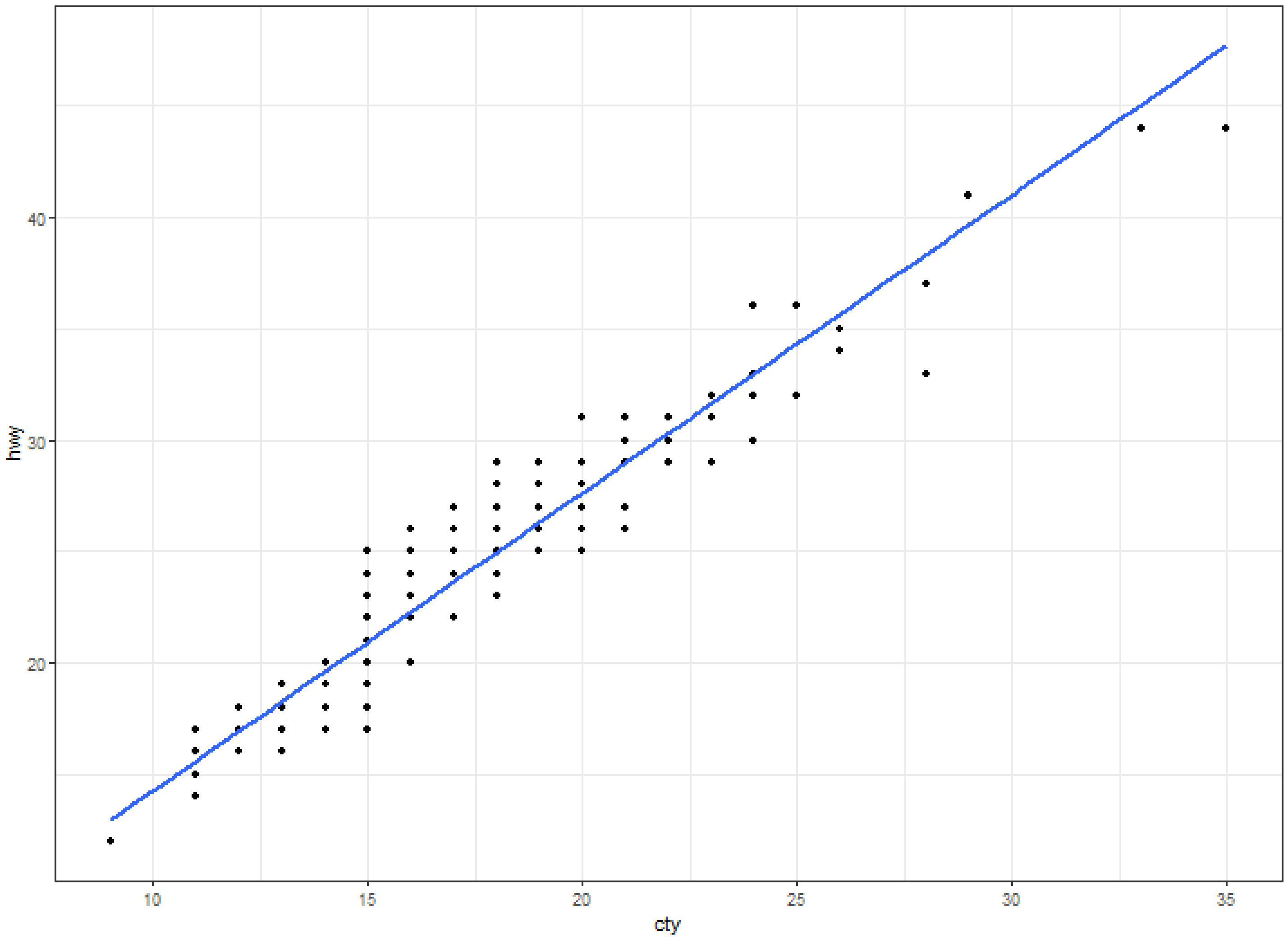

Scatterplot + Encircle
Area Vs Population
Population
5e+05
4e+05
3e+05
2e+05
1e+05
0e+00
0.000
0.025
0.050
0.075
0.100
Area
state
IL
IN
MI
OH
WI
popdensity
20000
40000
60000
80000
Source: midwest

Correlation, jitter plot

```r
ggplot(mpg, aes(cty, hwy)) + geom_point() +
  geom_smooth(method = "lm", se = F) +
  labs(
    subtitle = "mpg: city vs highway mileage",
    y = "hwy",
    x = "cty",
    title = "Scatterplot with overlapping points",
    caption = "Source: midwest"
  ) +
  theme_bw()
```

Scatterplot with overlapping points

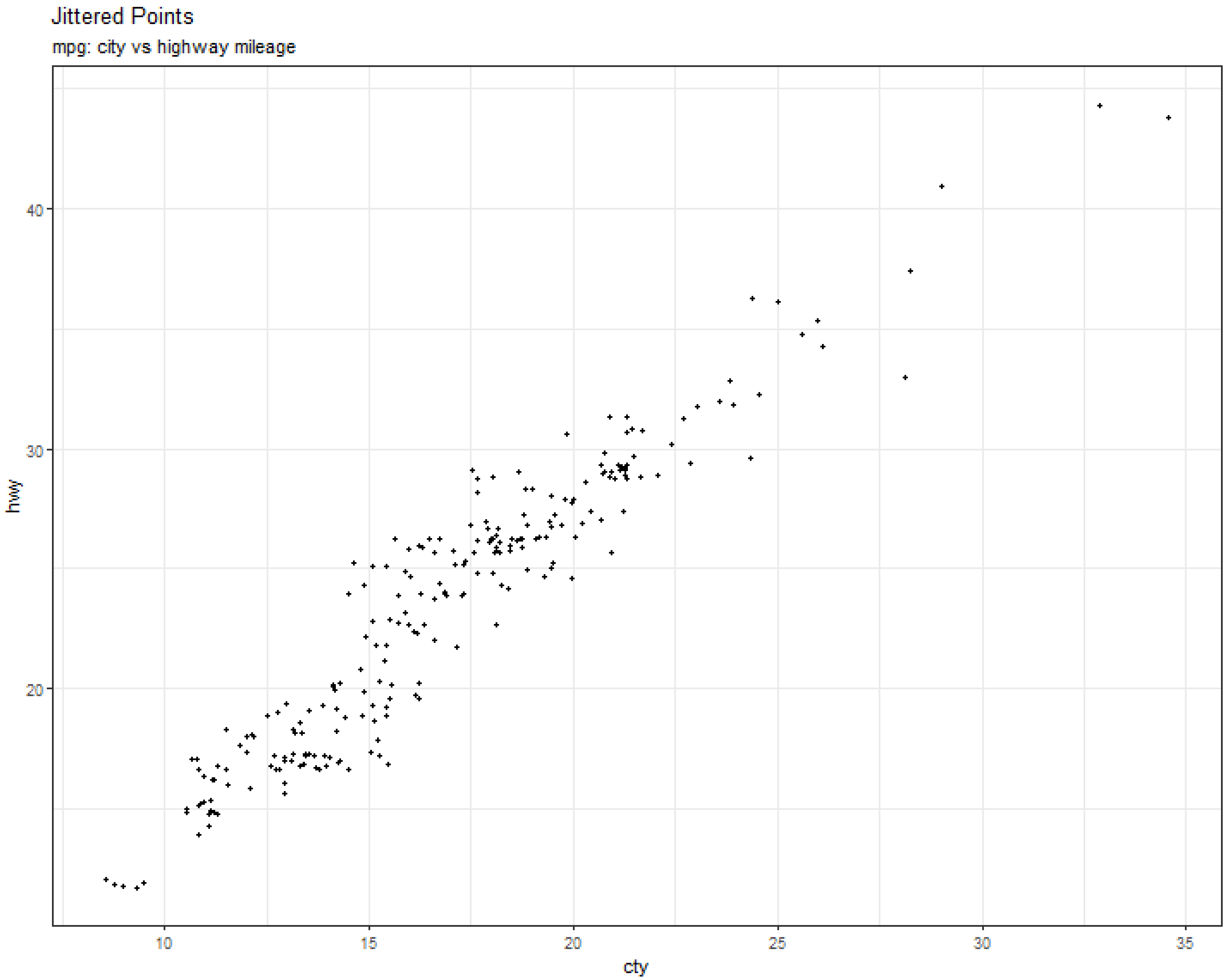
mpg: city vs highway mileage



Source: midwest

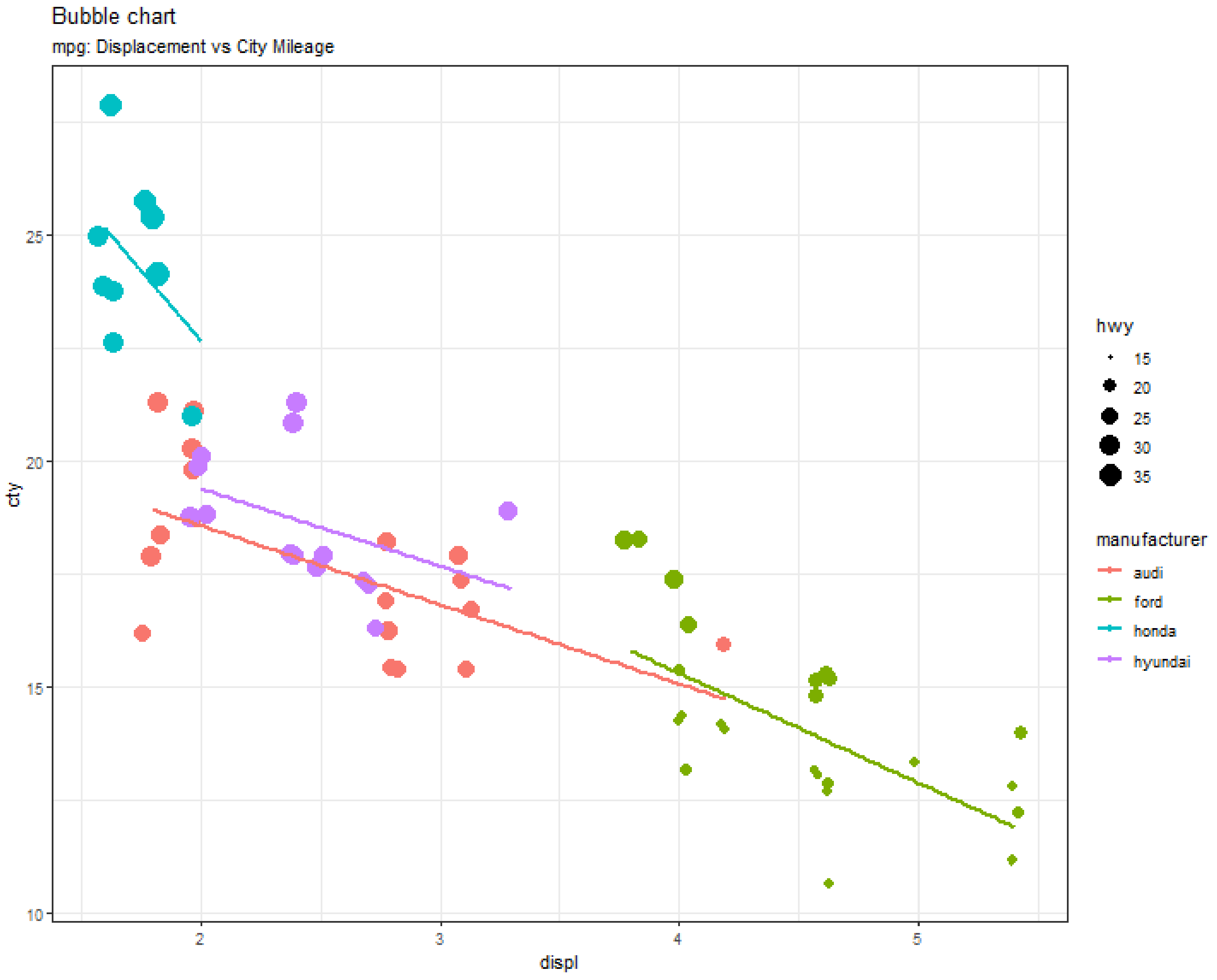
Correlation, jitter plot

```r
ggplot(mpg, aes(cty, hwy)) +
  geom_jitter(width = .5, size = 1) +
  labs(
    subtitle = "mpg: city vs highway mileage",
    y = "hwy",
    x = "cty",
    title = "Jittered Points"
  )+
  theme_bw()
```


Jittered Points
mpg: city vs highway mileage
hwy
40
30
20
10
15
20
25
30
35
cty

Correlation, bubble plot

```
mpg %>%
  filter(manufacturer %in% c("audi", "ford", "honda", "hyundai")) %>%
  ggplot(aes(displ, cty)) +
  labs(subtitle = "mpg: Displacement vs City Mileage",
  title = "Bubble chart") +
  geom_jitter(aes(col = manufacturer, size = hwy)) +
  geom_smooth(aes(col = manufacturer), method = "lm", se = F) +
  theme_bw()
```

Bubble chart
mpg: Displacement vs City Mileage
cty
25
20
15
10
displ
2
3
4
5
hwy
15
20
25
30
35
manufacturer
audi
ford
honda
hyundai

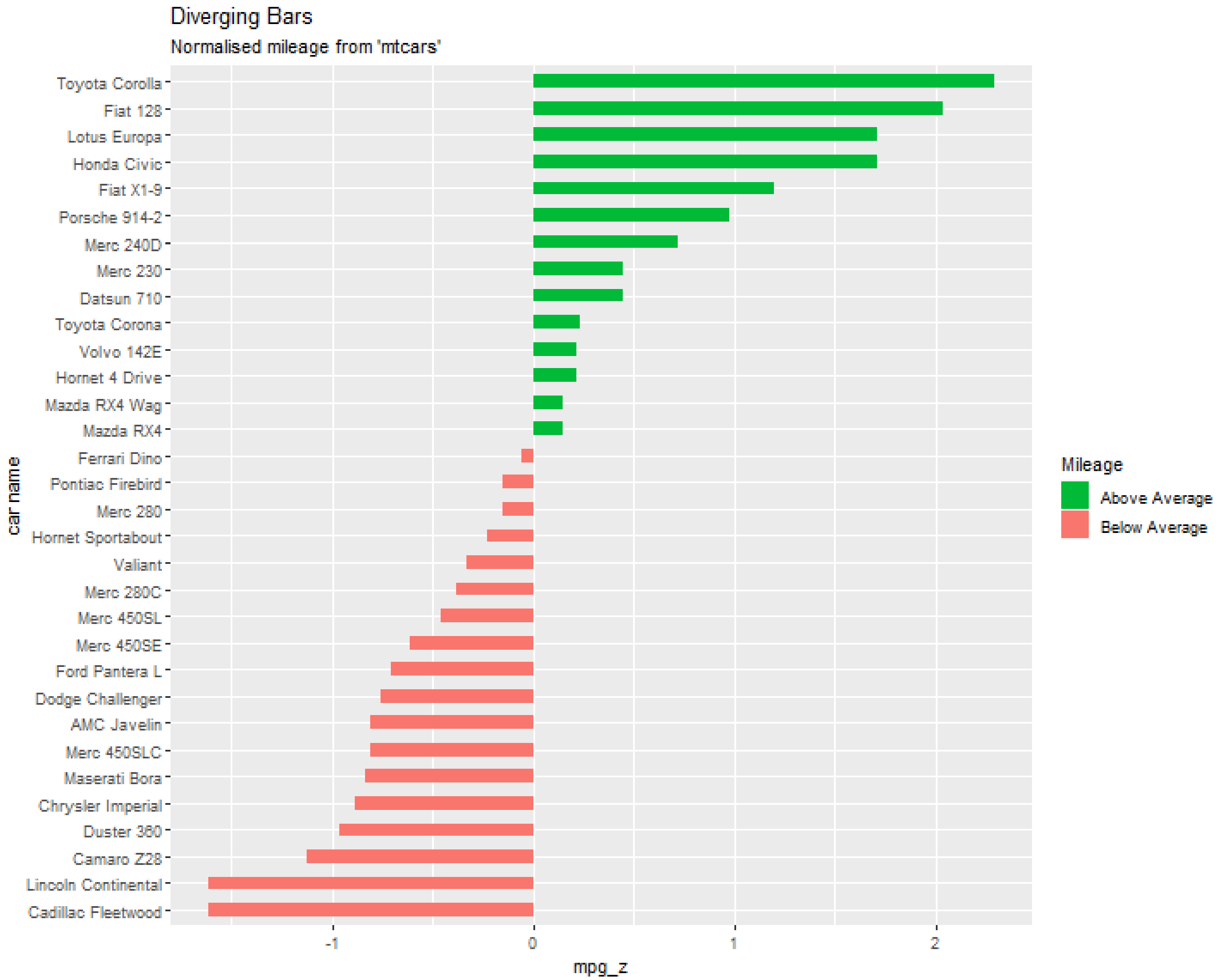
Correlation, correlogram

```r
library(ggcorrplot)
corr <- round(cor(mtcars), 1)
ggcorrplot(corr, hc.order = TRUE,
           type = "lower",
           lab = TRUE,
           lab_size = 3,
           method="circle",
           colors = c("tomato2", "white", "springgreen3"),
           title="Correlogram of mtcars",
           ggtheme=theme_bw)
```

```
## Error in library(ggcorrplot): there is no package called 'ggcorrplot'

## Error in ggcorrplot(corr, hc.order = TRUE, type = "lower", lab = TRUE, : could not find function "ggcorrplot"
```

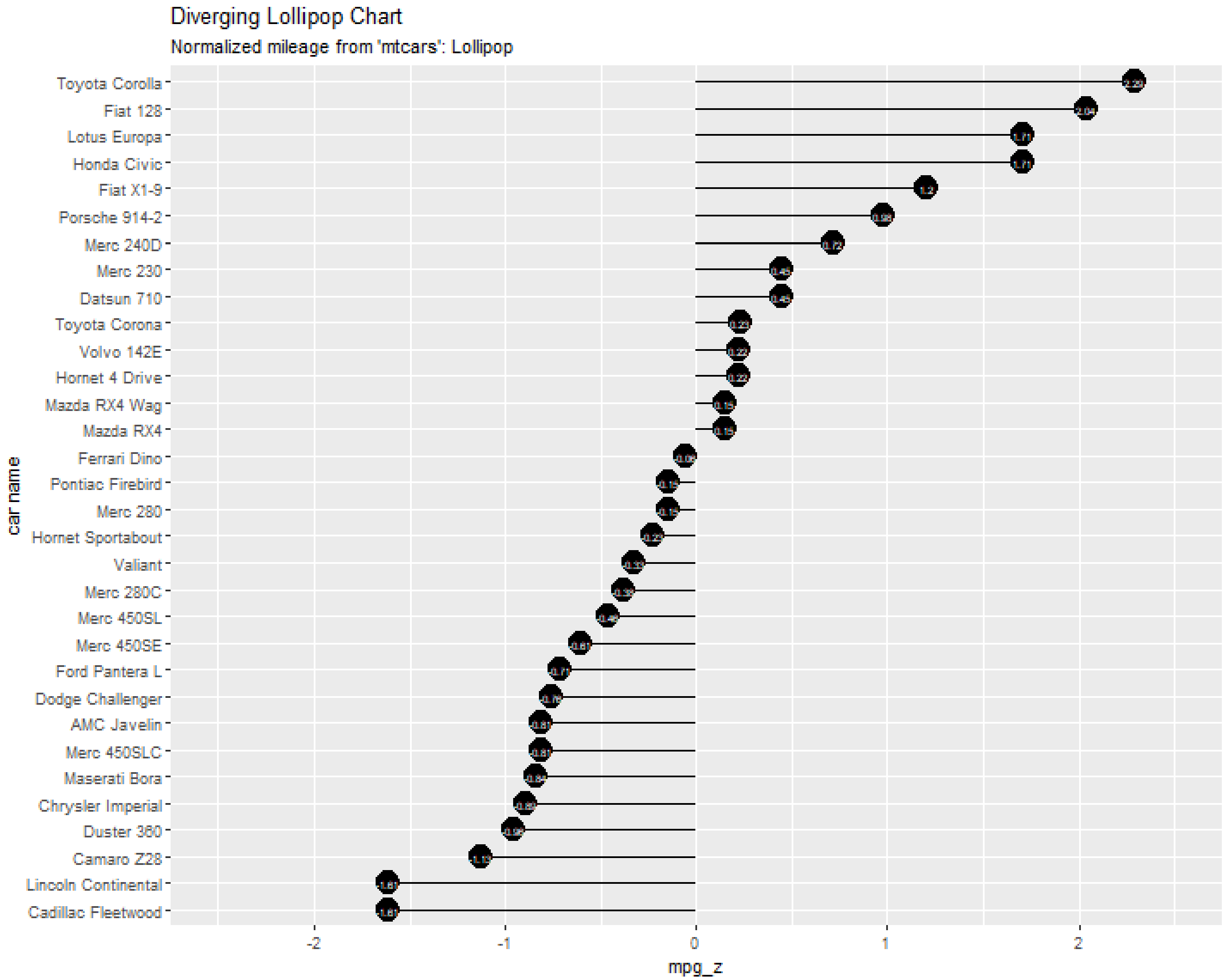
Deviation, diverging bar

```r
# Data Prep
data("mtcars")  # load data
mtcars$`car name` <- rownames(mtcars)  # create new column for car names
mtcars$mpg_z <- round((mtcars$mpg - mean(mtcars$mpg))/sd(mtcars$mpg), 2)  # compute normalized mpg
mtcars$mpg_type <- ifelse(mtcars$mpg_z < 0, "below", "above")  # above / below avg flag
mtcars <- mtcars[order(mtcars$mpg_z), ]  # sort
mtcars$`car name` <- factor(mtcars$`car name`, levels = mtcars$`car name`)  # convert to factor to retain sor

# Diverging Barcharts
ggplot(mtcars, aes(x=`car name`, y=mpg_z, label=mpg_z)) +
  geom_bar(stat='identity', aes(fill=mpg_type), width=.5)  +
  scale_fill_manual(name="Mileage",
                    labels = c("Above Average", "Below Average"),
                    values = c("above"="#00ba38", "below"="#f8766d")) +
  labs(subtitle="Normalised mileage from 'mtcars'",
       title= "Diverging Bars") +
  coord_flip()
```


Diverging Bars
Normalised mileage from 'mtcars'
car name
Toyota Corolla
Fiat 128
Lotus Europa
Honda Civic
Fiat X1-9
Porsche 914-2
Merc 240D
Merc 230
Datsun 710
Toyota Corona
Volvo 142E
Hornet 4 Drive
Mazda RX4 Wag
Mazda RX4
Ferrari Dino
Pontiac Firebird
Merc 280
Hornet Sportabout
Valiant
Merc 280C
Merc 450SL
Merc 450SE
Ford Pantera L
Dodge Challenger
AMC Javelin
Merc 450SLC
Maserati Bora
Chrysler Imperial
Duster 360
Camaro Z28
Lincoln Continental
Cadillac Fleetwood
-1
0
1
2
mpg_z
Mileage
🟩 Above Average
🟥 Below Average

Deviation, diverging lollipop bar

```
ggplot(mtcars, aes(x=`car name`, y=mpg_z, label=mpg_z)) +
  geom_point(stat='identity', fill="black", size=6)  +
  geom_segment(aes(y = 0,
                   x = `car name`,
                   yend = mpg_z,
                   xend = `car name`),
               color = "black") +
  geom_text(color="white", size=2) +
  labs(title="Diverging Lollipop Chart",
       subtitle="Normalized mileage from 'mtcars': Lollipop") +
  ylim(-2.5, 2.5) +
  coord_flip()
```

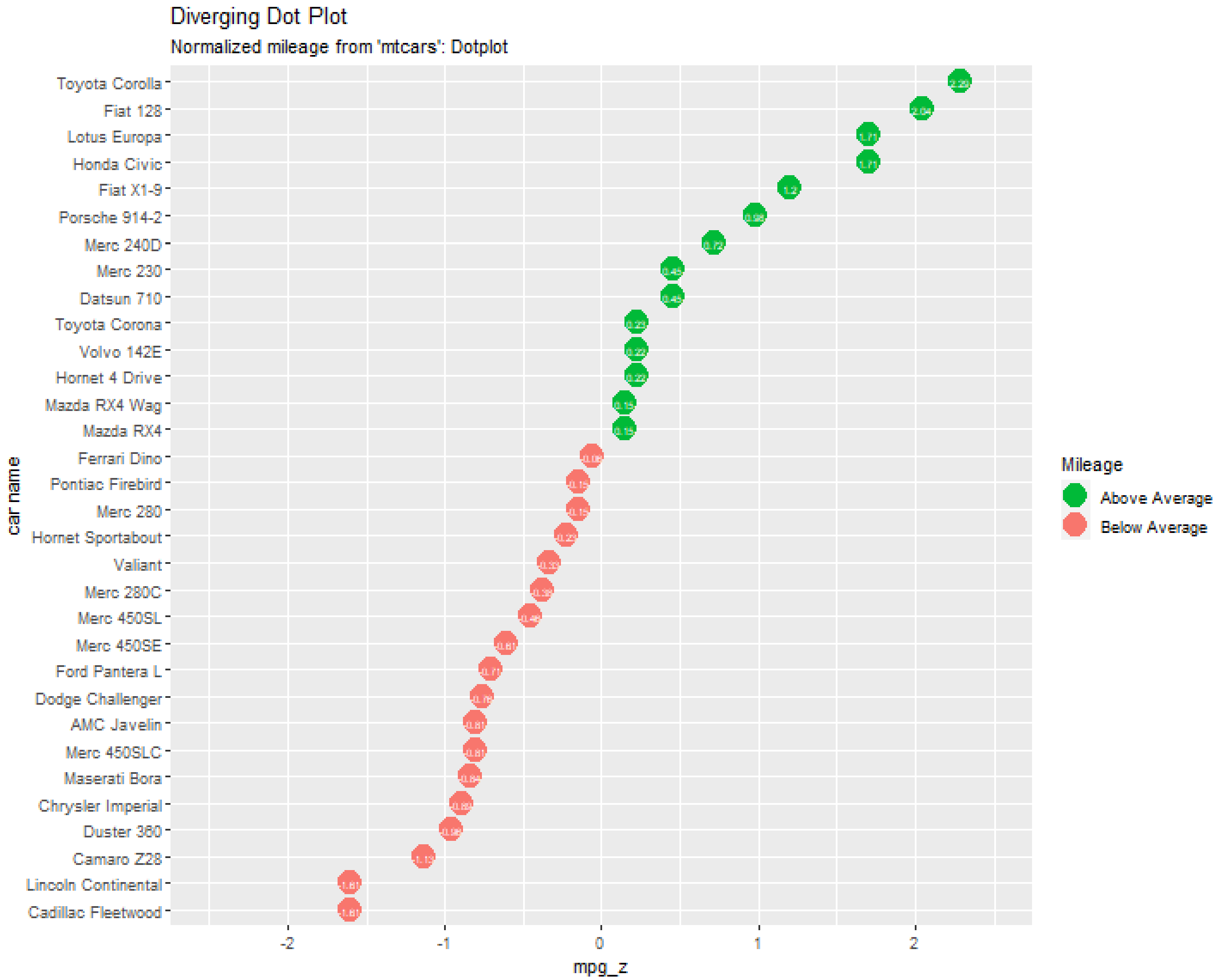


Diverging Lollipop Chart

Normalized mileage from 'mtcars': Lollipop

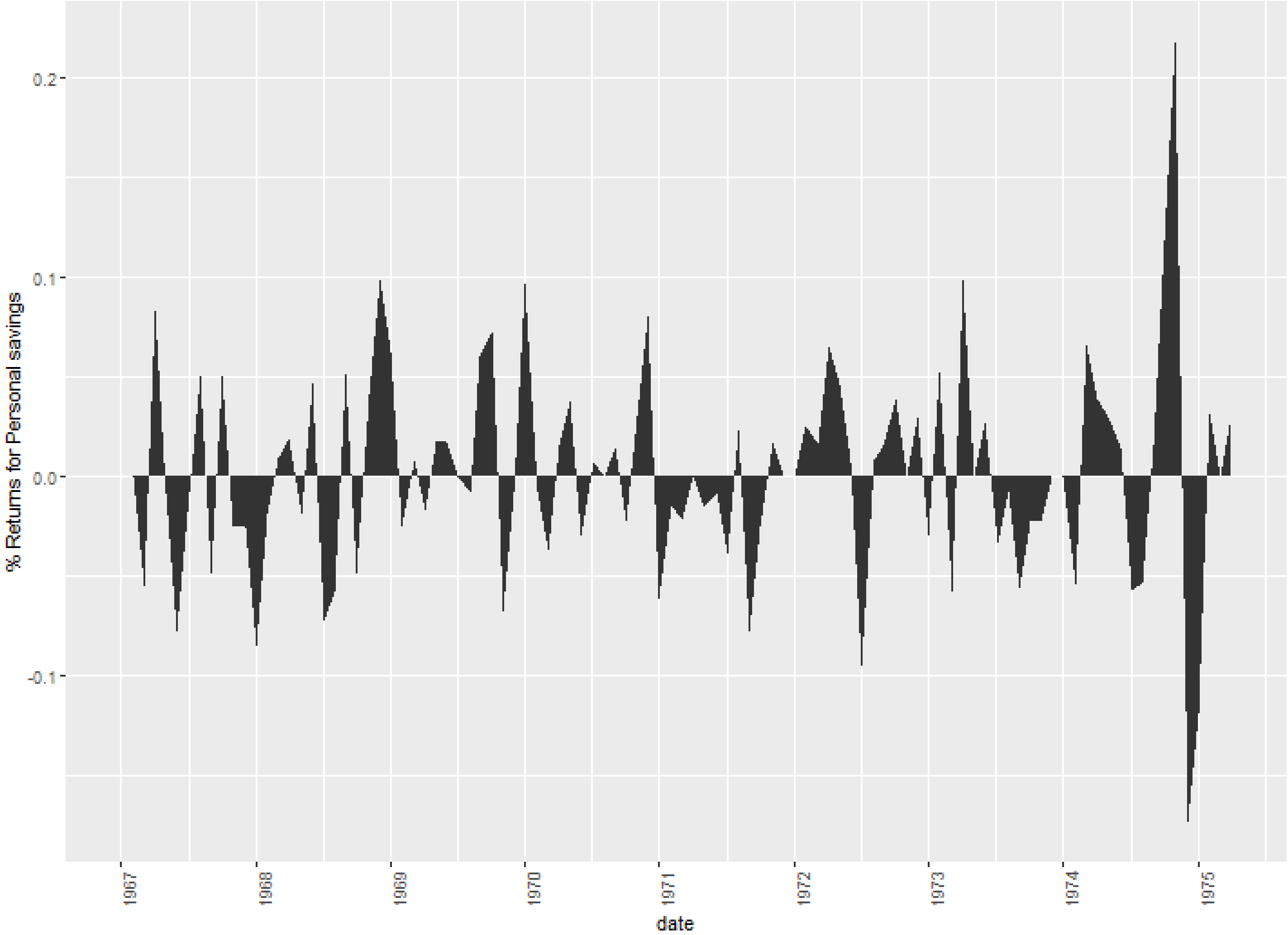
Deviation, diverging lollipop bar

```r
ggplot(mtcars, aes(x=`car name`, y=mpg_z, label=mpg_z)) +
  geom_point(stat='identity', aes(col=mpg_type), size=6)  +
  scale_color_manual(name="Mileage",
                     labels = c("Above Average", "Below Average"),
                     values = c("above"="#00ba38", "below"="#f8766d")) +
  geom_text(color="white", size=2) +
  labs(title="Diverging Dot Plot",
       subtitle="Normalized mileage from 'mtcars': Dotplot") +
  ylim(-2.5, 2.5) +
  coord_flip()
```


Diverging Dot Plot
Normalized mileage from 'mtcars': Dotplot
car name
Toyota Corolla
Fiat 128
Lotus Europa
Honda Civic
Fiat X1-9
Porsche 914-2
Merc 240D
Merc 230
Datsun 710
Toyota Corona
Volvo 142E
Hornet 4 Drive
Mazda RX4 Wag
Mazda RX4
Ferrari Dino
Pontiac Firebird
Merc 280
Hornet Sportabout
Valiant
Merc 280C
Merc 450SL
Merc 450SE
Ford Pantera L
Dodge Challenger
AMC Javelin
Merc 450SLC
Maserati Bora
Chrysler Imperial
Duster 360
Camaro Z28
Lincoln Continental
Cadillac Fleetwood
2.29
2.04
1.71
1.71
1.2
0.98
0.72
0.45
0.45
0.23
0.22
0.22
0.15
0.15
-0.06
-0.15
-0.15
-0.23
-0.33
-0.38
-0.46
-0.61
-0.71
-0.76
-0.81
-0.81
-0.84
-0.89
-0.96
-1.13
-1.61
-1.61
-2
-1
0
1
2
mpg_z
Mileage
Above Average
Below Average

Deviation, area chart

```r
economics$returns_perc <- c(0, diff(economics$psavert)/economics$psavert[-length(economics$psavert)])

# Create break points and labels for axis ticks
brks <- economics$date[seq(1, length(economics$date), 12)]
lbls <- lubridate::year(economics$date[seq(1, length(economics$date), 12)])

# Plot
ggplot(economics[1:100, ], aes(date, returns_perc)) +
  geom_area() +
  scale_x_date(breaks=brks, labels=lbls) +
  theme(axis.text.x = element_text(angle=90)) +
  labs(title="Area Chart",
       subtitle = "Perc Returns for Personal Savings",
       y="% Returns for Personal savings",
       caption="Source: economics")
```



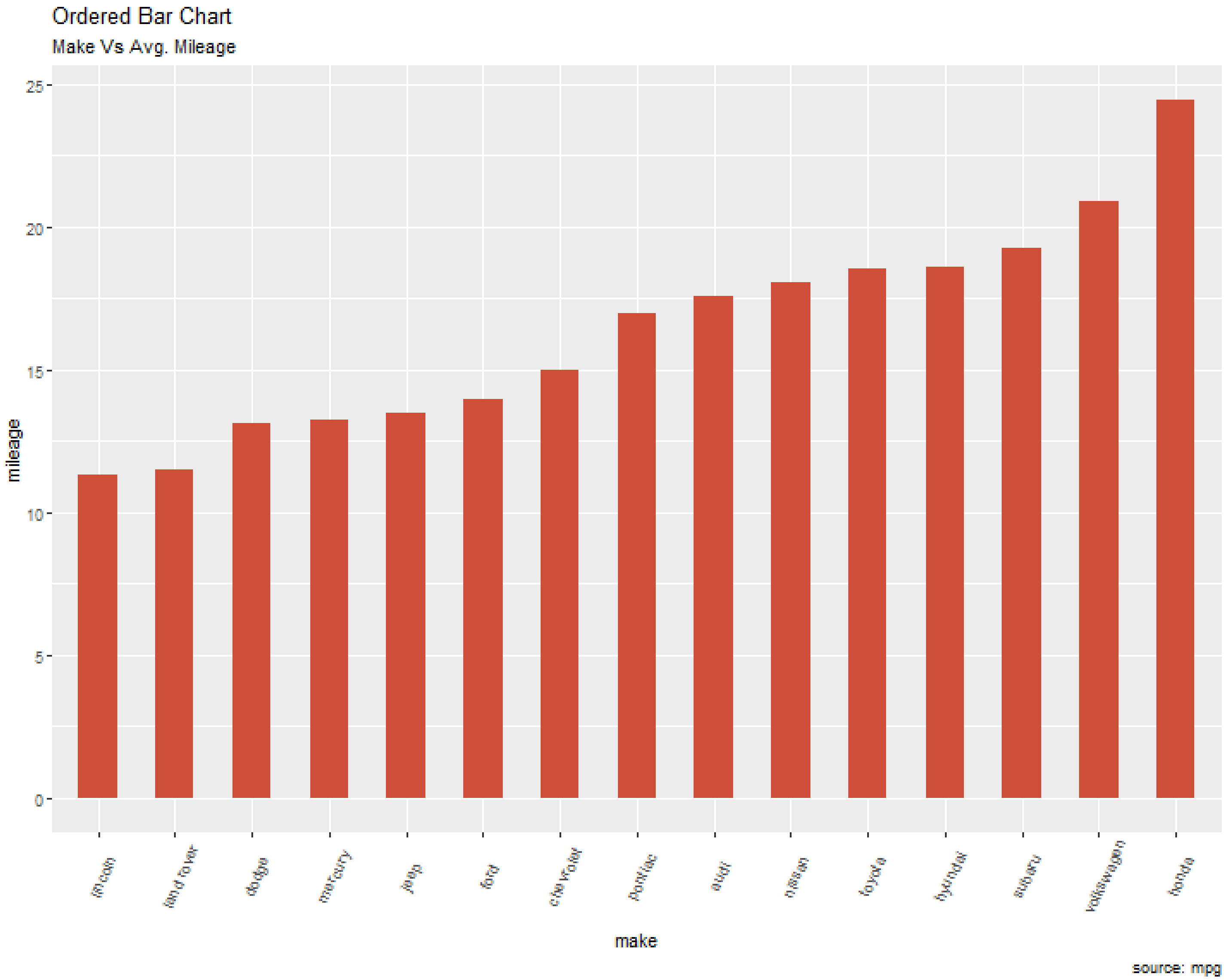
Area Chart

Perc Returns for Personal Savings

Source: economics

Ranking, ordered bar chart

```r
# Prepare data: group mean city mileage by manufacturer.
cty_mpg <- aggregate(mpg$cty, by=list(mpg$manufacturer), FUN=mean)  # aggregate
colnames(cty_mpg) <- c("make", "mileage")  # change column names
cty_mpg <- cty_mpg[order(cty_mpg$mileage), ]  # sort
cty_mpg$make <- factor(cty_mpg$make, levels = cty_mpg$make)  # to retain the order in plot.
ggplot(cty_mpg, aes(x=make, y=mileage)) +
  geom_bar(stat="identity", width=.5, fill="tomato3") +
  labs(title="Ordered Bar Chart",
       subtitle="Make Vs Avg. Mileage",
       caption="source: mpg") +
  theme(axis.text.x = element_text(angle=65, vjust=0.6))
```



Ordered Bar Chart

Make Vs Avg. Mileage

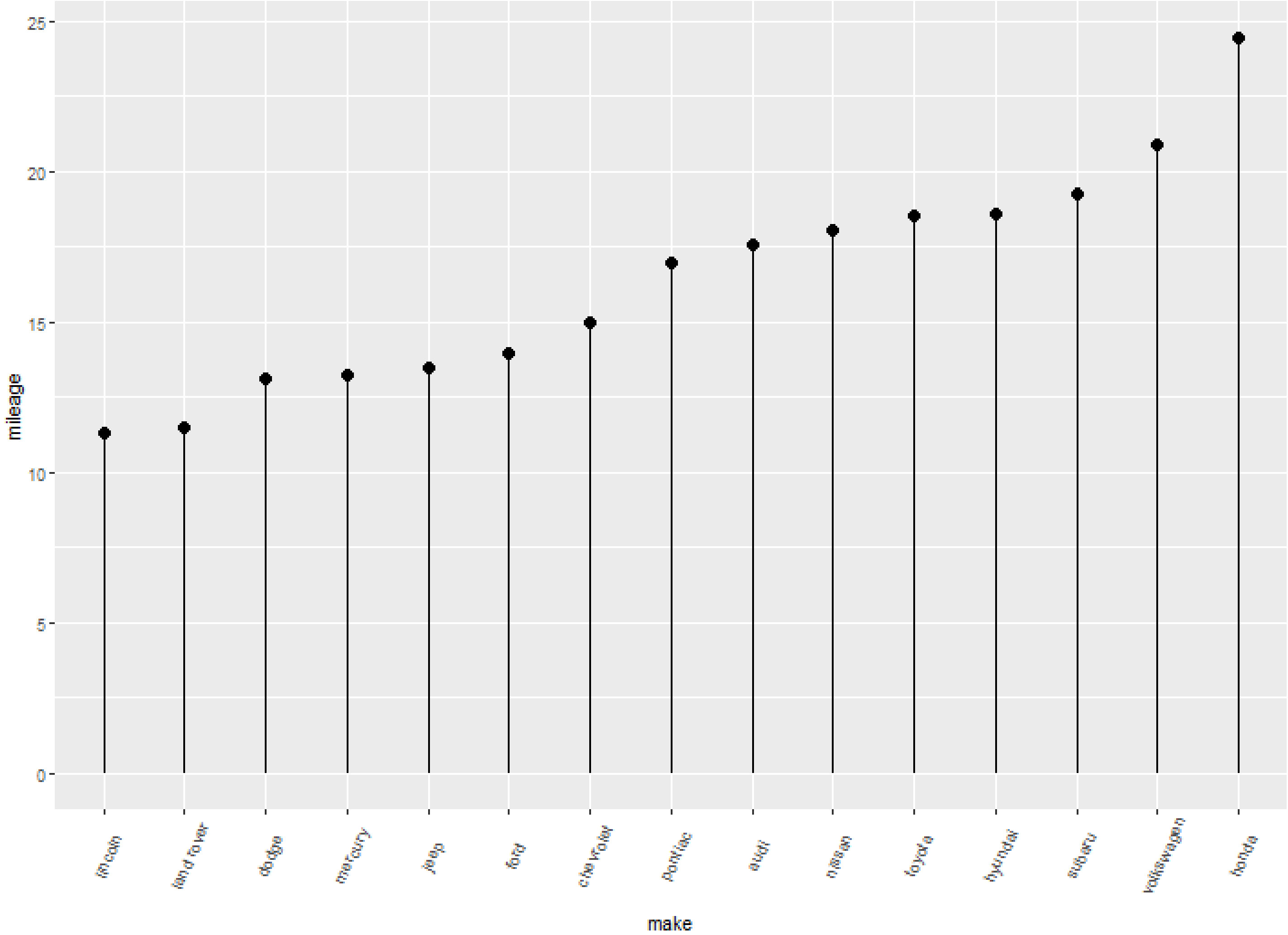
source: mpg

Ranking, lollipop chart

```
ggplot(cty_mpg, aes(x=make, y=mileage)) +
  geom_point(size=3) +
  geom_segment(aes(x=make,
                   xend=make,
                   y=0,
                   yend=mileage)) +
  labs(title="Lollipop Chart",
       subtitle="Make Vs Avg. Mileage",
       caption="source: mpg") +
  theme(axis.text.x = element_text(angle=65, vjust=0.6))
```

Lollipop Chart

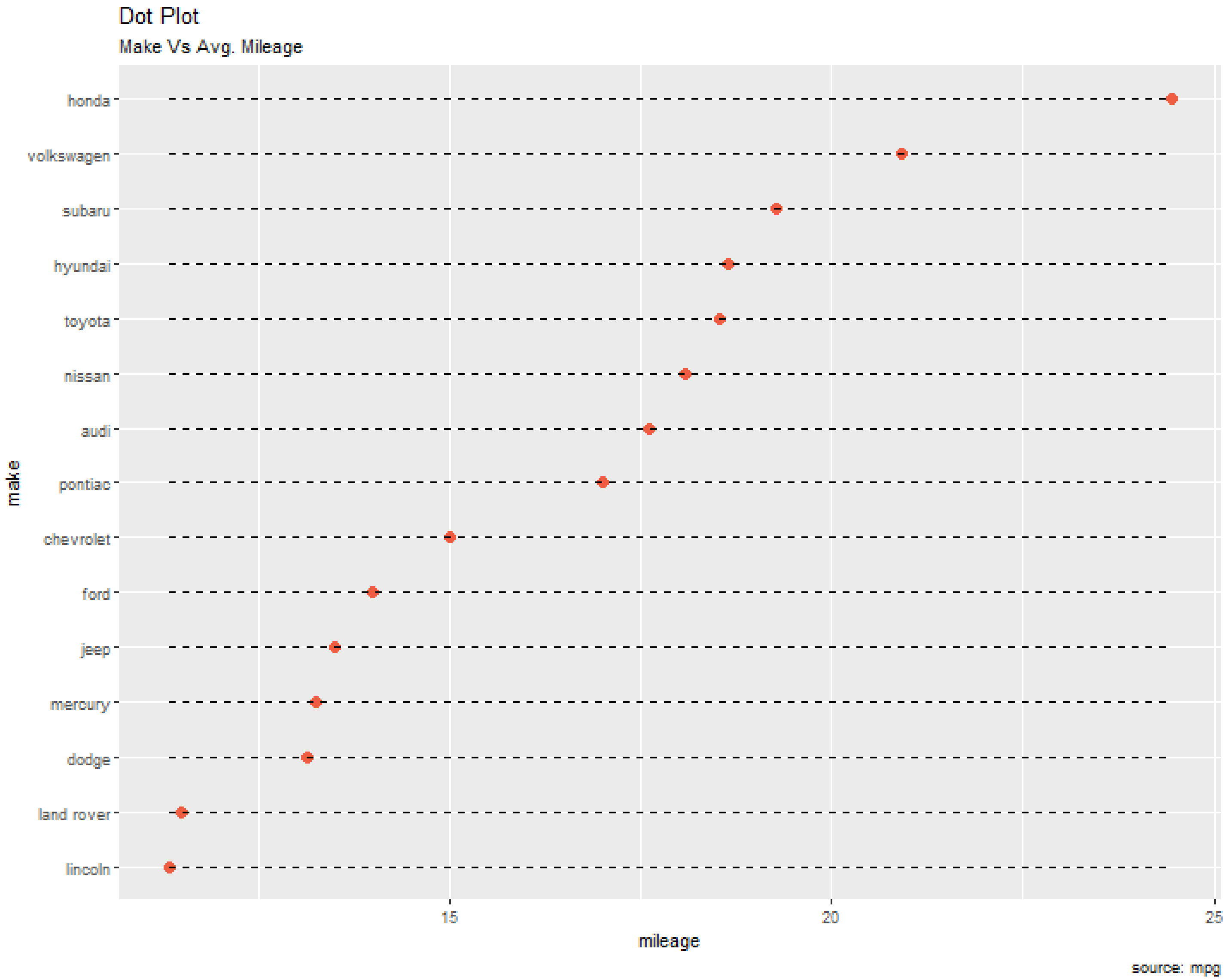
Make Vs Avg. Mileage



source: mpg

Ranking, dot plot

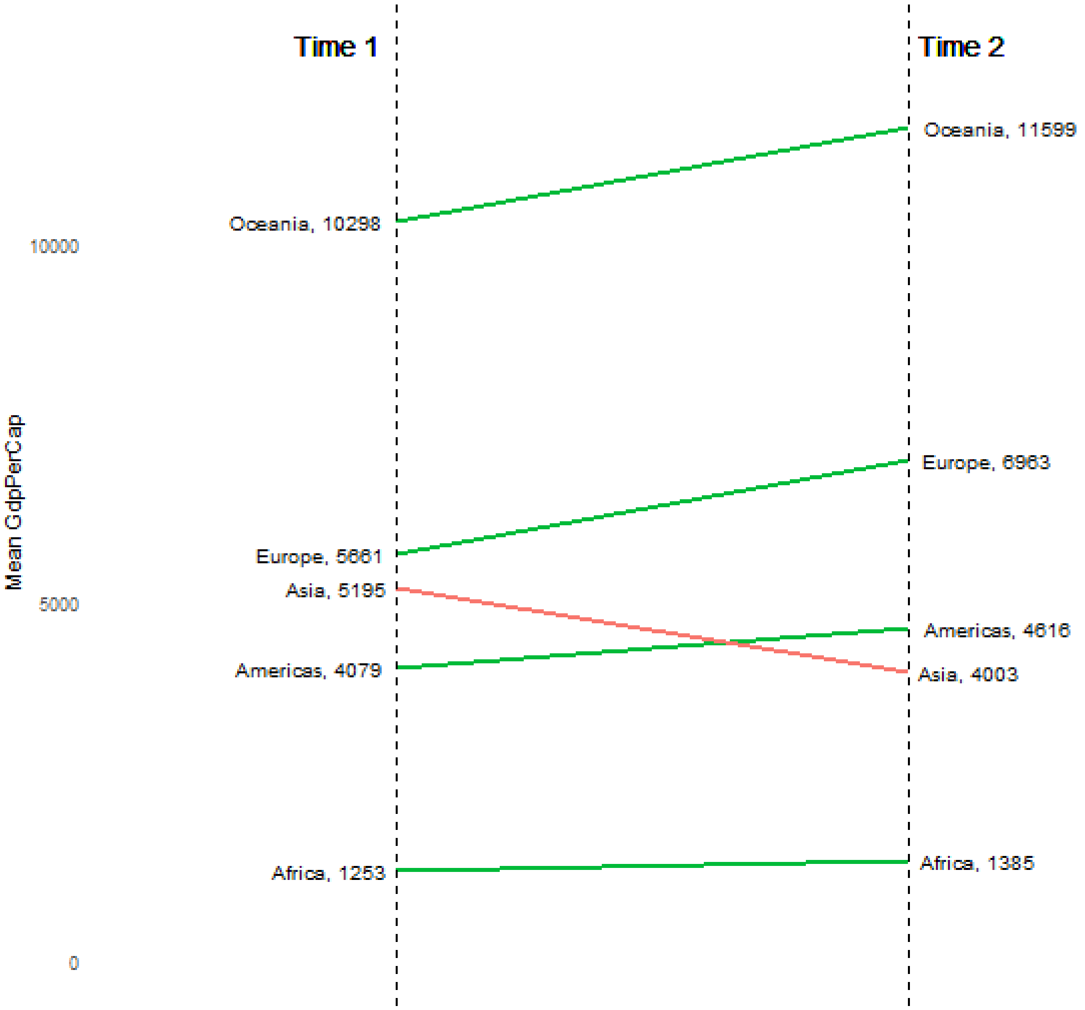
```r
# Plot
ggplot(cty_mpg, aes(x=make, y=mileage)) +
  geom_point(col="tomato2", size=3) +   # Draw points
  geom_segment(aes(x=make,
                   xend=make,
                   y=min(mileage),
                   yend=max(mileage)),
               linetype="dashed",
               size=0.1) +   # Draw dashed lines
  labs(title="Dot Plot",
       subtitle="Make Vs Avg. Mileage",
       caption="source: mpg") +
  coord_flip()
```



Dot Plot
Make Vs Avg. Mileage
honda
volkswagen
subaru
hyundai
toyota
nissan
audi
pontiac
chevrolet
ford
jeep
mercury
dodge
land rover
lincoln
make
15
20
25
mileage
source: mpg

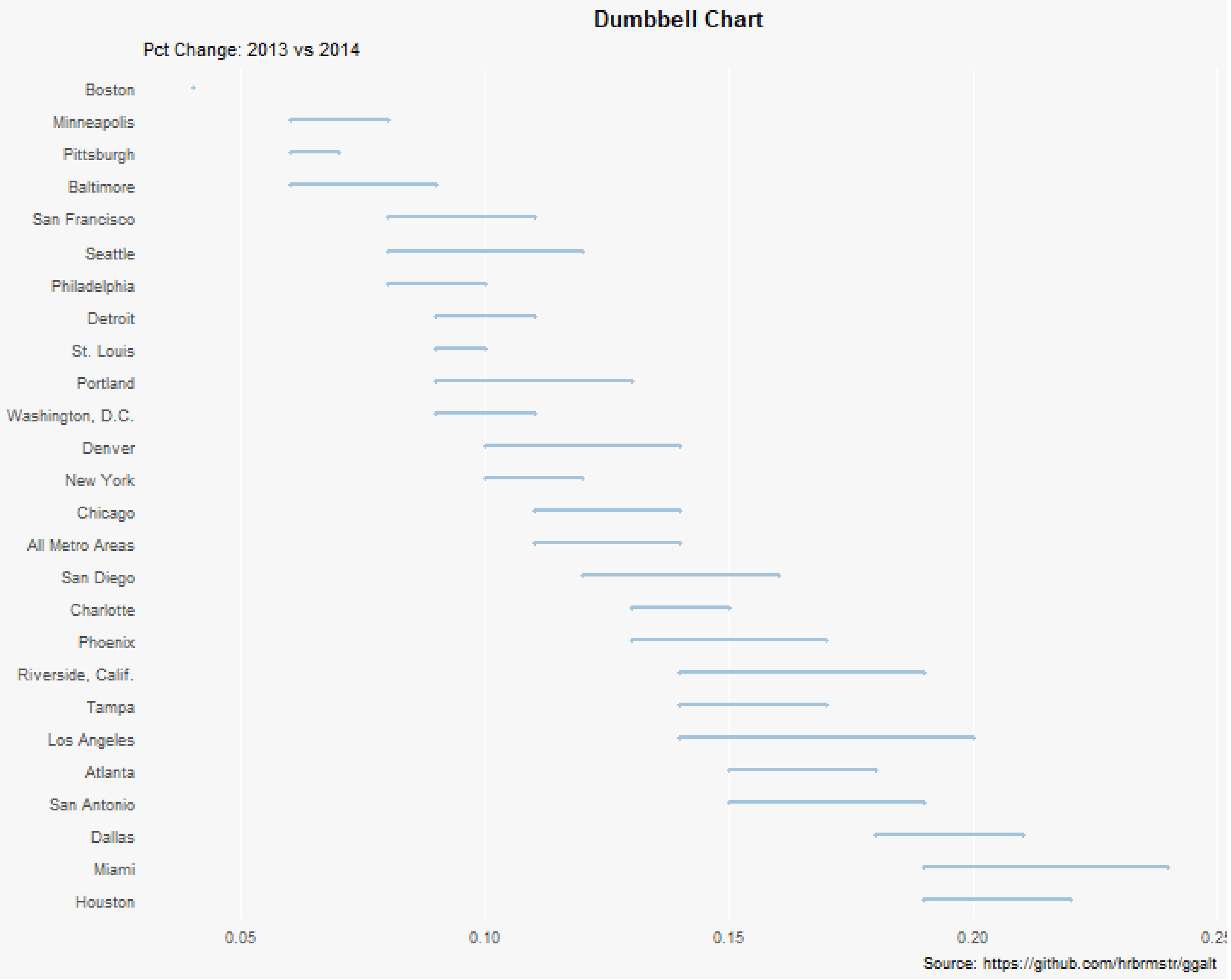
Ranking, slop chart

```r
# prep data
df <- read.csv("https://raw.githubusercontent.com/selva86/datasets/master/gdppercap.csv")
colnames(df) <- c("continent", "1952", "1957")
left_label <- paste(df$continent, round(df$`1952`),sep=", ")
right_label <- paste(df$continent, round(df$`1957`),sep=", ")
df$class <- ifelse((df$`1957` - df$`1952`) < 0, "red", "green")

# Plot
p <- ggplot(df) + geom_segment(aes(x=1, xend=2, y=`1952`, yend=`1957`, col=class), size=.75, show.legend=F) +
                  geom_vline(xintercept=1, linetype="dashed", size=.1) +
                  geom_vline(xintercept=2, linetype="dashed", size=.1) +
                  scale_color_manual(labels = c("Up", "Down"),
                                     values = c("green"="#00ba38", "red"="#f8766d")) +  # color of lines
                  labs(x="", y="Mean GdpPerCap") +  # Axis labels
                  xlim(.5, 2.5) + ylim(0,(1.1*(max(df$`1952`, df$`1957`))))  # X and Y axis limits

# Add texts
p <- p + geom_text(label=left_label, y=df$`1952`, x=rep(1, NROW(df)), hjust=1.1, size=3.5)
p <- p + geom_text(label=right_label, y=df$`1957`, x=rep(2, NROW(df)), hjust=-0.1, size=3.5)
p <- p + geom_text(label="Time 1", x=1, y=1.1*(max(df$`1952`, df$`1957`)), hjust=1.2, size=5)  # title
p <- p + geom_text(label="Time 2", x=2, y=1.1*(max(df$`1952`, df$`1957`)), hjust=-0.1, size=5)  # title

# Minify theme
p + theme(panel.background = element_blank(), panel.grid = element_blank(),
```


Time 1
Time 2
Oceania, 11599
Oceania, 10298
10000
Mean GdpPerCap
Europe, 6963
Europe, 5661
Asia, 5195
5000
Americas, 4616
Americas, 4079
Asia, 4003
Africa, 1385
Africa, 1253
0

Ranking, dumbbell

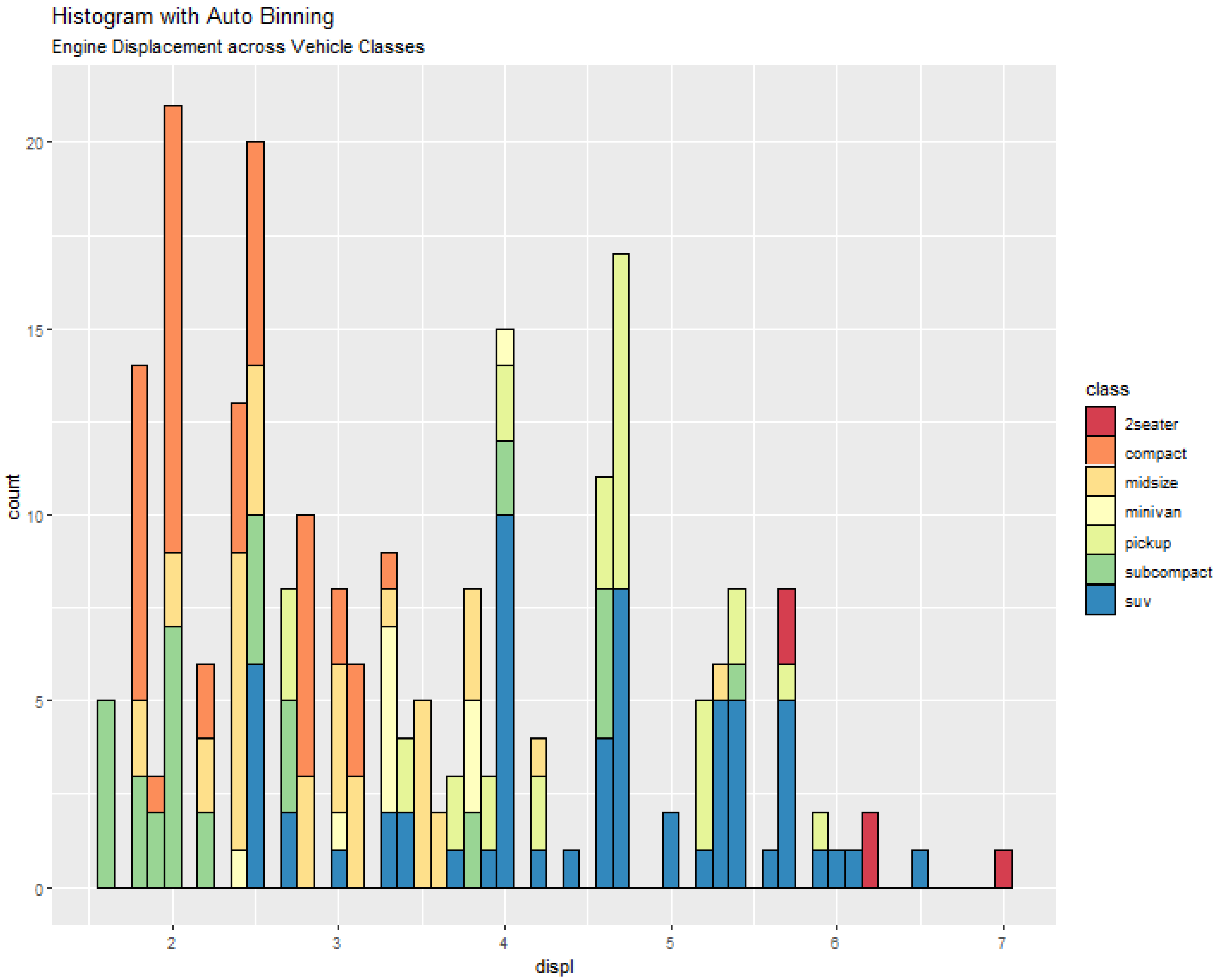
```r
library(ggalt)
health <-
  read.csv("https://raw.githubusercontent.com/selva86/datasets/master/health.csv")
health$Area <-
  factor(health$Area, levels = as.character(health$Area))  # for right ordering of the dumbells
ggplot(health, aes(
  x = pct_2013,
  xend = pct_2014,
  y = Area,
  group = Area
)) +
  geom_dumbbell(color = "#a3c4dc",
                size = 0.75,
                point.colour.l = "#0e668b") +
  labs(
    x = NULL, y = NULL,
    title = "Dumbbell Chart",
    subtitle = "Pct Change: 2013 vs 2014",
    caption = "Source: https://github.com/hrbrmstr/ggalt"
  ) +
  theme(
    plot.title = element_text(hjust = 0.5, face = "bold"),
    plot.background = element_rect(fill = "#f7f7f7"),
    panel.background = element_rect(fill = "#f7f7f7"),
```



Source: https://github.com/hrbrmstr/ggalt

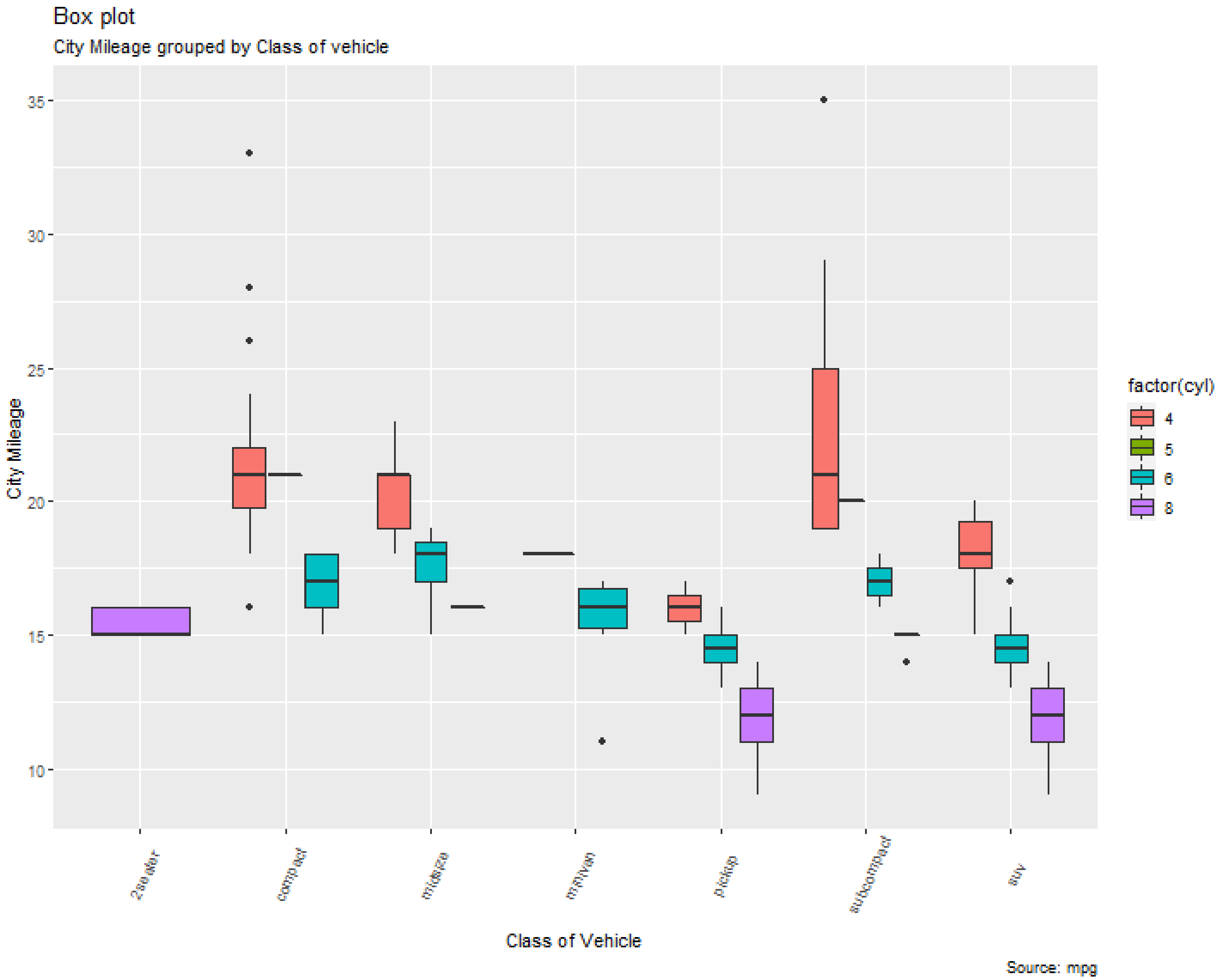
Distribution, histogram

```r
# Histogram on a Continuous (Numeric) Variable
g <- ggplot(mpg, aes(displ)) + scale_fill_brewer(palette = "Spectral")

g + geom_histogram(aes(fill=class),
                   binwidth = .1,
                   col="black",
                   size=.1) +  # change binwidth
  labs(title="Histogram with Auto Binning",
       subtitle="Engine Displacement across Vehicle Classes")
```


Histogram with Auto Binning
Engine Displacement across Vehicle Classes
count
0
5
10
15
20
2
3
4
5
6
7
displ
class
2seater
compact
midsize
minivan
pickup
subcompact
suv

Distribution, boxplot

```r
library(ggthemes)
g <- ggplot(mpg, aes(class, cty))
g + geom_boxplot(aes(fill=factor(cyl))) +
  theme(axis.text.x = element_text(angle=65, vjust=0.6)) +
  labs(title="Box plot",
       subtitle="City Mileage grouped by Class of vehicle",
       caption="Source: mpg",
       x="Class of Vehicle",
       y="City Mileage")
```



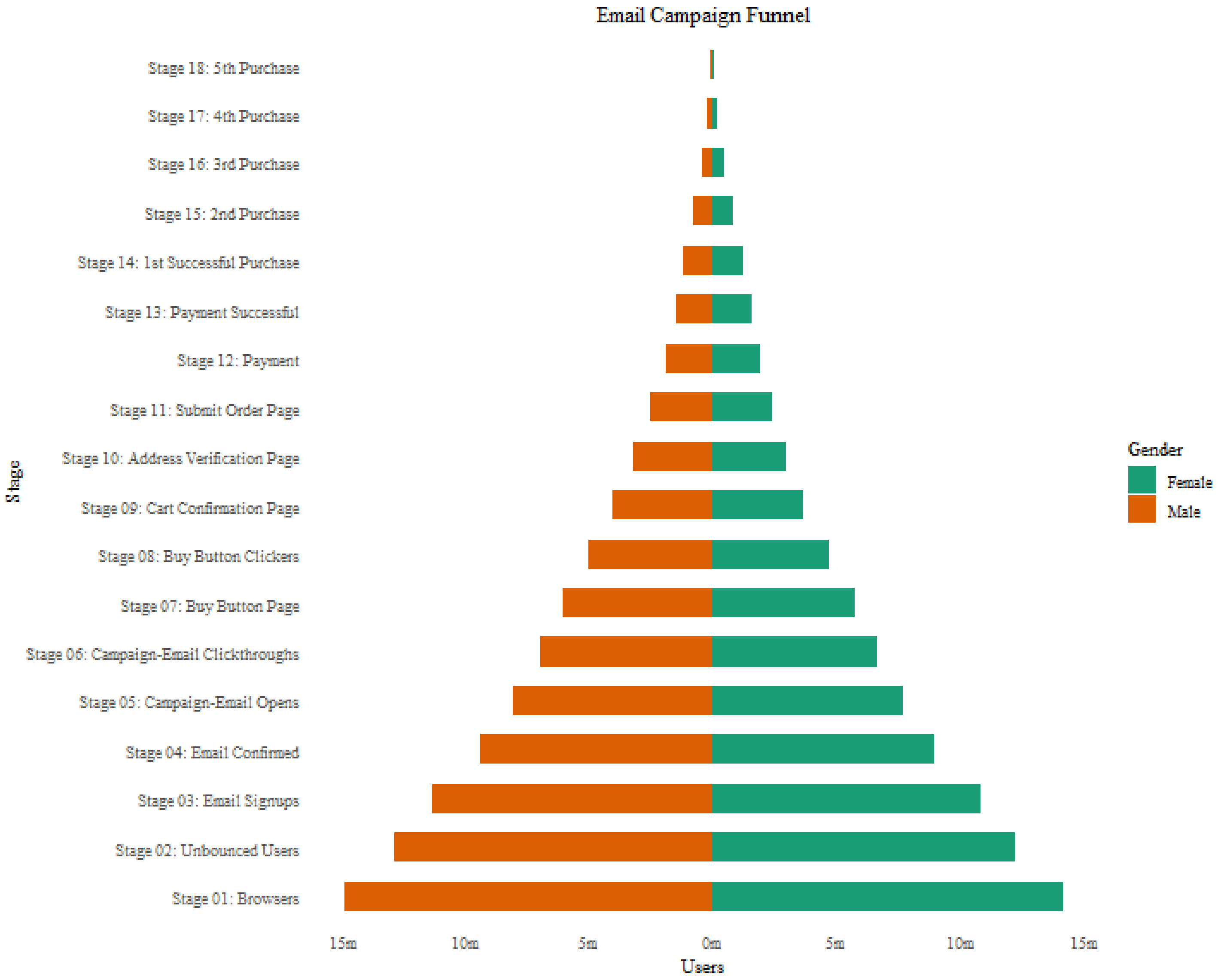
Box plot

City Mileage grouped by Class of vehicle

Source: mpg

Distribution, population pyramid

```r
library(ggplot2)
library(ggthemes)
# Read data
email_campaign_funnel <- read.csv("https://raw.githubusercontent.com/selva86/datasets/master/email_campaign_f

# X Axis Breaks and Labels
brks <- seq(-15000000, 15000000, 5000000)
lbls = paste0(as.character(c(seq(15, 0, -5), seq(5, 15, 5))), "m")

# Plot
ggplot(email_campaign_funnel, aes(x = Stage, y = Users, fill = Gender)) +   # Fill column
                              geom_bar(stat = "identity", width = .6) +   # draw the bars
                              scale_y_continuous(breaks = brks,   # Breaks
                                                 labels = lbls) + # Labels
                              coord_flip() +  # Flip axes
                              labs(title="Email Campaign Funnel") +
                              theme_tufte() +  # Tufte theme from ggfortify
                              theme(plot.title = element_text(hjust = .5),
                                    axis.ticks = element_blank()) +   # Centre plot title
                              scale_fill_brewer(palette = "Dark2")  # Color palette
```


Email Campaign Funnel
Stage
Stage 18: 5th Purchase
Stage 17: 4th Purchase
Stage 16: 3rd Purchase
Stage 15: 2nd Purchase
Stage 14: 1st Successful Purchase
Stage 13: Payment Successful
Stage 12: Payment
Stage 11: Submit Order Page
Stage 10: Address Verification Page
Stage 09: Cart Confirmation Page
Stage 08: Buy Button Clickers
Stage 07: Buy Button Page
Stage 06: Campaign-Email Clickthroughs
Stage 05: Campaign-Email Opens
Stage 04: Email Confirmed
Stage 03: Email Signups
Stage 02: Unbounced Users
Stage 01: Browsers
15m
10m
5m
0m
5m
10m
15m
Users
Gender
Female
Male

Change, time series

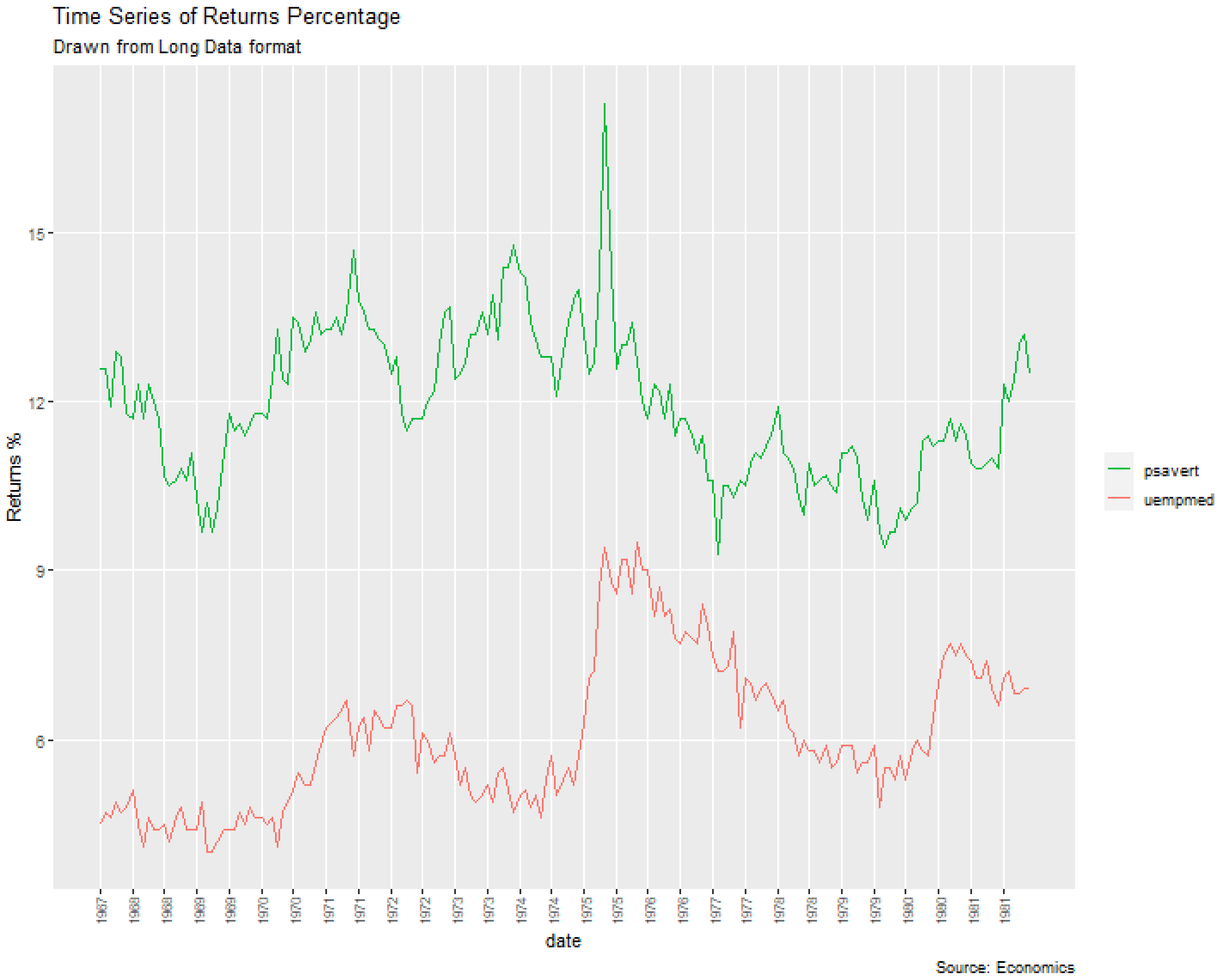
```r
library(ggfortify)
# Plot
autoplot(AirPassengers) +
  labs(title="AirPassengers") +
  theme(plot.title = element_text(hjust=0.5))
```

```
## Error in library(ggfortify): there is no package called 'ggfortify'
```

```
## Error in `autoplot()`:
## ! Objects of type ts not supported by autoplot.
```

Change, time series

```r
library(ggplot2)
library(lubridate)


df <- economics_long[economics_long$variable %in% c("psavert", "uempmed"), ]
df <- df[lubridate::year(df$date) %in% c(1967:1981), ]

# labels and breaks for X axis text
brks <- df$date[seq(1, length(df$date), 12)]
lbls <- lubridate::year(brks)

# plot
ggplot(df, aes(x=date)) +
  geom_line(aes(y=value, col=variable)) +
  labs(title="Time Series of Returns Percentage",
       subtitle="Drawn from Long Data format",
       caption="Source: Economics",
       y="Returns %",
       color=NULL) +  # title and caption
  scale_x_date(labels = lbls, breaks = brks) +  # change to monthly ticks and labels
  scale_color_manual(labels = c("psavert", "uempmed"),
                     values = c("psavert"="#00ba38", "uempmed"="#f8766d")) +  # line color
  theme(axis.text.x = element_text(angle = 90, vjust=0.5, size = 8),  # rotate x axis text
        panel.grid.minor = element_blank())  # turn off minor grid
```



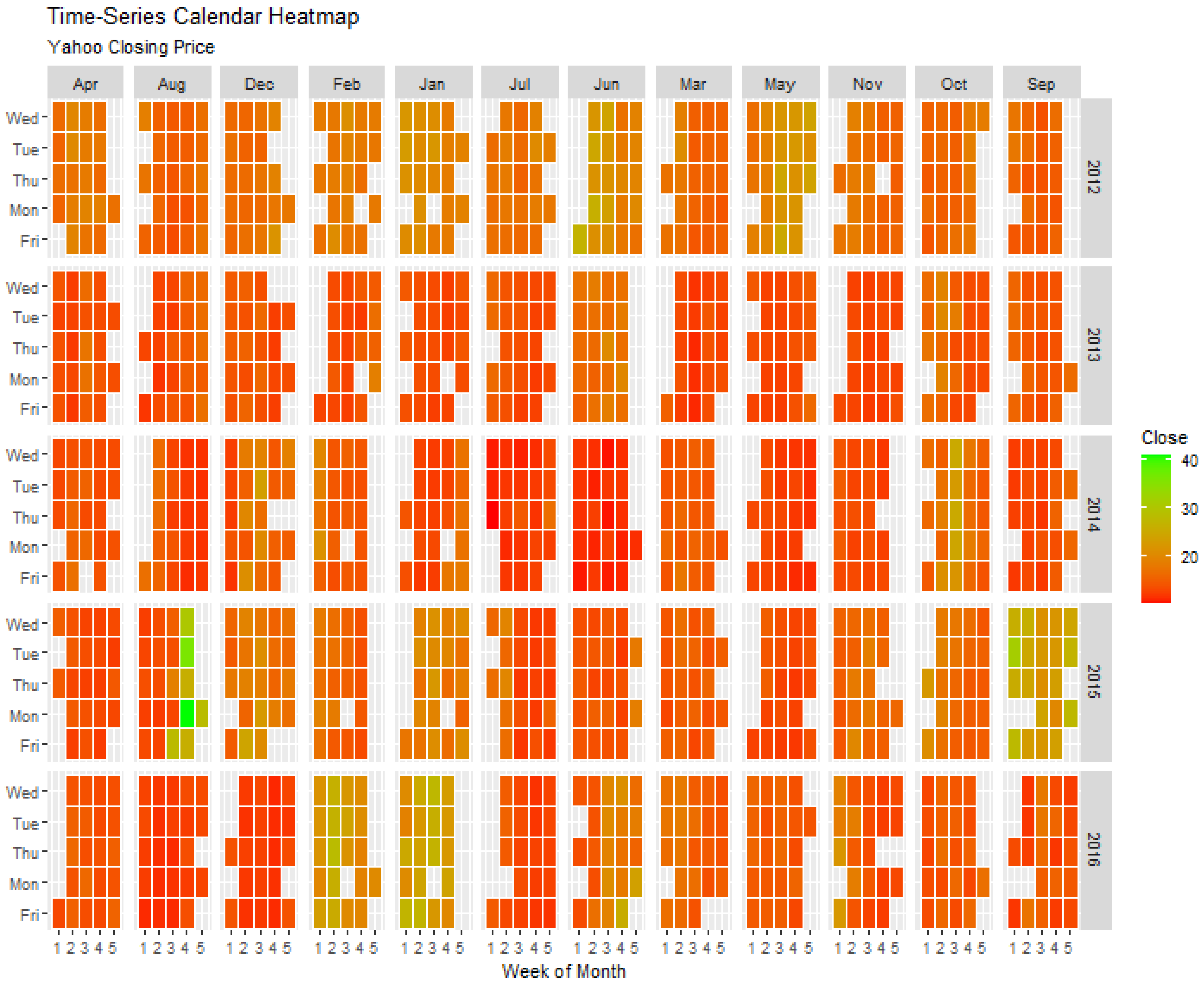
Time Series of Returns Percentage

Drawn from Long Data format

Source: Economics

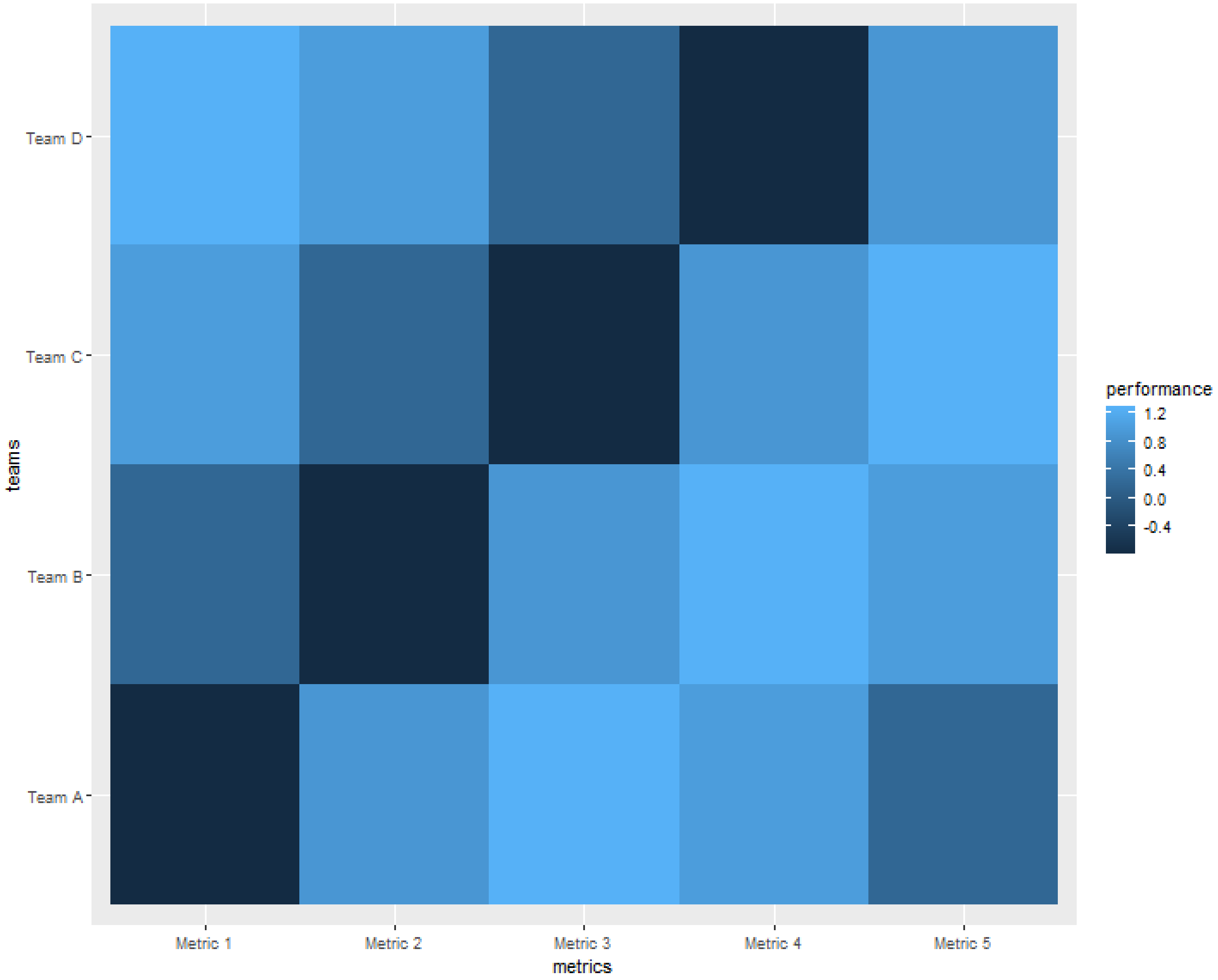
Change, calendar map

```r
# http://margintale.blogspot.in/2012/04/ggplot2-time-series-heatmaps.html
library(ggplot2)
library(plyr)
library(scales)
library(zoo)

df <- read.csv("https://raw.githubusercontent.com/selva86/datasets/master/yahoo.csv")
df$date <- as.Date(df$date)  # format date
df <- df[df$year >= 2012, ]  # filter reqd years

# Create Month Week
df$yearmonth <- as.yearmon(df$date)
df$yearmonthf <- factor(df$yearmonth)
df <- ddply(df,.(yearmonthf), transform, monthweek=1+week-min(week))  # compute week number of month
df <- df[, c("year", "yearmonthf", "monthf", "week", "monthweek", "weekdayf", "VIX.Close")]

# Plot
ggplot(df, aes(monthweek, weekdayf, fill = VIX.Close)) +
  geom_tile(colour = "white") +
  facet_grid(year~monthf) +
  scale_fill_gradient(low="red", high="green") +
  labs(x="Week of Month",
       y="",
       title = "Time-Series Calendar Heatmap",
```


Time-Series Calendar Heatmap
Yahoo Closing Price
Apr
Aug
Dec
Feb
Jan
Jul
Jun
Mar
May
Nov
Oct
Sep
Wed
Tue
Thu
Mon
Fri
2012
2013
2014
2015
2016
Close
40
30
20
1 2 3 4 5
Week of Month

Change, heat map

```
#Heat map
set.seed(41)
expand.grid(
  teams = c("Team A", "Team B", "Team C", "Team D")
  ,
  metrics = c("Metric 1", "Metric 2", "Metric 3", "Metric 4", "Metric 5")
) %>%
  mutate(performance = rnorm(5)) %>% # add variable: performance
  ggplot(aes(x = metrics, y = teams)) + geom_tile(aes(fill = performance))
```


Team D
Team C
Team B
Team A
teams
Metric 1
Metric 2
Metric 3
Metric 4
Metric 5
metrics
performance
1.2
0.8
0.4
0.0
-0.4

Change, seasonal plot

```r
library(ggplot2)
library(forecast)


# Subset data
window(nottem, start=c(1920, 1), end=c(1925, 12)) %>%   # subset a smaller timewindow
ggseasonplot() + labs(title="Seasonal plot: Air temperatures at Nottingham Castle")
```

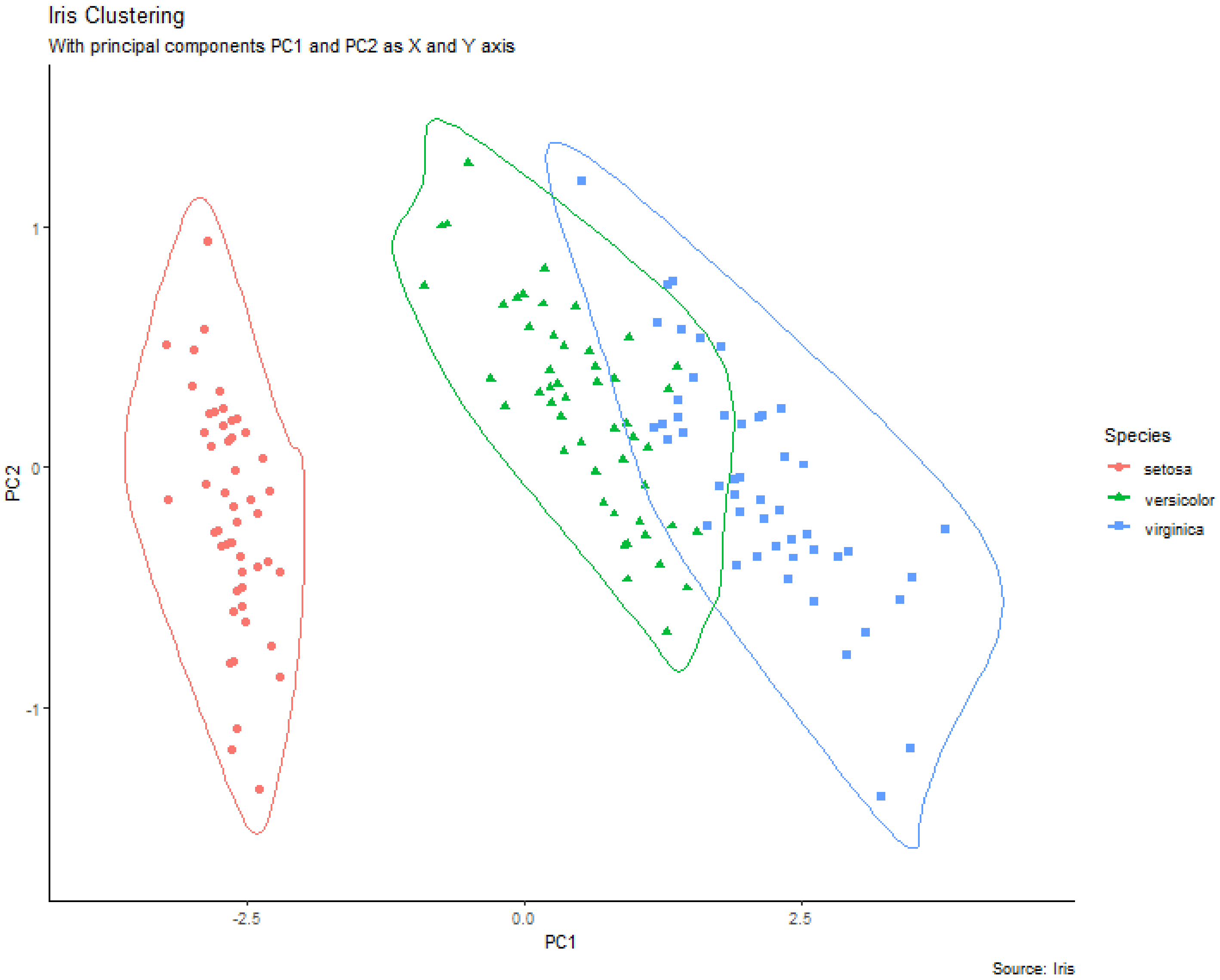
```
## Error in library(forecast): there is no package called 'forecast'
```

```
## Error in ggseasonplot(.): could not find function "ggseasonplot"
```

Advanced Visualization

Static, cluster

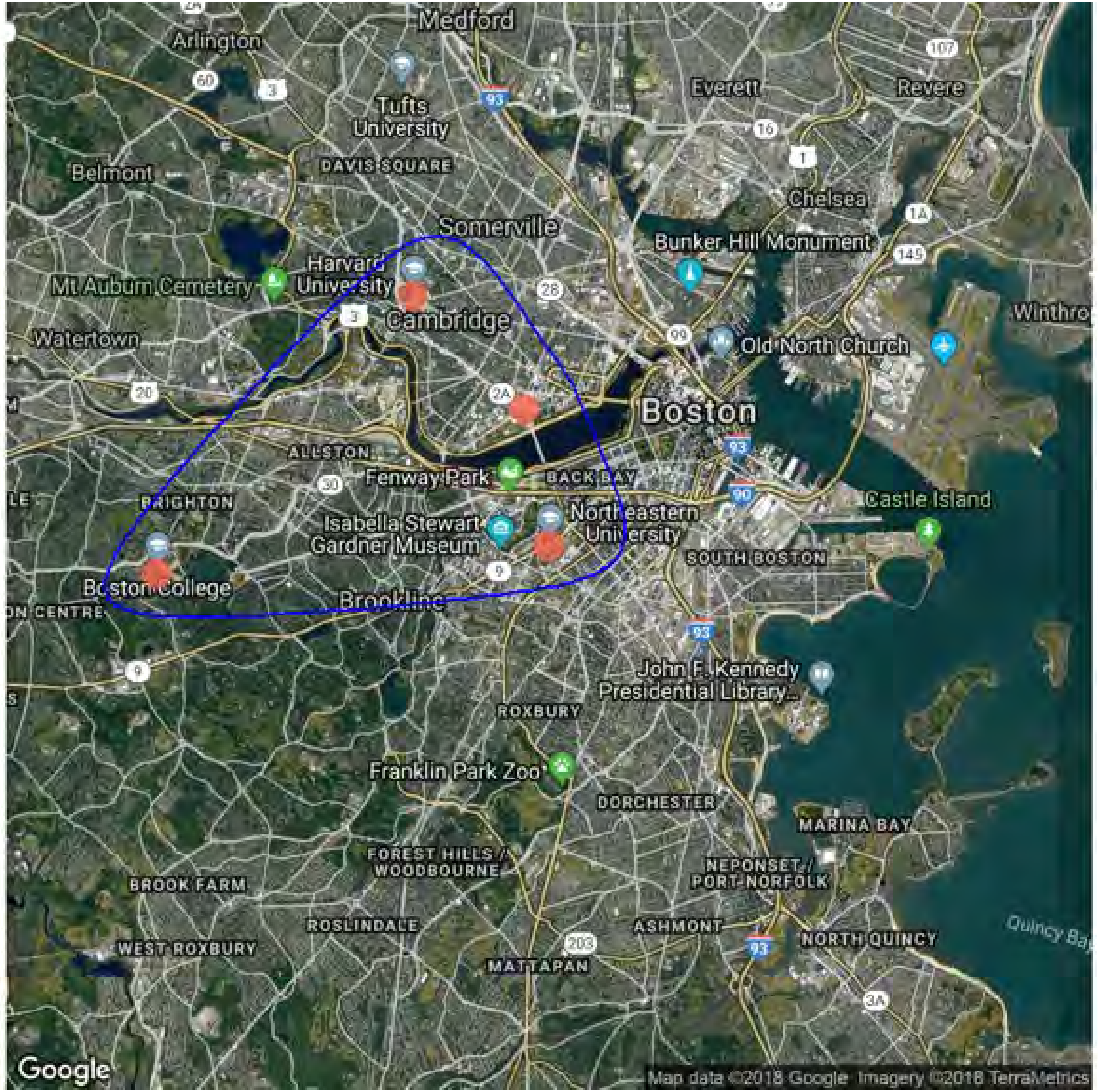
```r
library(ggplot2)
library(ggalt)
library(ggfortify)
theme_set(theme_classic())
# Compute data with principal components
df <- iris[c(1, 2, 3, 4)]
pca_mod <- prcomp(df)  # compute principal components

# Data frame of principal components
df_pc <- data.frame(pca_mod$x, Species=iris$Species)  # dataframe of principal components
df_pc_vir <- df_pc[df_pc$Species == "virginica", ]  # df for 'virginica'
df_pc_set <- df_pc[df_pc$Species == "setosa", ]  # df for 'setosa'
df_pc_ver <- df_pc[df_pc$Species == "versicolor", ]  # df for 'versicolor'

# Plot
ggplot(df_pc, aes(PC1, PC2, col=Species)) +
  geom_point(aes(shape=Species), size=2) +   # draw points
  labs(title="Iris Clustering",
       subtitle="With principal components PC1 and PC2 as X and Y axis",
       caption="Source: Iris") +
  coord_cartesian(xlim = 1.2 * c(min(df_pc$PC1), max(df_pc$PC1)),
                  ylim = 1.2 * c(min(df_pc$PC2), max(df_pc$PC2))) +   # change axis limits
  geom_encircle(data = df_pc_vir, aes(x=PC1, y=PC2)) +   # draw circles
  geom_encircle(data = df_pc_set, aes(x=PC1, y=PC2)) +
## Error in library(ggfortify): there is no package called 'ggfortify'
```



Static, spatial

google needs api key for retrieving the map now Use ?reigister_google for more information

```
library(ggplot2)
library(ggmap)
library(ggalt)

# Google Hybrid Map
neu_ggl_hybrid_map <-
  qmap("neu", zoom = 12,
       source = "google",
       maptype = "hybrid")
neu_places <- c("Northeastern University, MA",
                "MIT",
                "Harvard University",
                "Boston College, MA")
places_loc <- geocode(neu_places)
# Google Hybrid Map
neu_ggl_hybrid_map + geom_point(
  aes(x = lon, y = lat),
  data = places_loc,
  alpha = 0.7,
  size = 7,
  color = "tomato"
) +
```


Medford
Arlington
Tufts University
Everett
Revere
Belmont
DAVIS SQUARE
Chelsea
Somerville
Bunker Hill Monument
Harvard University
Mt Auburn Cemetery
Cambridge
Winthrop
Watertown
Old North Church
Boston
ALLSTON
Fenway Park
BACK BAY
Castle Island
BRIGHTON
Isabella Stewart Gardner Museum
Northeastern University
SOUTH BOSTON
Boston College
Brookline
ON CENTRE
John F. Kennedy Presidential Library...
ROXBURY
Franklin Park Zoo
DORCHESTER
MARINA BAY
FOREST HILLS / WOODBOURNE
NEPONSET / PORT NORFOLK
BROOK FARM
ROSLINDALE
ASHMONT
NORTH QUINCY
Quincy Bay
WEST ROXBURY
MATTAPAN
Google
Map data ©2018 Google Imagery ©2018 TerraMetrics

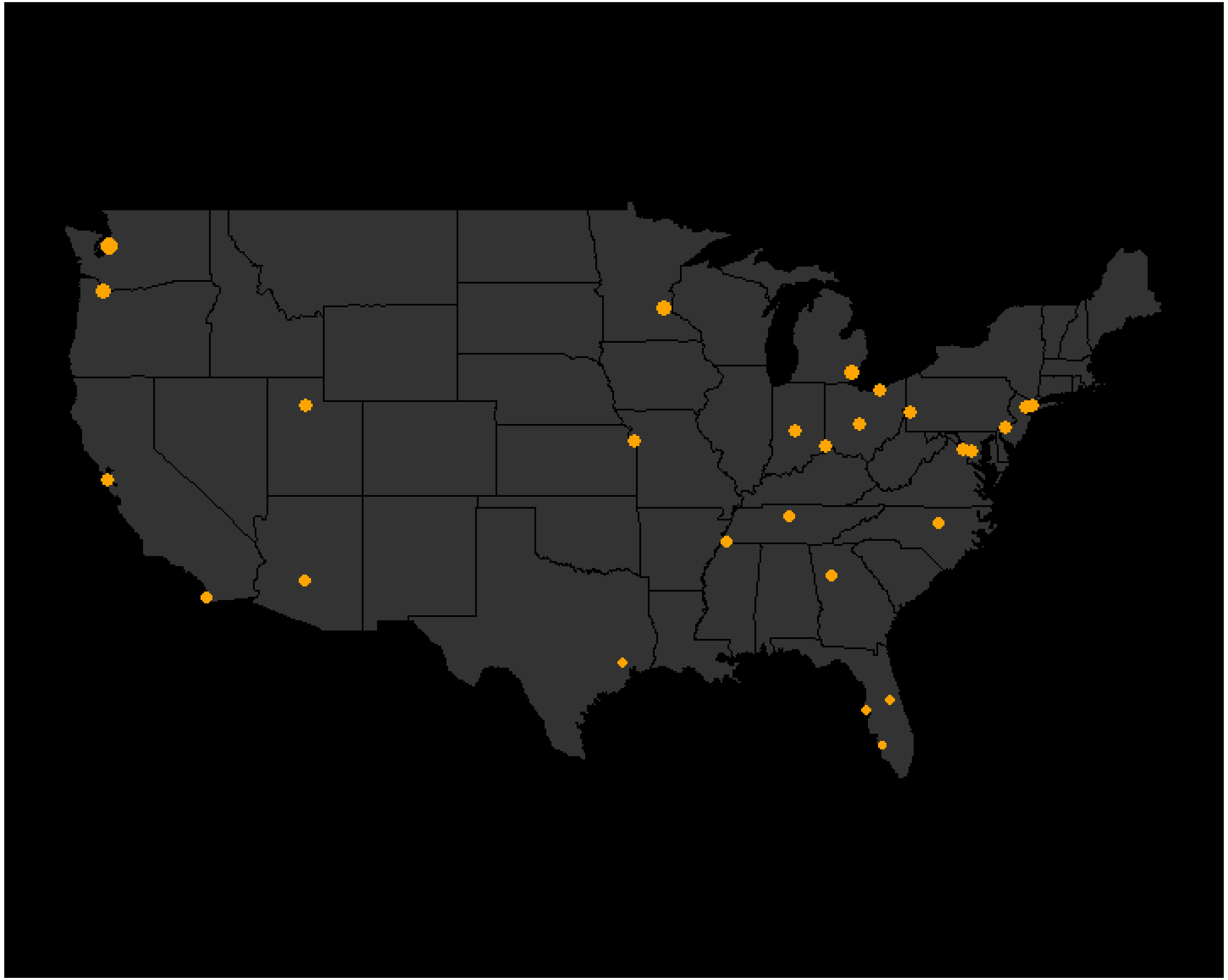
Static, network

```
#Network Map
library(maps)
library(geosphere)
airports <- read.csv("E:/IE6600/IE6600_SEA_Spring2021/R/R/Network Data Sets/Dataset3-Airlines-NODES.csv",
                     header=TRUE)
flights <- read.csv("E:/IE6600/IE6600_SEA_Spring2021/R/R/Network Data Sets/Dataset3-Airlines-EDGES.csv",
                    header=TRUE, as.is=TRUE)
# Select only large airports: ones with more than 10 connections in the data.
tab <- table(flights$Source)
big.id <- names(tab)[tab>10]
airports <- airports[airports$ID %in% big.id,]
flights  <- flights[flights$Source %in% big.id &
                      flights$Target %in% big.id, ]

# Plot a map of the united states:
map("state", col="grey20", fill=TRUE, bg="black", lwd=0.1)
```

Static, network (cont'd)

```r
# Add a point on the map for each airport:
points(x=airports$longitude, y=airports$latitude, pch=19,
       cex=airports$Visits/80, col="orange")
col.1 <- adjustcolor("orange red", alpha=0.4)
col.2 <- adjustcolor("orange", alpha=0.4)
edge.pal <- colorRampPalette(c(col.1, col.2), alpha = TRUE)
edge.col <- edge.pal(100)
for(i in 1:nrow(flights))  {
  node1 <- airports[airports$ID == flights[i,]$Source,]
  node2 <- airports[airports$ID == flights[i,]$Target,]

  arc <- gcIntermediate( c(node1[1,]$longitude, node1[1,]$latitude),
                         c(node2[1,]$longitude, node2[1,]$latitude),
                         n=1000, addStartEnd=TRUE )
  edge.ind <- round(100*flights[i,]$Freq / max(flights$Freq))

  lines(arc, col=edge.col[edge.ind], lwd=edge.ind/30)
}
```

```
## Error in library(geosphere): there is no package called 'geosphere'
```

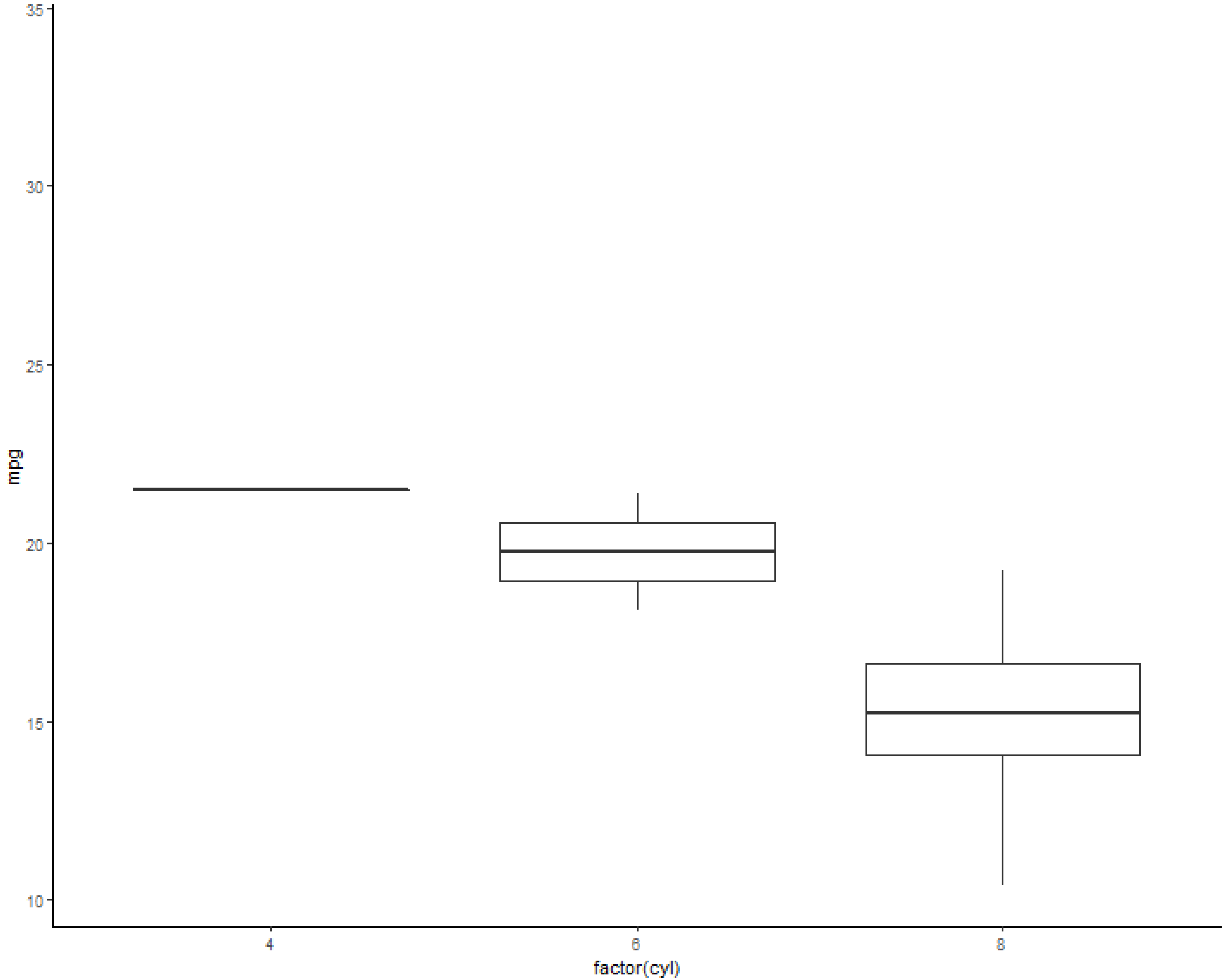


```
## Error in gcIntermediate(c(node1[1, ]$longitude, node1[1, ]$latitude), : could not find function "gcIntermediate"
```

Animated, Box plot

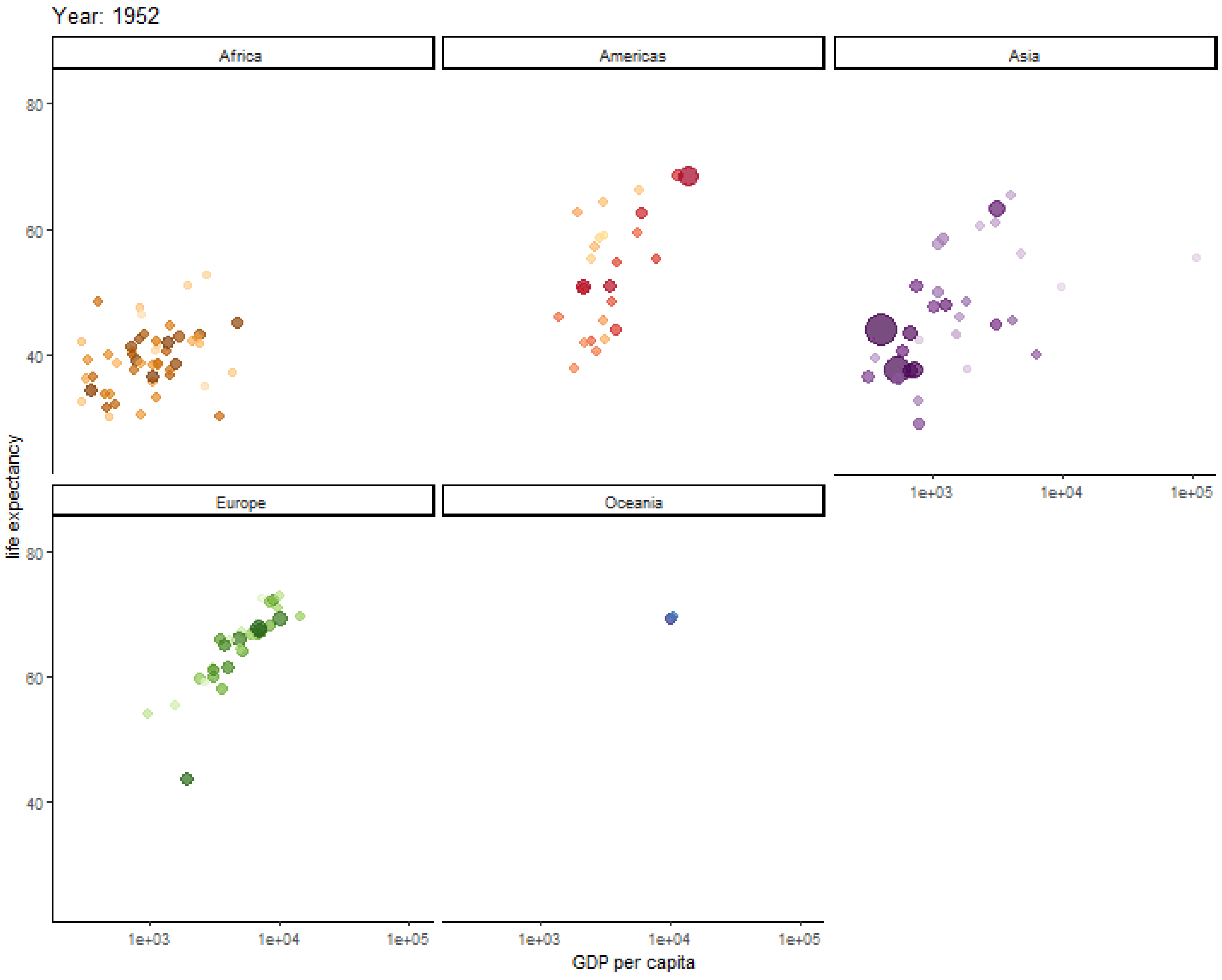
```
library(ggplot2)
library(gganimate)
library(gifski)
ggplot(mtcars, aes(factor(cyl), mpg)) +
  geom_boxplot() +
  # Here comes the gganimate code
  transition_states(
    gear,
    transition_length = 2,
    state_length = 1
  ) +
  enter_fade() +
  exit_shrink() +
  ease_aes('sine-in-out')
```

```
## Error in library(gifski): there is no package called 'gifski'
```



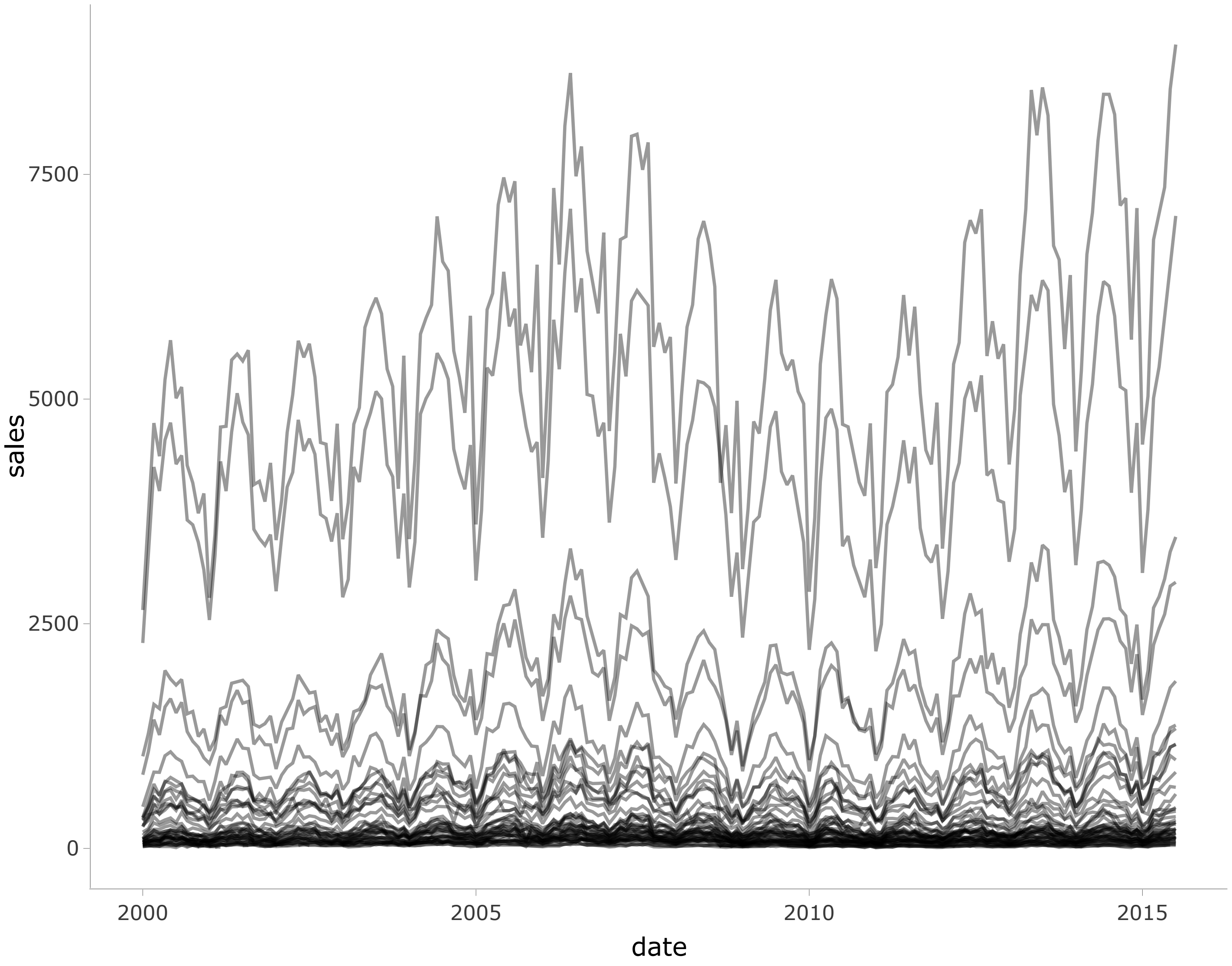
Animated, Dot Plot

```r
library(gapminder)

ggplot(gapminder, aes(gdpPercap, lifeExp, size = pop, colour = country)) +
  geom_point(alpha = 0.7, show.legend = FALSE) +
  scale_colour_manual(values = country_colors) +
  scale_size(range = c(2, 12)) +
  scale_x_log10() +
  facet_wrap(~continent) +
  # Here comes the gganimate specific bits
  labs(title = 'Year: {frame_time}', x = 'GDP per capita', y = 'life expectancy') +
  transition_time(year) +
  ease_aes('linear')
```


Year: 1952
Africa
Americas
Asia
Europe
Oceania
80
60
40
80
60
40
life expectancy
1e+03
1e+04
1e+05
1e+03
1e+04
1e+05
1e+03
1e+04
1e+05
GDP per capita

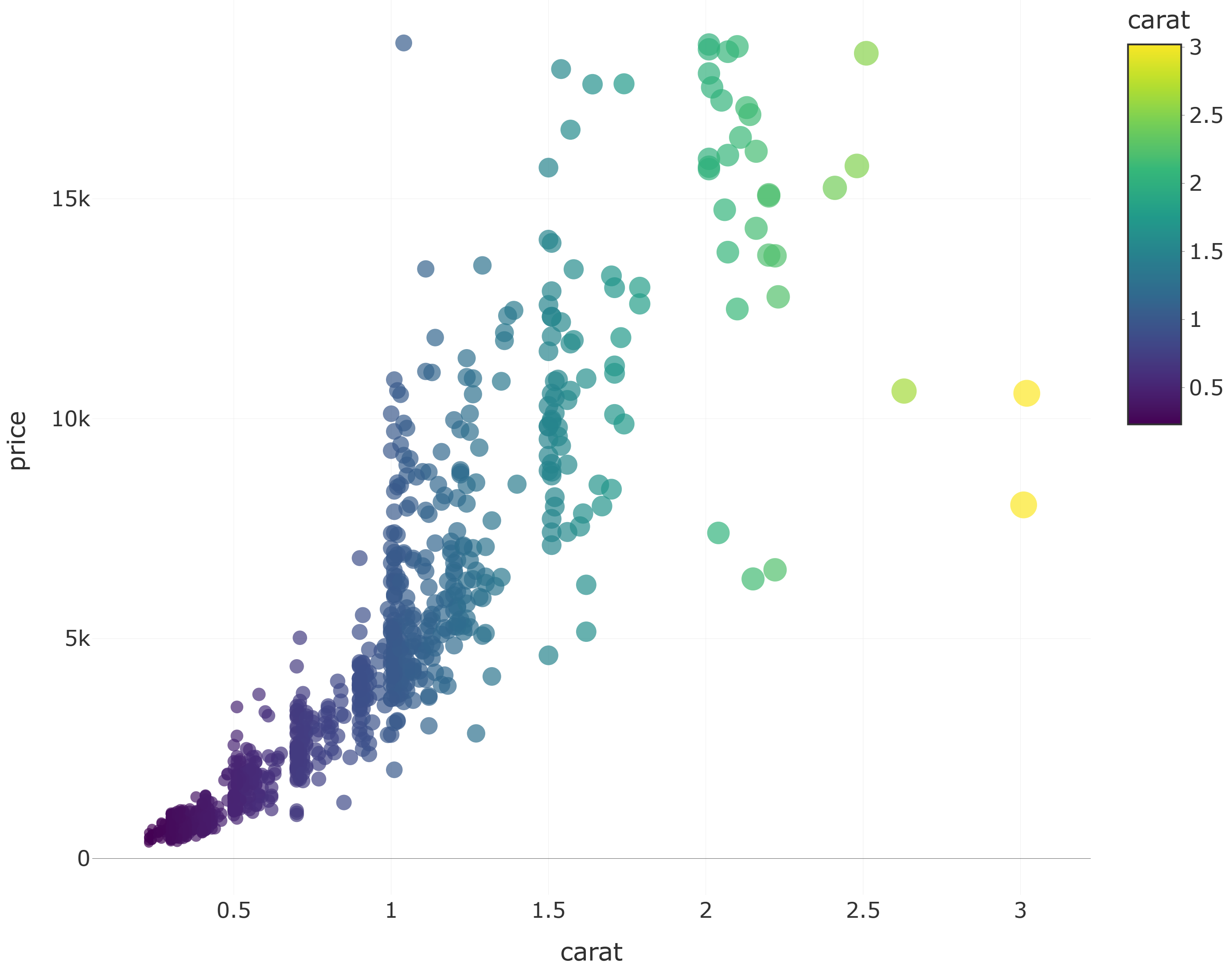
Interactive plot with ggplotlly()

```r
library(plotly)
g <- ggplot(txhousing, aes(x = date, y = sales, group = city)) +
  geom_line(alpha = 0.4)
ggplotly(g, tooltip = c("city"))
```


7500
5000
2500
0
sales
2000
2005
2010
2015
date

Interactive plot, plotly

```r
library(plotly)
diamonds[sample(nrow(diamonds), 1000), ] %>%
  plot_ly(
  x = ~carat, y = ~price,
  color = ~carat, size = ~carat
)
```


carat
3
2.5
2
1.5
1
0.5
price
15k
10k
5k
0
0.5
1
1.5
2
2.5
3
carat

Interactive plot, D3

```r
library(networkD3)
forceNetwork(Links = MisLinks, Nodes = MisNodes, Source = "source",
             Target = "target", Value = "value", NodeID = "name",
             Group = "group", opacity = 0.9, Nodesize = 3,
             linkDistance = 100, fontSize = 20)
```

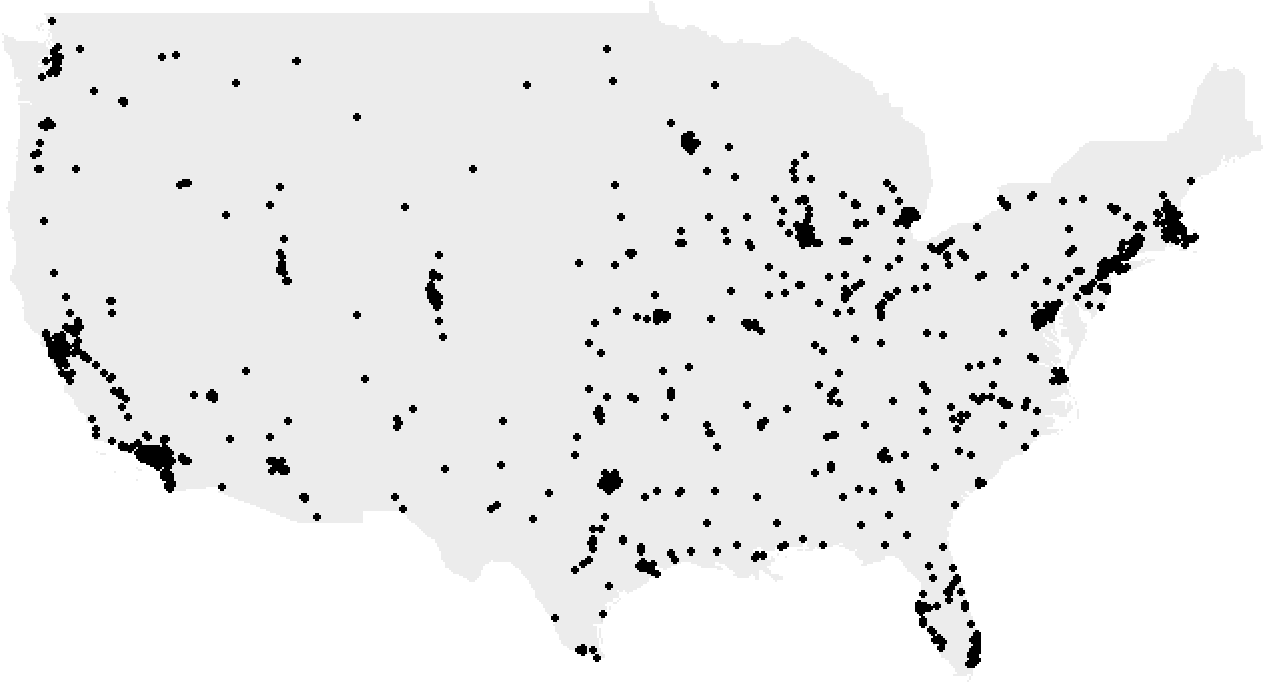
```
## Error in library(networkD3): there is no package called 'networkD3'
```

```
## Error in forceNetwork(Links = MisLinks, Nodes = MisNodes, Source = "source", : could not find function "forceNetw
```

Applicatin in real life

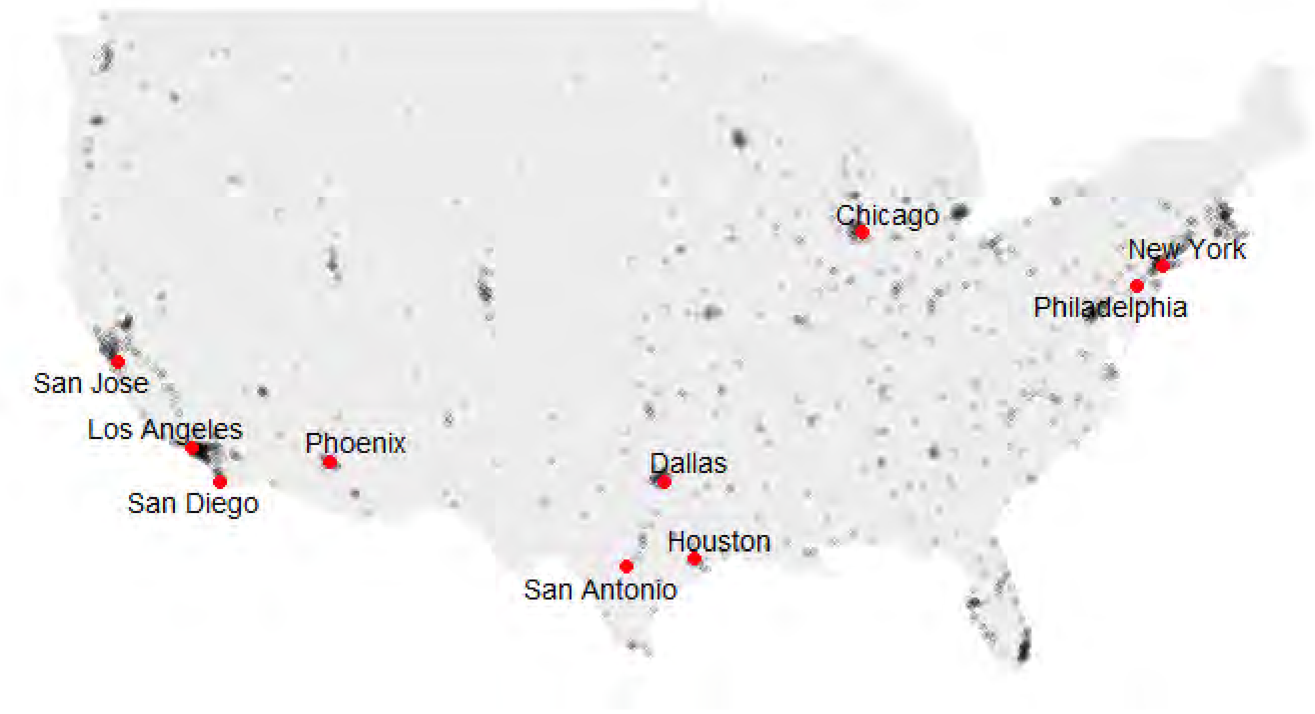
GB

```r
# Libraries
library(tidyverse)

# Get the world polygon and extract USA
library(maps)
USA <- map_data("world") %>% filter(region=="USA")
# Get a data frame with longitude, latitude, and size of bubbles (a bubble = a city)
data=world.cities %>% filter(country.etc=="USA")

# Left chart
ggplot() +
  geom_polygon(data = USA, aes(x=long, y = lat, group = group), fill="grey", alpha=0.3) +
  geom_point( data=data, aes(x=long, y=lat)) +
  theme_void() + ylim(20,56) + xlim(-125,-65)+coord_map()
```



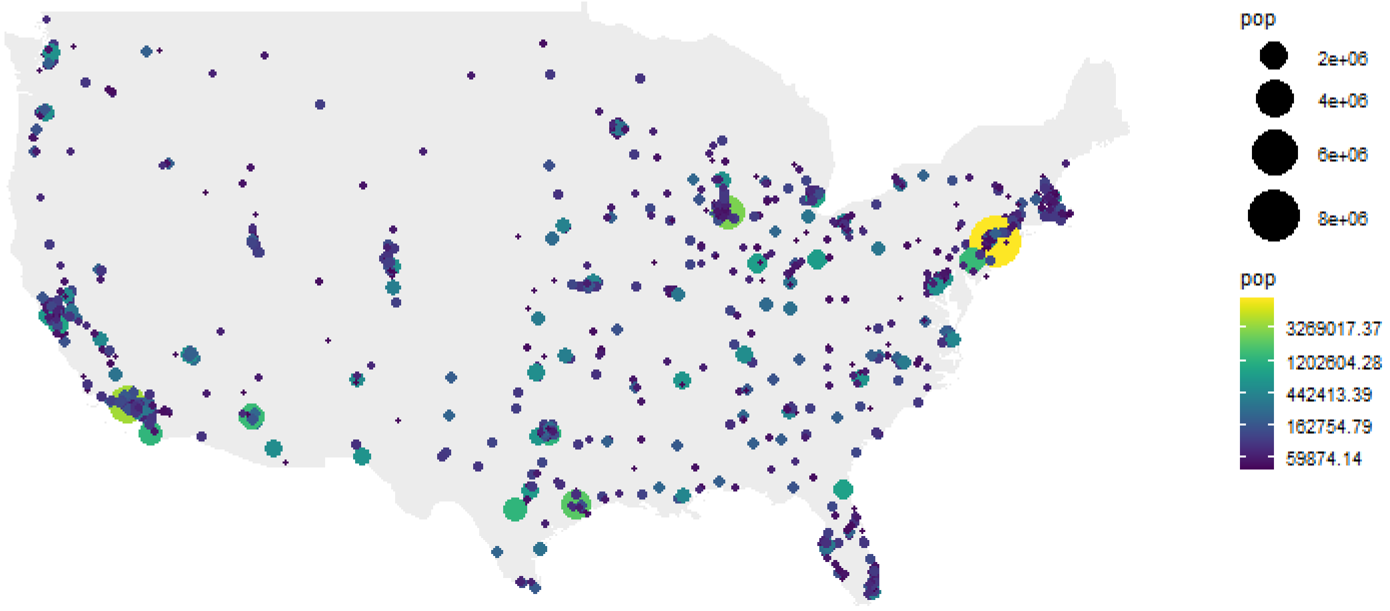
GB

```r
# Second graphic with names of the 10 biggest cities
library(ggrepel)
ggplot() +
  geom_polygon(data = USA, aes(x=long, y = lat, group = group), fill="grey", alpha=0.3) +
  geom_point( data=data, aes(x=long, y=lat, alpha=pop)) +
  geom_text_repel(data=data %>% arrange(pop) %>% tail(10), aes(x=long, y=lat, label=name), size=5) +
  geom_point( data=data %>% arrange(pop) %>% tail(10), aes(x=long, y=lat), color="red", size=3) +
  theme_void() + ylim(20,56) + xlim(-125,-65)+coord_map()  +
  theme(legend.position="none")
```

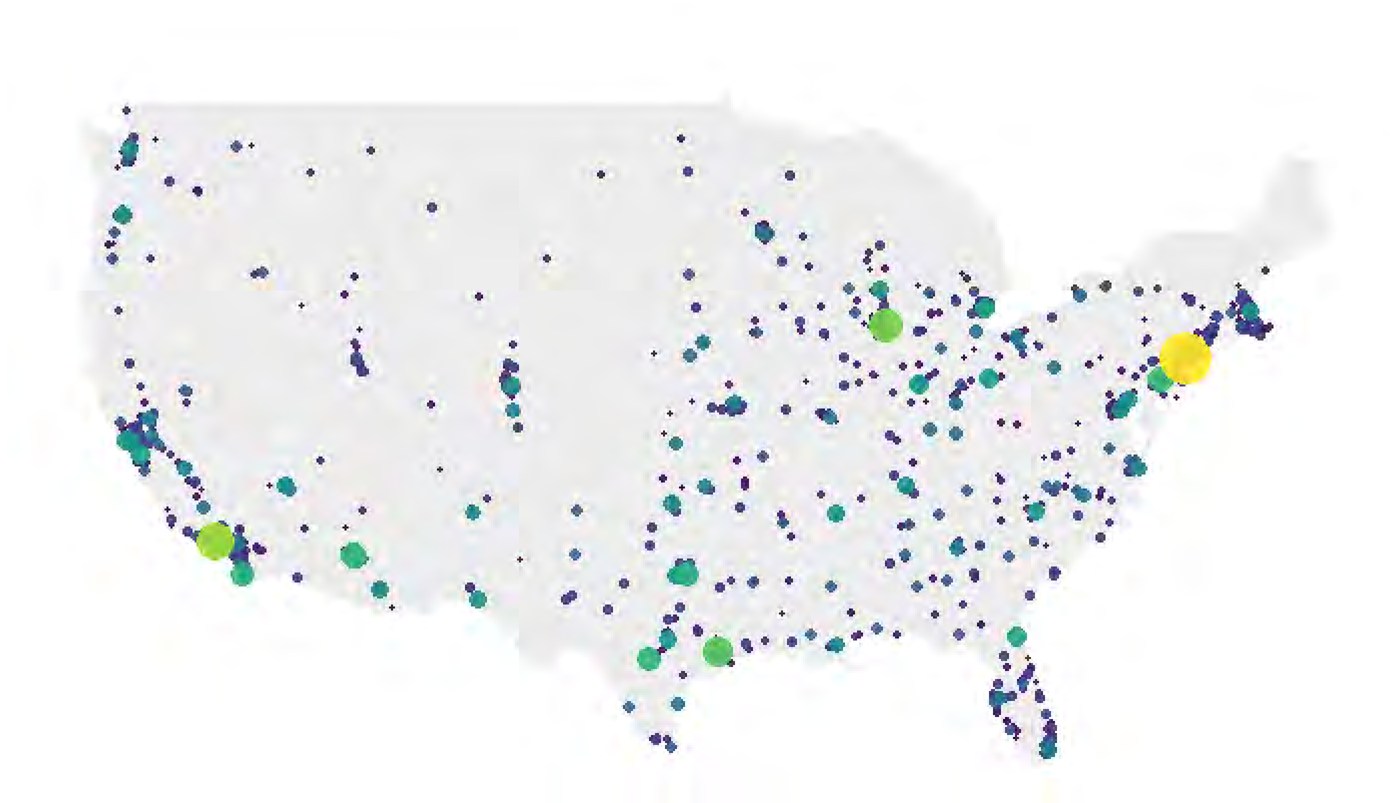

Chicago
New York
Philadelphia
San Jose
Los Angeles
Phoenix
San Diego
Dallas
Houston
San Antonio

GB

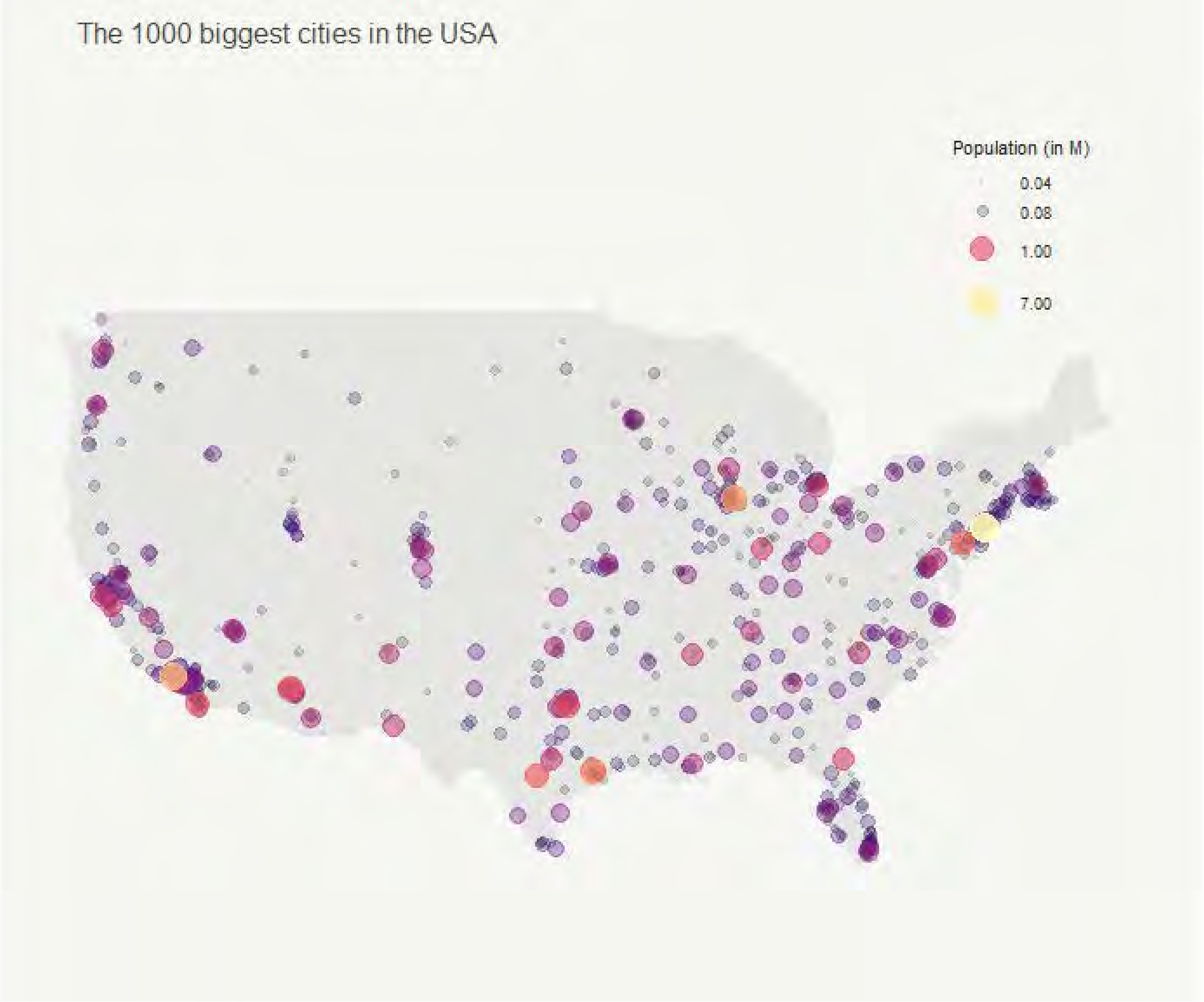
```r
library(viridis)
# first: use size and color
ggplot() +
  geom_polygon(data = USA, aes(x=long, y = lat, group = group), fill="grey", alpha=0.3) +
  geom_point( data=data, aes(x=long, y=lat, size=pop, color=pop)) +
  scale_size_continuous(range=c(1,12)) +
  scale_color_viridis(trans="log") +
  theme_void() + ylim(20,56) + xlim(-125,-65)+coord_map()
```


pop
2e+06
4e+06
6e+06
8e+06
pop
3269017.37
1202604.28
442413.39
162754.79
59874.14

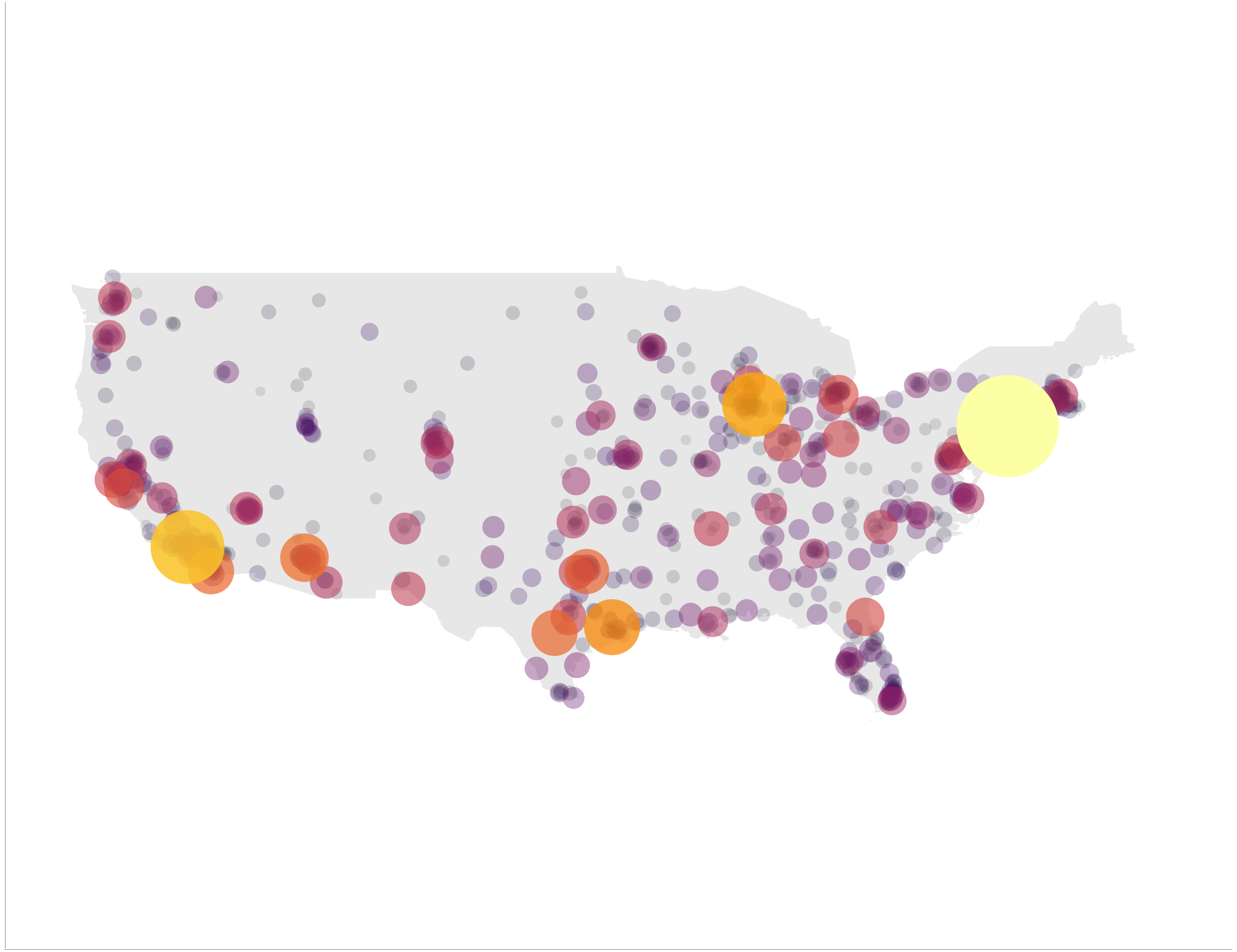
```
# second: reorder your dataset first! Big cities appear later = on top
data %>%
  arrange(pop) %>% # This reorder your data frame
  mutate( name=factor(name, unique(name))) %>% #this reorder the order of the levels of your factor --> this
  ggplot() +
  geom_polygon(data = USA, aes(x=long, y = lat, group = group), fill="grey", alpha=0.3) +
  geom_point( aes(x=long, y=lat, size=pop, color=pop), alpha=0.9) +
  scale_size_continuous(range=c(1,12)) +
  scale_color_viridis(trans="log") +
  theme_void() + ylim(20,56) + xlim(-125,-65)+coord_map()  + theme(legend.position="none")
```



```r
library(viridis)
mybreaks=c(0.02, 0.04, 0.08, 1, 7)
data %>%
  arrange(pop) %>%
  mutate( name=factor(name, unique(name))) %>%
  mutate(pop=pop/1000000) %>%
  ggplot() +
  geom_polygon(data =USA, aes(x=long, y = lat, group = group), fill="grey", alpha=0.3) +
  geom_point(  aes(x=long, y=lat, size=pop, color=pop, alpha=pop), shape=20, stroke=FALSE) +
  scale_size_continuous(name="Population (in M)", trans="log", range=c(1,12), breaks=mybreaks) +
  scale_alpha_continuous(name="Population (in M)", trans="log", range=c(0.1, .9), breaks=mybreaks) +
  scale_color_viridis(option="magma", trans="log", breaks=mybreaks, name="Population (in M)" ) +
   theme_void() + ylim(20,56) + xlim(-125,-65)+coord_map() +
  guides( colour = guide_legend()) +
  ggtitle("The 1000 biggest cities in the USA") +
  theme(
    legend.position = c(0.85, 0.8),
    text = element_text(color = "#22211d"),
    plot.background = element_rect(fill = "#f5f5f2", color = NA),
    panel.background = element_rect(fill = "#f5f5f2", color = NA),
    legend.background = element_rect(fill = "#f5f5f2", color = NA),
    plot.title = element_text(size= 16, hjust=0.1, color = "#4e4d47", margin = margin(b = -0.1, t = 0.4, l =
```


The 1000 biggest cities in the USA
Population (in M)
0.04
0.08
1.00
7.00

```r
# Easy to make it interactive!
library(plotly)

# plot
p=data %>%

  arrange(pop) %>%
  mutate( name=factor(name, unique(name))) %>%
  mutate( mytext=paste("City: ", name, "\n", "Population: ", pop, sep="")) %>% # This prepare the text displa

  # Makte the static plot calling this text:
  ggplot() +
  geom_polygon(data = USA, aes(x=long, y = lat, group = group), fill="grey", alpha=0.3) +
  geom_point(aes(x=long, y=lat, size=pop, color=pop, text=mytext, alpha=pop) ) +

  scale_size_continuous(range=c(1,15)) +
  scale_color_viridis(option="inferno", trans="log" ) +
  scale_alpha_continuous(trans="log") +
  theme_void() +
  ylim(0,100) + xlim(-125,-65) +
  coord_map() +
  theme(legend.position = "none")

ggplotly(p, tooltip="text")
```



References

Wickham, Hadley, and Garrett Grolemund. R For Data Science. OReilly, 2017.

Lu, Z. (2022). Data Visualization Tutorial in R. zhenyuanlu.github.io.

Lander, Jared P.R For Everyone - Advanced Analytics and Graphics. Pearson Education (Us), 2017

"Companies Using R."R-Bloggers, R-Bloggers, 28 Dec. 2016, www.r-bloggers.com/companies-using-r/. Kabacoff, Robert. "Access to

Database Management Systems (DBMS)."

www.statmethods.net/input/dbinterface.html.

www.r-bloggers.com

http://r-statistics.co

www.r-graph-gallery.com

Kabacoff, Robert.R In Action: Data Analysis and Graphics with R. Manning, 2015.