R Plotting with ggplot2 - Part I
Plotting with qplot() and ggplot()

Xuemao Zhang
Department of Mathematics
East Stroudsburg University

June 25, 2019

Outline

- Grammar of Graphics
  - Hadley Wickham and R:ggplot2
  - Layer-by-Layer Graphics
- Quick plots with qplot()
- The ggplot() grammar

Install the following packages if you don't have them.

```
install.packages("ggplot2");
install.packages("lattice");
install.packages("gridExtra");
```

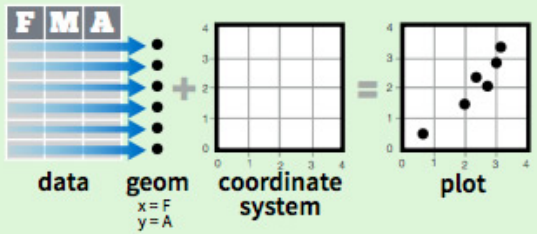
Grammar of Graphics - R:ggplot2

- The cheat sheet of `ggplot2` can be downloaded from Rstudio.

Data Visualization
with ggplot2
Cheat Sheet

Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same few components: a **data** set, a set of **geoms**—visual marks that represent data points, and a **coordinate system.**

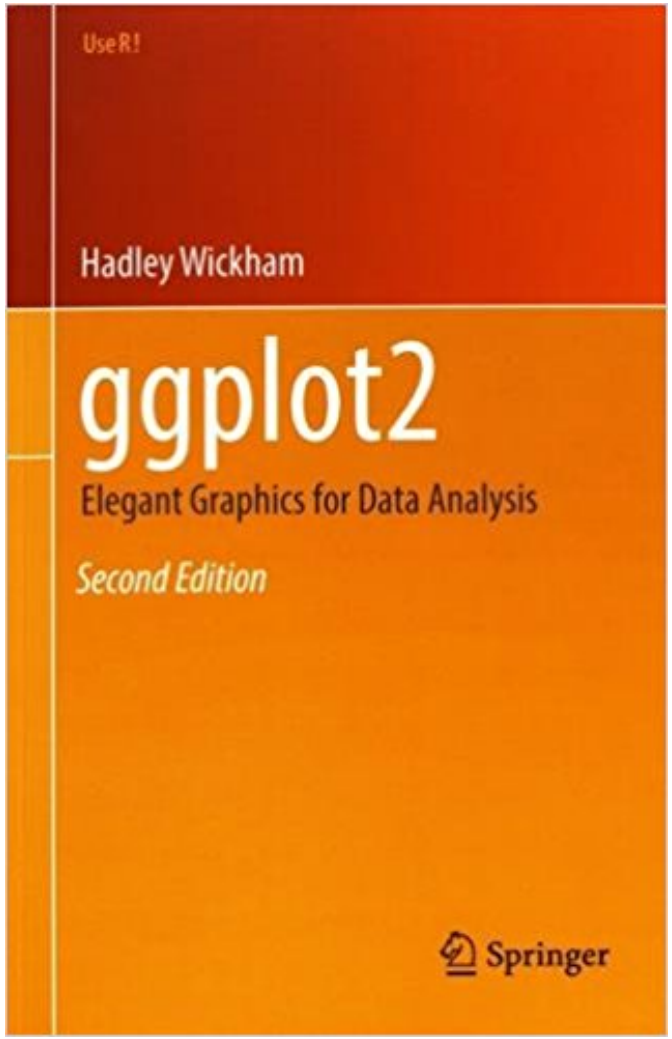


Download a copy from RStudio

Grammar of Graphics - R:ggplot2

- Hadley Wickham is the author of R: ggplot2.
- He is the chief scientist at RStudio, Creator of popular R packages: ggplot2, dplyr, tidyr, devtools, etc; "The man who r evolutionized R" according to Pricenomics (2015)
- R graphics: base -> lattice -> ggplot2
  "ggplot2, started in 2005, is an attempt to take the good things about base and lattice graphics and improve on them with a strong underlying model" (Hadley Wickham)
- R:ggplot2 is one of most commonly downloaded R packages
- Based on Grammar of Graphics by Wilkinson (2005; Springer 2ed)

Grammar of Graphics - R:ggplot2



Grammar of Graphics - R:ggplot2

Quote from the ggplot2 book that further quotes Wilkinson (2005):

> *In brief, the grammar tells us that a statistical graphic is a mapping from data to aesthetic attributes (colour, shape, size) of geometric objects (points, lines, bars). The plot may also contain statistical transformations of the data and is drawn on a specific coordinate system. Facetting can be used to generate the same plot for different subsets of the dataset. It is the combination of these independent components that make up a graphic.*

Keywords: mapping, aesthetic attributes, geometric objects, statistical transformations, coordinate system, facetting

Grammar of Graphics - R:ggplot2

Package 'ggplot2'

October 25, 2018

Version 3.1.0

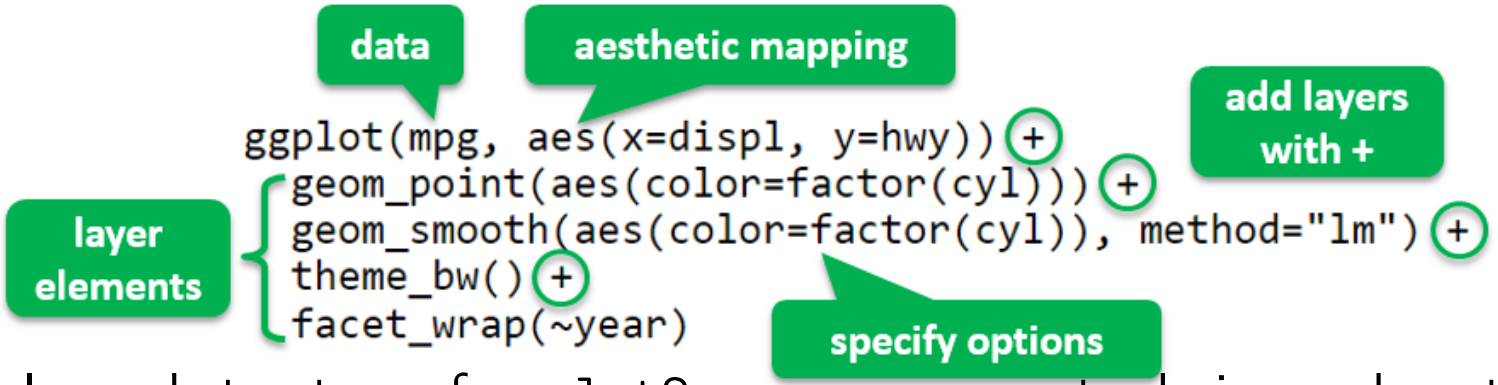
Title Create Elegant Data Visualisations Using the Grammar of Graphics

Description A system for 'declaratively' creating graphics,
based on ``The Grammar of Graphics". You provide the data, tell 'ggplot2'
how to map variables to aesthetics, what graphical primitives to use,
and it takes care of the details.

Depends R (>= 3.1)

- The most popular package for producing static visualizations in R; New upgrade to Version 3.1.0; See CRAN for updated information
- Online documentation at http://docs.ggplot2.org/
- Download the useful cheatsheet created by Rstudio, Inc.
- Also available in Python: http://ggplot.yhathq.com/

Layer-by-Layer Graphics



- The layered structure of `ggplot2` encourages you to design and contrust graphs in a structured manner.
- `ggplot2` uses the special "+" method to add layers to plots.
- **Aesthetics**: mapping data varialbes to aesthetic attributes (position, size, shape, color, . . .)
- **Geometric objects**: point, line, polygon, histogram, quantile, bar, . . .
- **Statistical transformations**: bin, boxplot, density, contour, function, . . .
- Other components for ggplots: **scales** (mapping values of the data to visual values for each aesthetic, e.g. position, color, fill and shape scales); **coordinate system** (cartesian, polar, map projection, ..); **facet** (conditioning display split data in multi-panels; theme (control non-data visual elements (title, axes, tick, . . .)

Layer-by-Layer Graphics

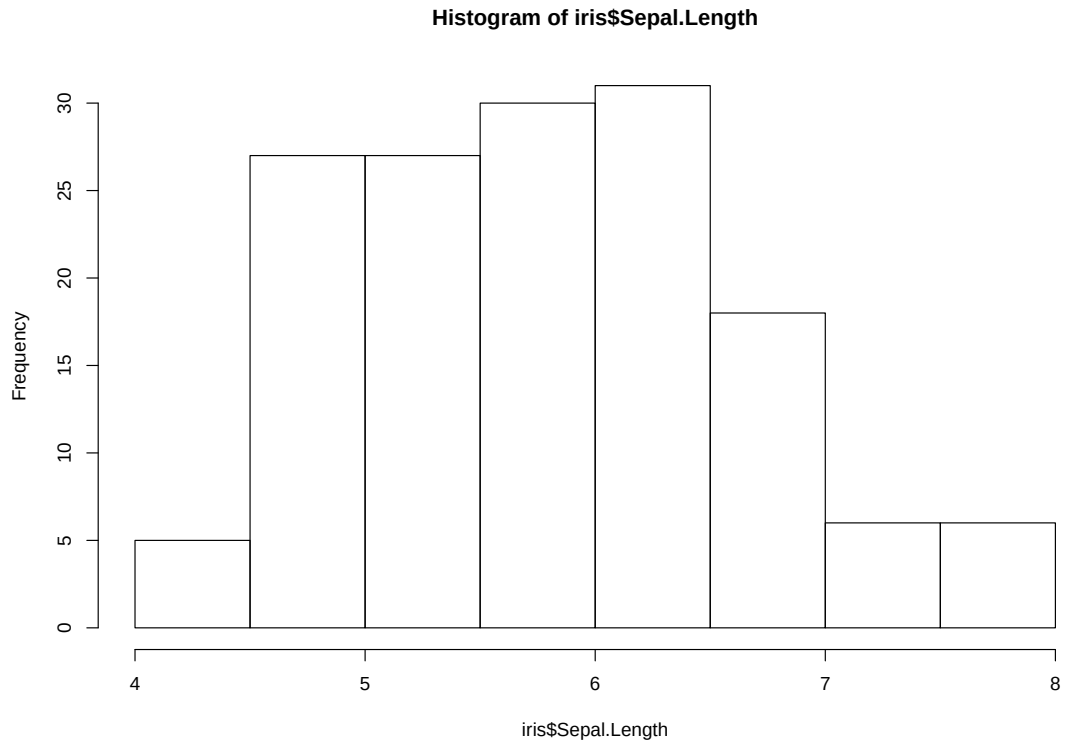
- In ggplot2, aesthetic means "something you can see". Examples include:
  - position (i.e., on the x and y axes)
  - color
  - fill
  - shape (of points)
  - linetype
  - size
- Aesthetic mappings are set with the `aes()` function.
- Each type of `geom_` accepts only a subset of all aesthetics. Use geom help pages to see what mappings each geom accepts.

```
help.search("geom_", package = "ggplot2");
```

```
## starting httpd help server ... done
```

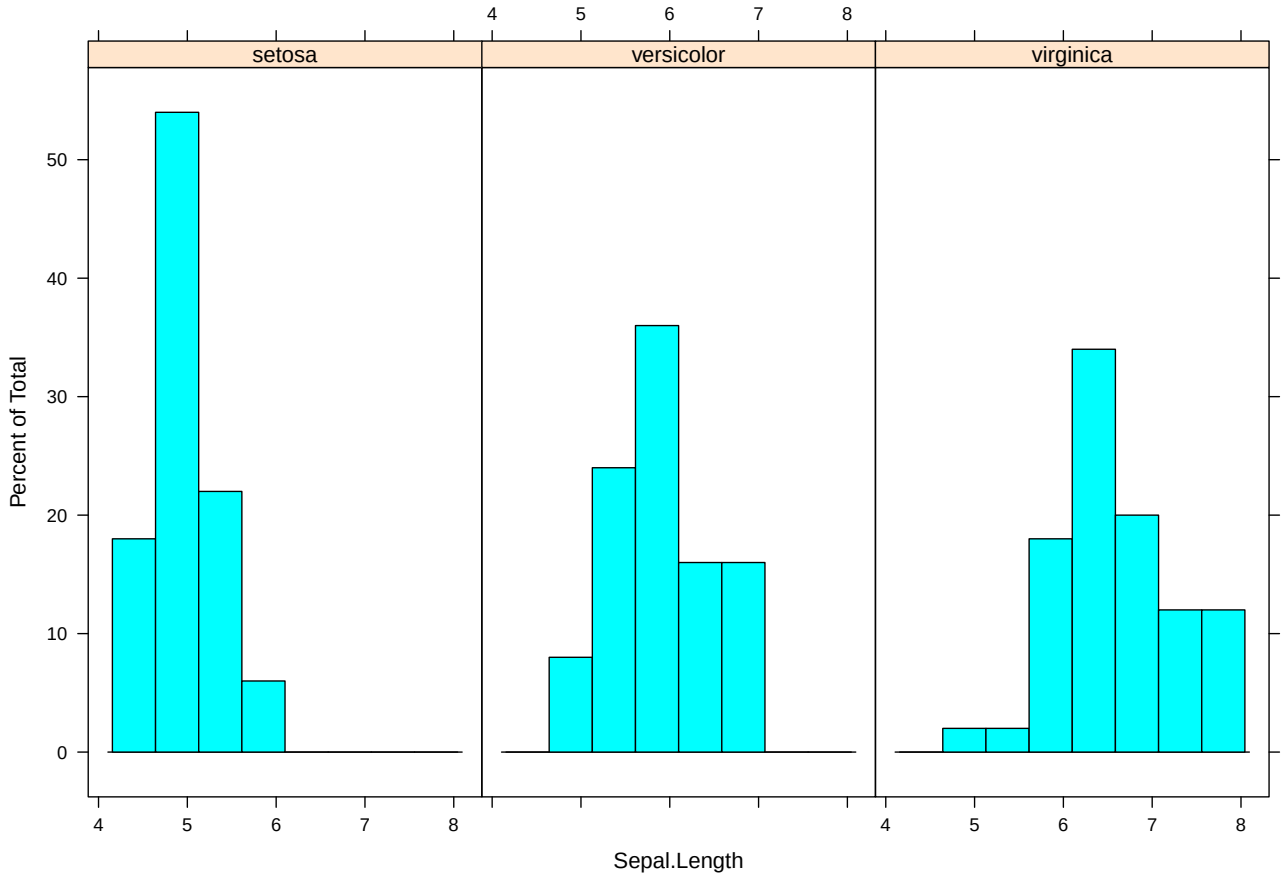
Base, Lattice and ggplot2 styles

```r
hist(iris$Sepal.Length);    # Base graphics
```



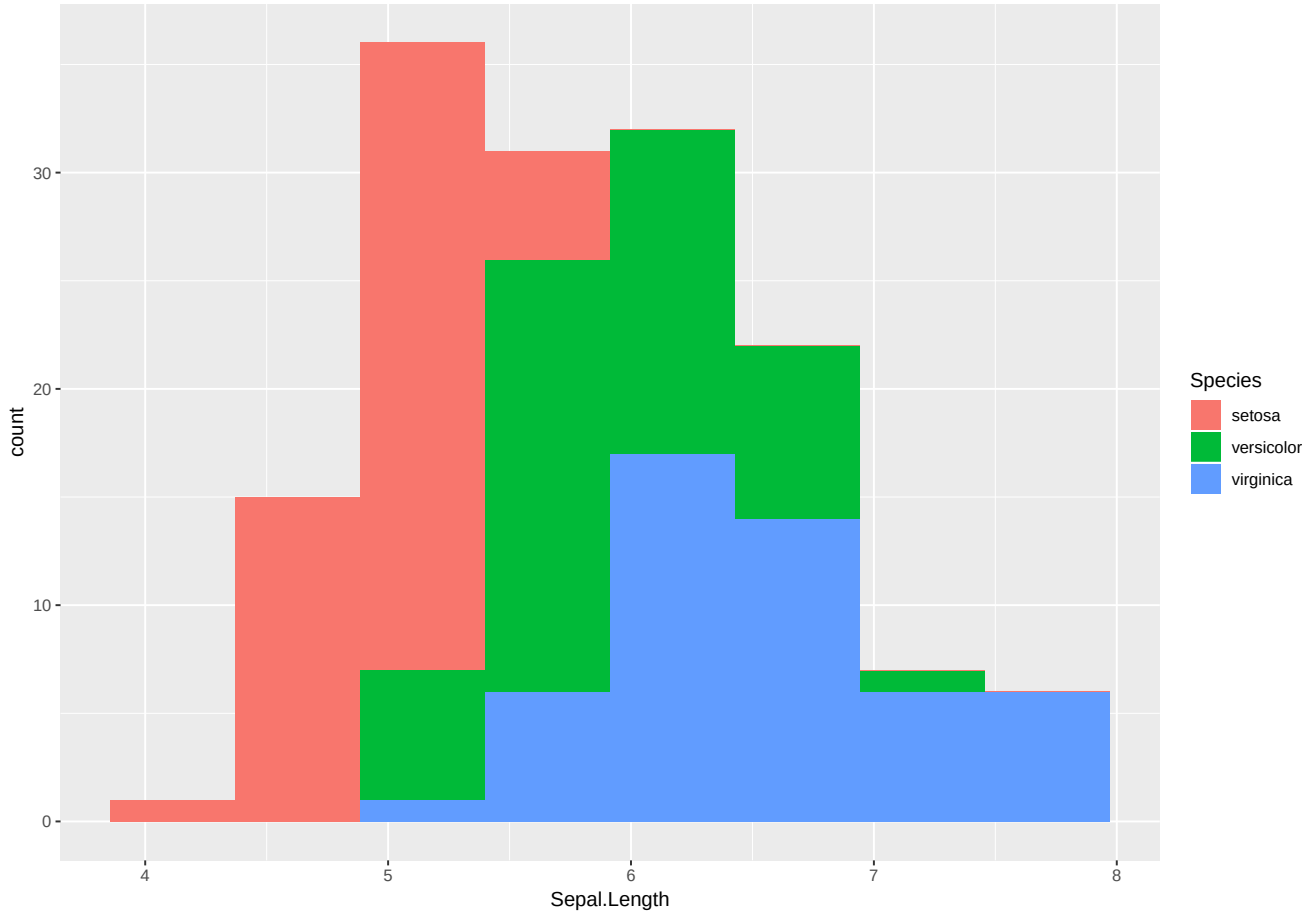
Base, Lattice and ggplot2 styles

```
library(lattice);
histogram(data=iris, ~Sepal.Length|Species);
```



Base, Lattice and ggplot2 styles

```
library(ggplot2);
ggplot(data=iris, aes(x=Sepal.Length,fill=Species)) +
  geom_histogram(bins=8);
```



Quick plots with qplot()

- There are three key components of every plot: data, aesthetics and geoms.
- `qplot()` is analog to base plot(), where "q" means quick.
- We can use `qplot()` when we just want to get a simple plot without thinking about the grammar at all. `ggplot()` function is more flexible and robust than qplot for building a plot piece by piece
- It defines a plot in a single call with the basic syntax:

  qplot(dataframe, variables, [geom], options);

- A sensible geom will be picked by default if it is not supplied.

Quick plots with qplot()

- Consider the data set mpg in the ggplot2 package.
- This dataset contains a subset of the fuel economy data that the EPA makes available on http://fueleconomy.gov. It contains only models which had a new release every year between 1999 and 2008 - this was used as a proxy for the popularity of the car.
- It is a data frame with 234 rows and 11 variables.

```
library(ggplot2);
str(mpg);
```

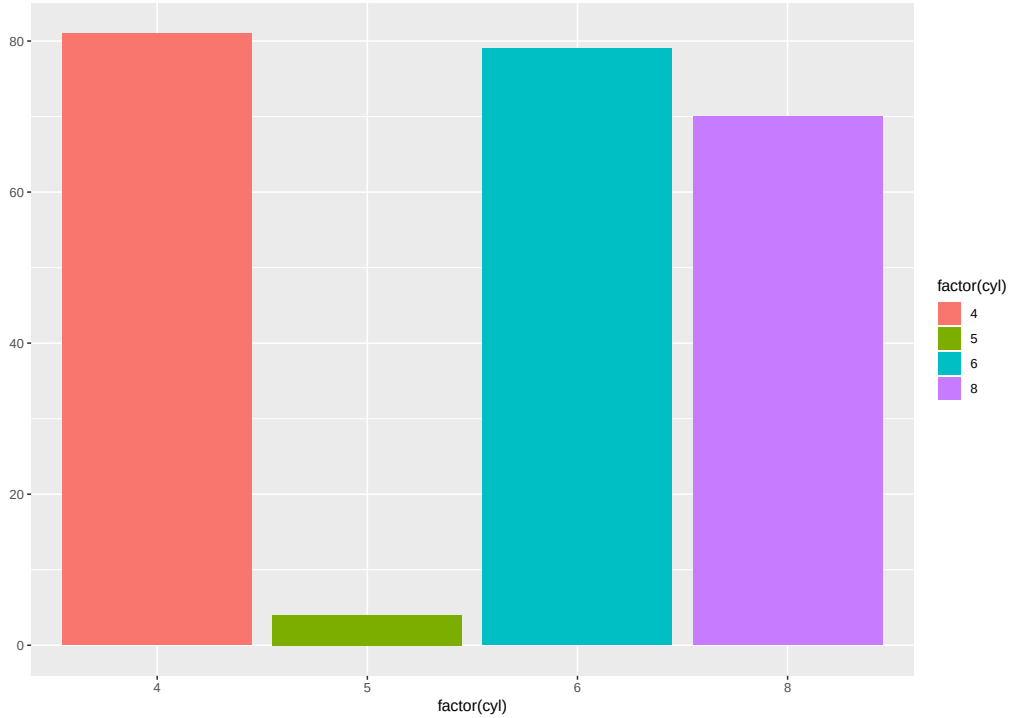
```
## Classes 'tbl_df', 'tbl' and 'data.frame':    234 obs. of  11 vari
##  $ manufacturer: chr  "audi" "audi" "audi" "audi" ...
##  $ model       : chr  "a4" "a4" "a4" "a4" ...
##  $ displ       : num  1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
##  $ year        : int  1999 1999 2008 2008 1999 1999 2008 1999 199
##  $ cyl         : int  4 4 4 4 6 6 6 4 4 4 ...
##  $ trans       : chr  "auto(l5)" "manual(m5)" "manual(m6)" "auto(
##  $ drv         : chr  "f" "f" "f" "f" ...
##  $ cty         : int  18 21 20 21 16 18 18 18 16 20 ...
##  $ hwy         : int  29 29 31 30 26 26 27 26 25 28 ...
```

Quick plots with qplot()

- The variables are mostly self-explanatory
  - `city` and `hwy` records miles per gallon for city and highway driving.
  - `displ`, engine displacement, in litres
  - `drv`: f = front-wheel drive, r = rear wheel drive, 4 = 4wd
  - `model`, model name
  - `year`, year of manufacture
  - `cyl`, number of cylinders
  - `trans`, type of transmission
  - `fl`, fuel type
  - `class`, "type" of car such as two-seater, SUV, compact, etc.
- `qplot()` tries to pick a sensible geometry and statistics based on the arguments provided.

Quick plots with qplot(): Bar plot

```
qplot(data = mpg, factor(cyl), geom="bar",  fill=factor(cyl));
```



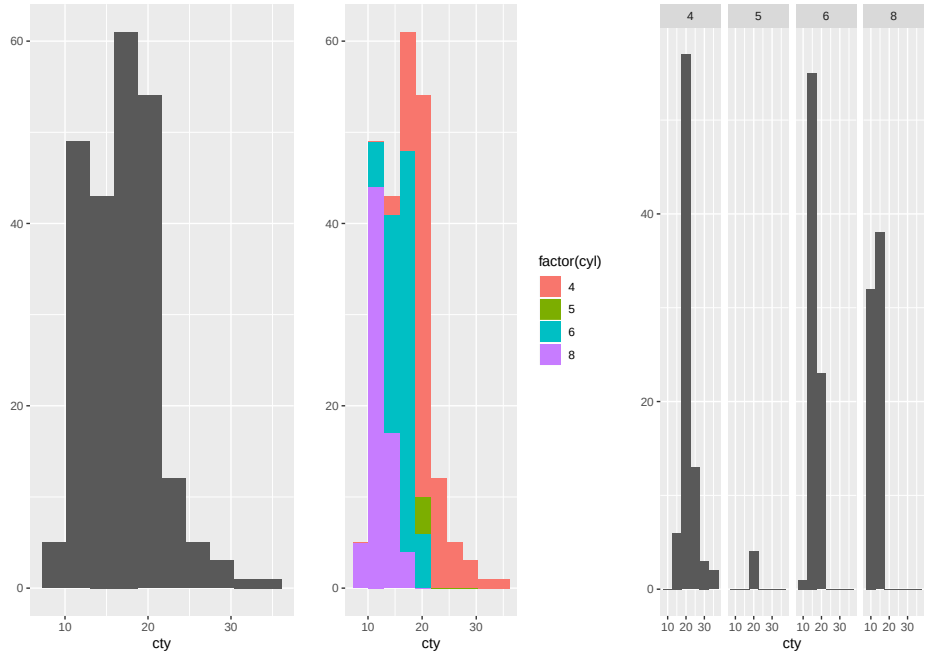

```
# qplot(data = mpg, factor(cyl),fill=factor(cyl));
```

- bar is the sensible geom for categorical variables

Quick plots with qplot(): Histogram

- The function `grid.arrange()` in the `gridExtra` package is used to arrange the following three plots.

```
library(gridExtra);
p1 = qplot(data =mpg, cty, geom="histogram",bins = 10);
p2 = qplot(data =mpg, cty, fill = factor(cyl), bins = 10);   # defau
p3 = qplot(data =mpg, cty, facets = .~factor(cyl), binwidth = 5);
grid.arrange(p1,p2,p3, ncol=3);
```

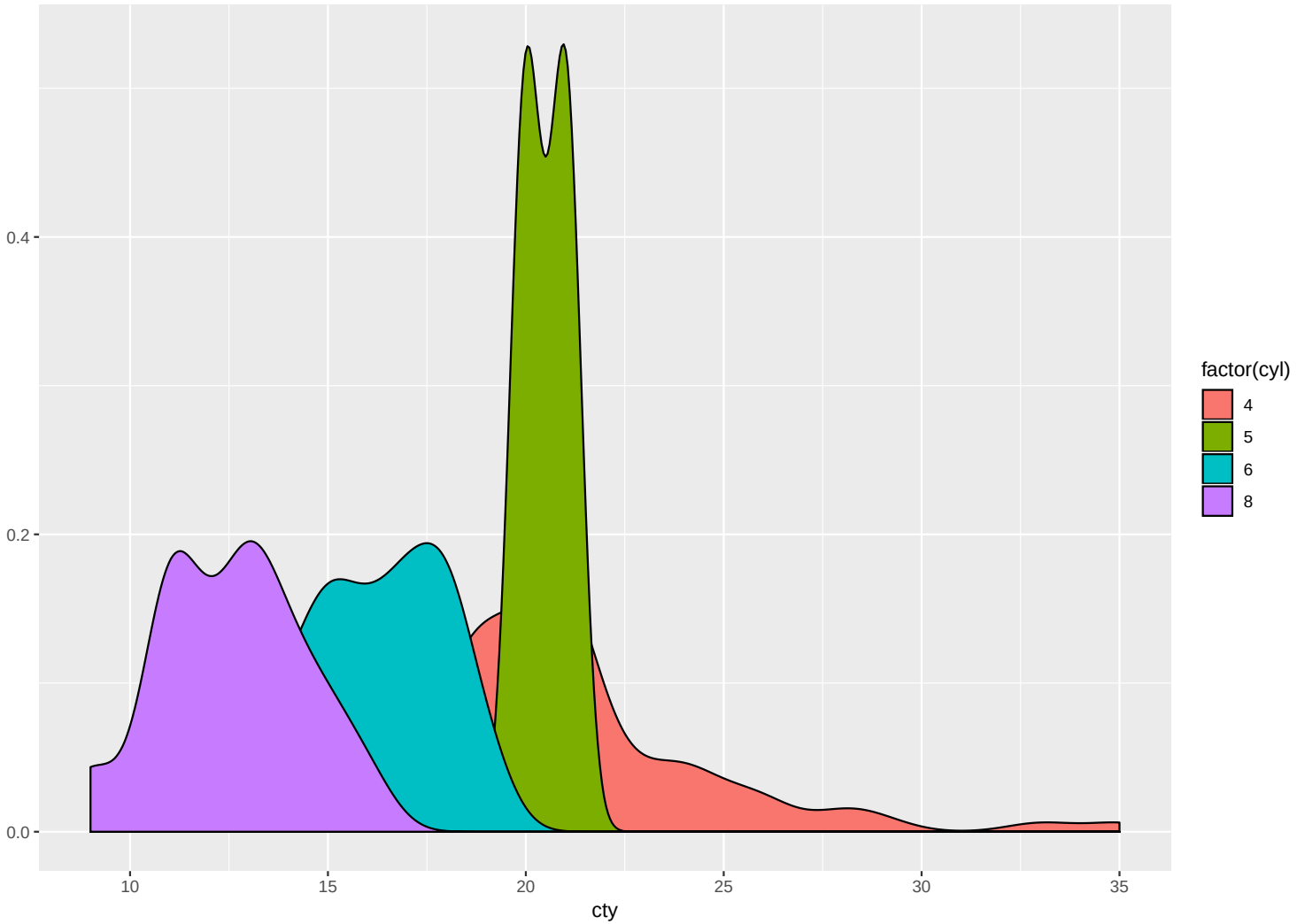


Quick plots with qplot(): Histogram

- Histogram is the sensible geom for continuous variables
- Automatic color setting (color/fill are grouping variables in ggplot2)
- Faceting is similar to the conditioning function | in the `Lattice` package.

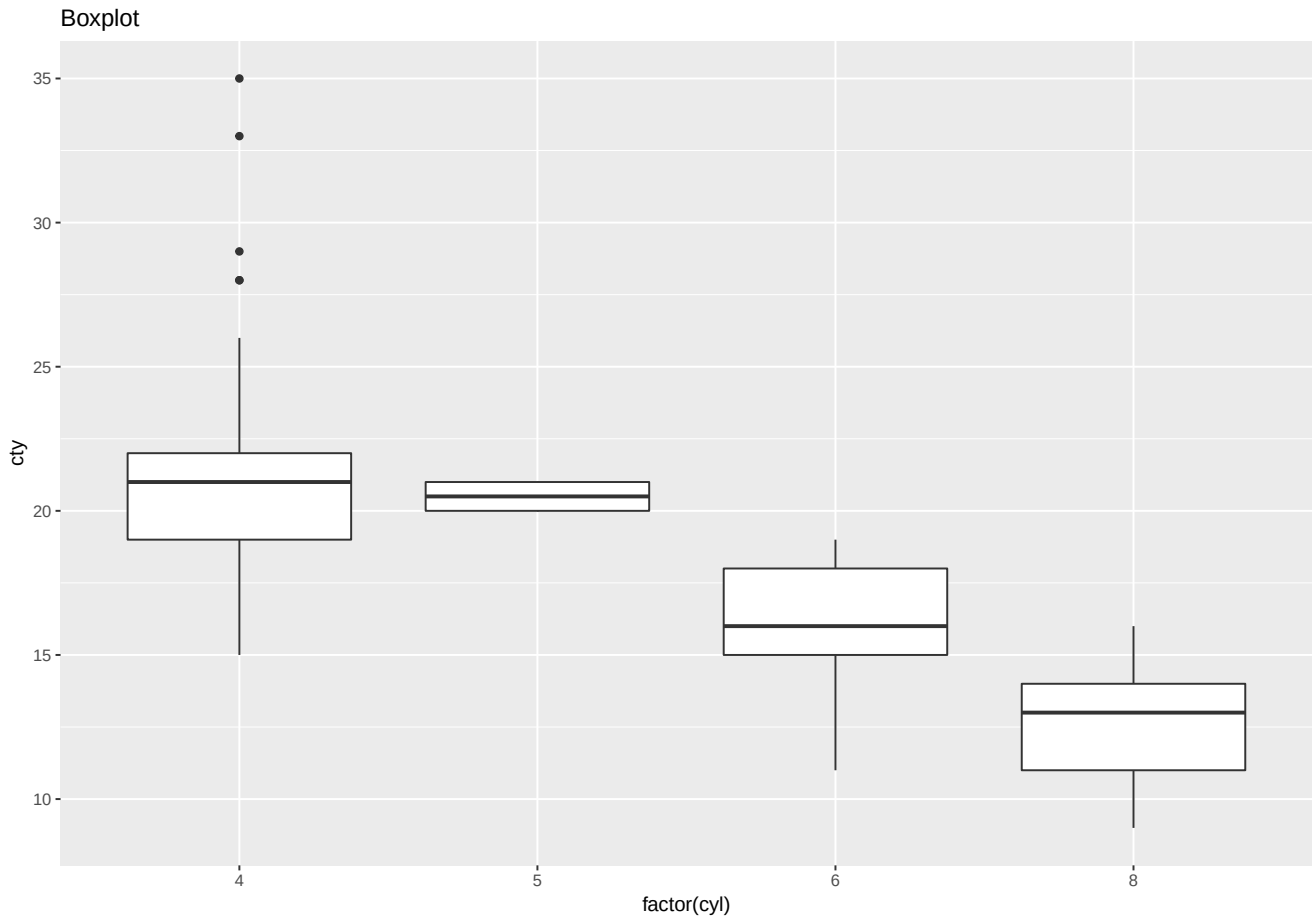
Quick plots with qplot(): Density plot

```
qplot(data = mpg, cty, geom = "density",  fill = factor(cyl));
```



Quick plots with qplot(): Boxplot with Grouping

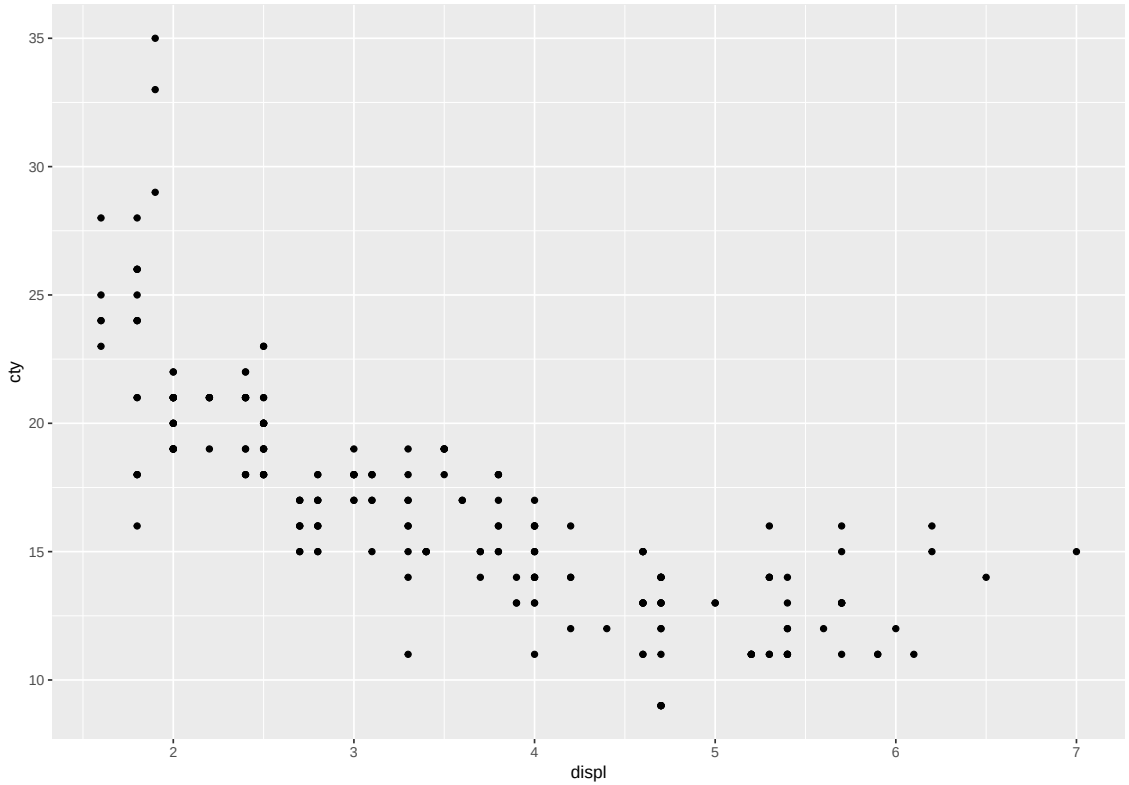
```
qplot(data = mpg, factor(cyl), cty, geom="boxplot", main="Boxplot");
```



- Following data are `x` (grouping) and `y` (response) variables.

Quick plots with qplot(): Scatter plot

```
qplot(data = mpg, displ, cty, geom = "point");
```



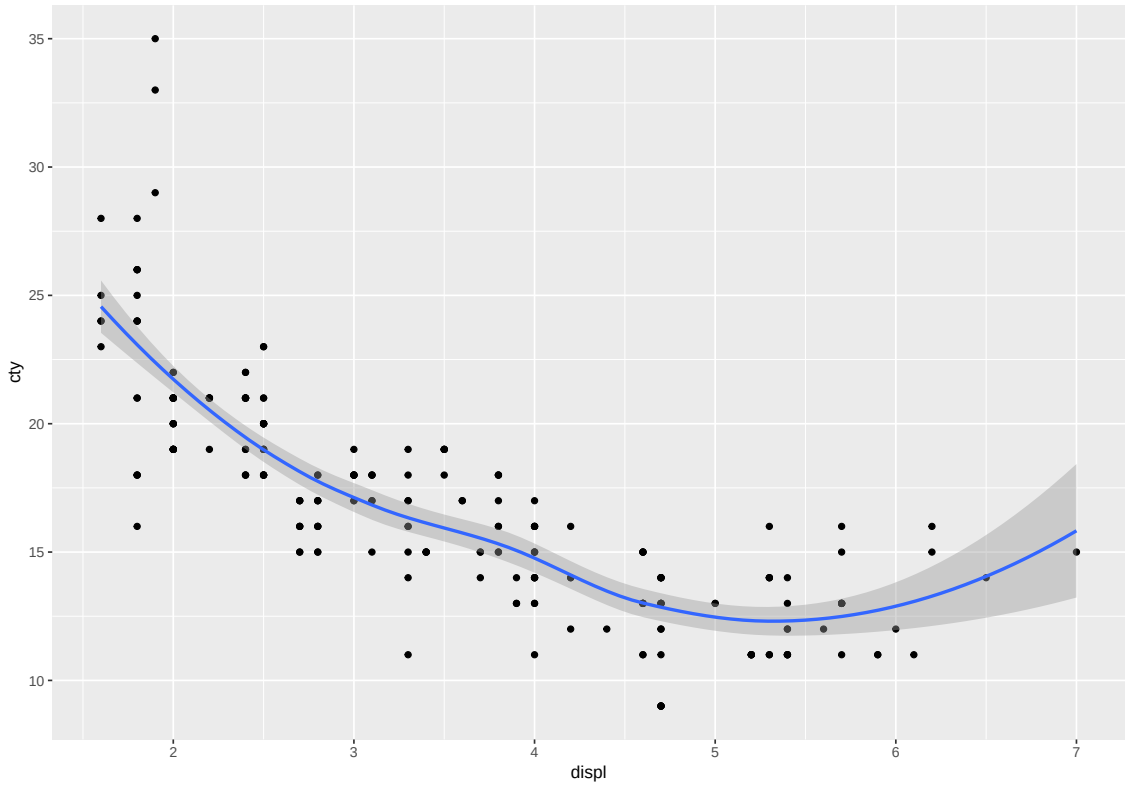

```
# qplot(data = mpg, displ, cty);
```

- `point` (scatter plot) is the sensible geom for two numerical variables

Quick plots with qplot(): Scatter plot

```
qplot(data = mpg, displ, cty, geom = c("point", "smooth"));
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

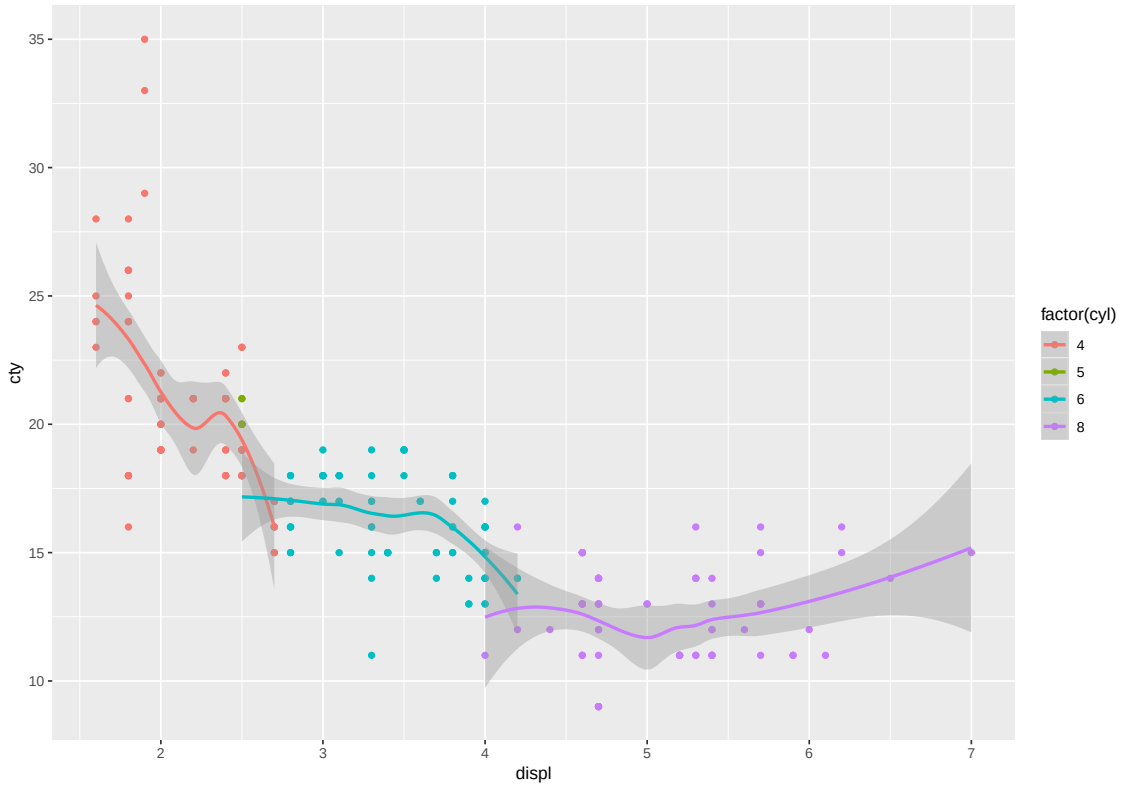


- Multiple geoms can be added together.

Quick plots with qplot(): Scatter plot

```
qplot(data = mpg, displ, cty, geom = c("point", "smooth"),
      color = factor(cyl)); # Color as a grouping variable
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



- Multiple geoms can be added together.

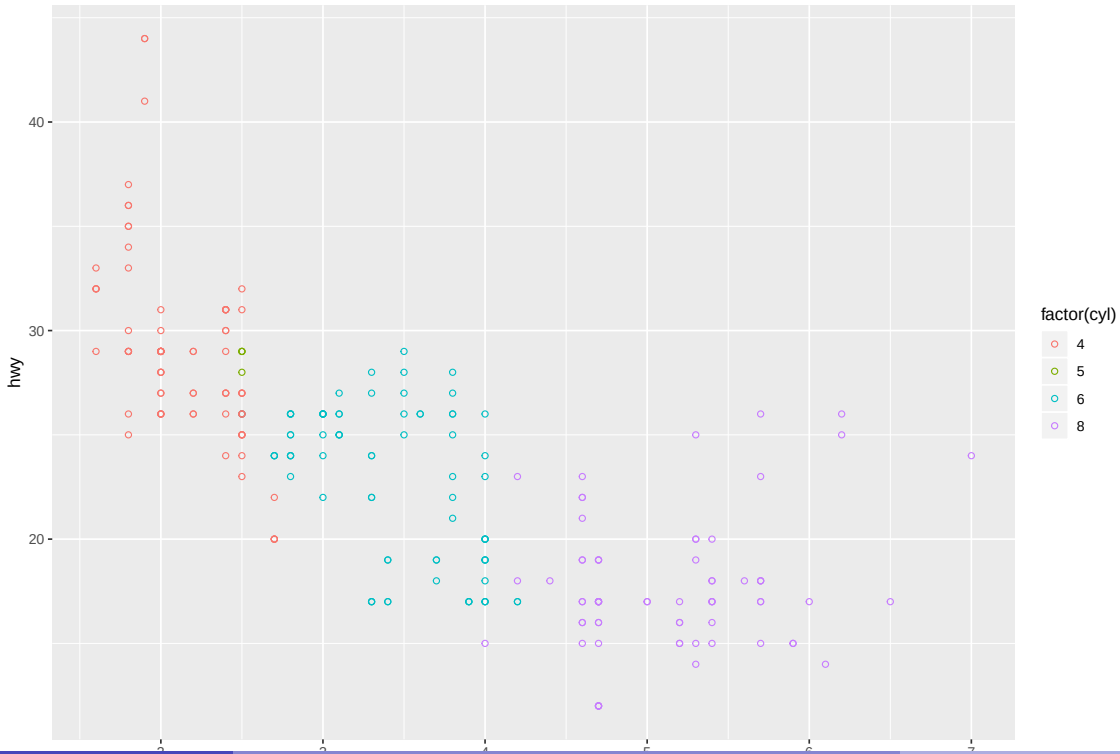
The `ggplot()` grammar

- `ggplot()` - grammar of graphics plot, which provides more controls than `qplot()`.
- Later in this data camp, `ggplot()` will also be used for animated plots.
- Key inputs for a ggplot graph:
  - **Data**: a data.frame to visualize
  - **Aesthetics**: mapping varialbes of the data to aesthetic attributes (position, size, shape, color, fill, transparency, . . .)
  - **Scales**: mapping values of the data to visual values for each aesthetic (e.g. position, color, fill and shape scales)
  - **Geometric objects**: point, line, polygon, histogram, quantile, bar, . . .
  - **Statistical transformations**: bin, boxplot, density, contour, function, . . .
  - **Coordinate system**: Cartesian, polar, map projection, . . .
  - **Facet**: display split data in multi-panels (aka conditioning)
  - **Theme**: control non-data visual elements (title, axes, tick, . . .)
- Every plot has three key components: data, aesthetics and geoms.

The ggplot() grammar

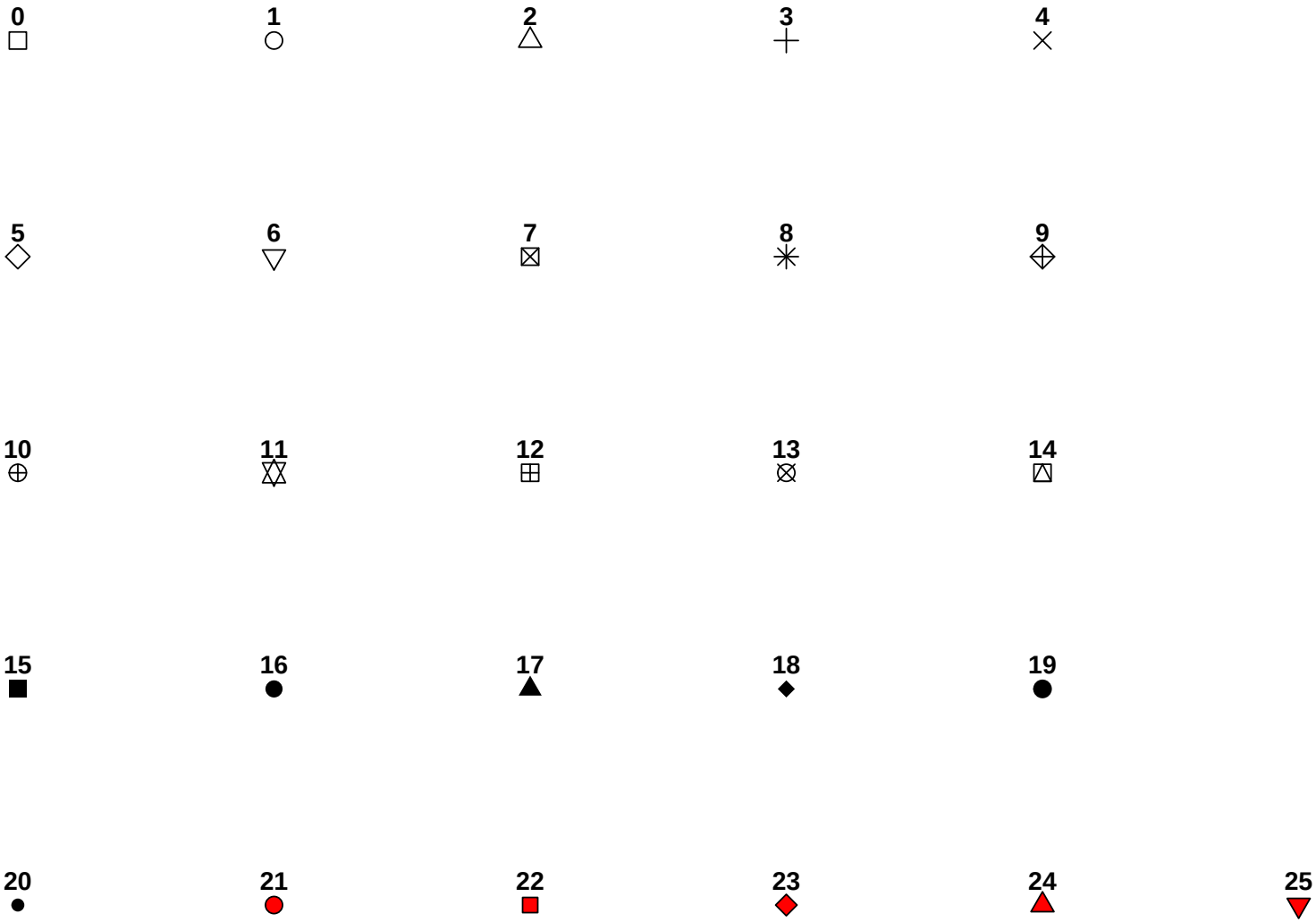
- We use an example of scatter plot of two variables and plot of a numerical variable to show the key components of ggplot().
- How are engine size and fuel economy related? Let's create a scatter plot of engine displacement and highway mpg with points colored by the number of cylinders.

```
ggplot(data=mpg, aes( displ, hwy, color=factor(cyl)) ) +
        geom_point(pch=21);
```



The ggplot() grammar

- The shapes of point available in R are as follows.



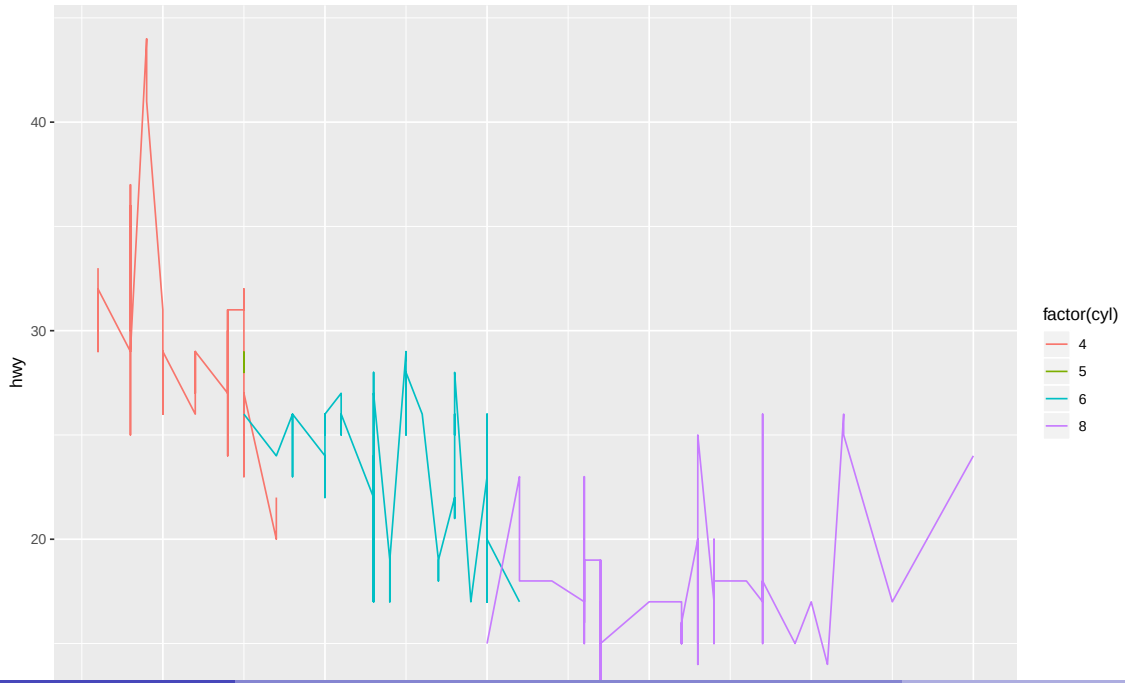
The `ggplot()` grammar: Mapping aesthetics to data

- How does `ggplot()` draw this plot?
- `aes()` function defines a mapping (by selecting the variables to be plotted and specifying how to present them in the graph, e.g. as x/y positions or characteristics such as size, shape, color, etc.
- Aesthetic mappings describe how variables in the data are mapped to visual properties (aesthetics) of geoms.
- In this example, the aesthetics are points according to the value of two variables, horizontal and vertical postion, point size, color and shape.

The ggplot() grammar: Mapping aesthetics to data

- **geoms** - graphical representations of the data in the plot (points, lines, bars). ggplot() offers many different geoms including:
  - geom_point() for scatter plots, dot plots, etc.
  - geom_boxplot() for boxplots
  - geom_line() for trend lines, time series, etc.
  - geom_smooth() for smoothing lines, produced by smoothing method in statistics

```
ggplot(data=mpg, aes(displ, hwy, color=factor(cyl)) ) +
        geom_line();
```



The ggplot() grammar: Faceting

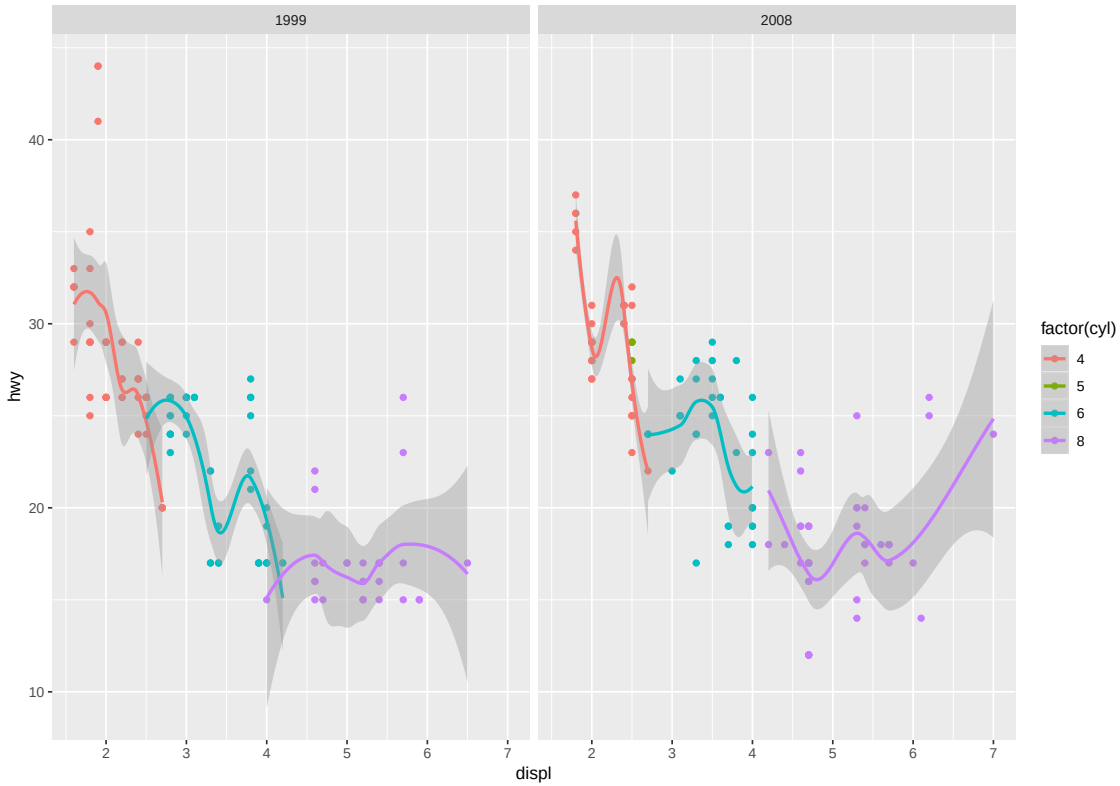
From the above graph, it can be seen that in ggplot2 we can produce many plots that don't make sense, yet are gramatically valid.

- **Facets** divide a plot into subplots based on the values of one or more categorical variables.
- There are two main functions for faceting:
  - **facet_grid()**: forms a matrix of panels defined by row and column faceting variables
  - **facet_wrap()**: wraps a 1d sequence of panels into 2d. This is generally a better use of screen space than `facet_grid()` because most displays are roughly rectangular.

The ggplot() grammar: Faceting

```
ggplot(data=mpg, aes(displ, hwy, color=factor(cyl)) ) +
  geom_point() +
  geom_smooth() +
  facet_grid(~year);
```

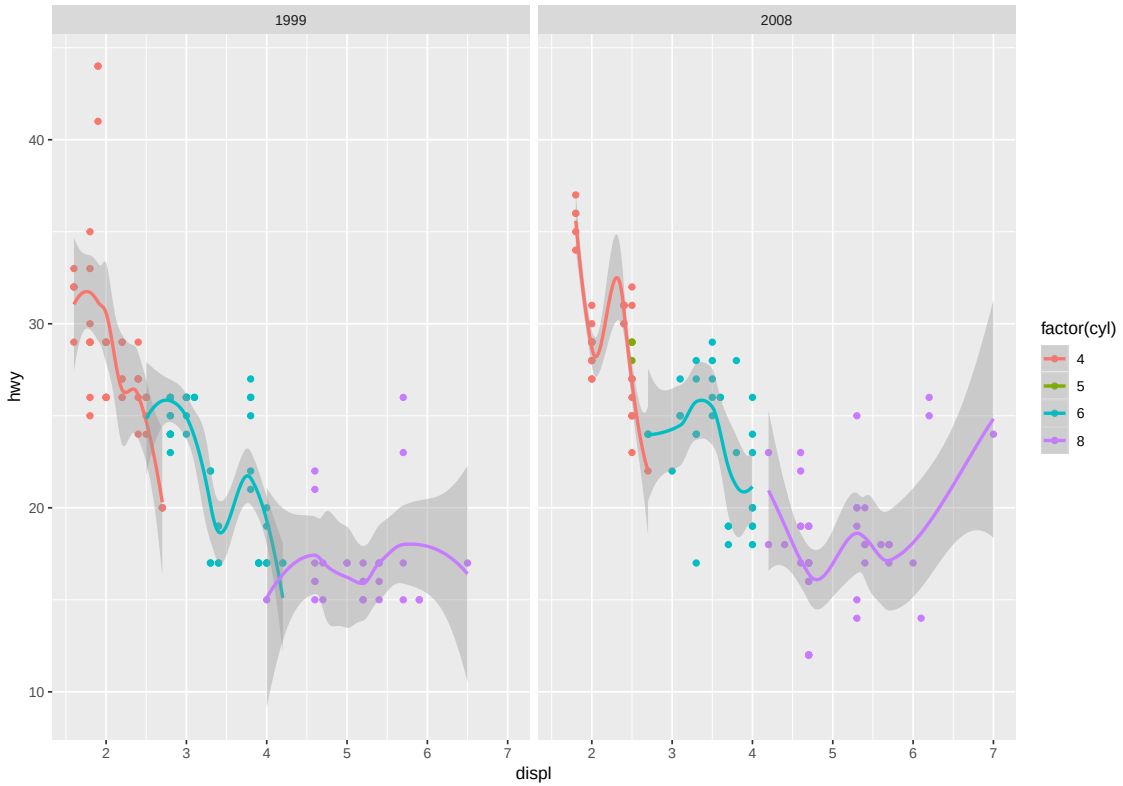
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



The ggplot() grammar: Faceting

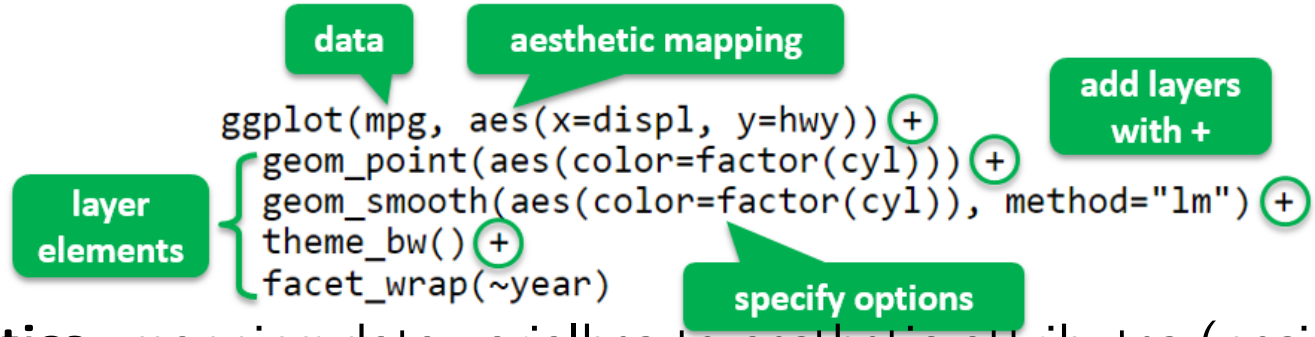
```
ggplot(data=mpg, aes(displ, hwy, color=factor(cyl)) ) +
  geom_point() +
  geom_smooth() +
  facet_wrap(~year);
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



The ggplot() grammar: layers

- ggplot graphics are built step by step by adding new elements. Adding layers in this fashion allows for extensive flexibility and customization of plots.

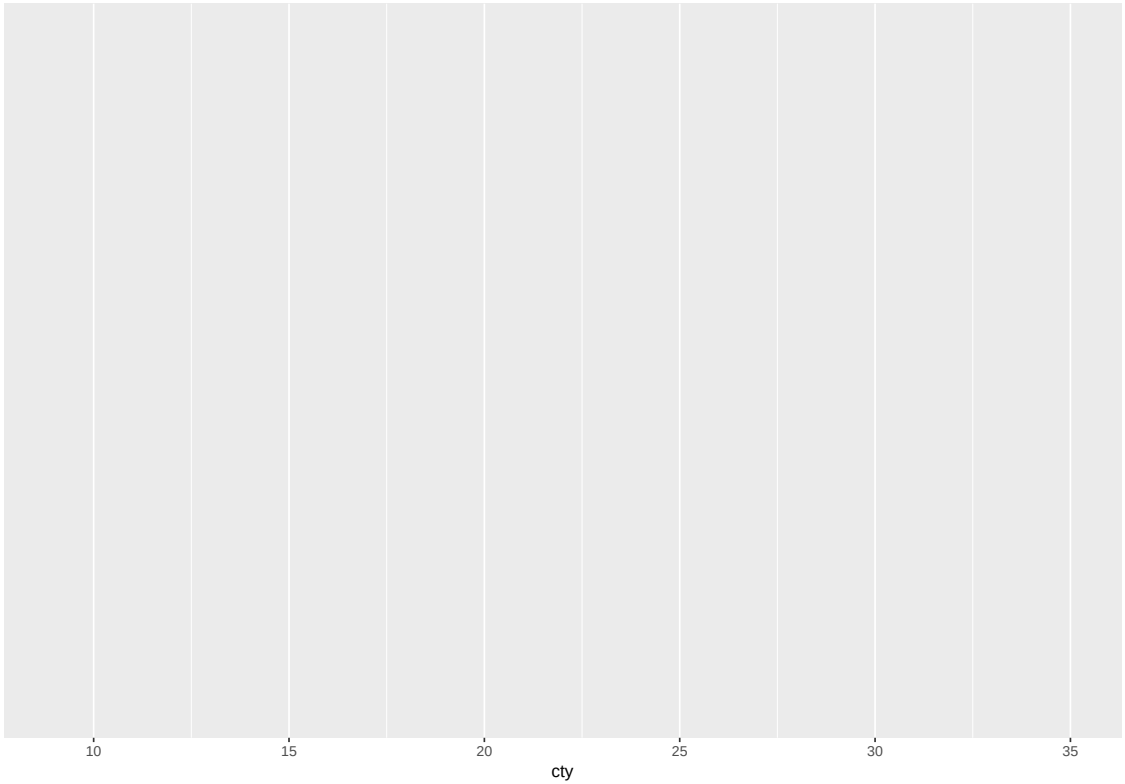


- **Aesthetics**: mapping data varialbes to aesthetic attributes (position, size, shape, color, . . .)
- **Geometric objects**: point, line, polygon, histogram, quantile, bar, . . .
- **Statistical transformations**: bin, boxplot, density, contour, function, . . .
- Other components for ggplots: **scales** (mapping values of the data to visual values for each aesthetic, e.g. position, color, fill and shape scales); **coordinate system** (cartesian, polar, map projection, ..); **facet** (conditioning display split data in multi-panels; theme (control non-data visual elements (title, axes, tick, . . .)

The ggplot() grammar: layers

- We start by creating a plot, named **P**, and finish by adding layers.

```
P = ggplot(data=mpg, aes(x = cty));
P;
```



The ggplot() grammar: layers

The following are some possible layers for a numerical variable:

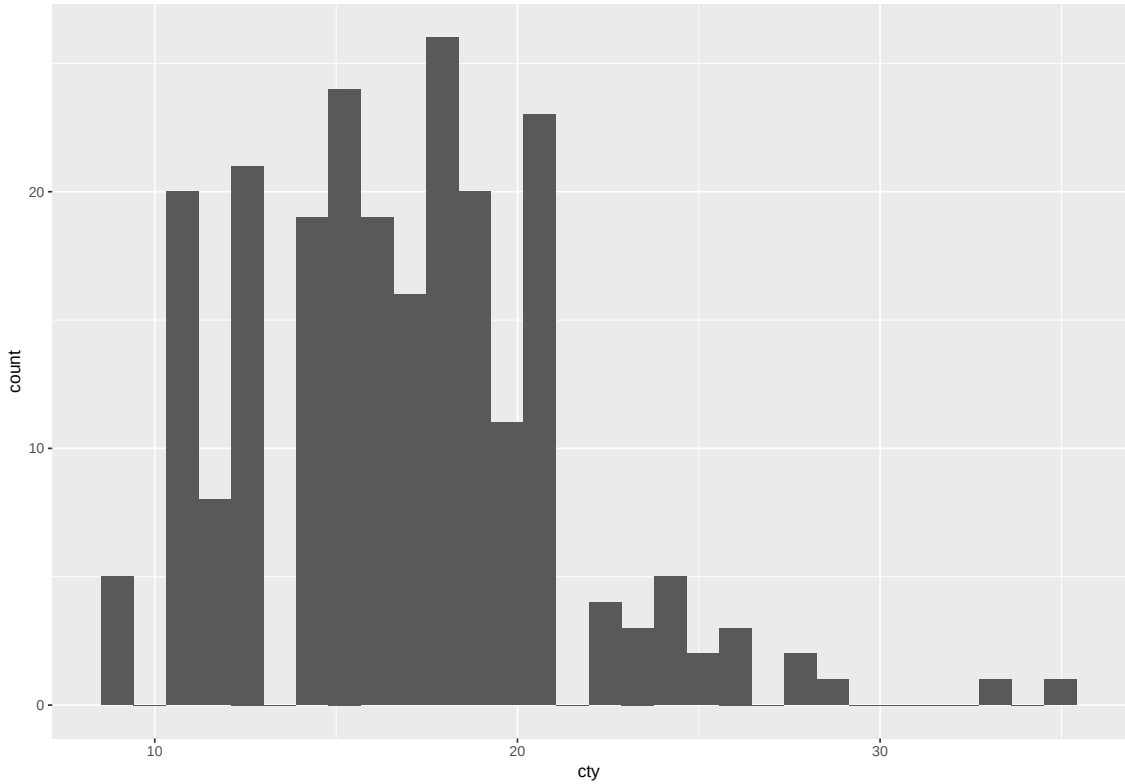
- geom_area() for area plot
- geom_density() for density plot
- geom_dotplot() for dot plot
- geom_freqpoly() for frequency polygon
- geom_histogram() for histogram plot
- stat_ecdf() for empirical cumulative density function
- stat_qq() for quantile - quantile plot

The ggplot() grammar: layers

- geom_histogram(): Histogram

```
P + geom_histogram();
```

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

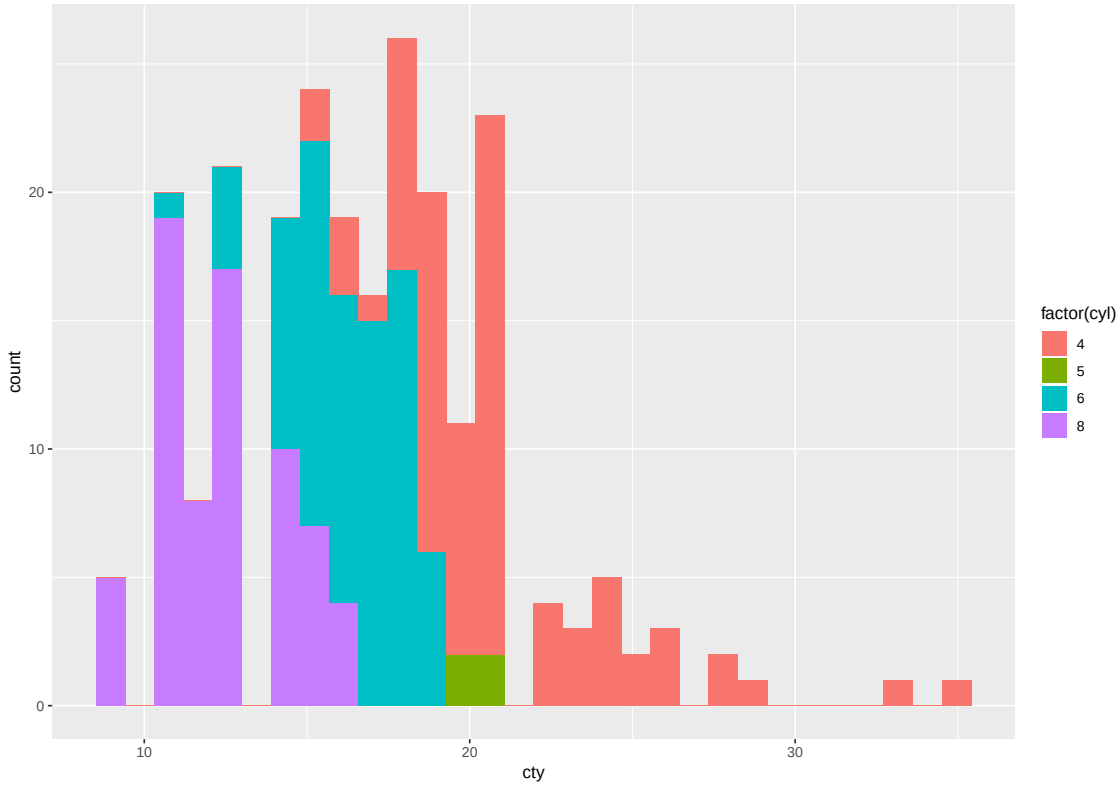


The ggplot() grammar: layers

- geom_histogram(): Histogram

```
# # change bin colors by cyl
P + geom_histogram(aes(fill = factor(cyl)));
```

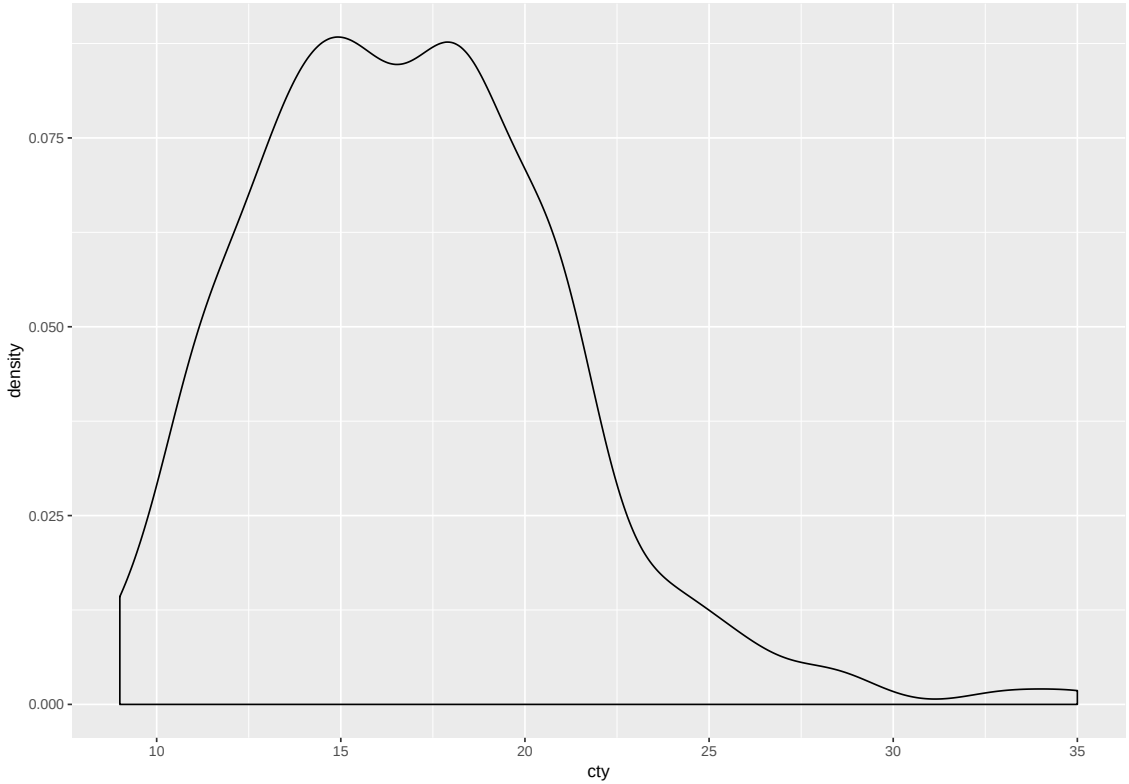
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`
```



The ggplot() grammar: layers

- geom_density(): kernal density estimate

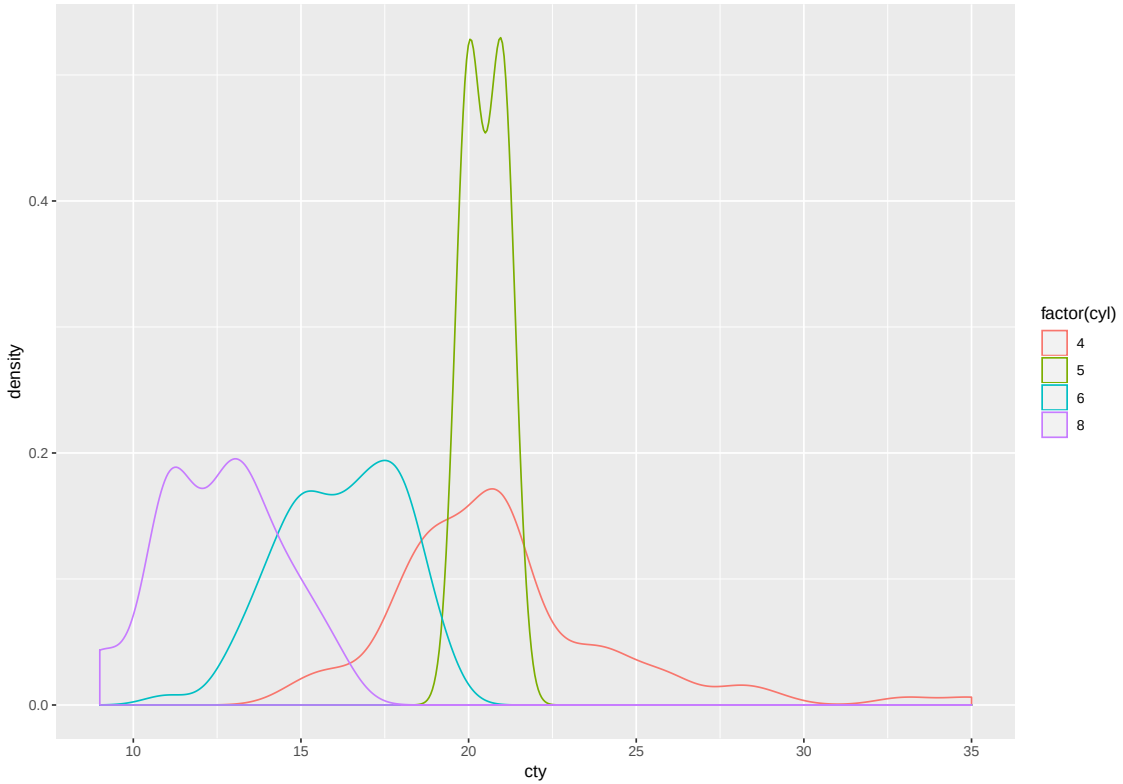
```
P + geom_density();
```



The ggplot() grammar: layers

- geom_density(): kernal density estimate

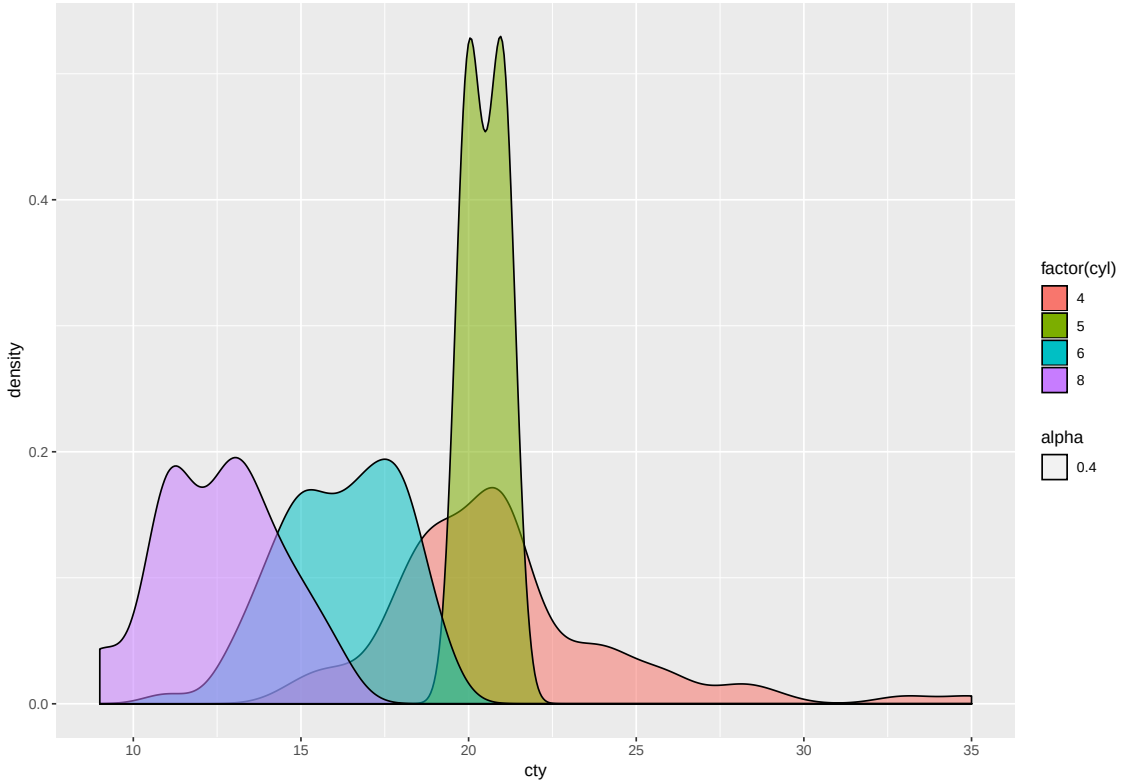
```
## change line colors by cyl
P + geom_density(aes(color = factor(cyl)));
```



The ggplot() grammar: layers

- geom_density(): kernal density estimate

```
# Use semi-transparent fill: alpha = 0.4
P + geom_density(aes(fill = factor(cyl),alpha=0.4));
```



- To customize the plot, these arguments can be used: alpha, color, fill, linetype, size. Learn more here: ggplot2 density plot.

Questions?