Visualisation with ggplot2

DSA8022

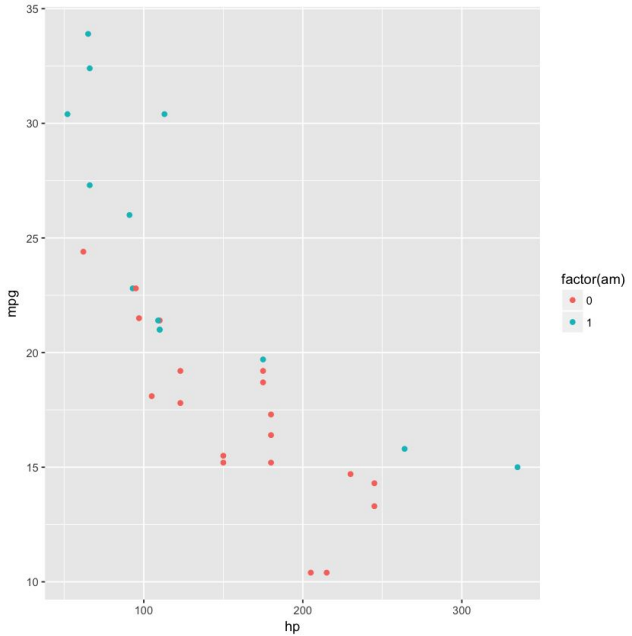
ggplot2 definition

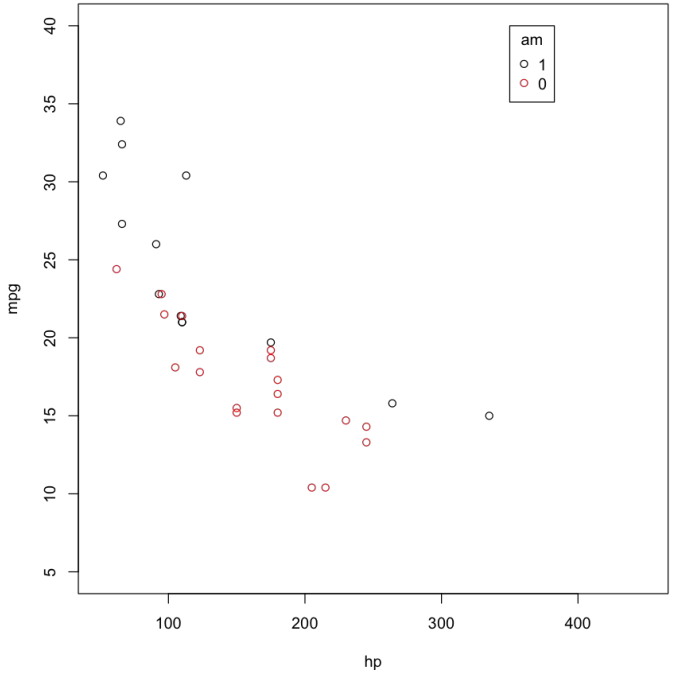
ggplot2 (grammar of graphics) is a data visualization package for R which splits graphs into semantic components such as scales and layers.

Created by Hadley Wickham in 2005.

Why using ggplot2?

- Consistent underlying grammar of graphics (Wilkinson, 2005)
- Plot specification at a high level of abstraction
- Theme system to refine plot appearance
- Very active development





ggplot2 (left) vs basic-graphics (right)

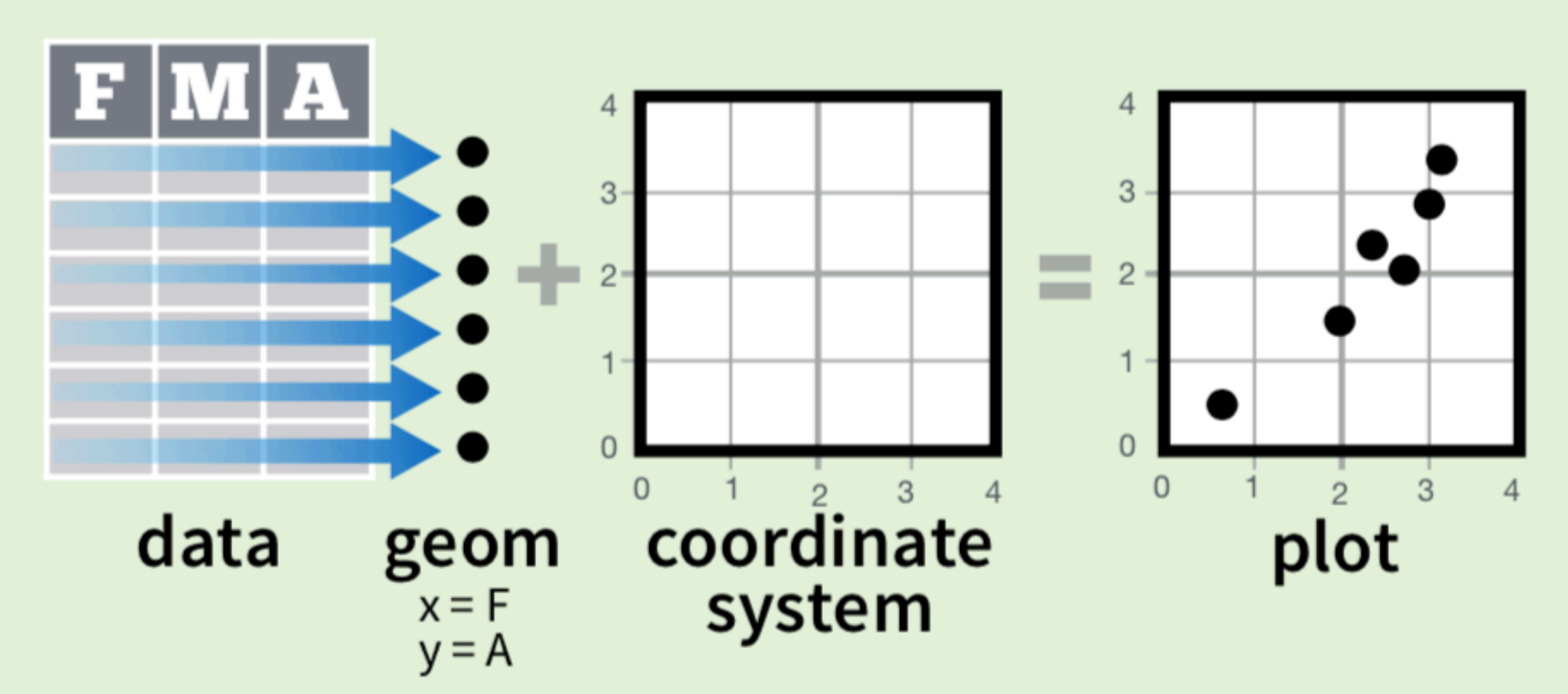
ggplot2 vs base-graphics

```
library(ggplot2)
ggplot(mtcars, aes(x=hp, y=mpg,
color=factor(am))) + geom_point()
```

```
par(mar = c(4,4,.1,.1))

plot(mpg ~ hp,
data=subset(mtcars, am==1),  xlim=
c(50, 450), ylim= c(5, 40))

points(mpg ~ hp, col="red",
data=subset(mtcars, am==0))

legend(350, 40,
c("1", "0"), title= "am",  col=
c("black", "red"),  pch= c(1, 1))
```

Build every graph from:

- A data set
- A coordinate system
- geoms – visual marks representing data points



Concept of ggplot2

Specifying a plot via building blocks:

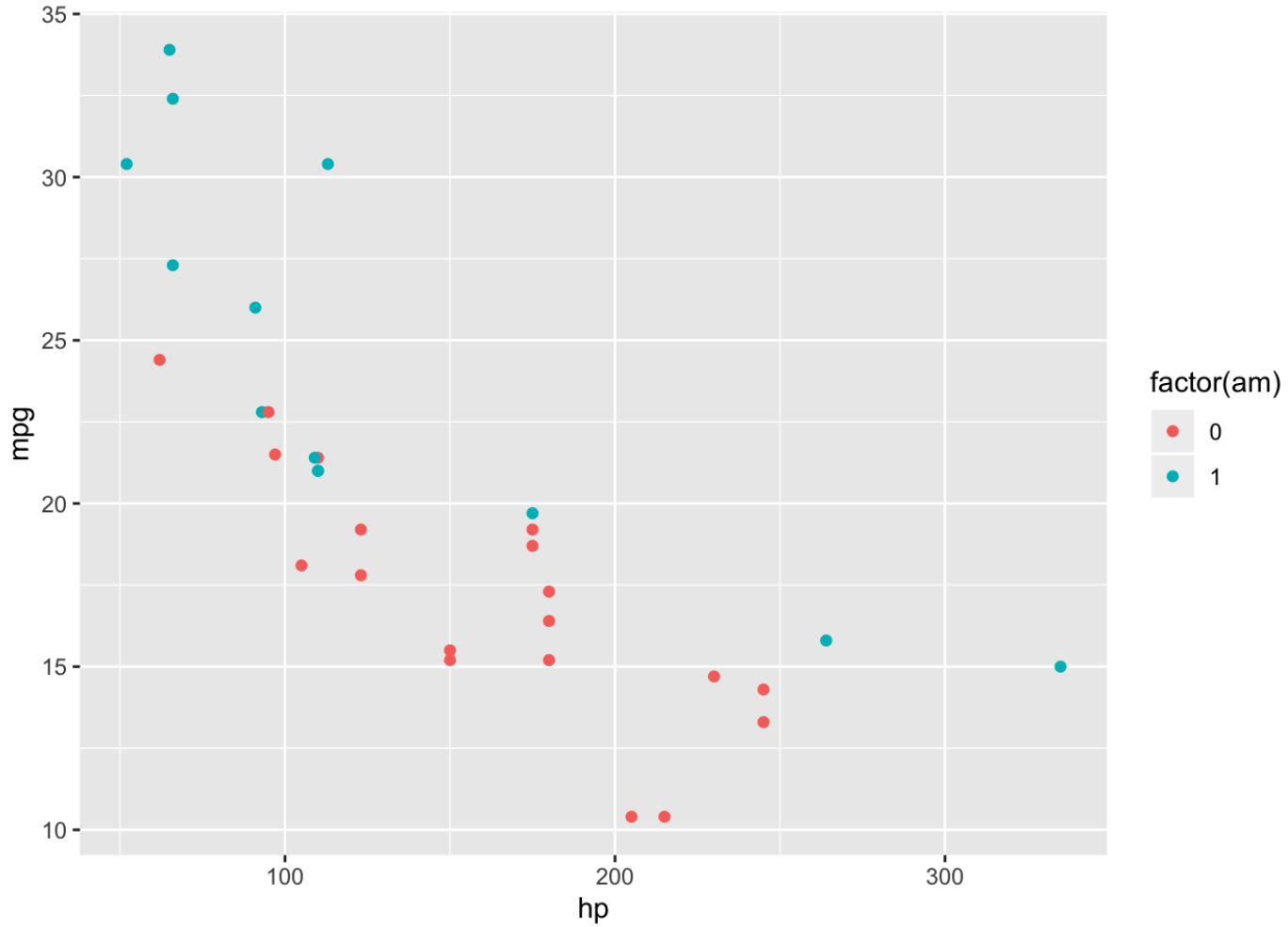
- Data
- Aesthetic mapping
- Geometric object
- Statistical transformations
- Scales
- Coordinate system
- Position adjustments
- Faceting

A plot includes multi layers

- Data
- Mappings between variables and aesthetics
- geometric object
- statistical transformation
- Scales to control the details of the mapping

```
library(ggplot2)
ggplot(mtcars, aes(x=hp,y=mpg,color=factor(am)))+geom_point()
```

- initializes a ggplot object
- declare the input data frame for a graphic
- specify the set of plot aesthetics intended to be common throughout all subsequent layers unless specifically overridden.

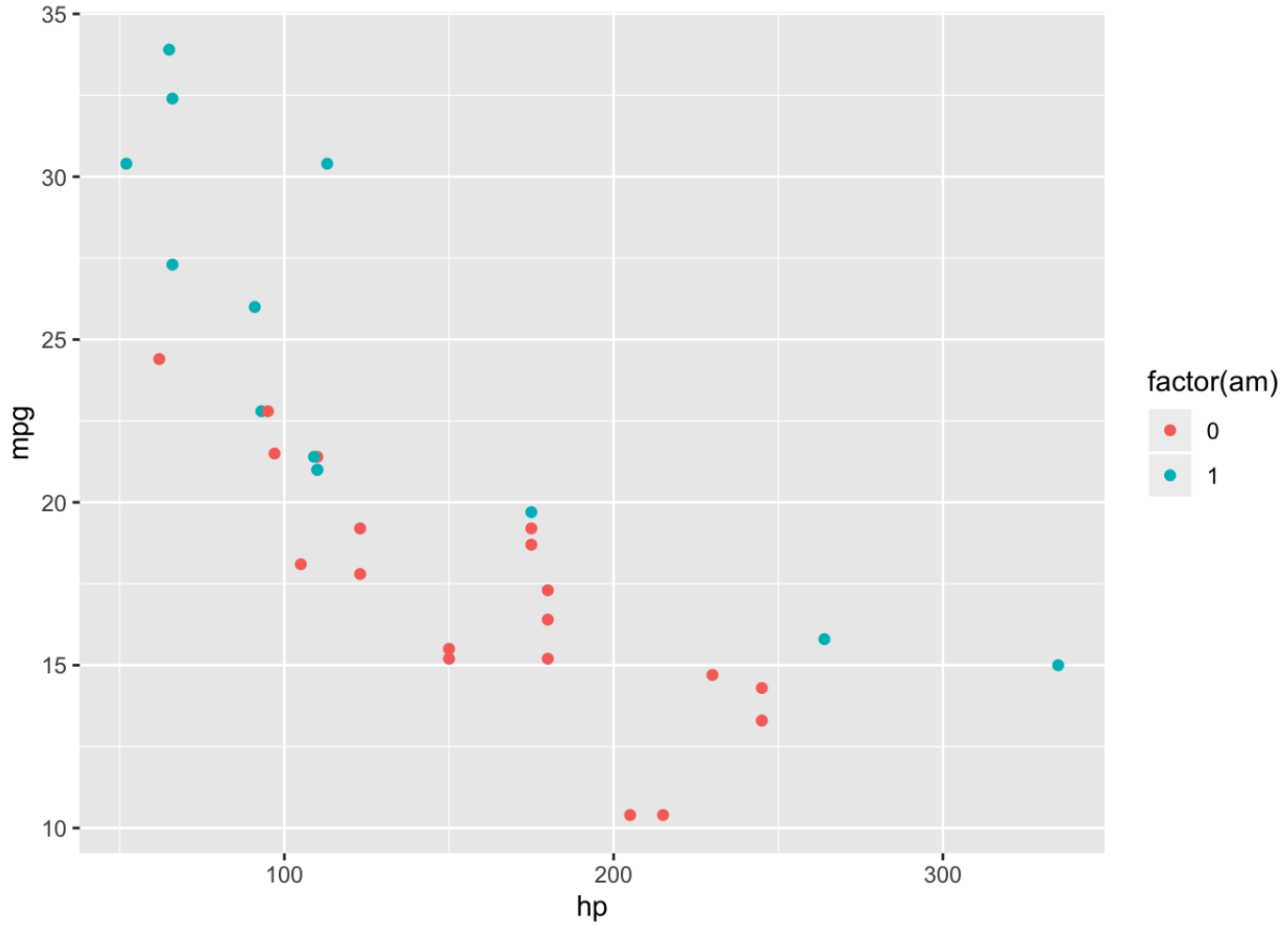


```r
library(ggplot2)
ggplot(mtcars, aes(x=hp,y=mpg,color=factor(am)))+geom_point()
```

```r
geom_point(mapping = NULL, data = NULL, stat = "identity",
  position = "identity", ..., na.rm = FALSE, show.legend = NA,
  inherit.aes = TRUE)
```

- Create scatterplots
- Useful for displaying the relationship between two continuous variables

- **mapping** Set of aesthetic mappings created by aes()
- **data** The data to be displayed in this layer
- **stat** The statistical transformation to use on the data for this layer, as a string.
- ... Other arguments passed on to layer().
- **na.rm** If FALSE, the default, missing values are removed with a warning
- **show.legend** Should this layer be included in the legends?
- **inherit.aes** If FALSE, overrides the default aesthetics, rather than combining with them.



ggplot2 structure

- Aesthetics and geometric Objects
- Statistical Transformations
- Scales
- Faceting
- Themes

Aesthetics and geometric Objects

Aesthetics

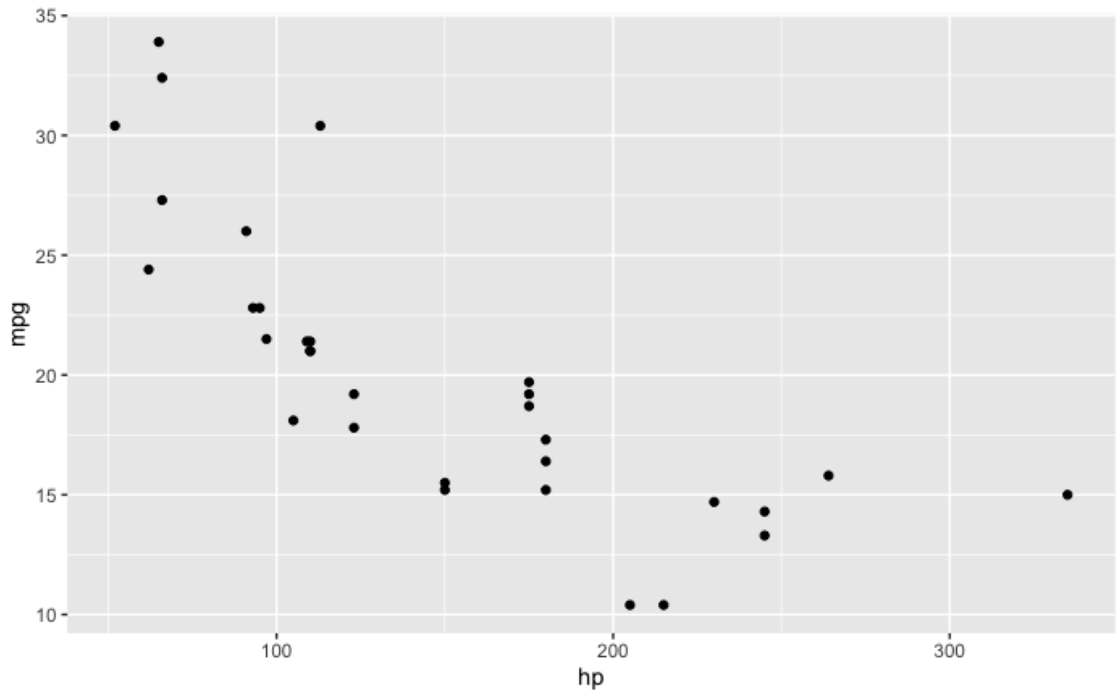
Aesthetic mappings describe how variables in the data are mapped to visual properties (aesthetics) of geoms

- Set with `aes()` function
- Position (x and y)
- Color (outline color)
- Fill
- Shape (point shapes)
- Line type
- Size

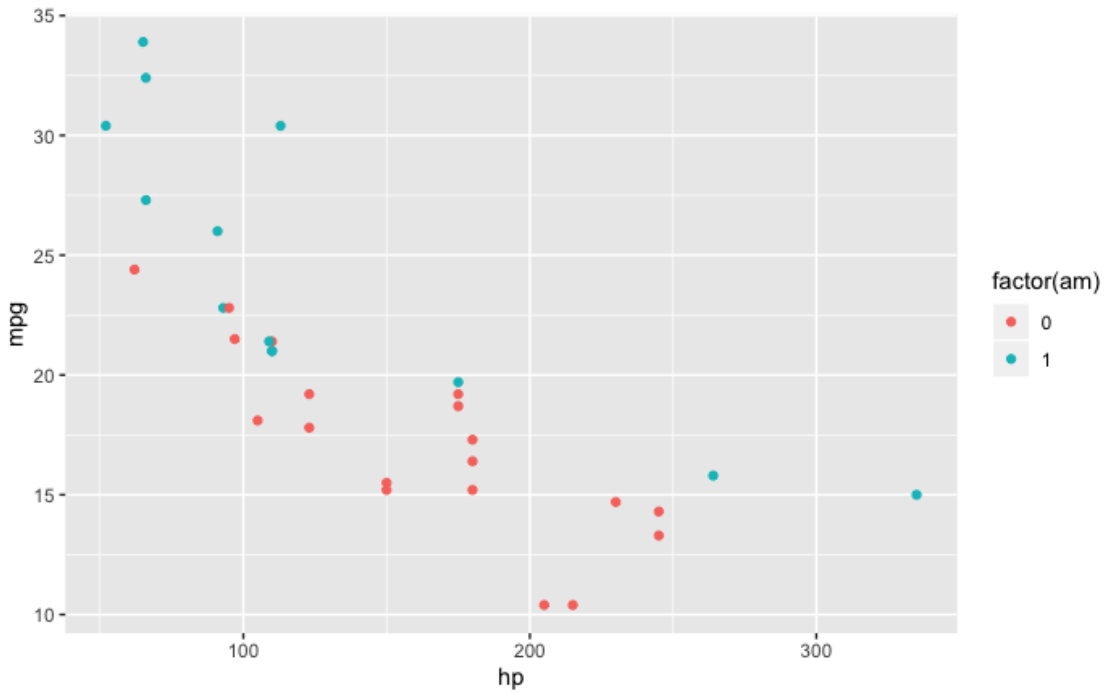
```
aes(x, y, ...)
```

```
ggplot(mtcars, aes(x=hp,y=mpg))
    +geom_point()

ggplot(mtcars, aes(x=hp,y=mpg,color=factor(am)))
    +geom_point()
```

```
ggplot(mtcars, aes(x=hp,y=mpg))
+geom_point()
```

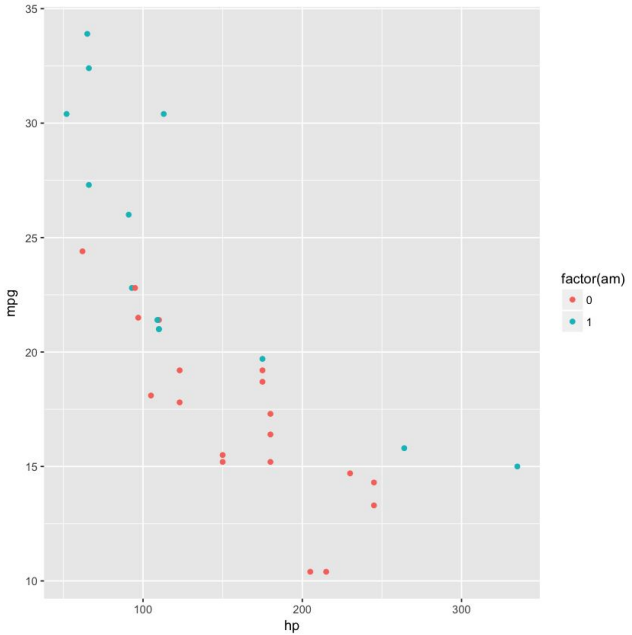



```
ggplot(mtcars, aes(x=hp,y=mpg,color=factor(am)))
+geom_point()
```



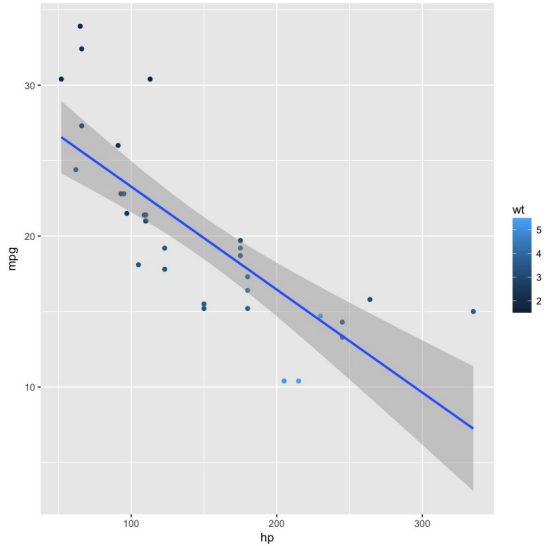
Geometric

- Points (`geom_point`: dot plots, scatter plots)
- Lines (`geom_line`: trend lines, time-series)
- Boxplot (`geom_boxplot`)
- At least one `geom` is required (no upper limit)
- Added to the plot using the **+** operator

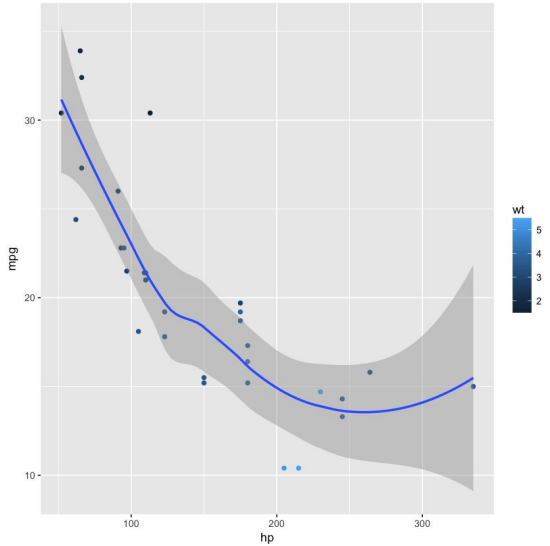


geom_smooth / geom_line

```
ggplot(mtcars, aes(x = hp, y = mpg)) +
geom_point(aes(color = wt)) +
geom_smooth(method = lm)
```

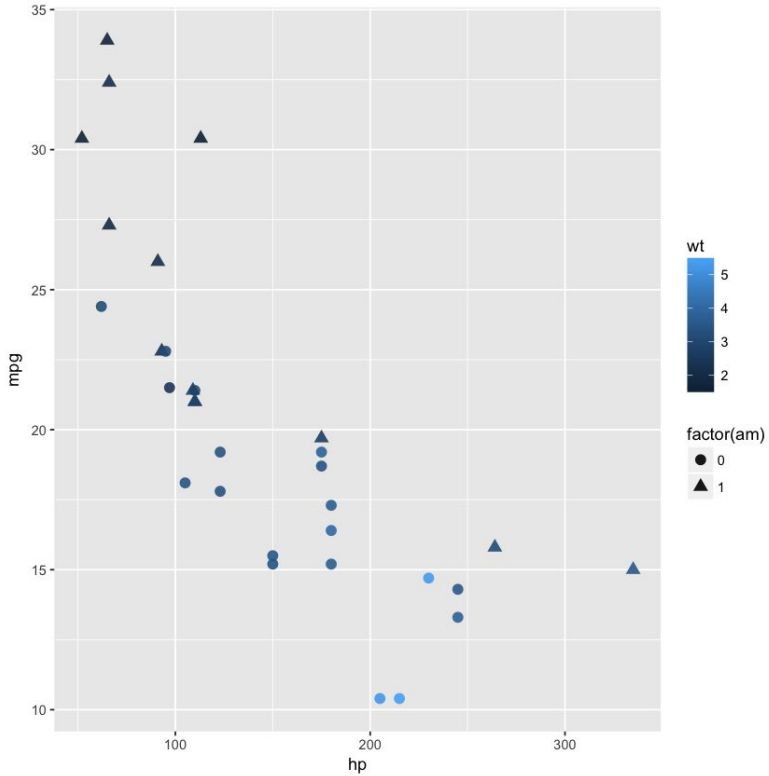



```
ggplot(mtcars, aes(x = hp, y = mpg)) +
geom_point(aes(color = wt)) +
geom_smooth(method = loess)
```



Map variables to aesthetics

```
ggplot(mtcars, aes(x= hp, y= mpg)) +
geom_point(aes(color= wt, shape= factor(am)),
size= 3, alpha= .9)
```



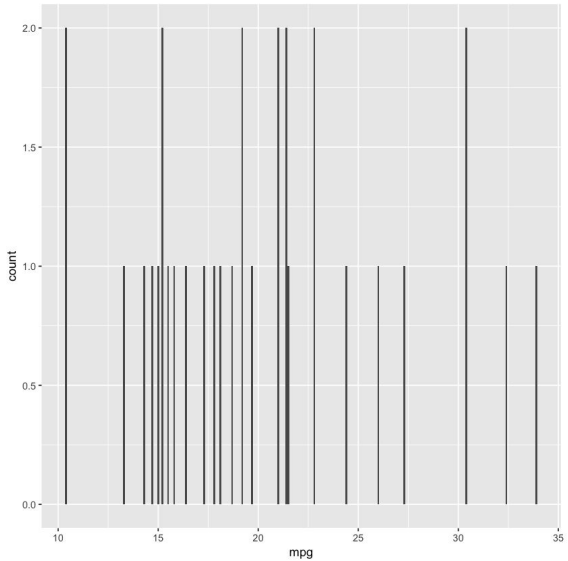
Statistical Transformations

Default Statistics for geom

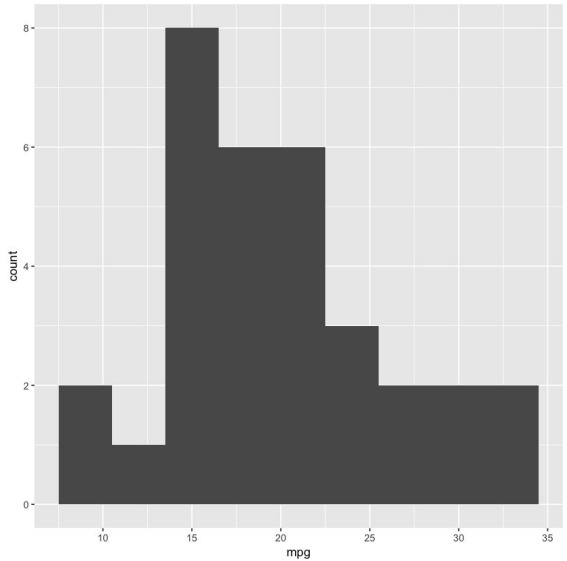
```
args(geom_bar)
function (mapping = NULL, data = NULL, stat = "bin", position = "stack", ...)

args(geom_boxplot)
function (mapping = NULL, data = NULL, stat = "boxplot", position = "dodge",
outlier.color = "black", outlier.shape = 16, outlier.size = 2,
notch = FALSE, notchwidth = 0.5, ...)

args(geom_histogram)
function (mapping = NULL, data = NULL, stat = "bin", position = "stack", ...)
```

Arguments that can be customised

```
ggplot(mtcars, aes(x= mpg)) +
geom_bar()
```




```
ggplot(mtcars, aes(x= mpg)) +
geom_histogram(stat= "bin", binwidth= 3)
```



Scales

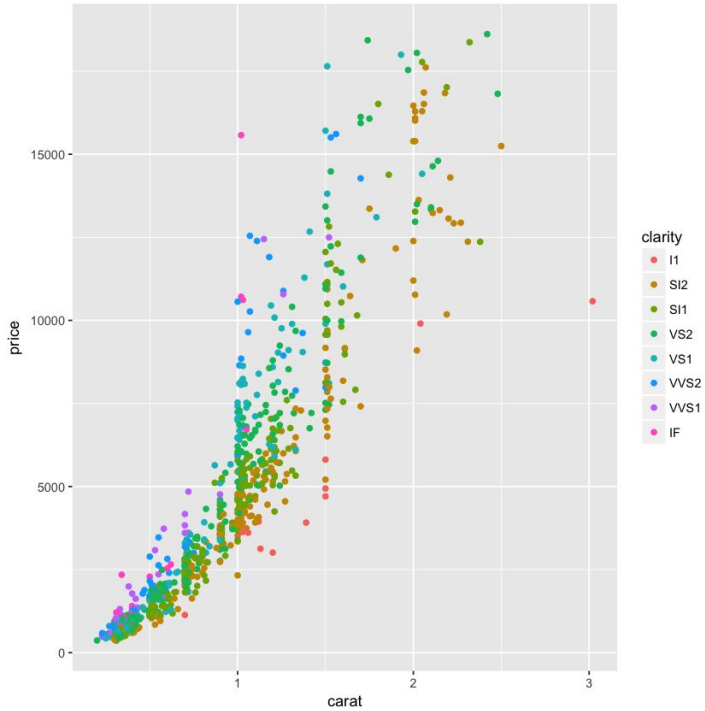
The scale contains:

position, color, shape, size and line type

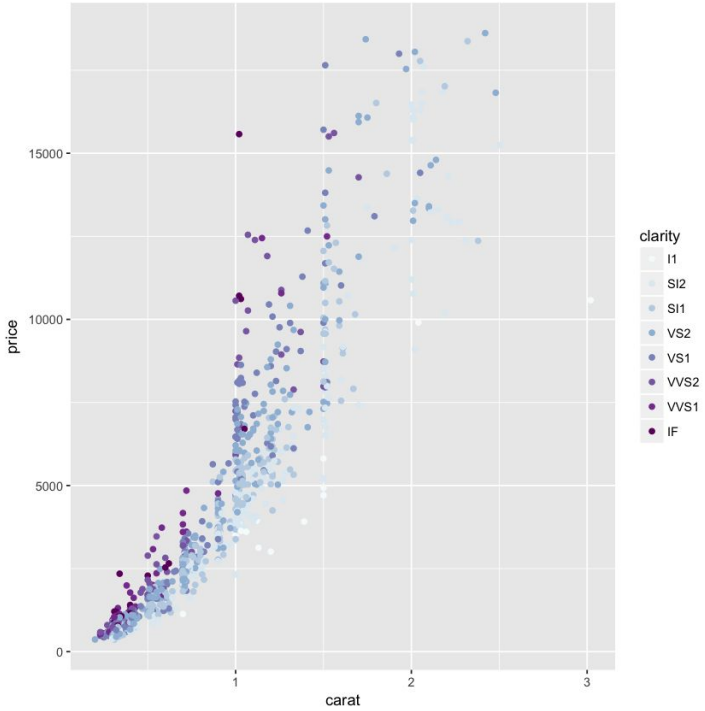
Range - Colour

```
dSample<-
diamonds[sample(nrow(diamonds),
1000),]
diamond <- ggplot(dSample,
aes(x= carat, y= price, color=
clarity)) + geom_point()
diamond
```

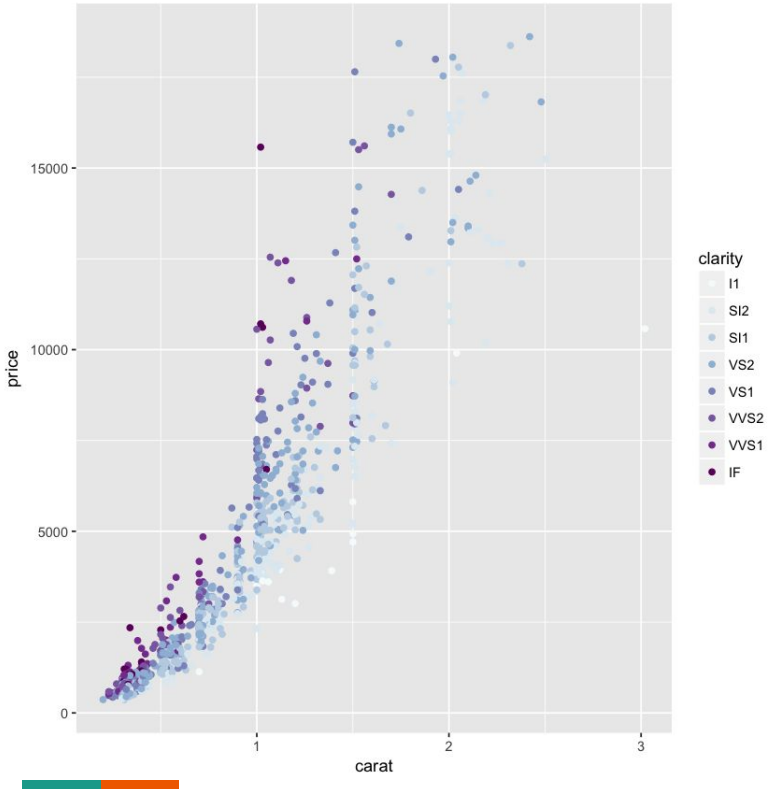
```
diamond +
scale_color_brewer(type= "seq",
palette= 3)
```

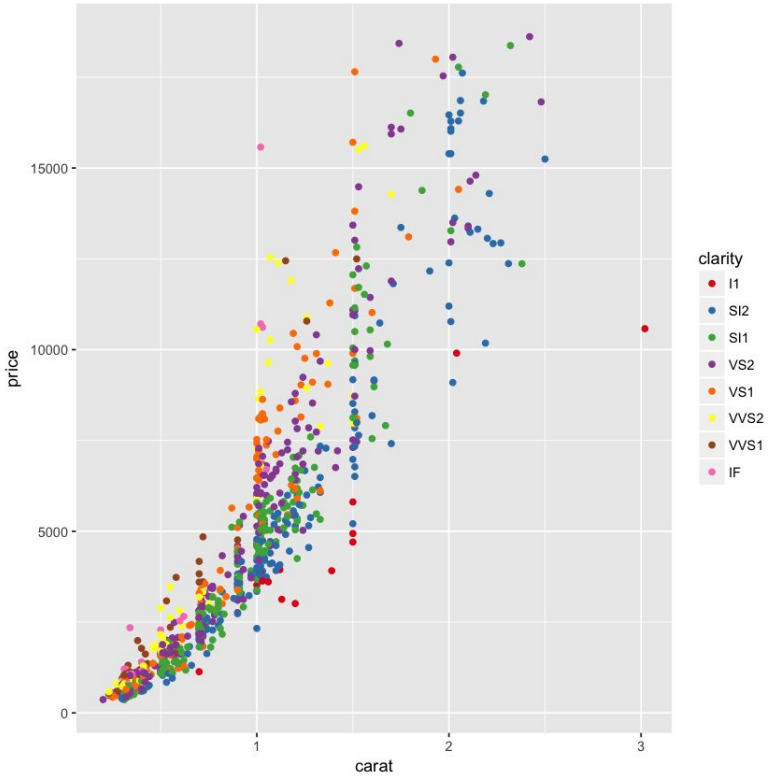



```
dSample<- diamonds[sample(nrow(diamonds), 1000),]
diamond <- ggplot(dSample, aes(x= carat, y= price,
color= clarity)) + geom_point()
diamond
```




```
diamond + scale_color_brewer(type= "seq",
palette= 3)
```



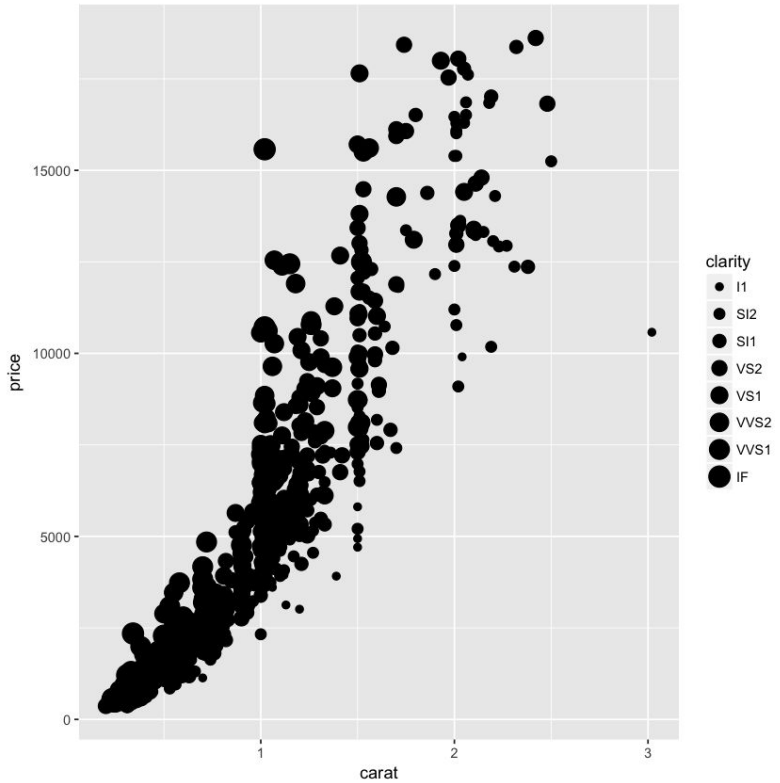


Using the sequential compare to the categorical colour

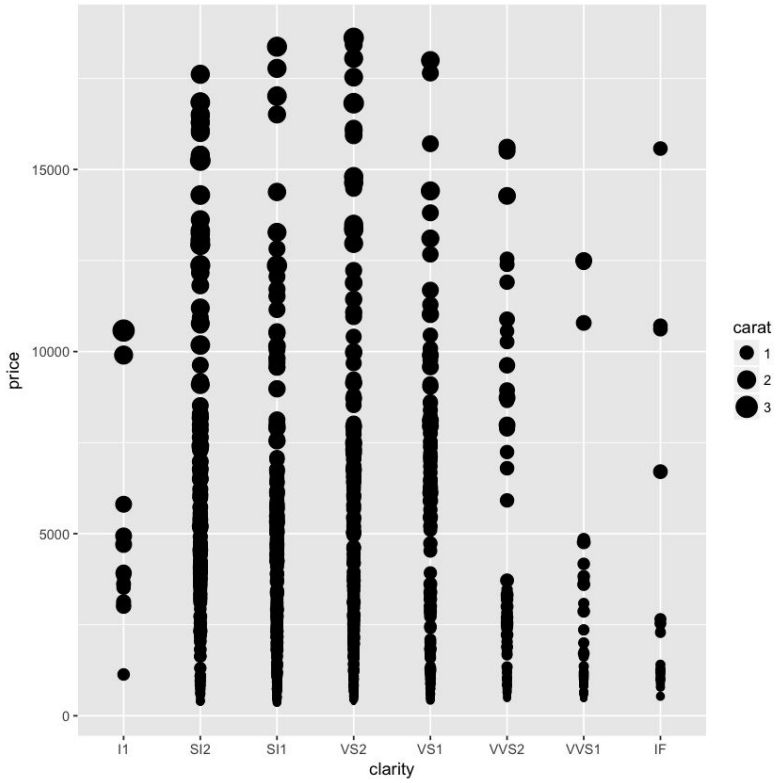
Range - Shapes

```
ggplot(dSample, aes(x= carat, y= price)) +
geom_point(aes(size= clarity))
```

```
ggplot(dSample, aes(x= clarity, y= price)) +
geom_point(aes(size= carat))
```




```
ggplot(dSample, aes(x= carat, y= price)) +
geom_point(aes(size= clarity))
```



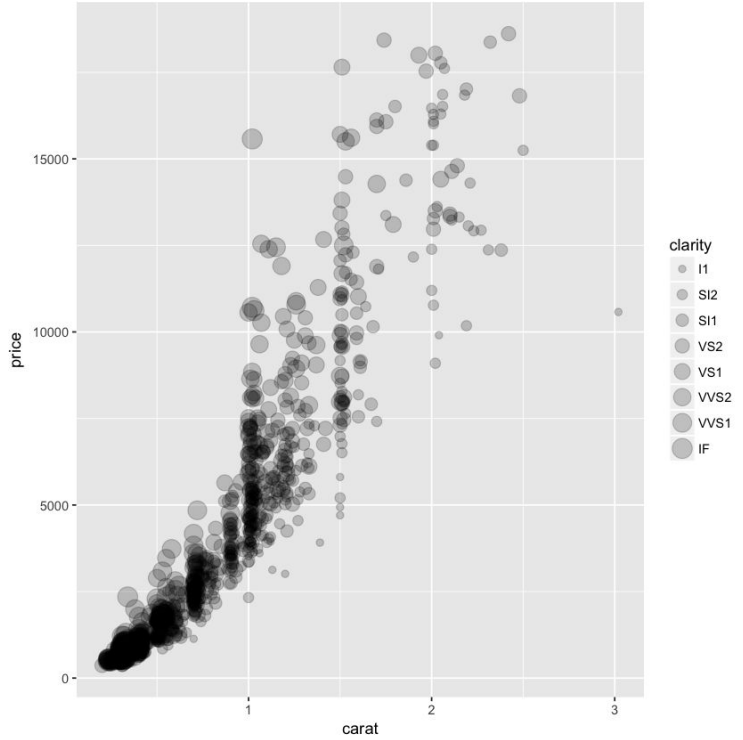

```
ggplot(dSample, aes(x= clarity, y= price)) +
geom_point(aes(size= carat))
```

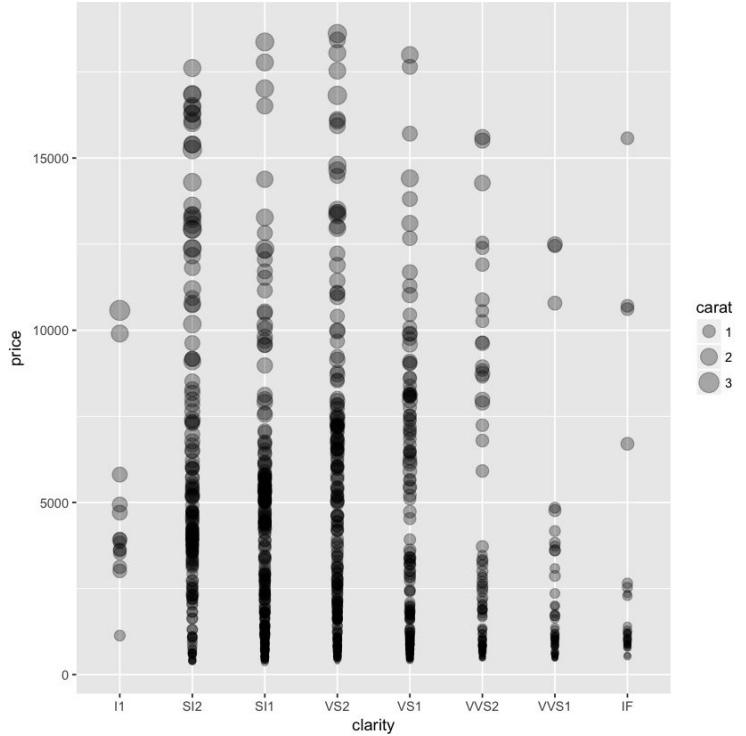
Question

Can we improve the plot?

Yes, by adding `alpha=0.5` to `geom_point`.

```
geom_point(aes(size= clarity), alpha= .5)
```





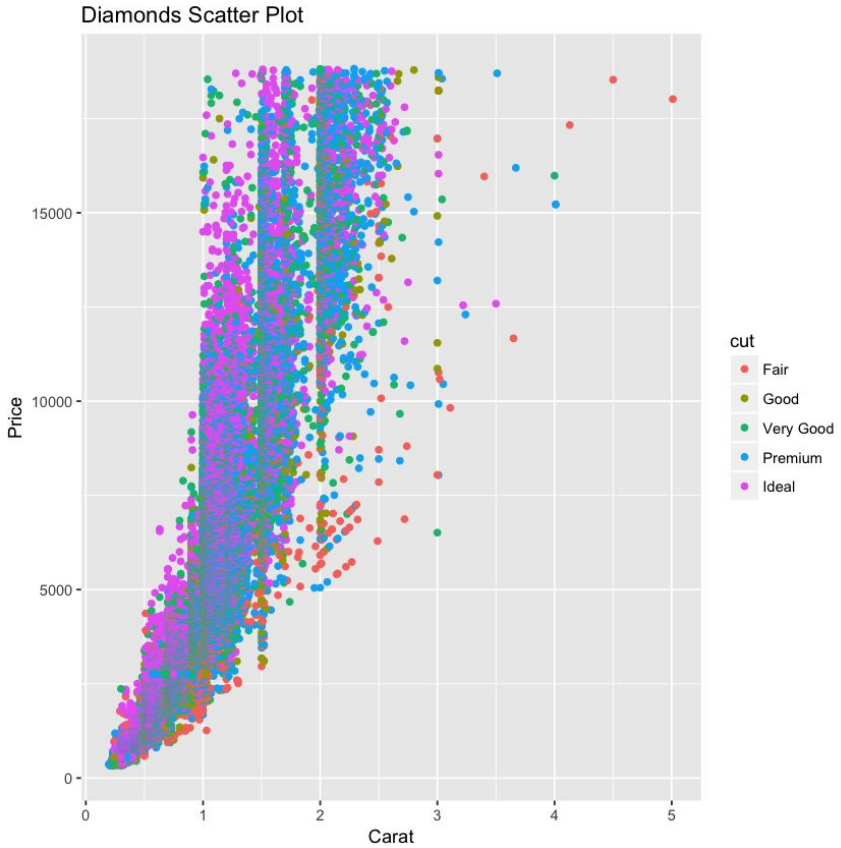
Using alpha to overcome the overlapping issue

```
geom_point(aes(size= clarity), alpha= .5)
```

Facet

Without Facet

```
ggplot(diamonds, aes(x= carat, y= price, color= cut)) +
geom_point()
```



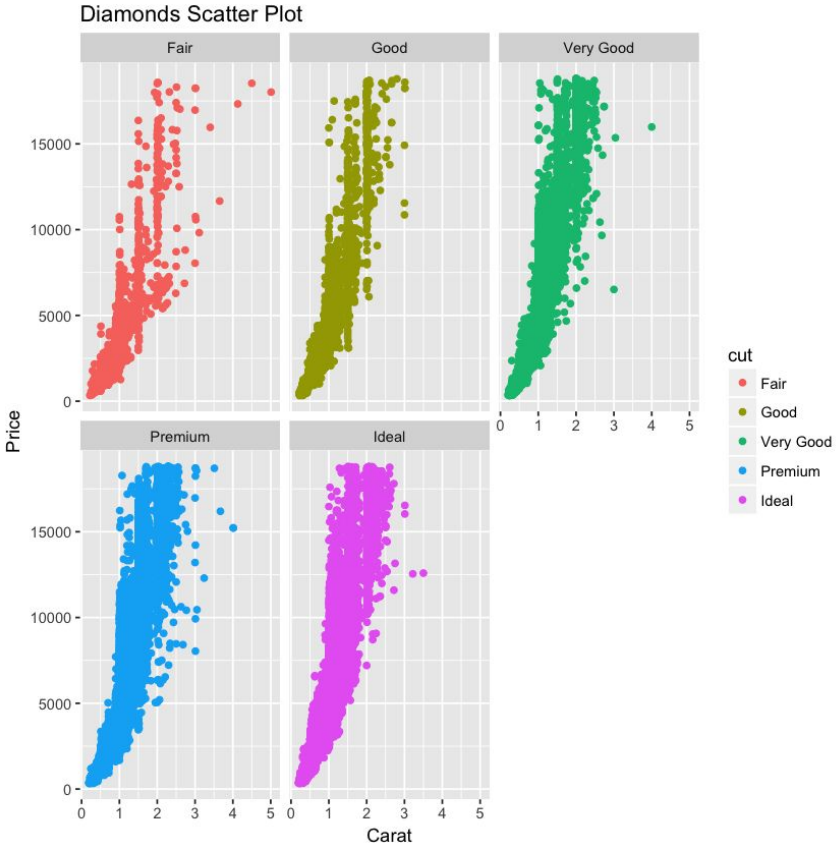
Try it

Add `+ facet_wrap(~ cut, ncol = 3)`

to the main ggplot2.

Adding Facet

```
ggplot(diamonds, aes(x= carat, y= price, color= cut)) +
geom_point() + facet_wrap(~ cut, ncol= 3)
```



Theme

The theme handles non-data elements within the plot:

- Axis labels
- Facet label background
- Legend
- Plot background

More details:
https://github.com/hadley/ggplot2/wiki/Themes

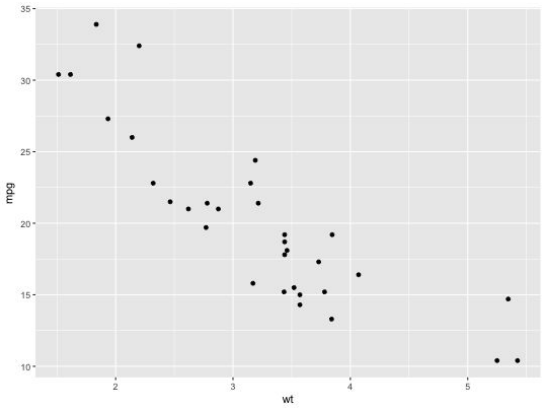
https://github.com/jrnold/ggthemes

ggplot2 Practice

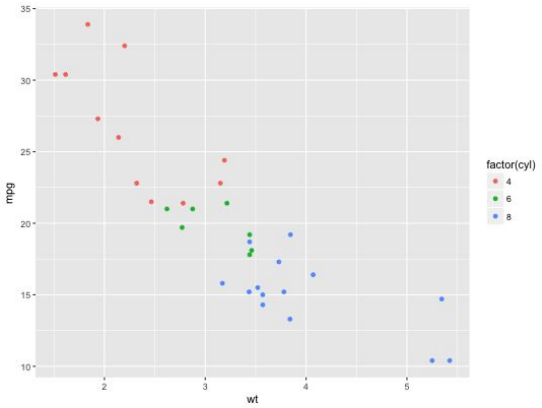
We are going to do coding...

Practical

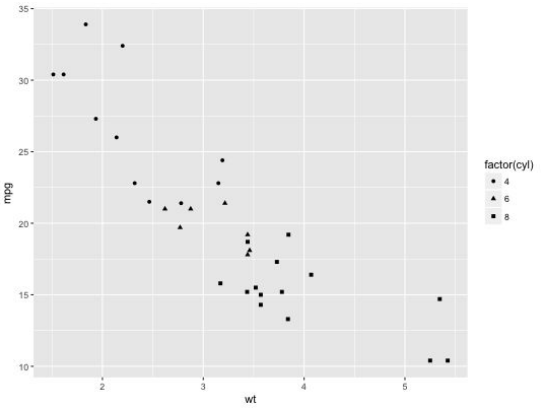
Let's plot the `mtcars` dataset




```
p <- ggplot(mtcars, aes(wt, mpg))
p + geom_point()
```



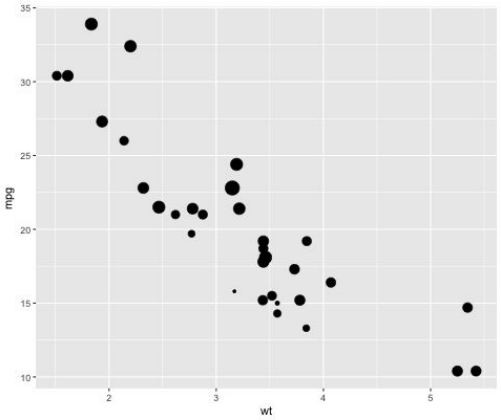

```
p + geom_point(aes(colour = factor(cyl)))
```

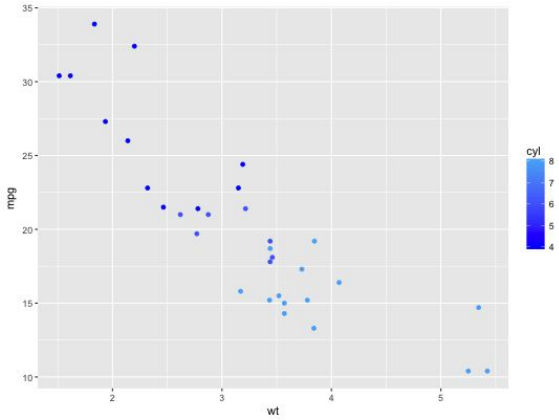


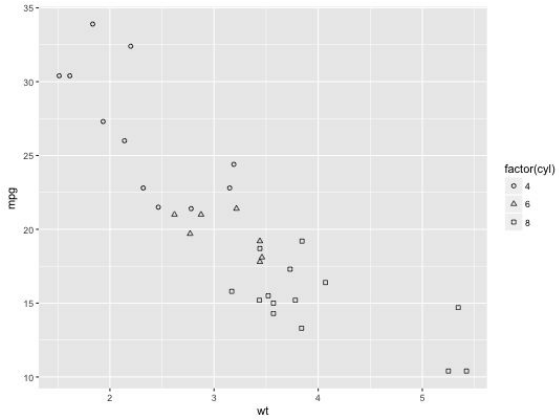

```
p + geom_point(aes(shape = factor(cyl)))
```

Practical

Now use `size, colour, shape`







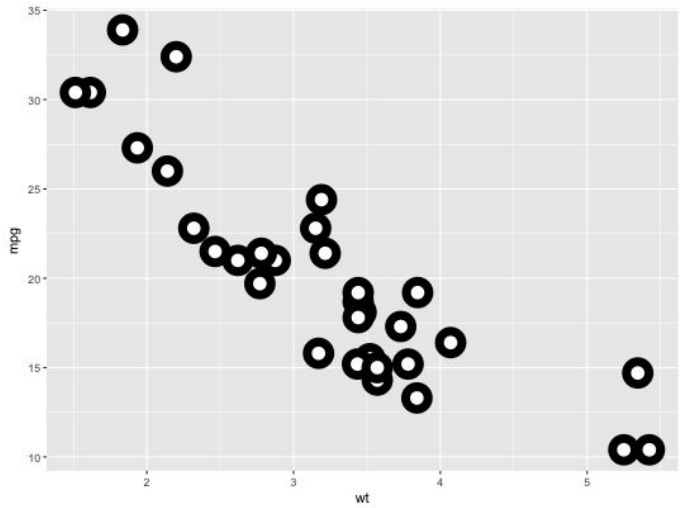

```
p + geom_point(aes(size = qsec))
```

```
p + geom_point(aes(colour = cyl))
```

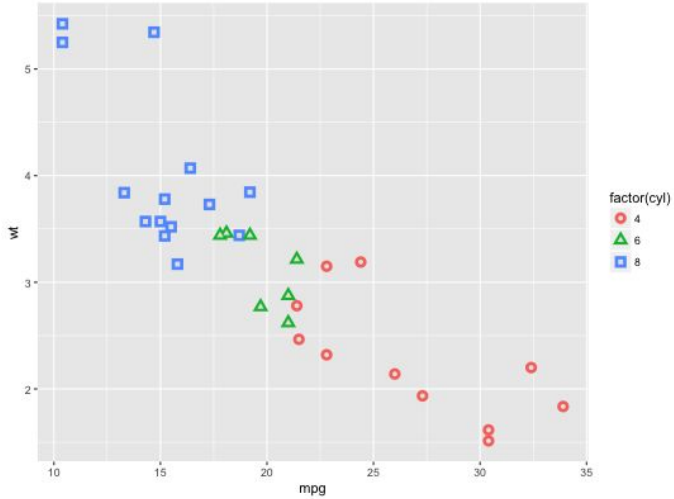
```
p + geom_point(aes(shape = factor(cyl))) +
scale_shape(solid = FALSE)
```

Practical

Now customise the colour/shape



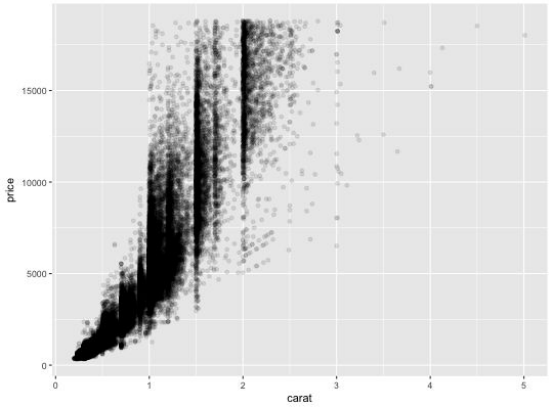

```
ggplot(mtcars, aes(wt, mpg)) +
  geom_point(shape= 21, colour= "black",
fill= "white", size= 5, stroke= 5)
```



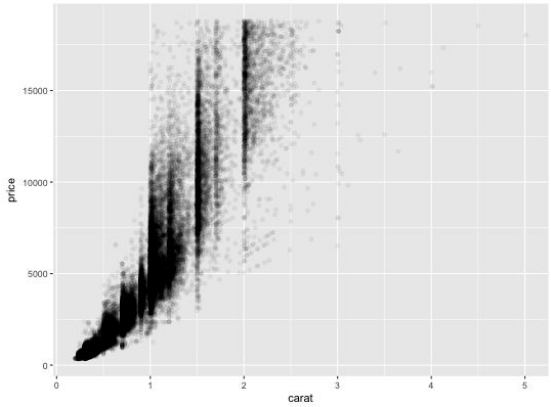

```
ggplot(mtcars, aes(mpg, wt, shape=
factor(cyl))) + geom_point(aes(colour=
factor(cyl)), size= 4) + geom_point(colour=
"grey90", size= 1.5)
```

Practical

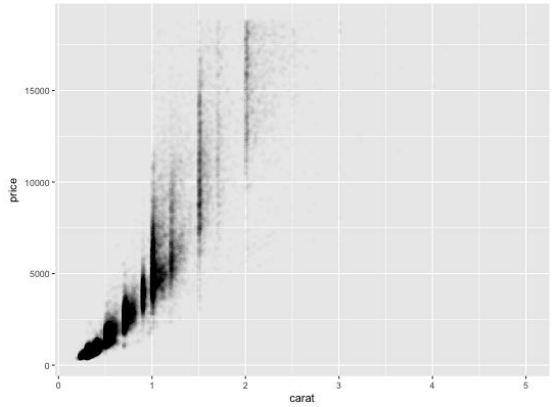
Plot large-scale diamonds datasets




```
d + geom_point(alpha= 1/20)
```




```
d <- ggplot(diamonds,
aes(carat, price))
d + geom_point()
```

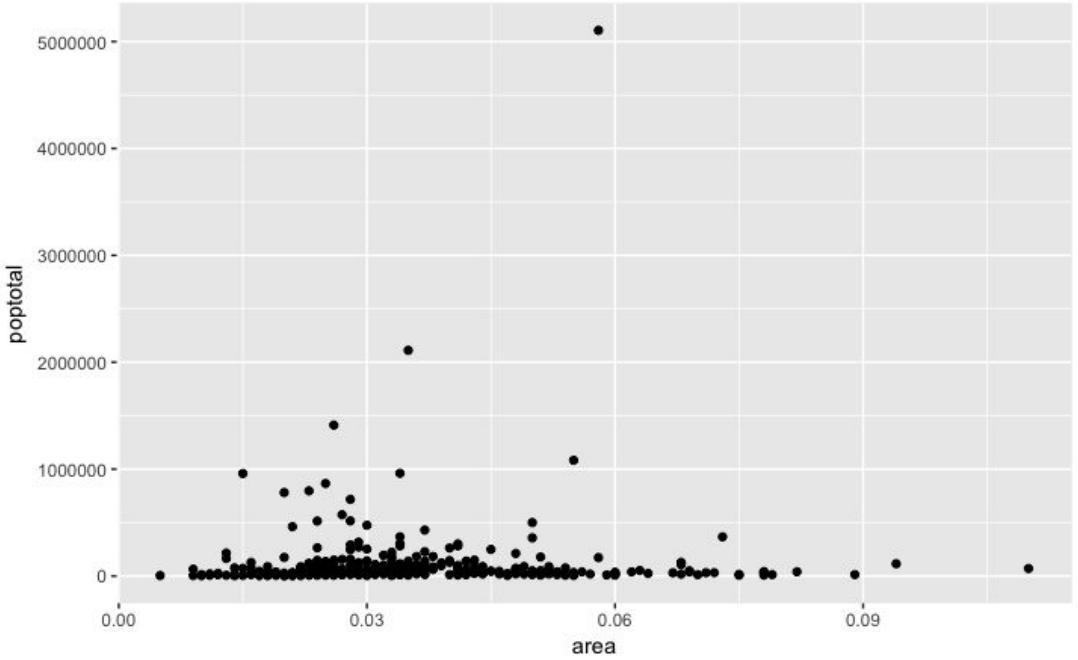



```
d + geom_point(alpha= 1/100)
```

Practical

Scatterplot - Simple

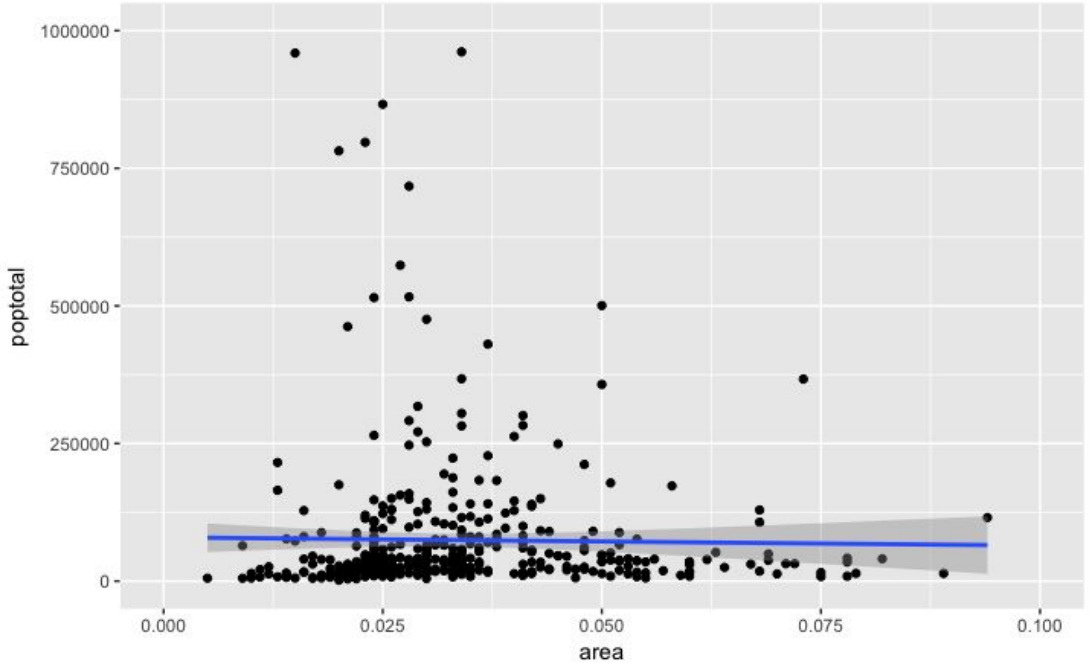
```
library(ggplot2)
theme_set(theme_bw())
data("midwest", package =
"ggplot2")


g <- ggplot(midwest,
aes(x=area, y=poptotal)) +
geom_point()
```



Practical

Scatterplot - Simple + line fitting

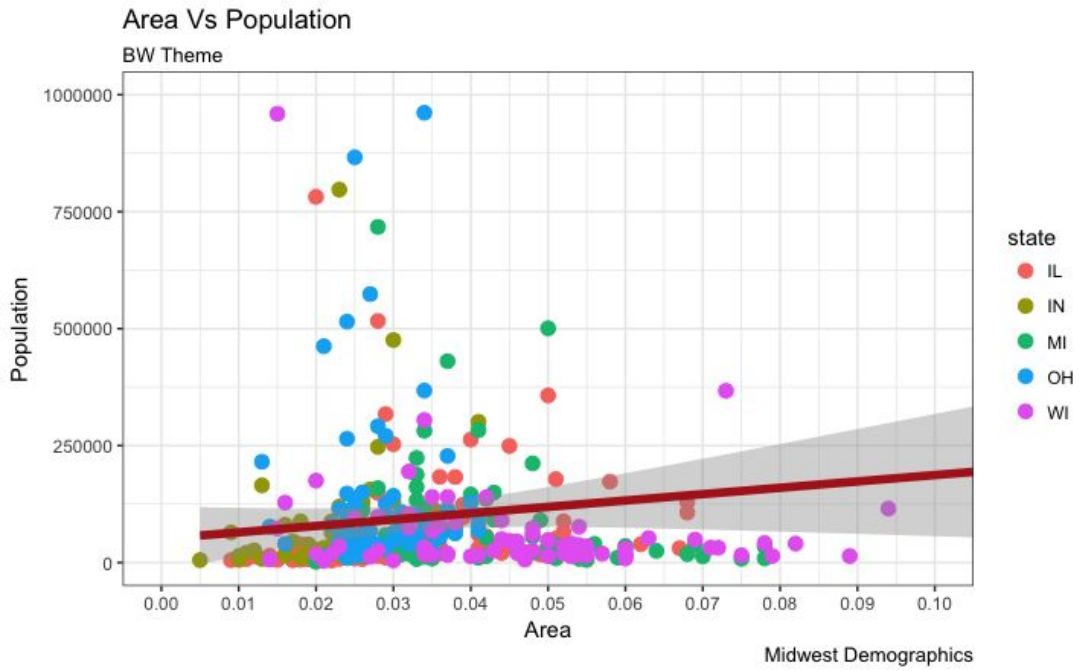
```
library(ggplot2)
theme_set(theme_bw())
data("midwest", package =
"ggplot2")


g <- ggplot(midwest,
aes(x=area, y=poptotal)) +
geom_point() +
geom_smooth(method="lm")

# Not outside the limits
g + xlim(0, 0.1) + ylim(0,
1000000)
```



Practical

Scatterplot - Simple + theme

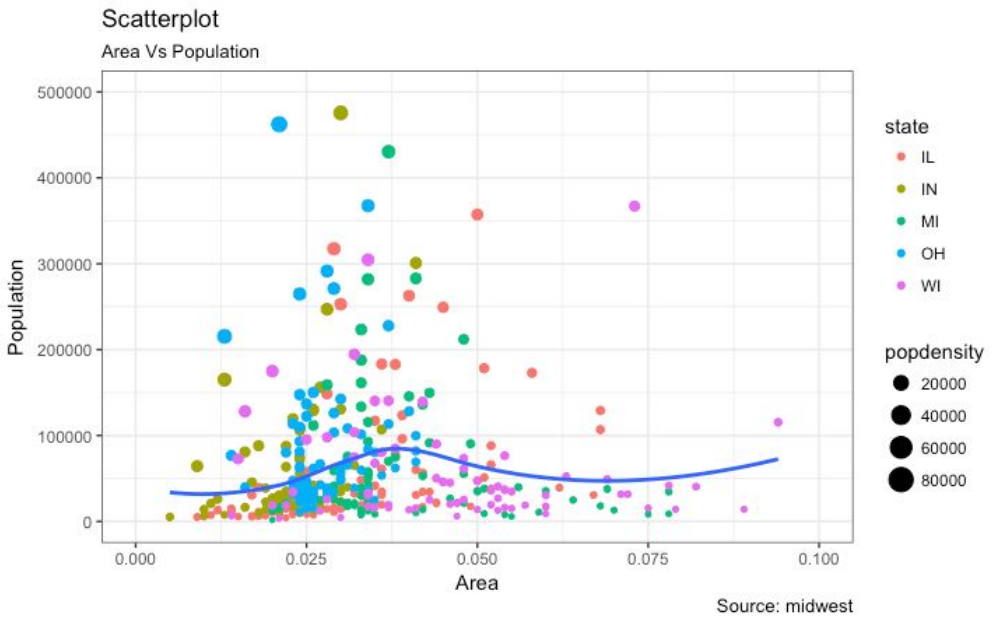
```
library(ggplot2)
theme_set(theme_bw())
data("midwest", package =
"ggplot2")


g <- ggplot(midwest,
aes(x=area, y=poptotal)) +
geom_point(aes(col=state),
size=3) +
geom_smooth(method="lm",
col="firebrick", size=2
)

g + xlim(0, 0.1) + ylim(0,
1000000)

g + theme_bw()
```



Practical

Scatterplot - Pro

```
library(ggplot2)
theme_set(theme_bw())
data("midwest", package = "ggplot2")

gg <- 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")

plot(gg)
```



Thank you

Useful sources

http://r-statistics.co/ggplot2-Tutorial-With-R.html

http://r-statistics.co/Complete-Ggplot2-Tutorial-Part1-With-R-Code.html

http://r-statistics.co/Complete-Ggplot2-Tutorial-Part2-Customizing-Theme-With-R-Code.html

http://r-statistics.co/Top50-Ggplot2-Visualizations-MasterList-R-Code.html

http://r-statistics.co/ggplot2-cheatsheet.html

http://tutorials.iq.harvard.edu/R/Rgraphics/Rgraphics.html

References

Kevin Davenport, Data visualization with R, San Diego R Users Group, June 2013.

Jessie Kennedy, Principles of Information Visualization Tutorial, Institute for Informatics & Digital Innovation, Edinburgh Napier University, 2012.

Martin Krzywinski, visual design principles, Vizbi 2013.

Jeffrey Heer, Interactive Data Analysis, University of Washington.

R. Jordan Crouser, Introduction to Visual Analytics, Computer Science Department at Tufts University, 2015.

Colin Ware. 2004. Information Visualization: Perception for Design. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Boris Naujoks, Jörg Stork, Martin Zaefferer, Thomas Bartz-Beielstein, Meta-Model Assisted (Evolutionary) Optimization Tutorial at PPSN 2016, 18.09.2016

Hadley's ggplot2 book
http://amzn.com/0387981403

ggplot2 google group
http://groups.google.com/group/ggplot2

stackoverflow
http://stackoverflow.com/tags/ggplot2

Lattice to ggplot2 conversion
http://learnr.wordpress.com/?s=lattice

Winston Chang's Cookbook for common graphics
http://wiki.stdout.org/rcookbook/Graphs/