Lecture 13 - Implementing A Layered Grammar of Graphics using R and ggplot2

Tutorial

Most of today will be spent working through examples of a variety of plot types that can be generated using R and ggplot2. Work through the provided code and talk to your neighbor about what is happening and why it works. Make sure to ask any questions you may have.

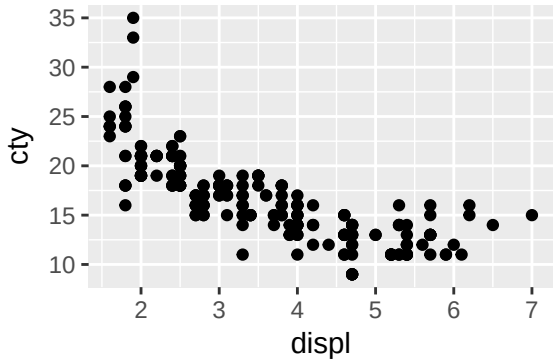
```
library(ggplot2)
library(grid)
library(gridExtra)

mpg = read.table("mpg.txt", header = TRUE, sep = "\t", stringsAsFactors = FALSE)

dim(mpg)
```

```
## [1] 234   9
```

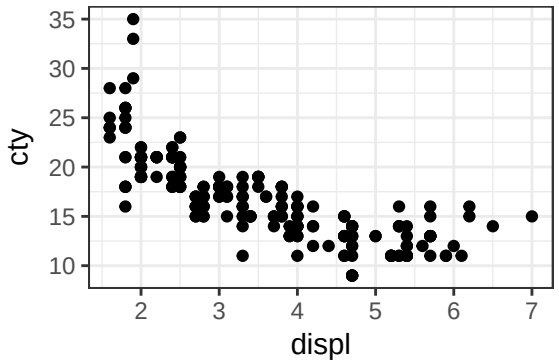
```
head(mpg)
```

```
##   manufacturer model displ year cyl      trans drv cty hwy
## 1         audi    a4   1.8 1999   4   auto(l5)   f  18  29
## 2         audi    a4   1.8 1999   4 manual(m5)   f  21  29
## 3         audi    a4   2.0 2008   4 manual(m6)   f  20  31
## 4         audi    a4   2.0 2008   4   auto(av)   f  21  30
## 5         audi    a4   2.8 1999   6   auto(l5)   f  16  26
## 6         audi    a4   2.8 1999   6 manual(m5)   f  18  26
```

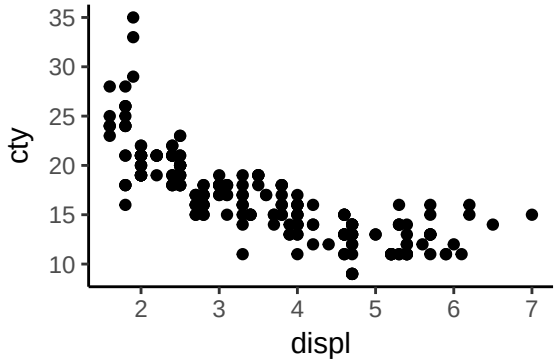
```
# plot of displacement (engine size) vs. city miles per
# gallon (cty)
a = ggplot(data = mpg, aes(x = displ, y = cty))
a + geom_point() + coord_cartesian()
```



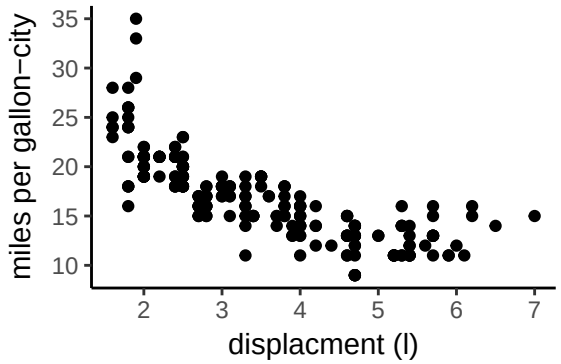

```
# remove grey background
a + geom_point() + coord_cartesian() + theme_bw()
```



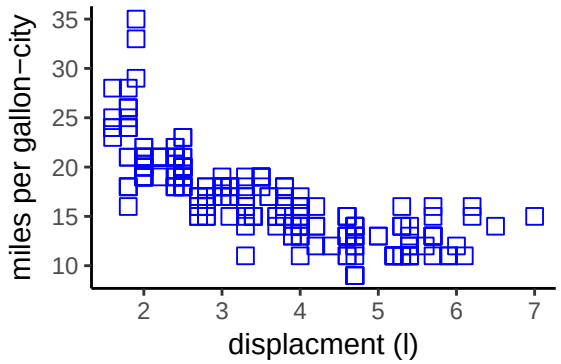

```
# remove grey background and gridlines
a + geom_point() + coord_cartesian() + theme_classic()
```



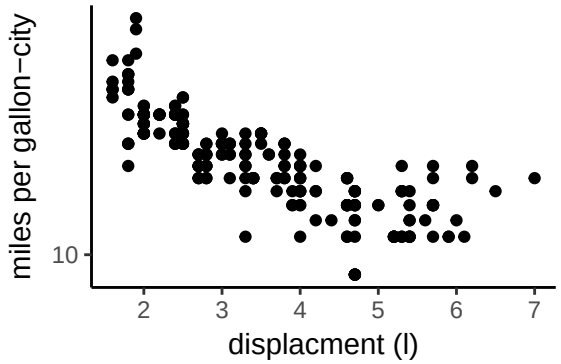

```
# change the x and y labels; cartesian coordinates are
# default
a + geom_point() + theme_classic() + xlab("displacment (l)") +
    ylab("miles per gallon-city")
```



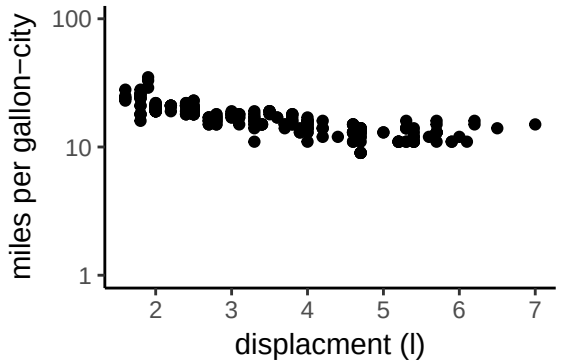

```
# arguments to geom_point() can alter the appearance of the
# points
a + geom_point(color = "blue", shape = 22, size = 3) + theme_classic() +
    xlab("displacment (l)") + ylab("miles per gallon-city")
```



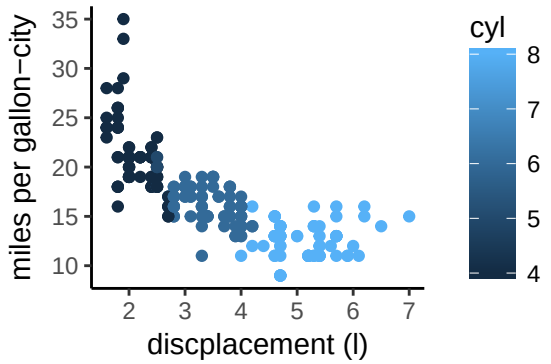

```
# log transform the y-axis
a + geom_point() + theme_classic() + xlab("displacment (l)") +
    ylab("miles per gallon-city") + scale_y_log10()
```



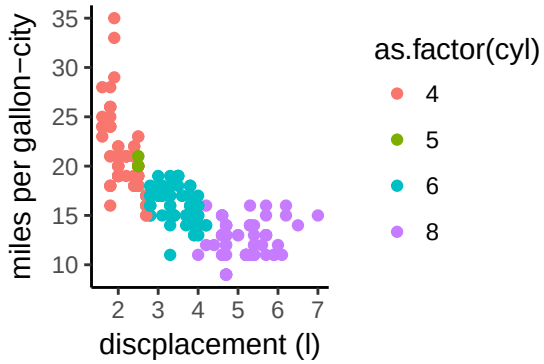

```
# arguments to scale can also customize the range and tick
# locations
a + geom_point() + theme_classic() + xlab("displacment (l)") +
    ylab("miles per gallon-city") + scale_y_log10(limits = c(1,
    100), breaks = c(1, 10, 100))
```



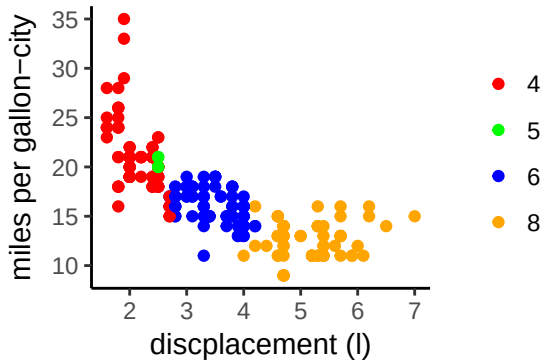

```
# we can also color code points based on continuous or
# categorical variables continuous
a + geom_point(aes(color = cyl)) + theme_classic() + xlab("discplacement (l)") +
    ylab("miles per gallon-city")
```



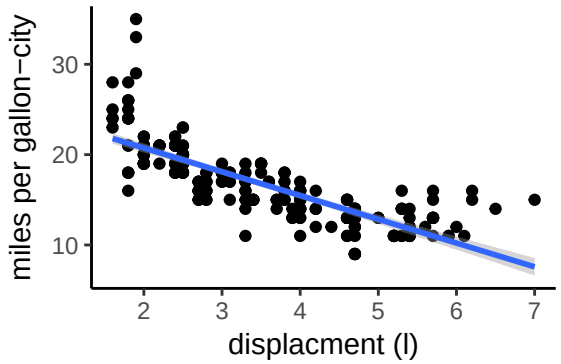

```
# categorical
a + geom_point(aes(color = as.factor(cyl))) + theme_classic() +
    xlab("discplacement (l)") + ylab("miles per gallon-city")
```



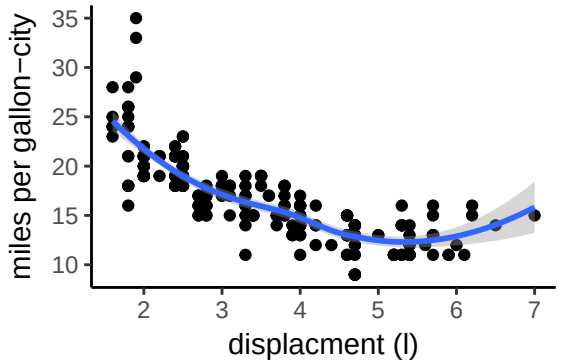

```
# categorical - the default colors are a bit odd
a + geom_point(aes(color = as.factor(cyl))) + theme_classic() +
    xlab("discplacement (l)") + ylab("miles per gallon-city") +
    scale_color_manual(values = c("red", "green", "blue", "orange")) +
    theme(legend.title = element_blank())
```




```
# add a linear trendline with a new layer
a + geom_point() + theme_classic() + xlab("displacment (l)") +
    ylab("miles per gallon-city") + stat_smooth(method = "lm")
```

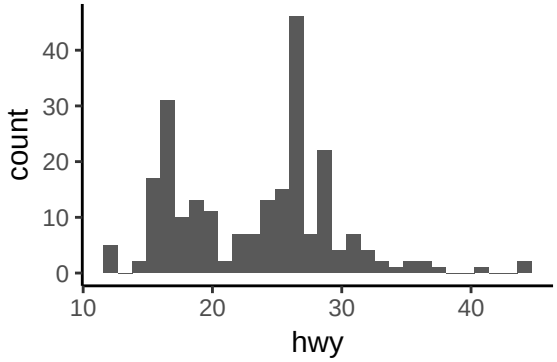



```
# add a spline with a new layer
a + geom_point() + theme_classic() + xlab("displacment (l)") +
    ylab("miles per gallon-city") + stat_smooth(method = "loess")
```



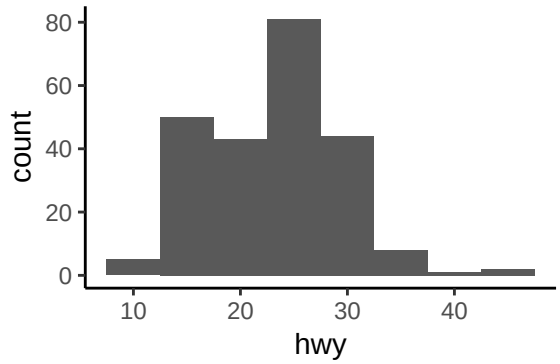

```
# histogram of mpg hwy
b = ggplot(data = mpg, aes(x = hwy))
b + geom_histogram() + theme_classic()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

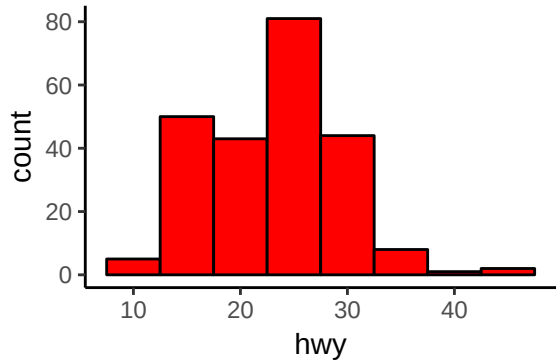



```
# change bins
b + geom_histogram() + theme_classic() + stat_bin(binwidth = 5)
```

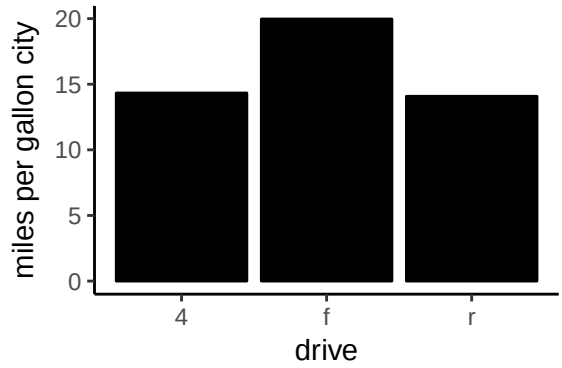
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
# change bins; the stat_bin is a default for geom_histogram,
# so it can be skippped color can also be specified in the
# geom_histogram call
b + geom_histogram(binwidth = 5, fill = "red", color = "black") +
    theme_classic()
```




```
# we can generate a barplot of means too
d = ggplot(data = mpg)
d + geom_bar(aes(x = as.factor(drv), y = cty), stat = "summary",
    fun.y = "mean", fill = "black", color = "black") + theme_classic() +
    xlab("drive") + ylab("miles per gallon city")
```

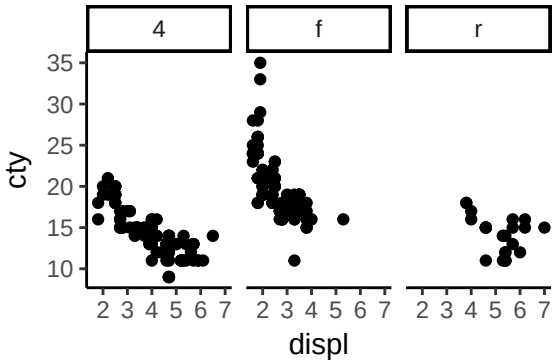



```
tapply(X = mpg$cty, INDEX = mpg$drv, FUN = mean)
```

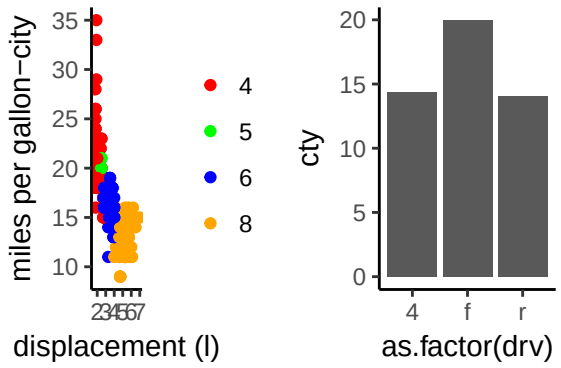
```
##       4       f       r
## 14.3301 19.9717 14.0800
```

```
# faceting allows the same plot for different classes to be
# generated
f = ggplot(data = mpg, aes(x = displ, y = cty)) + geom_point()
```

```
f + facet_wrap(~drv) + theme_classic()
```




```
# grid.arrange from grid/gridExtra libraries allows for
# multiple panels with different plot types you'll need to
# install the grid and gridExtra libraries to accomplish
# this!
g = ggplot(data = mpg, aes(x = displ, y = cty)) + geom_point(aes(color = as.factor(cyl))) +
    theme_classic() + xlab("displacement (l)") + ylab("miles per gallon-city") +
    scale_color_manual(values = c("red", "green", "blue", "orange")) +
    theme(legend.title = element_blank())

h = ggplot(data = mpg) + geom_bar(aes(x = as.factor(drv), y = cty),
    stat = "summary", fun.y = "mean") + theme_classic()

grid.arrange(g, h, ncol = 2)
```



Challenge

Practice using the syntax demonstrated above by writing a script to generate the following plots using the mpg data.

1. A scatter plot of miles per gallon city versus miles per gallon highway. Color code the points by 'drv' (four-wheel drive vs. front-wheel drive vs. rear-wheel drive). Add a linear trendline to the plot.

2. A "density plot" of engine displacement.

3. A barplot of mean displacement for different numbers of cylinders (cyl).