

R package ggplot2

STAT 133

Gaston Sanchez

Department of Statistics, UC–Berkeley

`gastonsanchez.com`

`github.com/gastonstat/stat133`

Course web: gastonsanchez.com/teaching/stat133

ggplot2

Scatterplot with "ggplot2"

Terminology

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

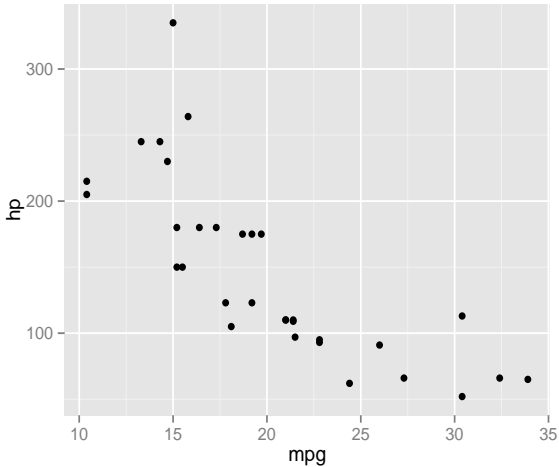
Considerations

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)

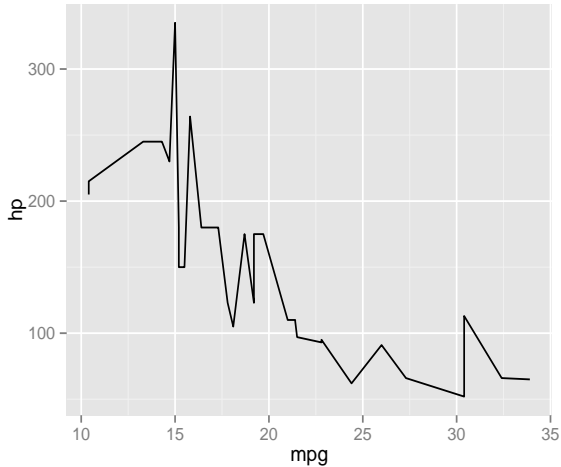
Scatterplot with geom_point

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



Another geom

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



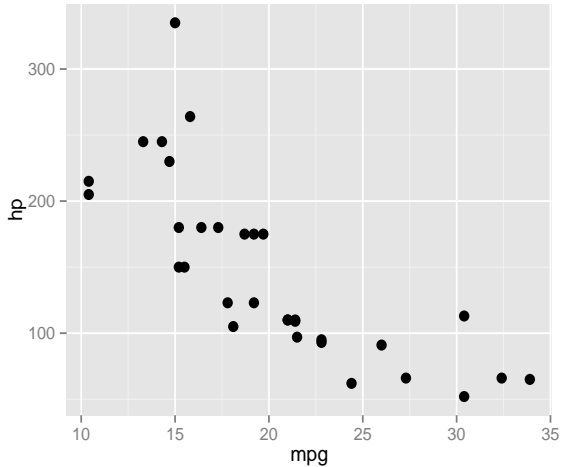
Mapping Attributes

-vs-

Setting Attributes

Increase size of points

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



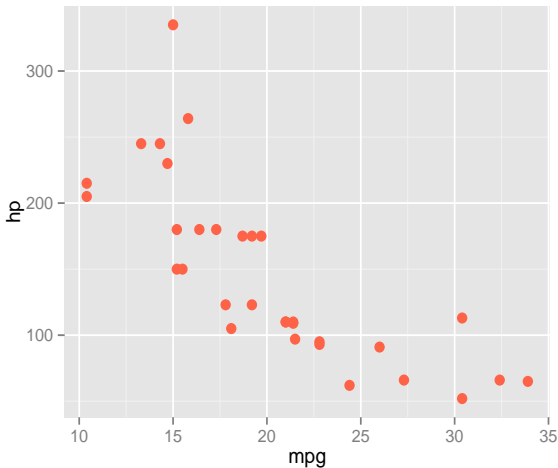
How does it work?

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)
```

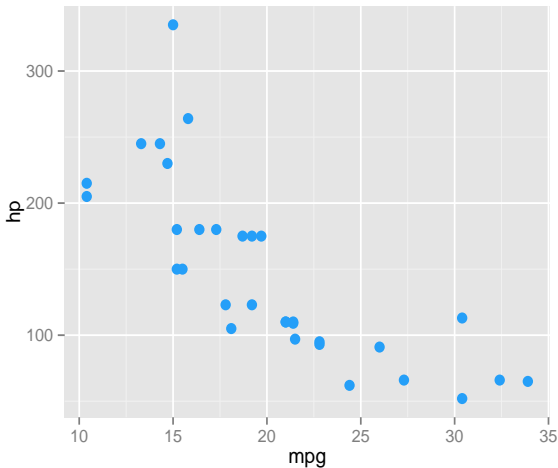
Adding color

```
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")
```



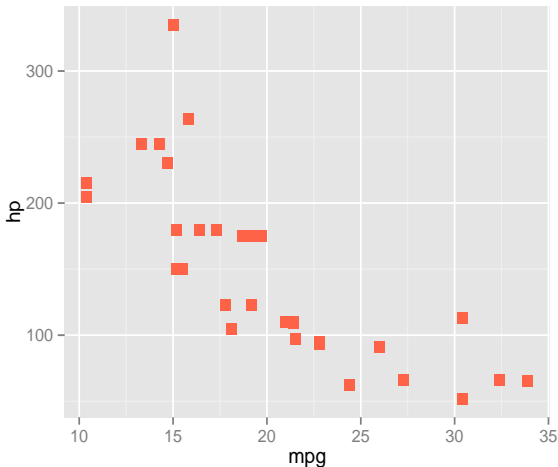
Test your knowledge

Identify the valid hex-color

- A) "345677"
- B) "#1234567"
- C) "#AAAAAA"
- D) "#GG0033"

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

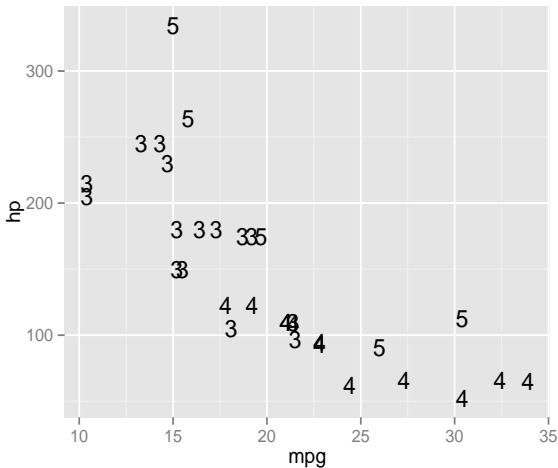
Aesthetic attributes can be either **mapped** —via `aes()`— or **set**

```
# 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")
```

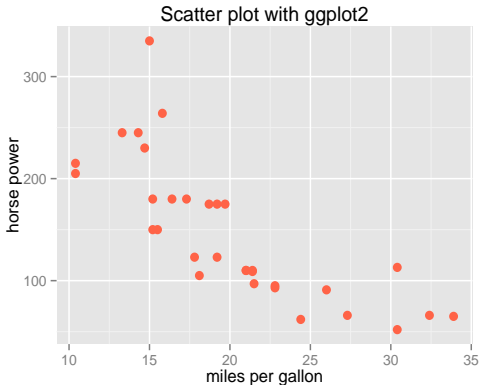
Geom text, and mapping labels

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



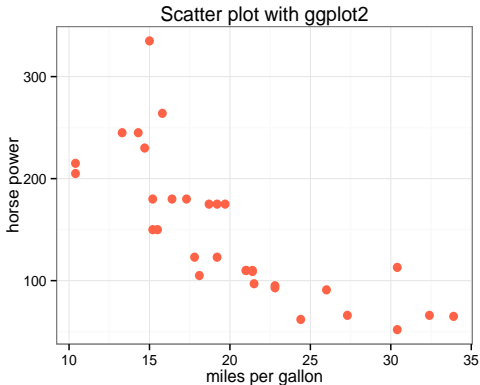
Changing axis labels and title

```
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")
```

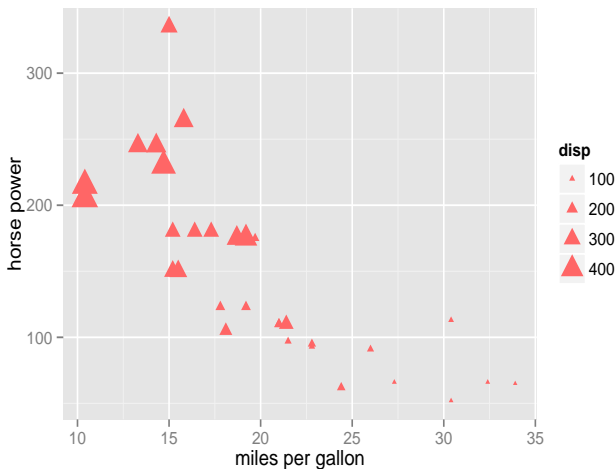


Changing background theme

```
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

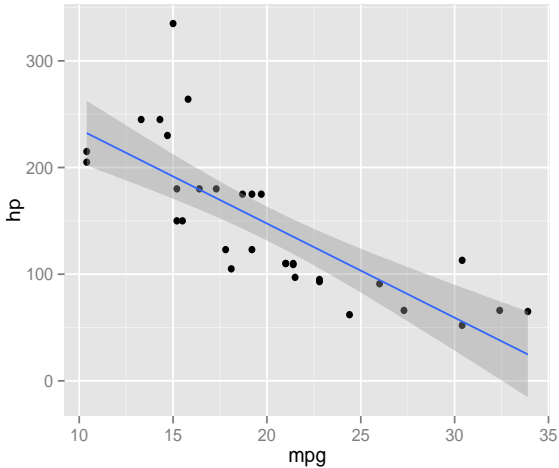
- ▶ Specify a color in hex notation
- ▶ Change the shape of the point symbol
- ▶ Map `disp` to attribute size of points
- ▶ Add axis labels

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

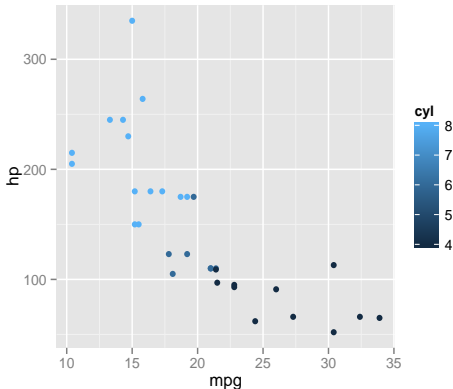
```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point() +  
  geom_smooth(method = "lm")
```



More geoms

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

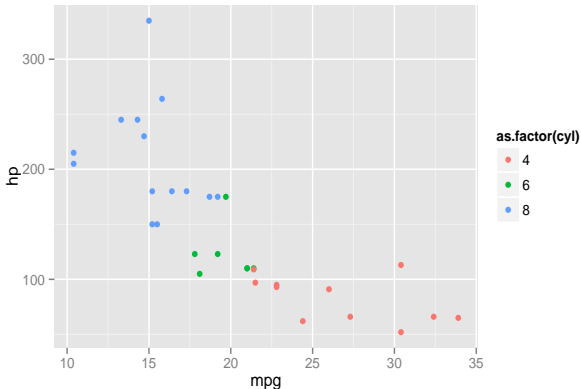
```
ggplot(data = mtcars, aes(x = mpg, y = hp)) +  
  geom_point(aes(color = cyl))
```



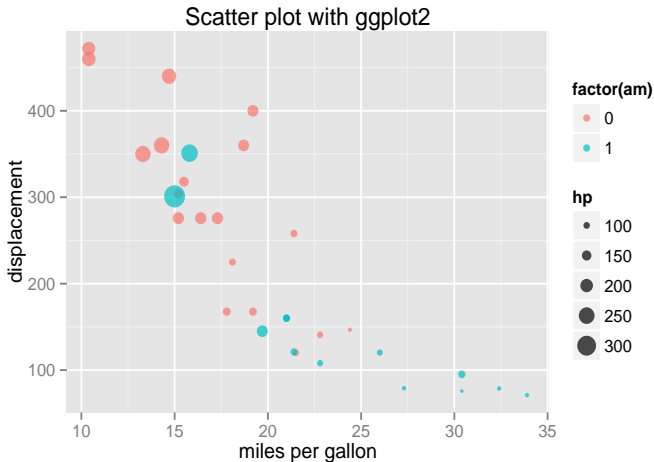
More geoms

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

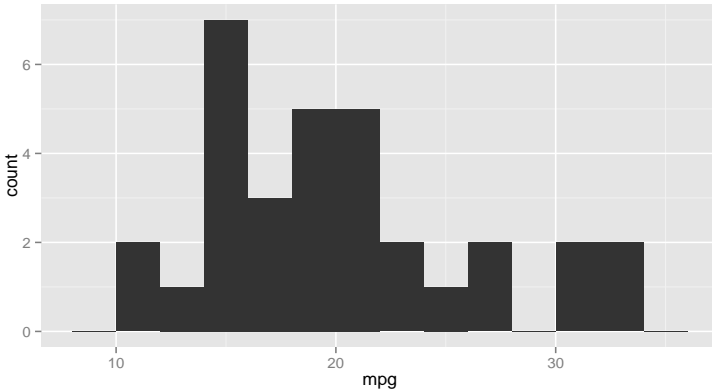
- ▶ Map `hp` to attribute size of points
- ▶ Map `am` (as factor) to attribute color points
- ▶ Add an alpha transparency of 0.7
- ▶ Change the shape of the point symbol
- ▶ Add axis labels
- ▶ Add a title

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")
```

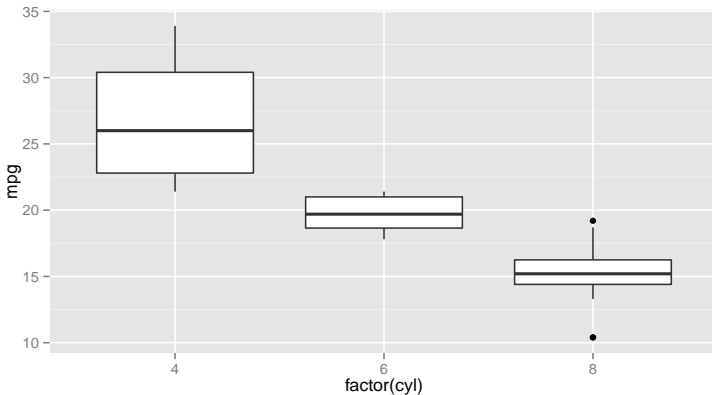
Histogram

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



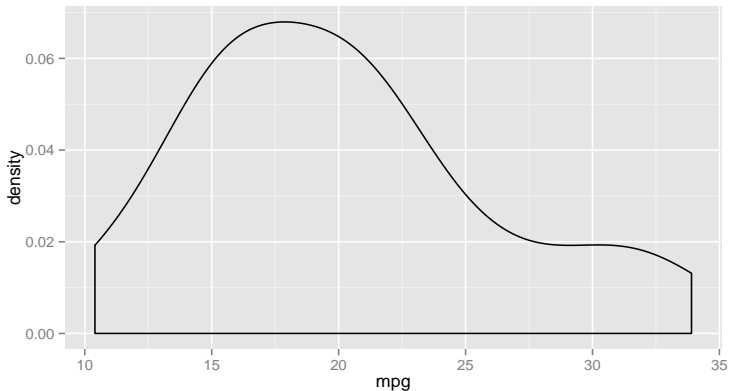
Boxplots

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



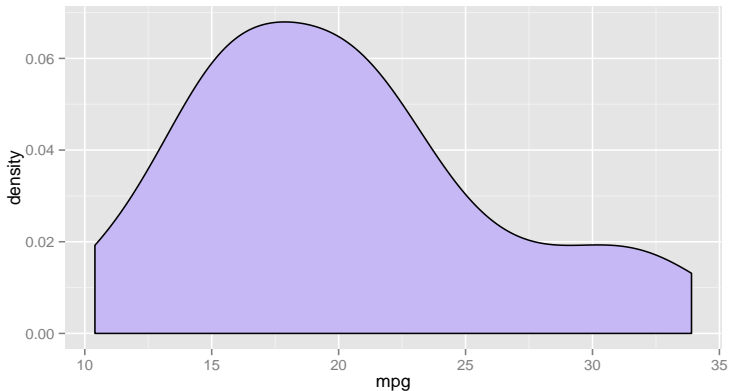
Density Curves

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



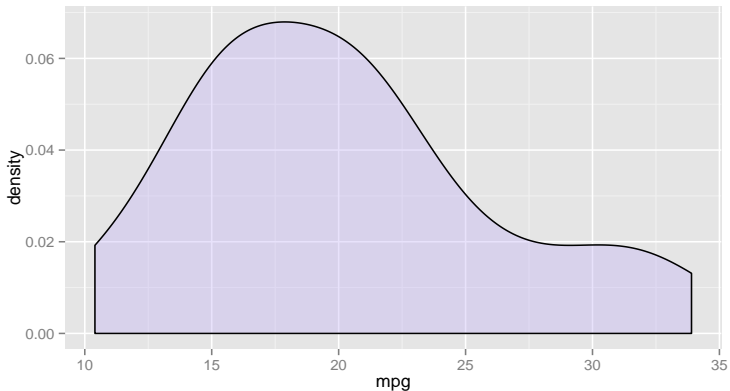
Density Curves

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



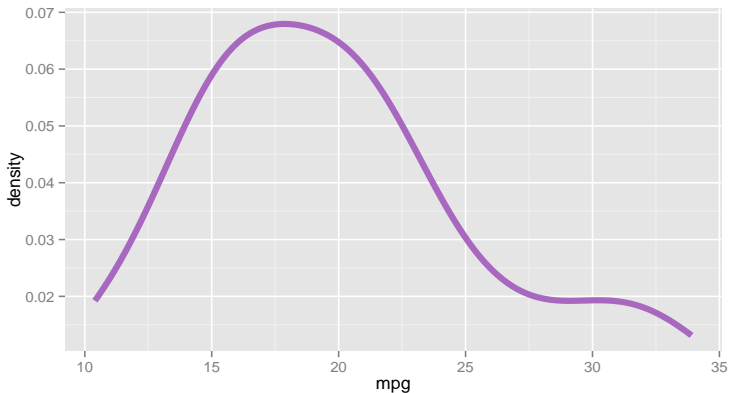
Density Curves

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



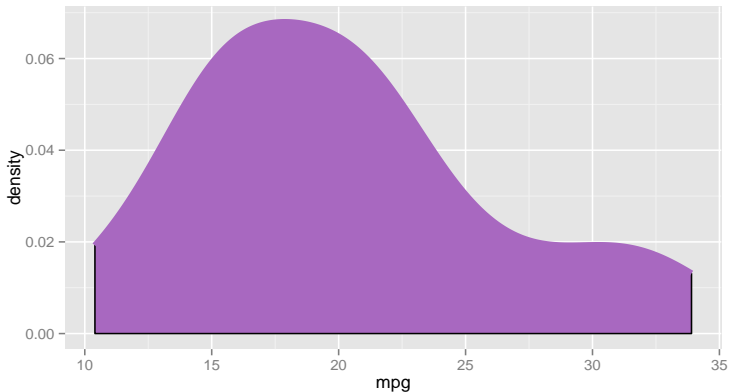
Density Curves

```
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)
```

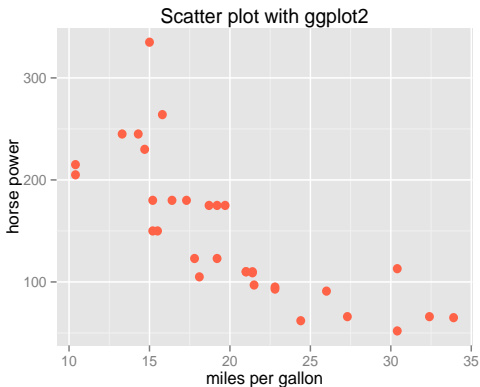
"ggplot2" objects

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")
```



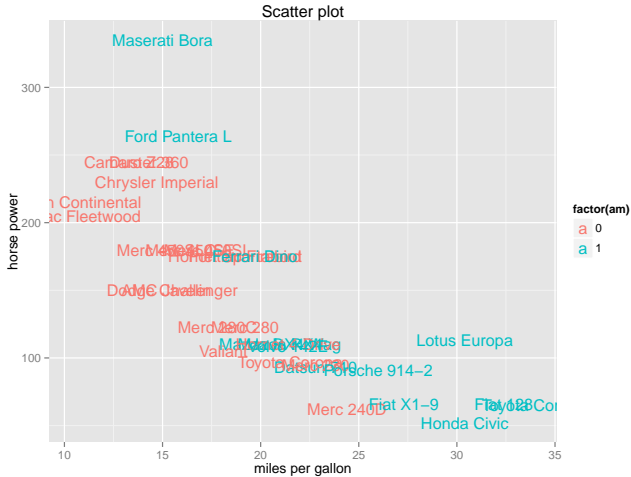
Your turn: example 3

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

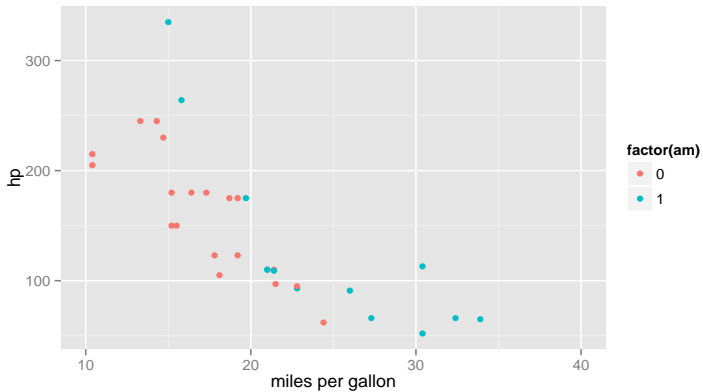
- ▶ The **scales** component encompasses the ideas of both axes and legends on plots, e.g.:
- ▶ Axes can be continuous or discrete
- ▶ Legends involve colors, symbol shapes, size, etc
 - `scale_x_continuous`
 - `scale_y_continuous`
 - `scale_color_manual`
- ▶ **scales** will often automatically generate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

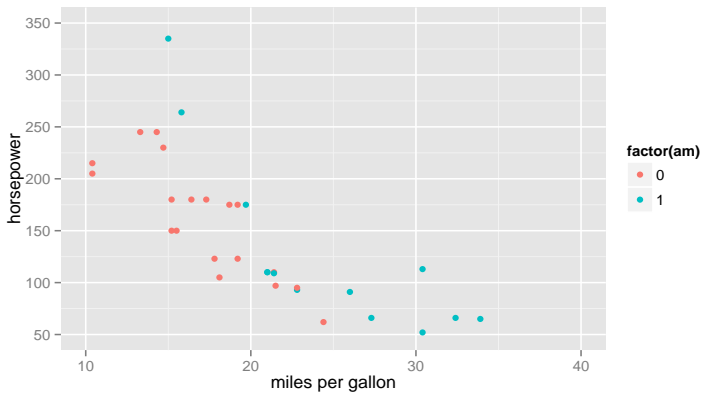


Continuous axis scales

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

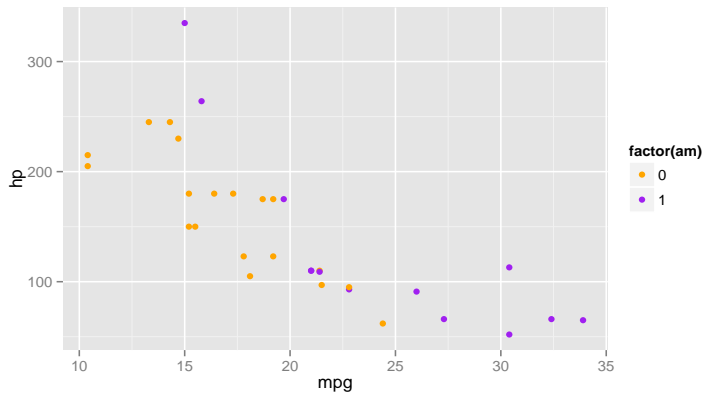


Example: color scale

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

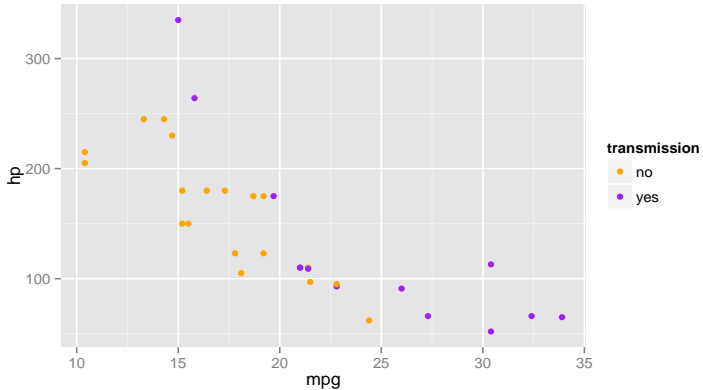


Example: modifying legend

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'))
```

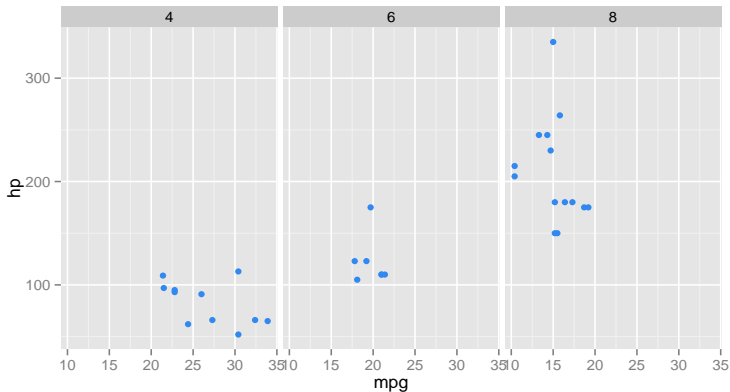
Example: modifying legend



Faceting

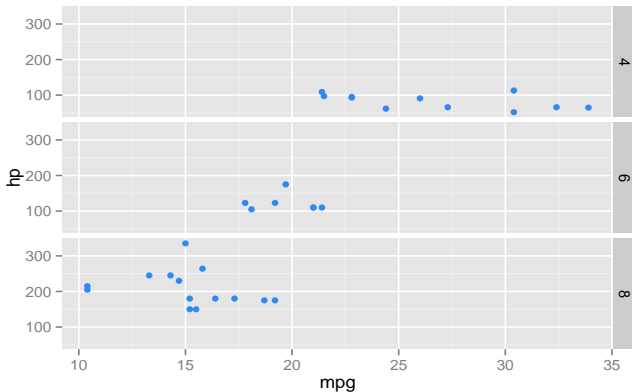
Faceting with facet_wrap()

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



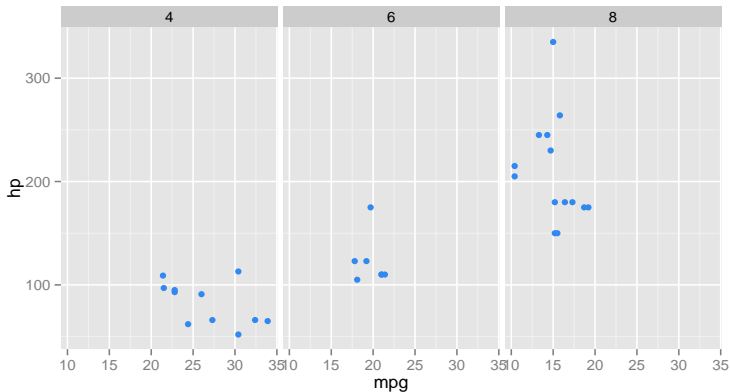
Faceting with facet_grid()

```
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