

Getting started with ggplot2

STAT 133

Gaston Sanchez

`github.com/ucb-stat133/stat133-fall-2016`

ggplot2

Resources for "ggplot2"

- ▶ Documentation: <http://docs.ggplot2.org/>
- ▶ Book: **ggplot2: Elegant Graphics for Data Analysis** (by Hadley Wickham)
- ▶ Book: **R Graphics Cookbook** (by Winston Chang)
- ▶ RStudio ggplot2 cheat sheet

<https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf>

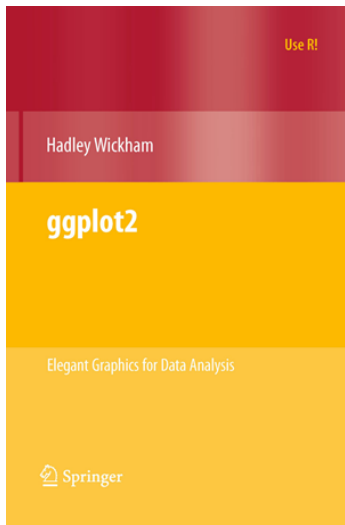
package "ggplot2"

```
# remember to install ggplot2  
# (just once)  
install.packages("ggplot2")
```

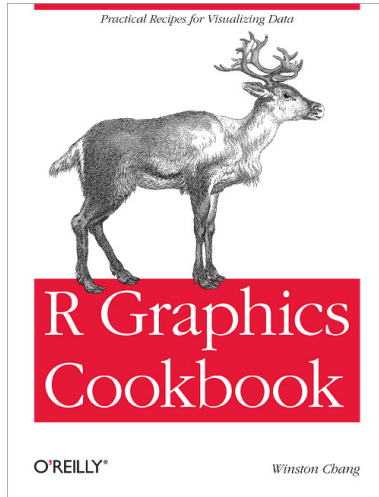
```
# load ggplot2  
library(ggplot2)
```

```
# see basic documentation  
?ggplot
```

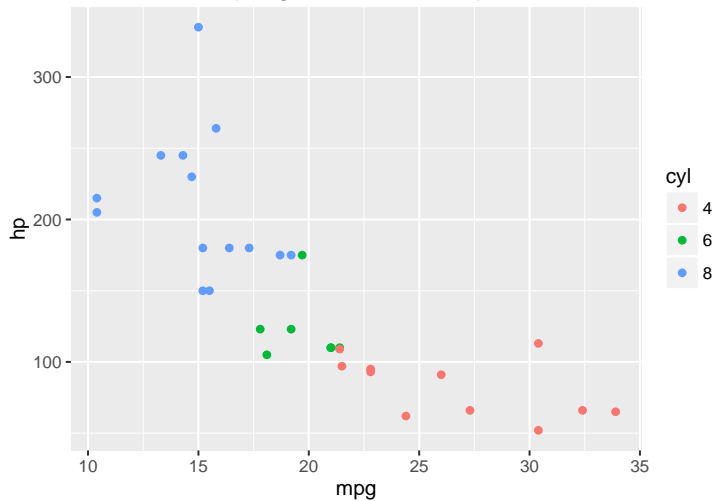
ggplot2 book



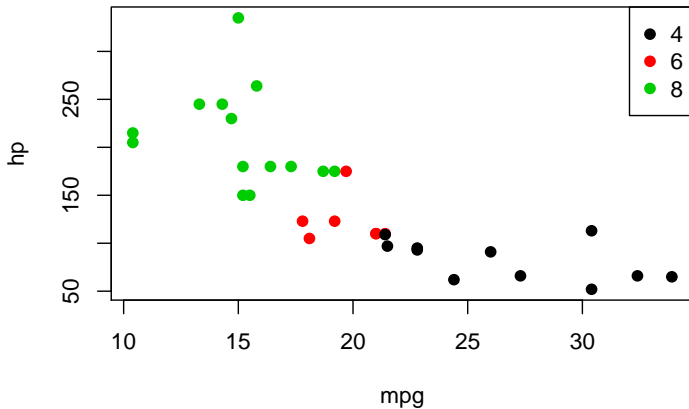
R Graphics Cookbook



Miles per gallon –vs– Horsepower



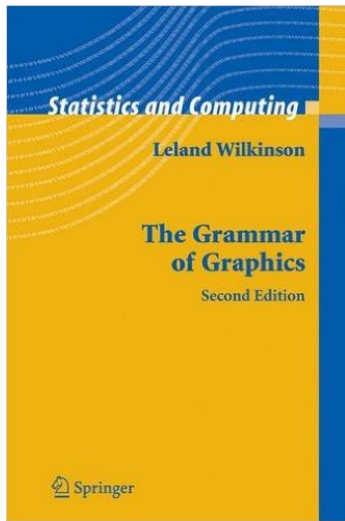
Miles per gallon –vs– Horsepower



About "ggplot2"

- ▶ "ggplot2" (by Hadley Wickham) is an R package for producing statistical graphics
- ▶ It provides a framework based on Leland Wilkinson's **Grammar of Graphics**
- ▶ "ggplot2" provides beautiful plots while taking care of fiddly details like legends, axes, colors, etc.
- ▶ "ggplot2" is built on the R graphics package "grid"
- ▶ Underlying philosophy is to describe a wide range of graphics with a compact syntax and independent components

The Grammar of Graphics



About the Grammar of Graphics

- ▶ *The Grammar of Graphics* is Wilkinson's attempt to define a theoretical framework for graphics
- ▶ **Grammar:** Formal system of rules for generating graphics
 - Some rules are mathematic
 - Some rules are aesthetic

About the Grammar of Graphics

3 Stages of Graphic Creation

- ▶ **Specification:** link data to graphic objects
- ▶ **Assembly:** put everything together
- ▶ **Display:** render of a graphic

About the Grammar of Graphics

Specification

Link data to graphic objects

- ▶ Data
- ▶ Transformation of variables (e.g. aggregation)
- ▶ Scale transformations (e.g. log)
- ▶ Coordinate system (e.g. cartesian)
- ▶ Graphic Elements (e.g. points, lines)
- ▶ Guides (e.g. labels, legends)

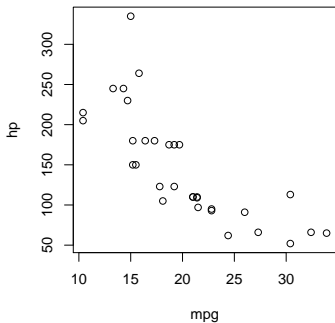
R package "ggplot2"

About "ggplot2"

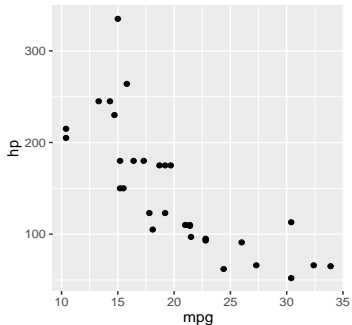
- ▶ Default appearance of plots carefully chosen
- ▶ Designed with visual perception in mind
- ▶ Inclusion of some components, like legends, are automated
- ▶ Great flexibility for annotating, editing, and embedding output

Base graphics -vs- "ggplot2"

base graphics



ggplot2



About "ggplot2"

- ▶ "ggplot2" is the name of the package
- ▶ The gg in "ggplot2" stands for *Grammar of Graphics*
- ▶ Inspired in the **Grammar of Graphics** by Lee Wilkinson
- ▶ "ggplot" is the class of objects (plots)
- ▶ ggplot() is the main function in "ggplot2"

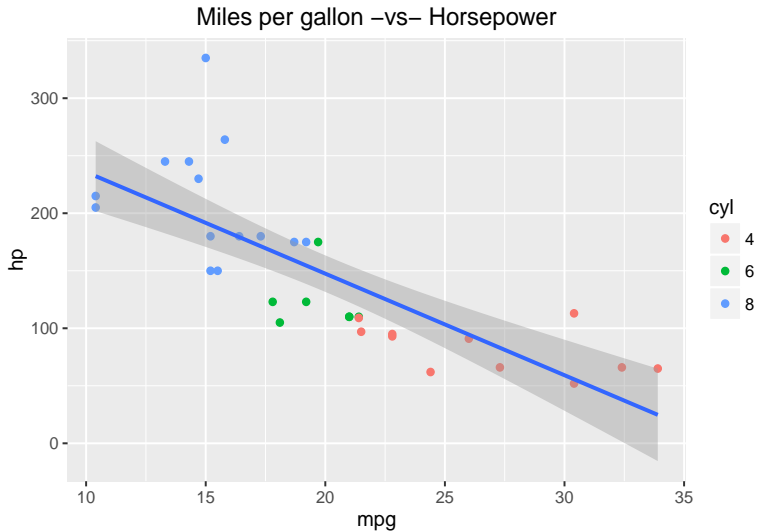
What is a Statistical Graphic?

Some Data set

mtcars

##	mpg	hp	cyl
## Mazda RX4	21.0	110	6
## Mazda RX4 Wag	21.0	110	6
## Datsun 710	22.8	93	4
## Hornet 4 Drive	21.4	110	6
## Hornet Sportabout	18.7	175	8
## Valiant	18.1	105	6
## Duster 360	14.3	245	8
## Merc 240D	24.4	62	4
## Merc 230	22.8	95	4
## Merc 280	19.2	123	6

What is a statistical graphic?



What is a statistical graphic?

Elements to draw the chart “manually”

What is a statistical graphic?

Elements to draw the chart “manually”

- ▶ coordinate system
- ▶ x and y axis (intervals)
- ▶ axis tick marks
- ▶ axis labels, and title
- ▶ points (with colors)
- ▶ regression line (and ribbon)
- ▶ legend

What is a statistical graphic?

Simply put, a statistical graphic is:

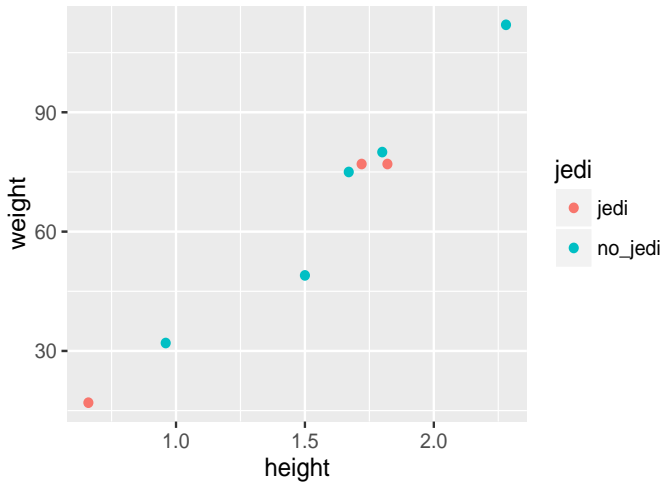
- ▶ A mapping from data to aesthetic attributes (color, shape, size) of geometric objects (points, lines, bars)
- ▶ A plot may also contain statistical transformations of the data
- ▶ A plot is drawn on a specific coordinate system
- ▶ Sometimes faceting can be used to get the same plot for different subsets of the dataset

Starting with "ggplot2"

starwarstoy.csv

##		name	gender	height	weight	jedi	species	weapon
## 1	Luke Skywalker	male	1.72	77	jedi	human	lightsaber	
## 2	Leia Skywalker	female	1.50	49	no_jedi	human	blaster	
## 3	Obi-Wan Kenobi	male	1.82	77	jedi	human	lightsaber	
## 4	Han Solo	male	1.80	80	no_jedi	human	blaster	
## 5	R2-D2	male	0.96	32	no_jedi	droid	unarmed	
## 6	C-3PO	male	1.67	75	no_jedi	droid	unarmed	
## 7	Yoda	male	0.66	17	jedi	yoda	lightsaber	
## 8	Chewbacca	male	2.28	112	no_jedi	wookiee	bowcaster	

Scatterplot



Main steps in creating ggplot graphics

1 Dataset

A	B	C	D	E	F

2 Which variables

A	B	C	D	E	F

3 Geometric objects

● *points*

abcd *text*

~ *lines*

■ *bars*

4 Aesthetics

x = A

y = B

color = C

size = *default*

shape = *default*

Building a scatterplot

User specifications

- ▶ Dataset: `starwars`
- ▶ Variables: `height`, `weight`, `jedi`
- ▶ Geoms: `points`
- ▶ Aesthetics (attributes):
 - `x`: `height`
 - `y`: `weight`
 - **`color`**: `jedi`

Scatterplot with "ggplot2"

```
ggplot(data = starwars) +  
  geom_point(aes(x = height, y = weight, color = jedi))
```

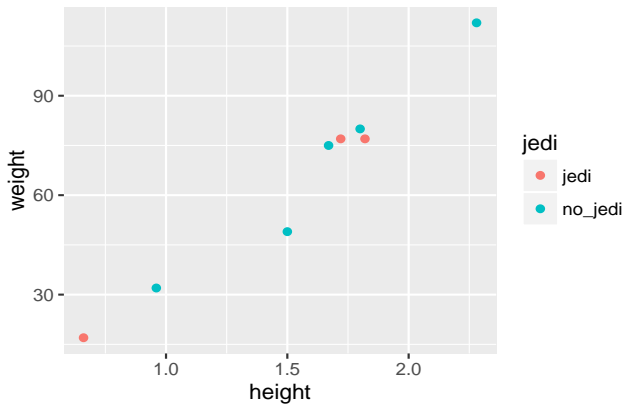
Scatterplot with "ggplot2"

```
ggplot(data = starwars) +  
  geom_point(aes(x = height, y = weight, color = jedi))
```

- ▶ `ggplot()` initializes a "ggplot" object
- ▶ specify the dataset with `data`
- ▶ type of geometric object: `geom_point()`
- ▶ mapping aesthetic attributes to variables with `aes()`
 - x-position: `height`
 - y-position: `weight`
 - color: `jedi`

Scatterplot with "ggplot2"

```
ggplot(data = starwars) +  
  geom_point(aes(x = height, y = weight, color = jedi))
```



Scatterplot with "ggplot2"

Automated things in "ggplot2"

- ▶ Axis labels
- ▶ Legends (position, labels, symbols)
- ▶ Choose of colors for points
- ▶ Background color (e.g. gray)
- ▶ Grid lines (major and minor)
- ▶ Axis tick marks

you can always change the automated elements

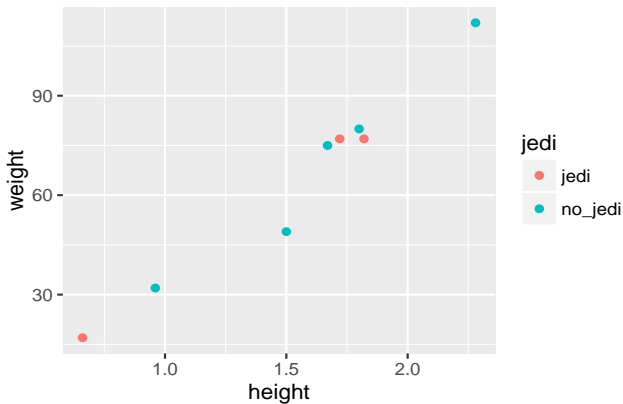
"ggplot2" graphics

Philosophy of "ggplot2"

A graphic is a **mapping** from **data** to **aesthetic attributes** (color, shape, size) of **geometric objects** (points, lines, bars)

Scatterplot with "ggplot2"

```
ggplot(data = starwars) +  
  geom_point(aes(x = height, y = weight, color = jedi))
```



Mapping

data values

height	weight	jedi
1.72	77	jedi
1.50	49	no_jedi
1.82	77	jedi
1.80	80	no_jedi
0.96	32	no_jedi
1.67	75	no_jedi
0.66	17	jedi
2.28	112	no_jedi



aesthetic attributes

x	y	color
x_1	y_1	#F8766D
x_2	y_2	#00BFC4
x_3	y_3	#F8766D
x_4	y_4	#00BFC4
x_5	y_5	#00BFC4
x_6	y_6	#00BFC4
x_7	y_7	#F8766D
x_8	y_8	#00BFC4

"ggplot2" graphics

Philosophy of "ggplot2"

A graphic is a **mapping** from **data** to **aesthetic attributes** (color, shape, size) of **geometric objects** (points, lines, bars)

- ▶ `ggplot(data, ...)`
- ▶ `aes()`
- ▶ `geom_objects()`

Scatterplot with "ggplot2"

How does "ggplot2" work?

- ▶ plots are created piece-by-piece
- ▶ plot components added with **+** operator
- ▶ aesthetic attributes mapped to data values
- ▶ computation of scales for aesthetic attributes

How does it work?

Usually, we specify the data and variables inside the function `ggplot()`

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

Note the use of the internal function `aes()` to *map* x to mpg, and y to hp.

Then we **add a layer** of geometric objects: points in this case

```
+ geom_point()
```

Some alternative options

```
# option A  
ggplot(data = starwars,  
       aes(x = height, y = weight, color = jedi)) +  
  geom_point()
```

Some alternative options

```
# option A  
ggplot(data = starwars,  
       aes(x = height, y = weight, color = jedi)) +  
  geom_point()
```

```
# option B  
ggplot(data = starwars) +  
  geom_point(aes(x = height, y = weight, color = jedi))
```

Some alternative options

option A

```
ggplot(data = starwars,  
       aes(x = height, y = weight, color = jedi)) +  
  geom_point()
```

option B

```
ggplot(data = starwars) +  
  geom_point(aes(x = height, y = weight, color = jedi))
```

option C

```
ggplot() +  
  geom_point(data = starwars,  
            aes(x = height, y = weight, color = jedi))
```


Main inquiries

Always ask yourself ...

- ▶ What is the **data** set of interest?
- ▶ What **variables** will be used to make the plot?
- ▶ What **graphics shapes** will be used to display?
- ▶ What **features** of the shapes will be used to represent the data values?

"ggplot2" basics

- ▶ The data must be in a `data.frame`
- ▶ Variables are mapped to aesthetic attributes
- ▶ Aesthetic attributes belong to geometric objects **geoms** (points, lines, polygons)

Basic Terminology

- ▶ **ggplot()** - The main function where you specify the dataset and variables to plot
- ▶ **geoms** - geometric objects
 - `geom_point()`, `geom_bar()`, `geom_line()`, `geom_density()`
- ▶ **aes** - aesthetics (i.e. attributes)
 - shape, color, fill, linetype

Warning

"ggplot2" comes with the function `qplot()` (i.e. *quick plot*).
Avoid using it!

As Karthik Ram says: “you’ll end up unlearning and relearning a good bit”

Scatterplot with "ggplot2"

Terminology

- ▶ aesthetic mappings
- ▶ geometric objects
- ▶ statistical transformations
- ▶ scales
- ▶ non-data elements (themes & elements)
- ▶ facets

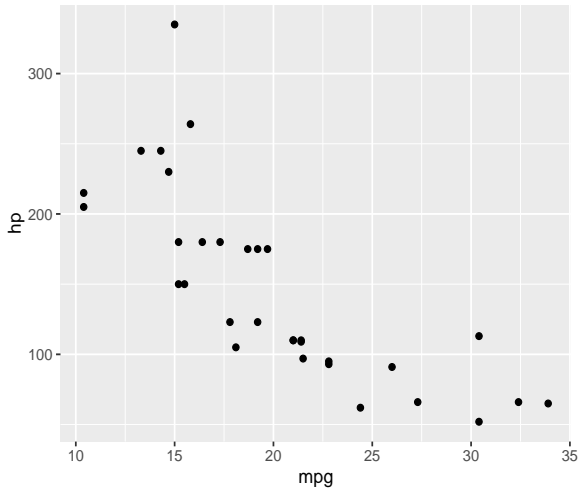
Considerations

Specifying graphical elements from 3 sources:

- ▶ The data values (represented by the geometric objects)
- ▶ The scales and coordinate system (axes, legends)
- ▶ Plot annotations (background, title, grid lines)

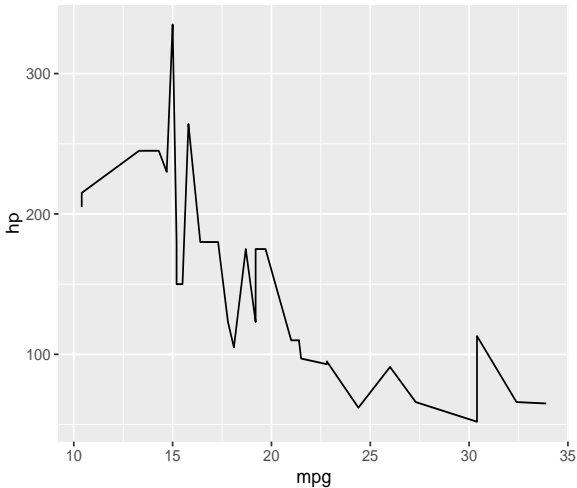
Scatterplot with `geom_point`

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



Another geom

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



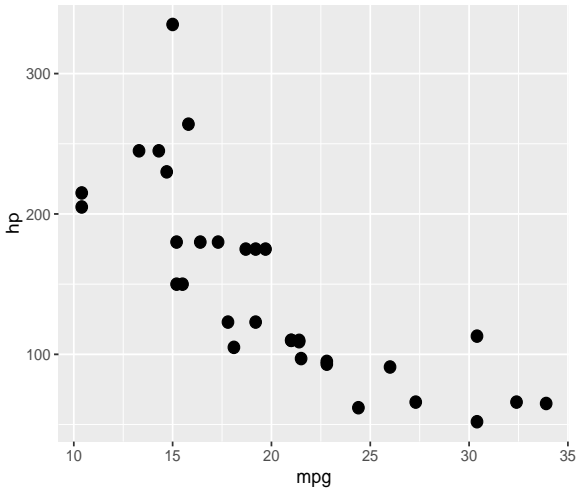
Mapping Attributes

-vs-

Setting Attributes

Increase size of points

```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(size = 3)
```



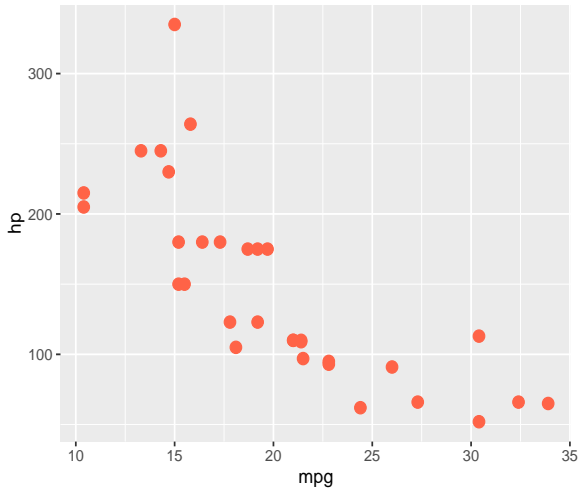
How does it work?

To increase the size of points, we **set** the aesthetic size to a constant value of 3 (inside the *geoms* function):

```
+ geom_point(size = 3)
```

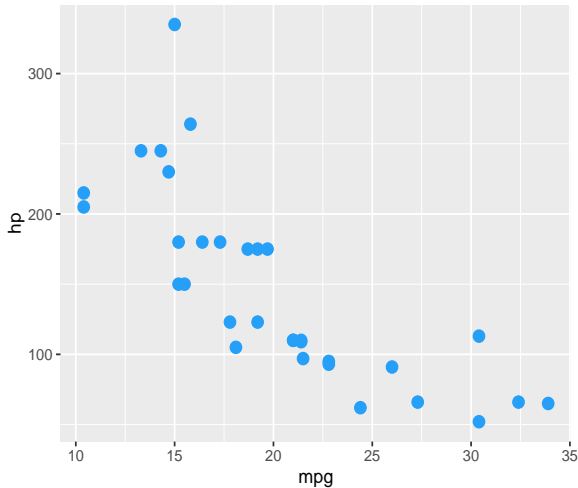
Adding color

```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(size = 3, color = "tomato")
```



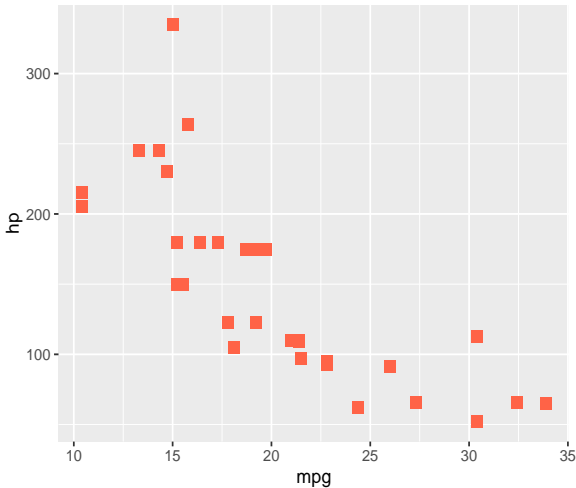
Adding color

```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(size = 3, color = "#259ff8")
```



Changing points shape

```
# 'shape' accepts 'pch' values  
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(size = 3, color = "tomato", shape = 15)
```



Setting and Mapping

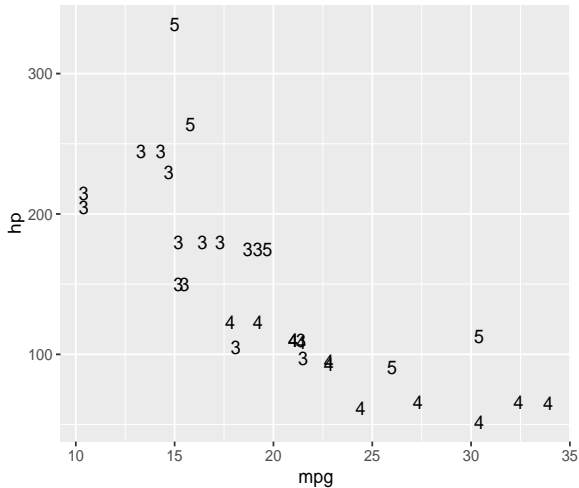
Aesthetic attributes can be either **mapped** —via `aes()`— or **set**

```
# mapping aesthetic color  
ggplot(mtcars, aes(x = mpg, y = hp)) +  
  geom_point(aes(color = cyl))
```

```
# setting aesthetic color  
ggplot(mtcars, aes(x = mpg, y = hp)) +  
  geom_point(color = "blue")
```

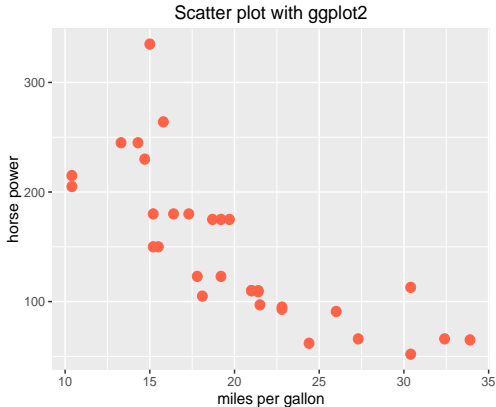
Geom text, and mapping labels

```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_text(aes(label = gear))
```



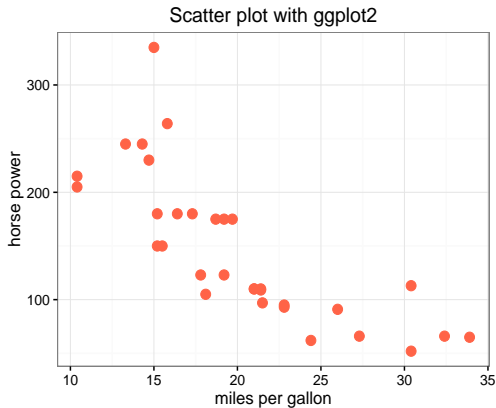
Changing axis labels and title

```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(size = 3, color = "tomato") +  
  xlab("miles per gallon") +  
  ylab("horse power") +  
  ggtitle("Scatter plot with ggplot2")
```

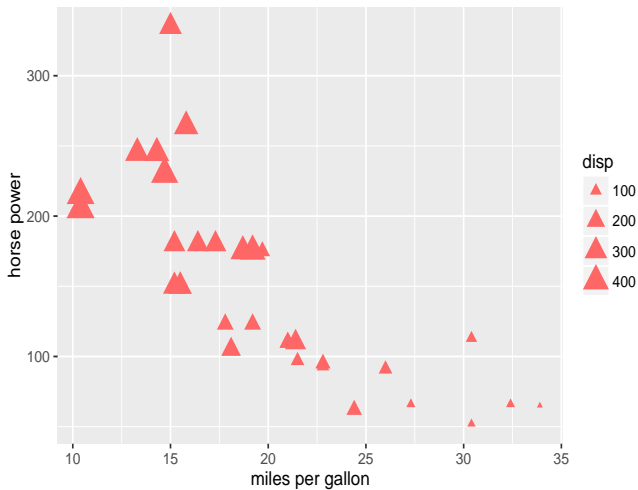


Changing background theme

```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(size = 3, color = "tomato") +  
  xlab("miles per gallon") +  
  ylab("horse power") +  
  ggtitle("Scatter plot with ggplot2") +  
  theme_bw()
```



Your turn: Replicate this figure



Your turn: Replicate this figure

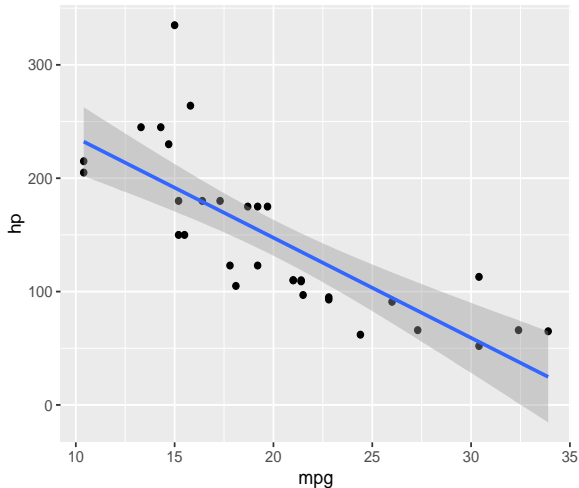
- ▶ Specify a color in hex notation
- ▶ Change the shape of the point symbol
- ▶ Map `disp` to attribute size of points
- ▶ Add axis labels

Your turn

```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(aes(size = disp),  
             color = "#ff6666", shape = 17) +  
  xlab("miles per gallon") +  
  ylab("horse power")
```

More geoms

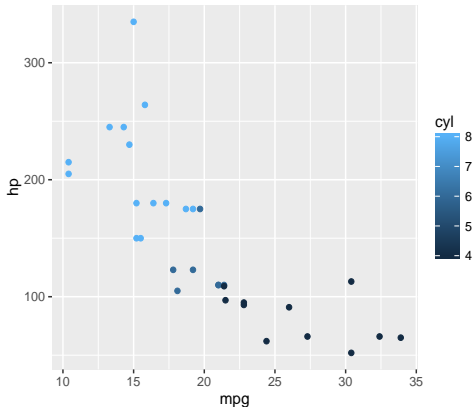
```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point() +  
  geom_smooth(method = "lm")
```



More geoms

We can map variable to a color aesthetic. Here we map color to `cyl` (cylinders)

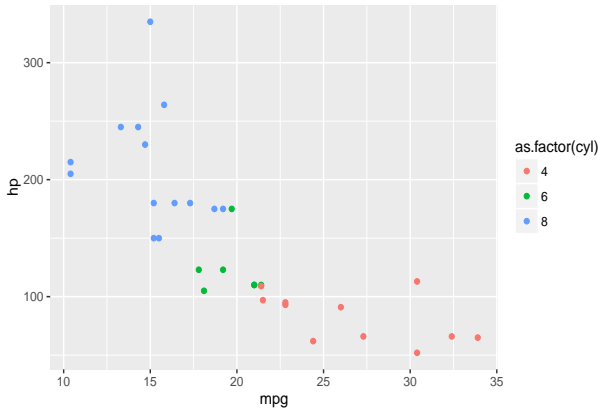
```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(aes(color = cyl))
```



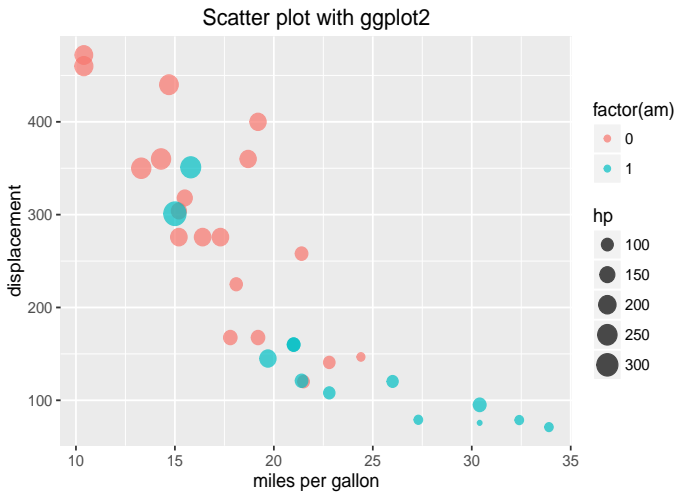
More geoms

If the variable that maps to color is a factor, then the color scale will change

```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(aes(color = as.factor(cyl)))
```



Your turn: Replicate this figure



Your turn: example 2

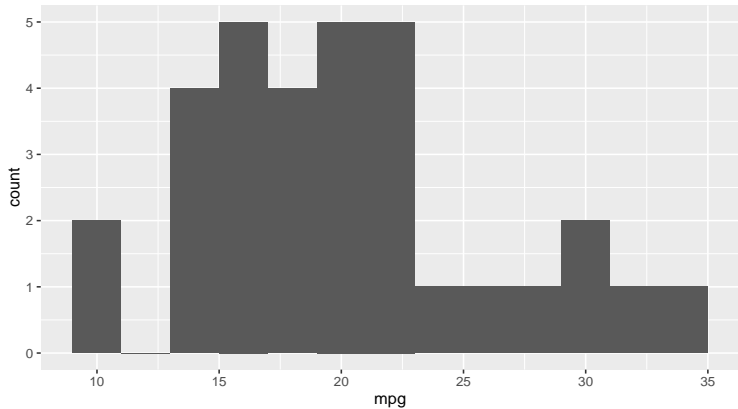
- ▶ Map `hp` to attribute size of points
- ▶ Map `am` (as factor) to attribute color points
- ▶ Add an alpha transparency of 0.7
- ▶ Change the shape of the point symbol
- ▶ Add axis labels
- ▶ Add a title

Your turn: example 2

```
ggplot(data = mtcars, aes(x = mpg, y = disp)) +  
  geom_point(aes(size = hp, color = factor(am)),  
             alpha = 0.7) +  
  xlab("miles per gallon") +  
  ylab("displacement") +  
  ggtitle("Scatter plot with ggplot2")
```

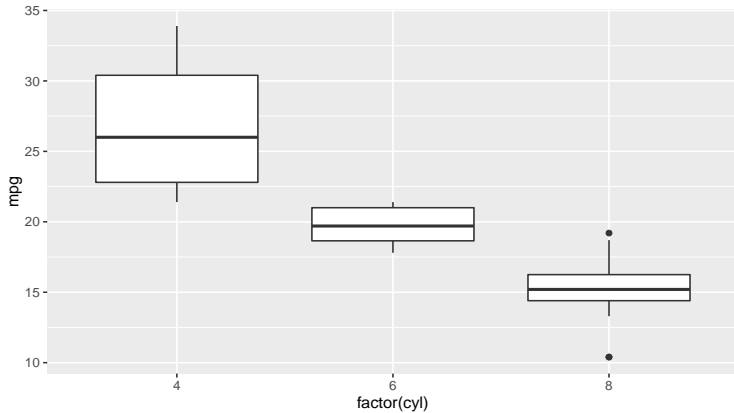
Histogram

```
ggplot(data = mtcars, aes(x = mpg)) +  
  geom_histogram(binwidth = 2)
```



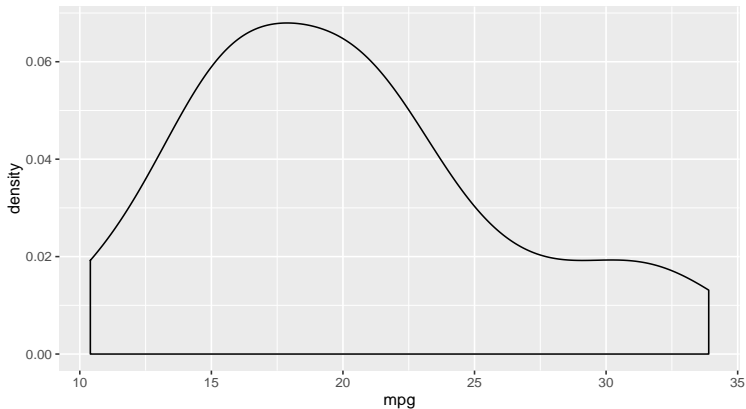
Boxplots

```
ggplot(data = mtcars, aes(x = factor(cyl), y = mpg)) +  
  geom_boxplot()
```



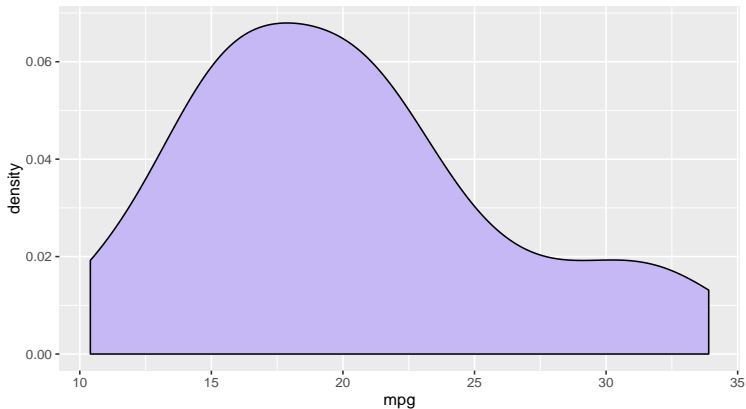
Density Curves

```
ggplot(data = mtcars, aes(x = mpg)) +  
  geom_density()
```



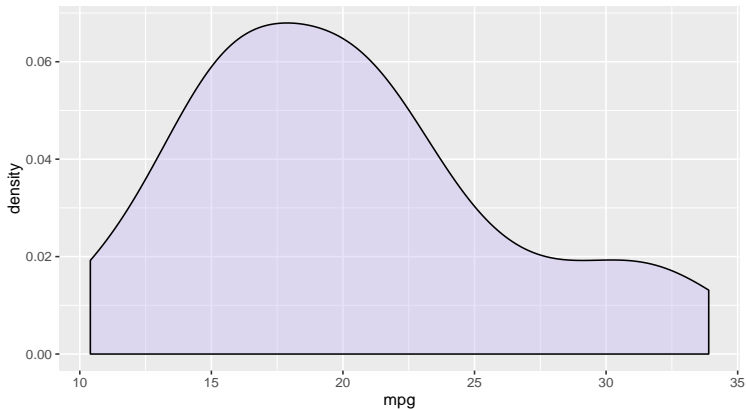
Density Curves

```
ggplot(data = mtcars, aes(x = mpg)) +  
  geom_density(fill = "#c6b7f5")
```



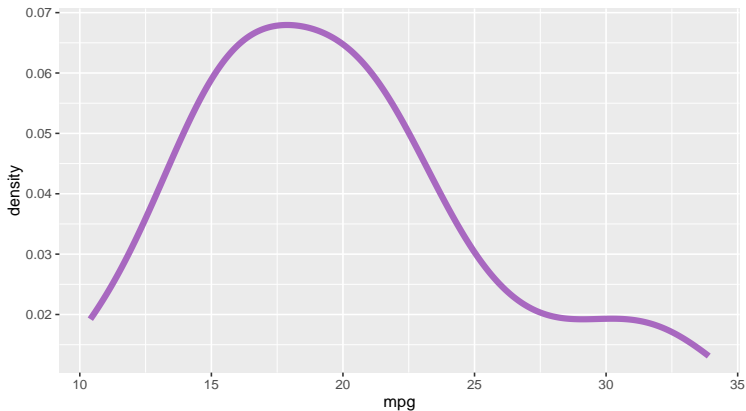
Density Curves

```
ggplot(data = mtcars, aes(x = mpg)) +  
  geom_density(fill = "#c6b7f5", alpha = 0.4)
```



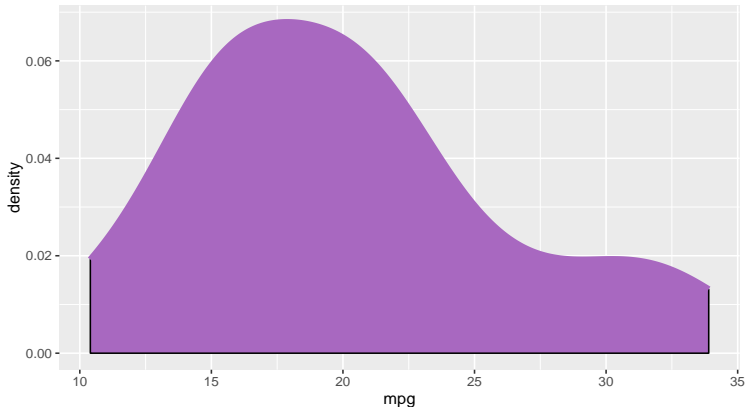
Density Curves

```
ggplot(data = mtcars, aes(x = mpg)) +  
  geom_line(stat = 'density', col = "#a868c0", size = 2)
```



Density Curves

```
ggplot(data = mtcars, aes(x = mpg)) +  
  geom_density(fill = '#a868c0') +  
  geom_line(stat = 'density', col = "#a868c0", size = 2)
```



ggplot objects

Plot objects

You can assign a plot to a new object (this won't plot anything):

```
mpg_hp <- ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(size = 3, color = "tomato")
```

To show the actual plot associated to the object `mpg_hp` use the function `print()`

```
print(mpg_hp)
```

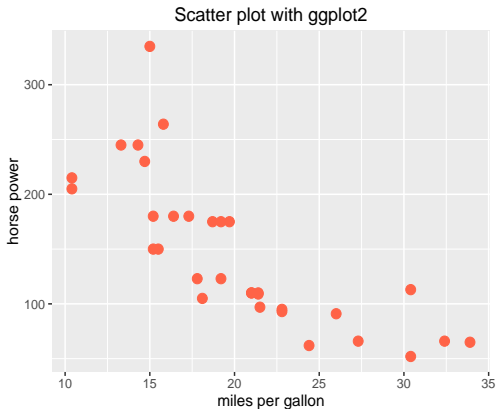
"ggplot2" objects

working with ggplot objects, we can ...

- ▶ define a basic plot, to which we can add or change layers without typing everything again
- ▶ render it on screen with `print()`
- ▶ describe its structure with `summary()`
- ▶ render it to disk with `ggsave()`
- ▶ save a cached copy to disk with `save()`

Adding a title and axis labels to a ggplot2 object:

```
mpg_hp + ggtitle("Scatter plot with ggplot2") +  
  xlab("miles per gallon") + ylab("horse power")
```



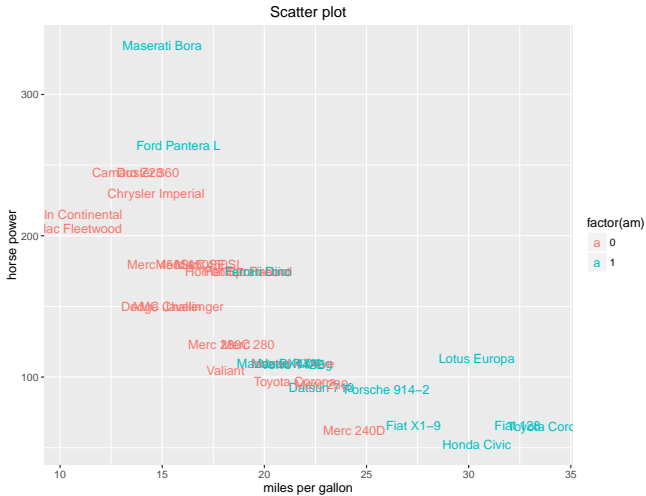
Your turn: example 3

Create the following ggplot object:

```
# ggplot object  
obj <- ggplot(data = mtcars,  
              aes(x = mpg, y = hp, label = rownames(mtcars)))
```

Add more layers to the object "obj" in order to replicate the figure in the following slide:

Your turn: example 3



Your turn: example 3

```
obj +  
  geom_text(aes(color = factor(am))) +  
  ggtitle("Scatter plot") +  
  xlab("miles per gallon") +  
  ylab("horse power")
```

Scales

Scales

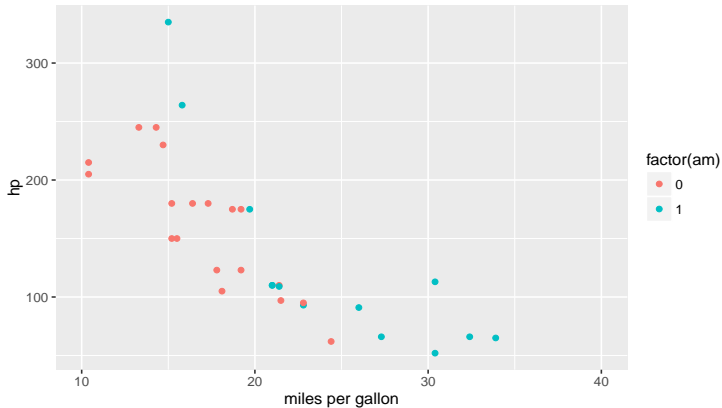
- ▶ The **scales** component encompasses the ideas of both axes and legends on plots, e.g.:
- ▶ Axes can be continuous or discrete
- ▶ Legends involve colors, symbol shapes, size, etc
 - `scale_x_continuous`
 - `scale_y_continuous`
 - `scale_color_manual`
- ▶ **scales** will often automatically generate appropriate scales for plots
- ▶ Explicitly adding a scale component overrides the default scale

Continuous axis scales

Use `scale_x_continuous()` to modify the default values in the *x* axis

```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(aes(color = factor(am))) +  
  scale_x_continuous(name = "miles per gallon",  
                     limits = c(10, 40),  
                     breaks = c(10, 20, 30, 40))
```

Continuous axis scales

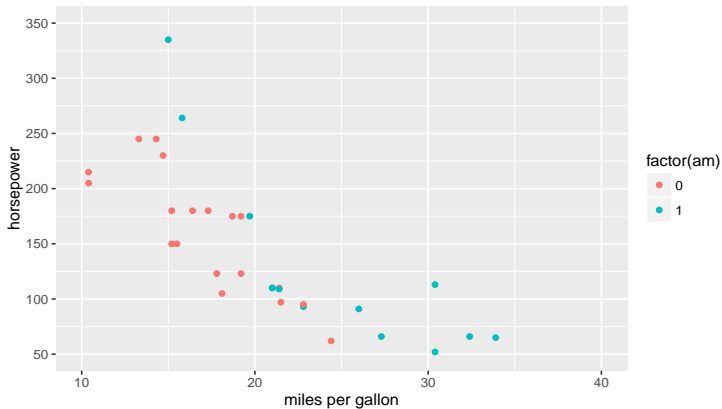


Continuous axis scales

Use `scale_y_continuous()` to modify the default values in the *y* axis

```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(aes(color = factor(am))) +  
  scale_x_continuous(name = "miles per gallon",  
                     limits = c(10, 40),  
                     breaks = c(10, 20, 30, 40)) +  
  scale_y_continuous(name = "horsepower",  
                     limits = c(50, 350),  
                     breaks = seq(50, 350, by = 50))
```

Continuous axis scales

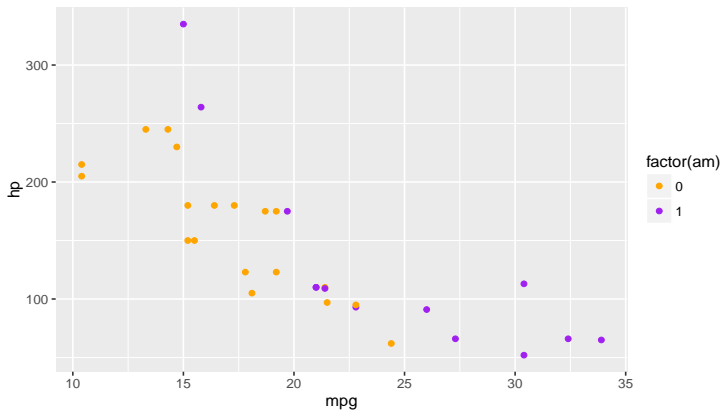


Example: color scale

Use `scale_color_manual()` to modify the colors associated to a factor

```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(aes(color = factor(am))) +  
  scale_color_manual(values = c("orange", "purple"))
```


Example: color scale

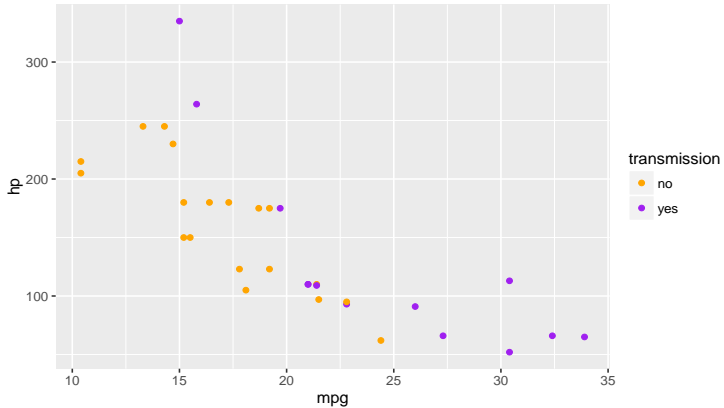


Example: modifying legend

Modifying legends depends on the type of scales (e.g. color, shapes, size, etc)

```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(aes(color = factor(am))) +  
  scale_color_manual(values = c("orange", "purple"),  
                     name = "transmission",  
                     labels = c('no', 'yes'))
```

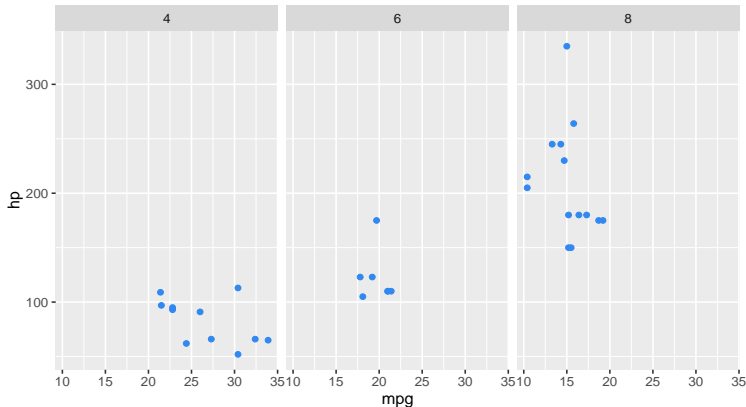
Example: modifying legend



Faceting

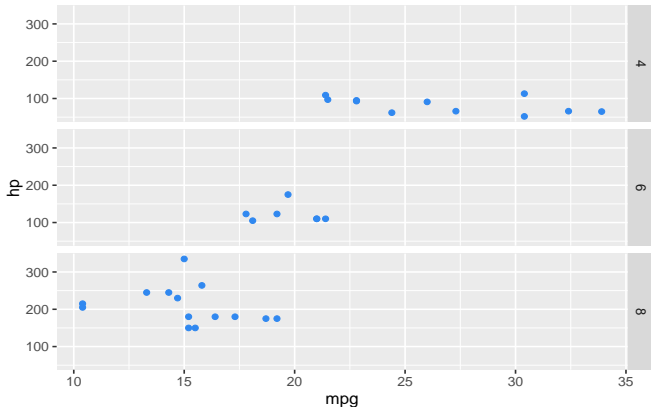
Faceting with facet_wrap()

```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(color = "#3088f0") +  
  facet_wrap(~ cyl)
```



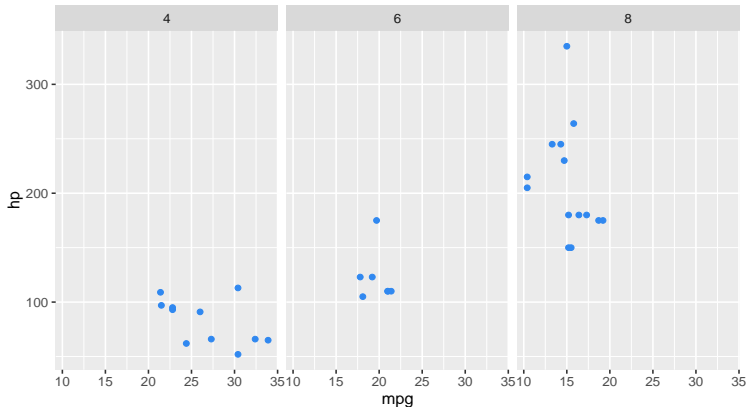
Faceting with facet_grid()

```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(color = "#3088f0") +  
  facet_grid(cyl ~ .)
```



Faceting with facet_grid()

```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(color = "#3088f0") +  
  facet_grid(. ~ cyl)
```



Layered Grammar

About "ggplot2"

- ▶ Key concept: **layer** (layered grammar of graphics)
- ▶ Designed to work in a layered fashion
- ▶ Starting with a layer showing the data
- ▶ Then adding layers of annotations and statistical transformations
- ▶ Core idea: independent components combined together

Some Concepts

- ▶ the **data** to be visualized
- ▶ a set of **aesthetic mappings** describing how variables are mapped to aesthetic attributes
- ▶ geometric objects, **geoms**, representing what you see on the plot (points, lines, etc)
- ▶ statistical transformations, **stats**, summarizing data in various ways
- ▶ **scales** that map values in the data space to values in an aesthetic space
- ▶ a coordinate system, **coord**, describing how data coordinates are mapped to the plane of the graphic
- ▶ a **faceting** specification describing how to break up the data into subsets and to displays those subsets