

# R package ggplot2

STAT 133

Gaston Sanchez

Department of Statistics, UC–Berkeley

`gastonsanchez.com`

`github.com/gastonstat/stat133`

Course web: [gastonsanchez.com/teaching/stat133](https://gastonsanchez.com/teaching/stat133)

# ggplot2

# Resources for "ggplot2"

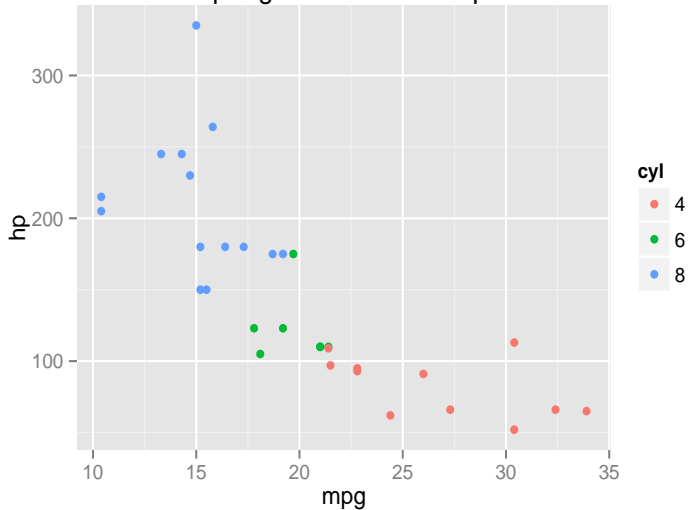
- ▶ Documentation: <http://docs.ggplot2.org/>
- ▶ Book: **ggplot2: Elegant Graphics for Data Analysis** (by Hadley Wickham)
- ▶ Book: **R Graphics Cookbook** (by Winston Chang)

```
install.packages("ggplot2")  
library(ggplot2)
```

# 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

Miles per gallon –vs– Horsepower



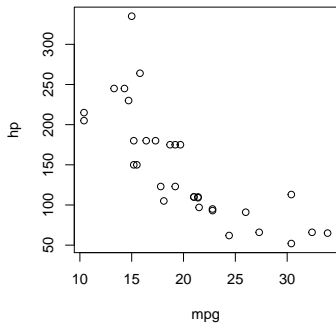
# R package "ggplot2"

## About "ggplot2"

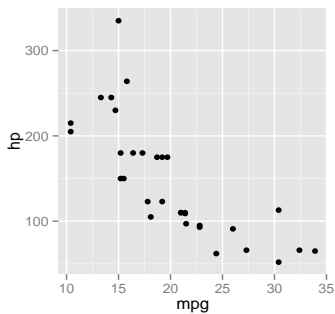
- ▶ Default appearance of plots carefully chosen
- ▶ Designed with visual perception in mind
- ▶ Inclusion of some components, like legends, are automated
- ▶ Great flexibility for annotating, editing, and embedding output

# Base graphics -vs- "ggplot2"

base graphics



ggplot2



# Preliminary Concepts

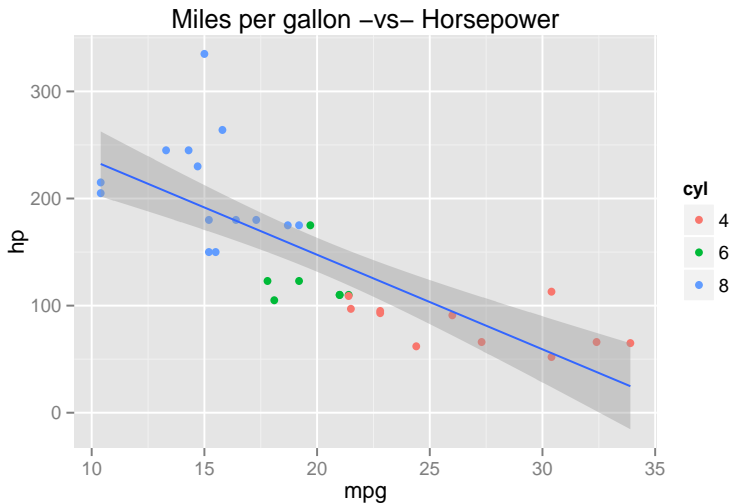


# Grammar of Graphics

## About "ggplot2"

- ▶ The gg in "ggplot2" stands for *Grammar of Graphics*
- ▶ "ggplot2" has a deep underlying grammar
- ▶ inspired in the **Grammar of Graphics** by Lee Wilkinson
- ▶ Grammar of Graphics describes the features that underlie all statistical graphics

# What is a statistical graphic?



# What is a statistical graphic?

##	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?

## 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

# 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)

# 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)

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

Starting with "ggplot2"

# mtcars dataset

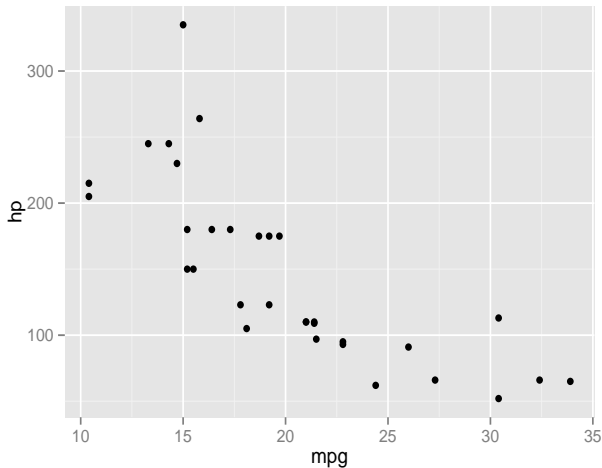
```
head(mtcars, n = 10)
```

##	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
## Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
## Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
## Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
## Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
## Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
## Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
## Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
## Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
## Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
## Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4



# Scatter plot (Option 1)

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



# How does it work?

We specify the data and variables inside the function `ggplot()`. Note the use of the internal function `aes()` to *map* `x` to `mpg`, and `y` to `hp`.

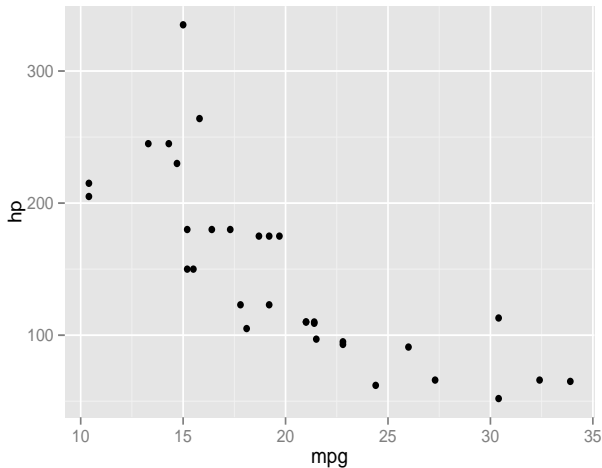
```
ggplot(data = mtcars, aes(x = mpg, y = hp))
```

Then we add a layer of geometric objects: points in this case. Note the use of `"+"` to **add** the layer to the plot

```
+ geom_point()
```

## Scatter plot (Option 2)

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



# "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
  - shape, color, fill, linetype

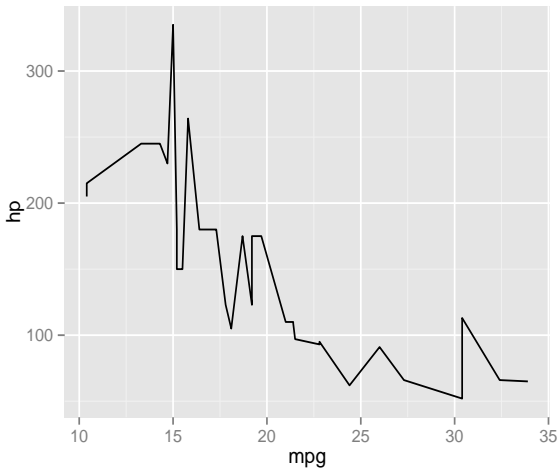
# Warning

"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”

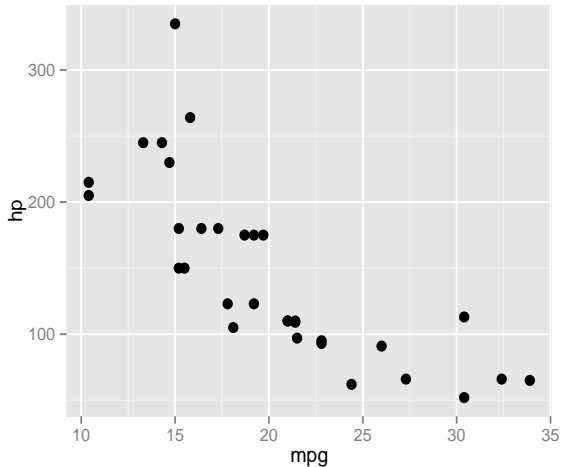
## Another geom

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



# Increase size of points

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





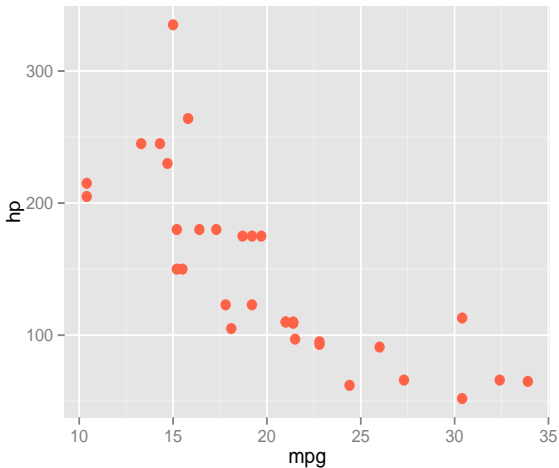
# How does it work?

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)
```

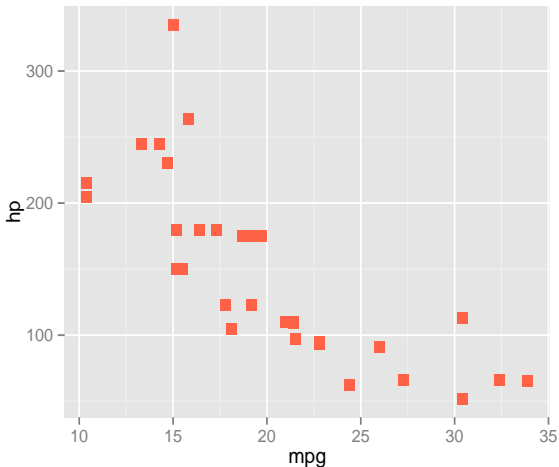
# Adding color

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



# 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

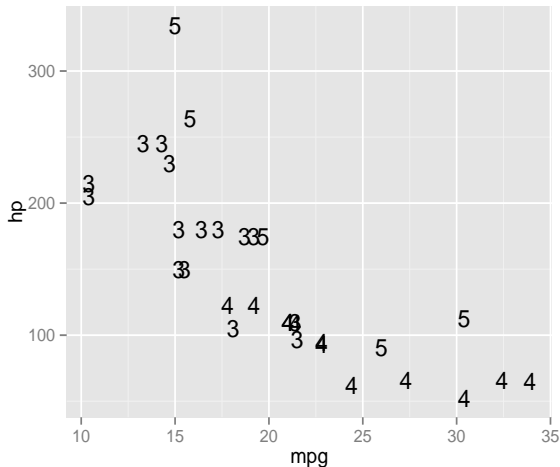
Aesthetic attributes can be either **mapped** —via `aes()`— or **set**

```
# 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")
```

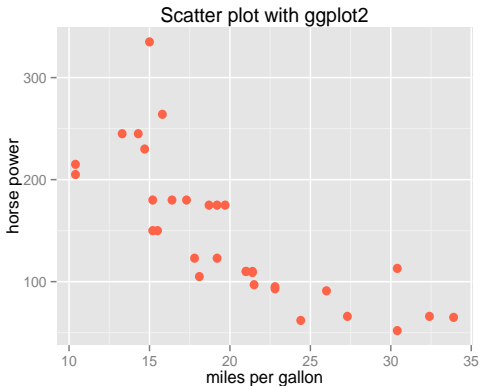
# Geom text, and mapping labels

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



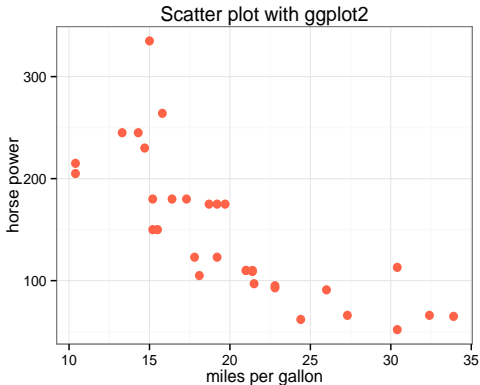
# Changing axis labels and title

```
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")
```

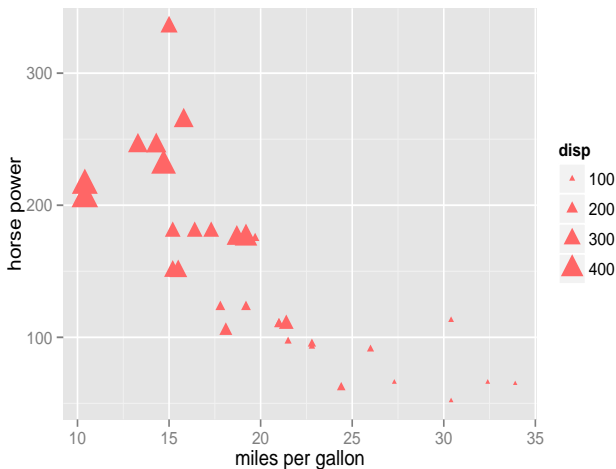


# Changing background theme

```
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

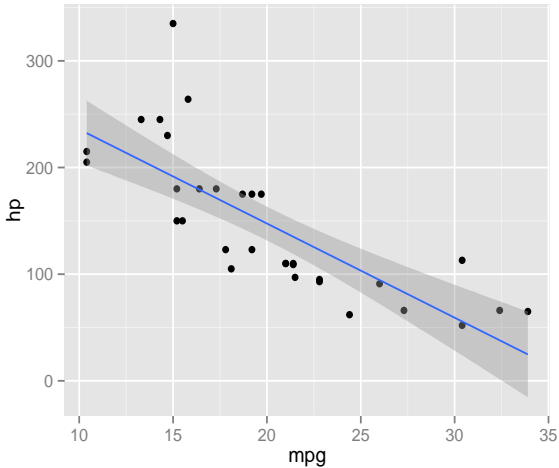
- ▶ Specify a color in hex notation
- ▶ Change the shape of the point symbol
- ▶ Map `disp` to attribute size of points
- ▶ Add axis labels

# 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

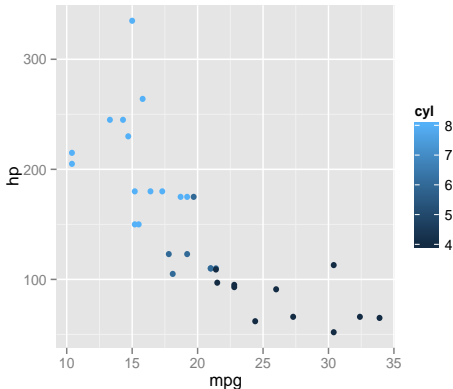
```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point() +  
  geom_smooth(method = "lm")
```



# More geoms

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

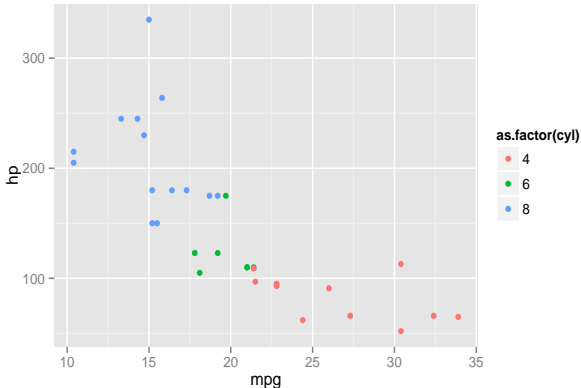
```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(aes(color = cyl))
```



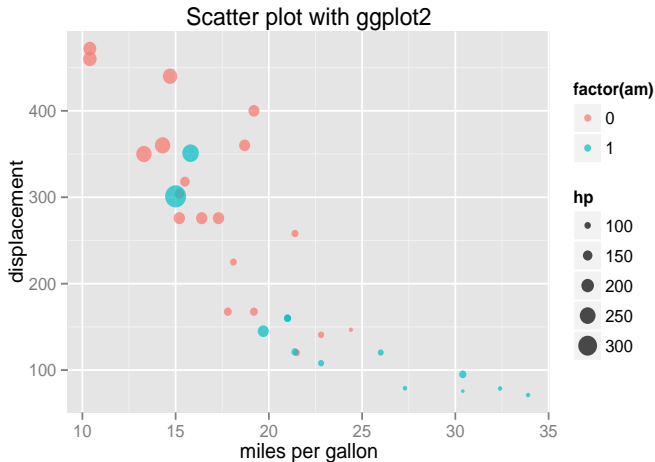
# More geoms

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

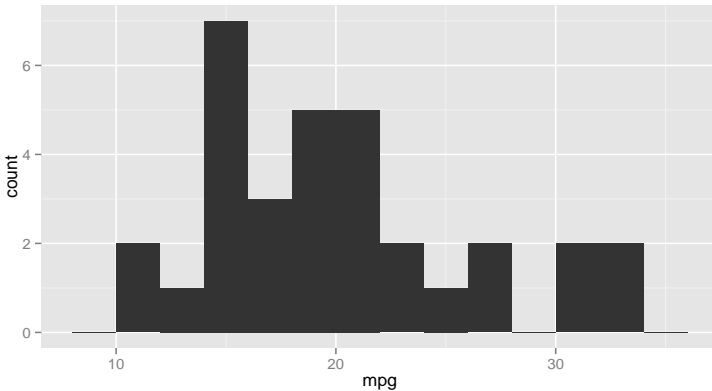
## Your turn: example 2

```
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")
```



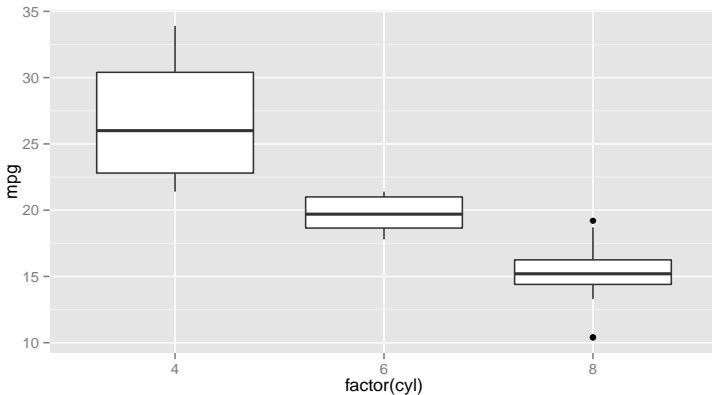
# Histogram

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



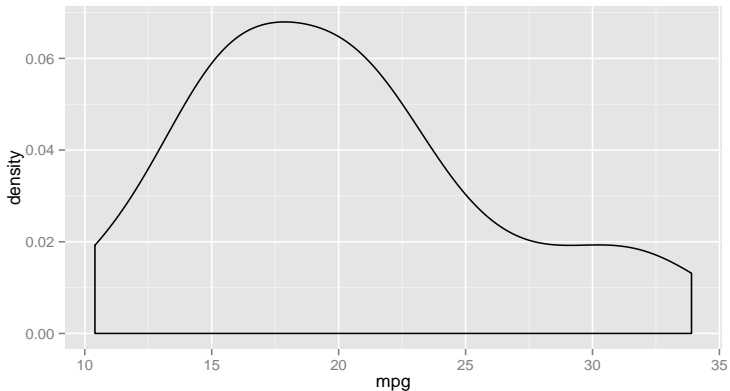
# Boxplots

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



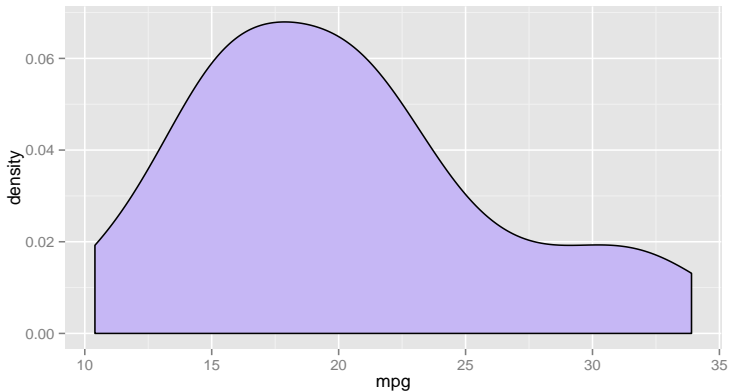
# Density Curves

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



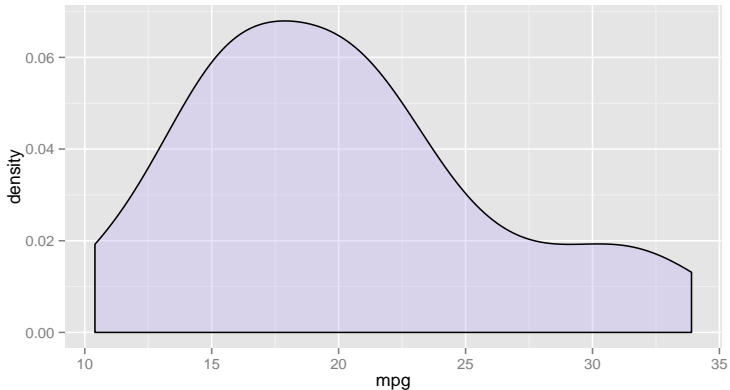
# Density Curves

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



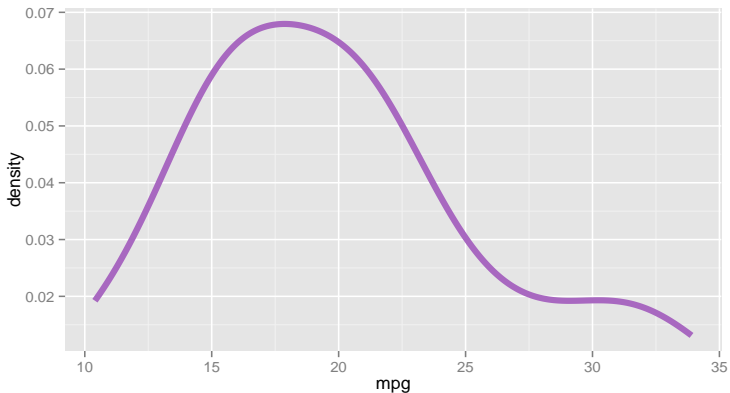
# Density Curves

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



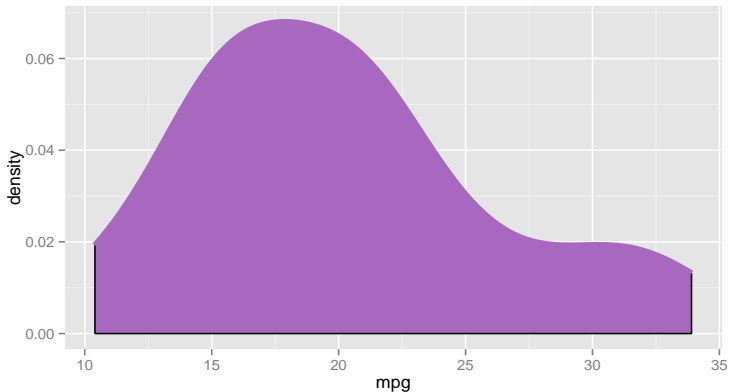
# Density Curves

```
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)
```

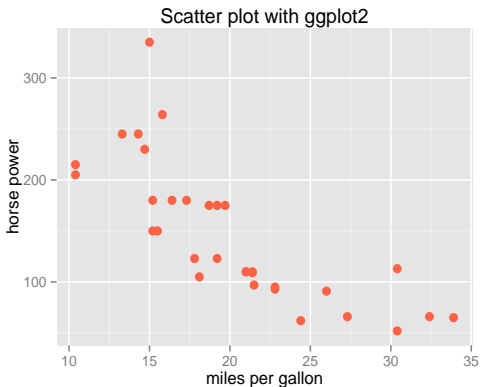
# "ggplot2" objects

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")
```



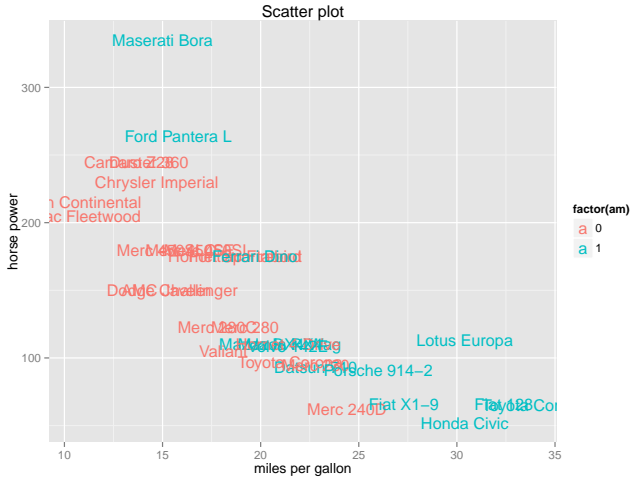
## Your turn: example 3

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

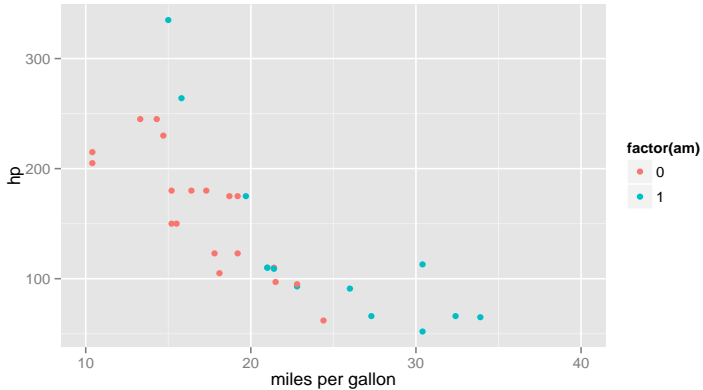


# Continuous axis scales

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

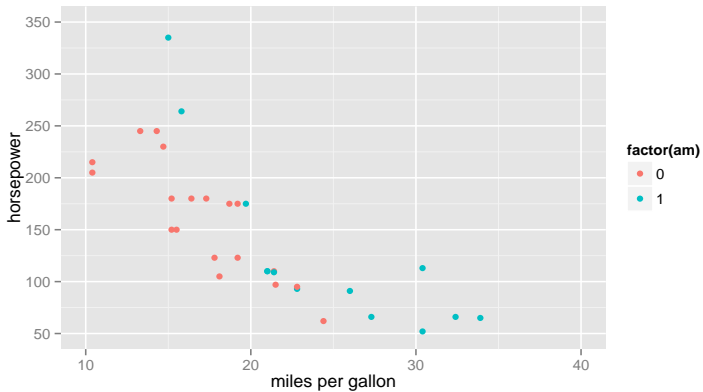


# Continuous axis scales

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

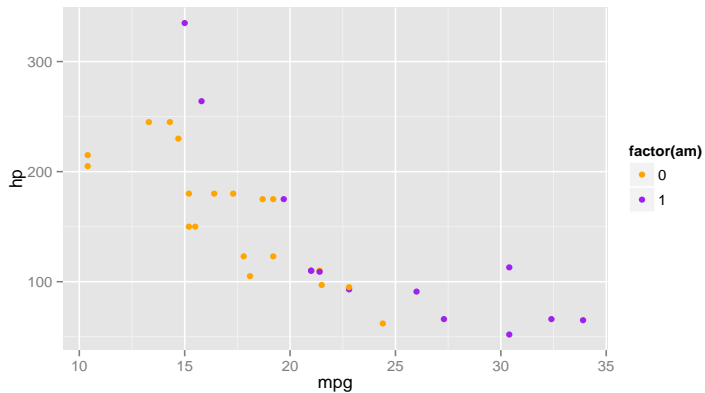


## Example: color scale

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

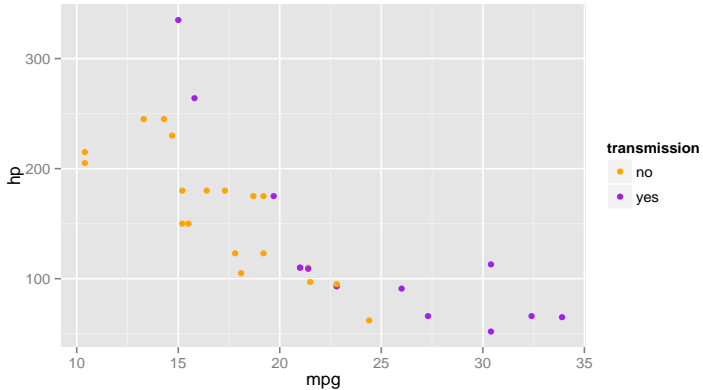


## Example: modifying legend

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

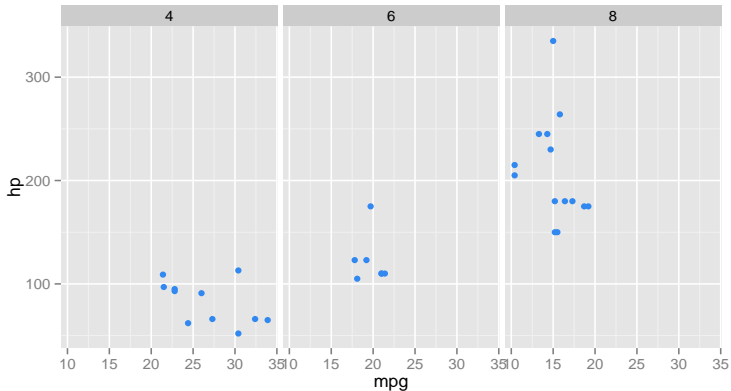




# Faceting

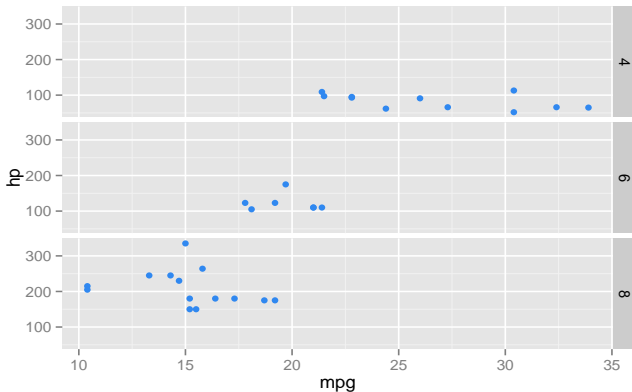
# Faceting with facet\_wrap()

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



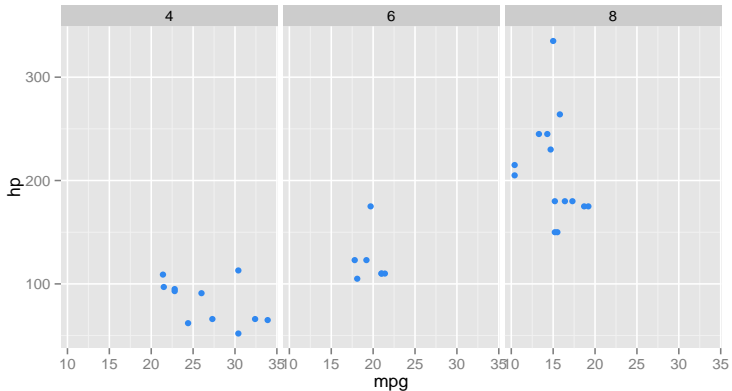
# Faceting with facet\_grid()

```
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