R Plotting with ggplot2 - Part II
Graphical parameters

Xuemao Zhang
Department of Mathematics
East Stroudsburg University

June 25, 2019

Outline

- Titles
- Legend
- Colors
- Points
- Axis scales

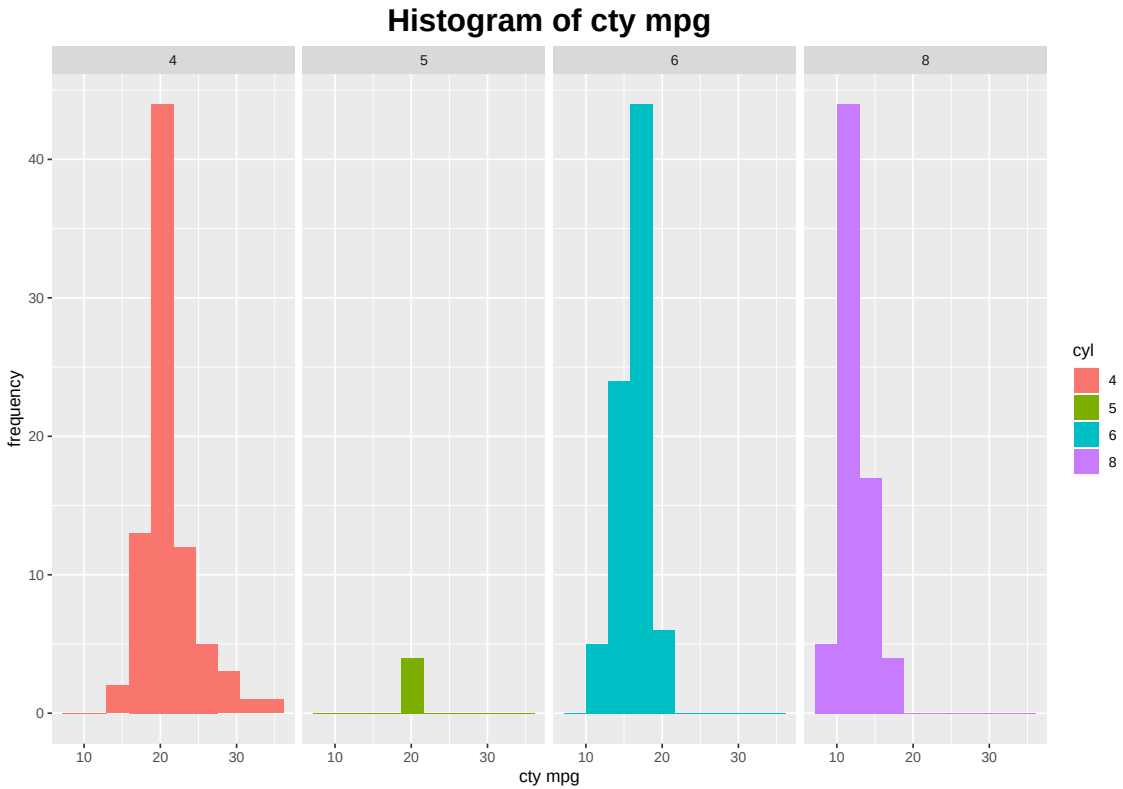
Titles

- We use the data mpg to show how to modify plot titles.
- The functions to be used are
  - ggtitle(label) # for the main title
  - xlab(label) # for the x axis label
  - ylab(label) # for the y axis label
  - labs(...) # for the main title, axis labels and legend titles

```
library(ggplot2);
mpg$cyl=as.factor(mpg$cyl); ## convert cyl from a int to a factor
attach(mpg);
P=ggplot(data=mpg, aes(x = cty)) +
  geom_histogram(aes(fill = cyl),bins = 10) +
  facet_grid(~cyl);
```

Titles

```
P  +   ggtitle("Histogram of cty mpg") +
 theme(plot.title = element_text(hjust = 0.5,size = 20,
 face = "bold")) + #centering the title and set the size
 xlab("cty mpg") + ylab("frequency");
```



Titles

- Main title and, x and y axis labels can be customized using the functions theme() and element_text().

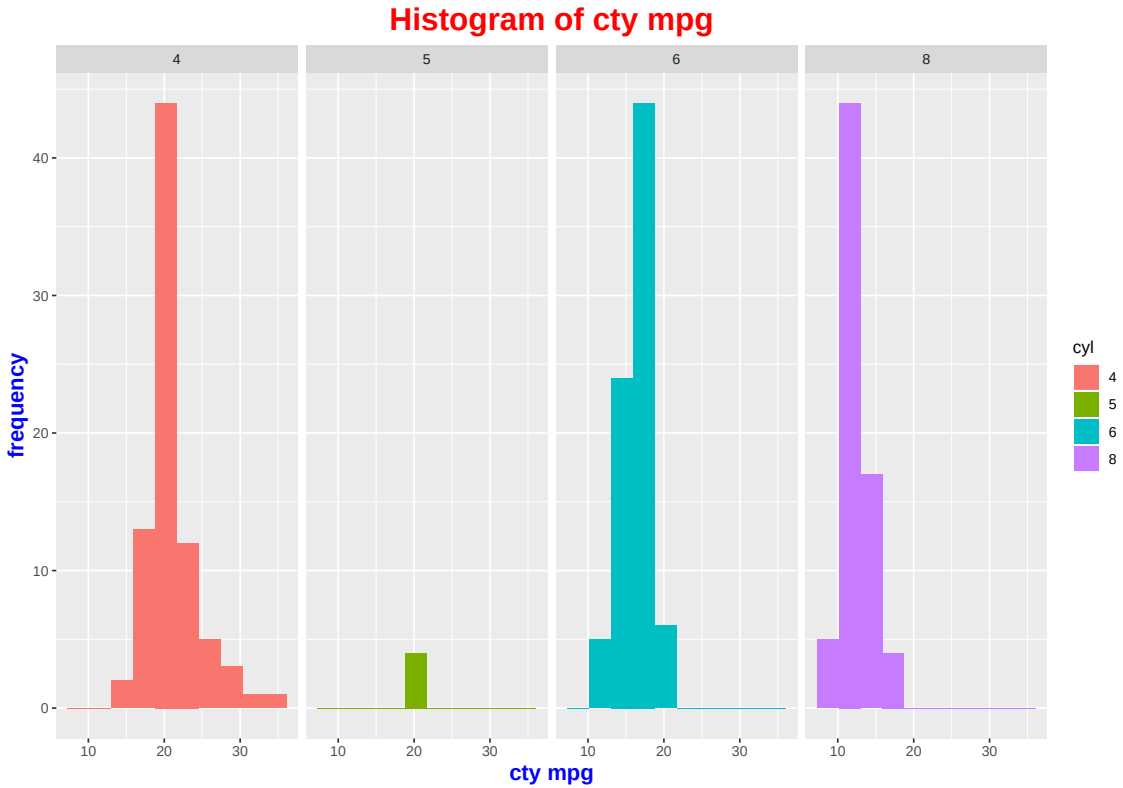
```
# main title
P + theme(plot.title = element_text(family, face, color, size))
# x axis title
P + theme(axis.title.x = element_text(family, face, color, size))
# y axis title
P + theme(axis.title.y = element_text(family, face, color, size))
```

Titles

- The arguments are
  - **family**: font family
  - **face** : font face. Possible values are “plain”, “italic”, “bold” and “bold.italic”
  - **color** : text color
  - **size** : text size in pts
  - **hjust** : horizontal justification (in [0, 1])
  - **vjust** : vertical justification (in [0, 1])
  - **lineheight** : line height. In multi-line text, the lineheight argument is used to change the spacing between lines.
  - **color** : an alias for color

Titles

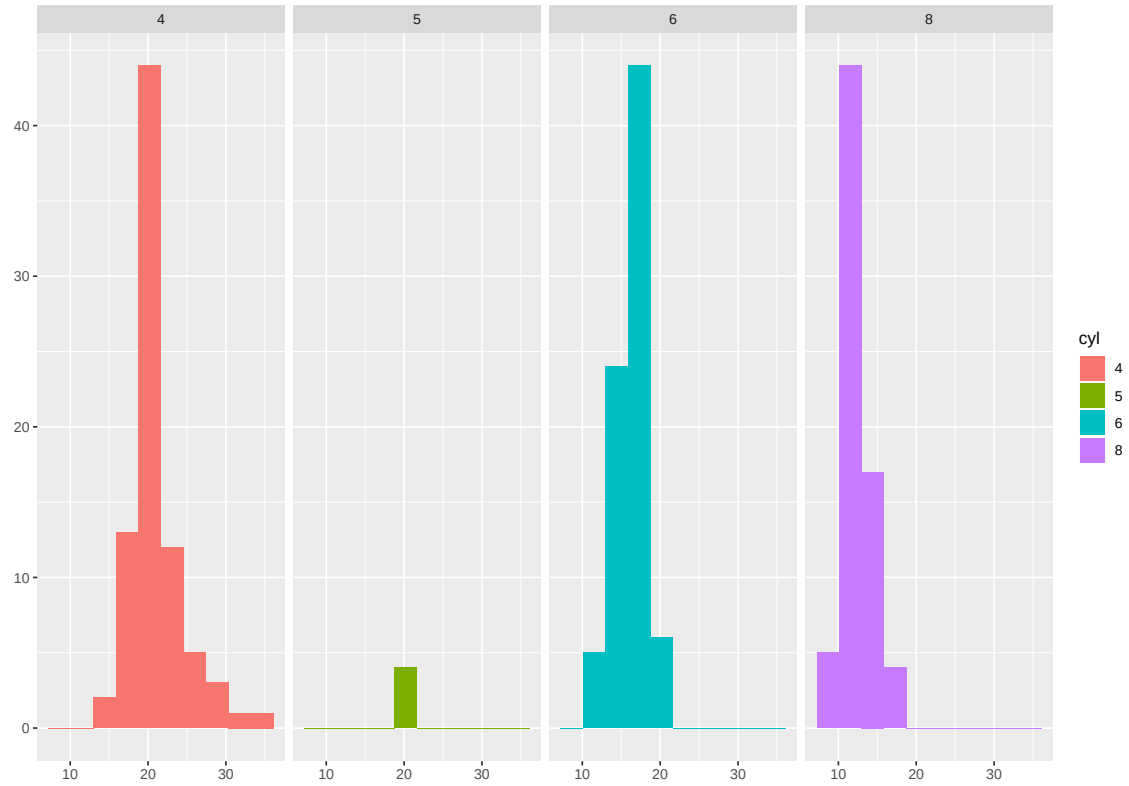
```
P + ggtitle("Histogram of cty mpg") +
  theme(plot.title = element_text(color="red", hjust = 0.5,
  size = 20, face = "bold"),
axis.title.x = element_text(color="blue", size=14, face="bold"),
axis.title.y = element_text(color="blue", size=14, face="bold")) +
  xlab("cty mpg") + ylab("frequency");
```



Titles

- It's possible to hide the main title and axis labels using the function `element_blank()`.

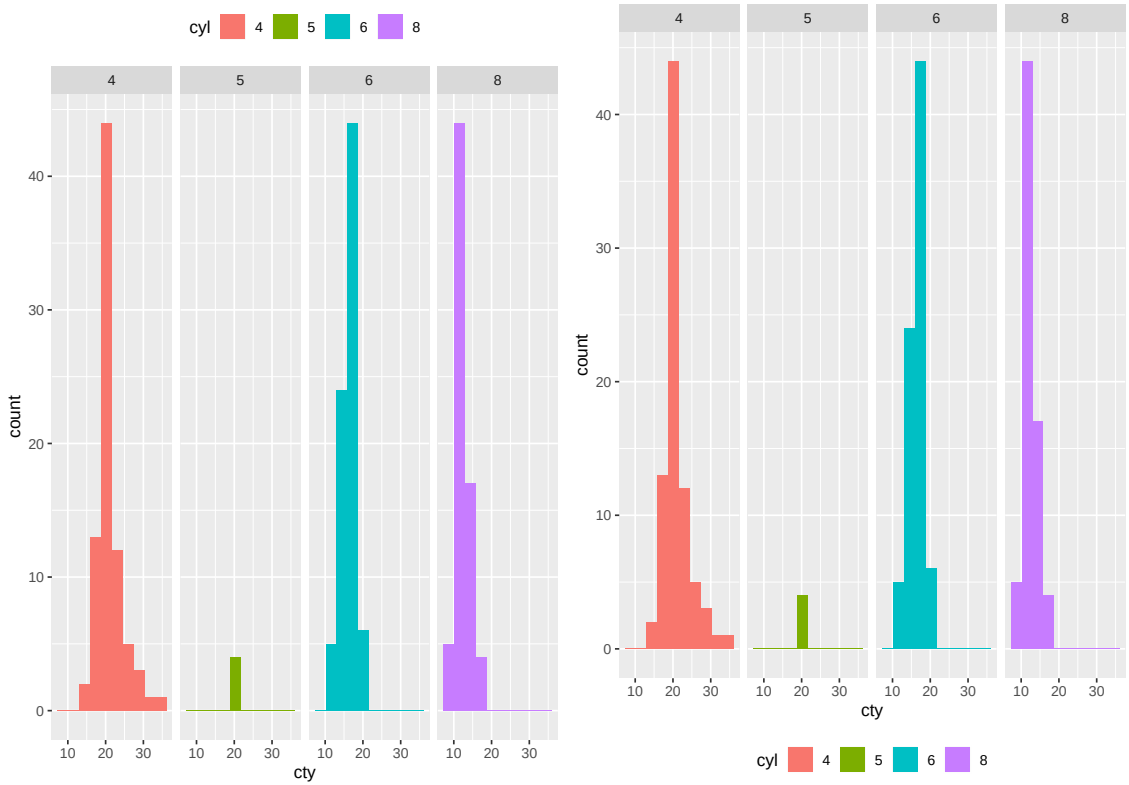
```
P + theme(
  plot.title = element_blank(),
  axis.title.x = element_blank(),
  axis.title.y = element_blank() );
```



Legend

- Change the legend position

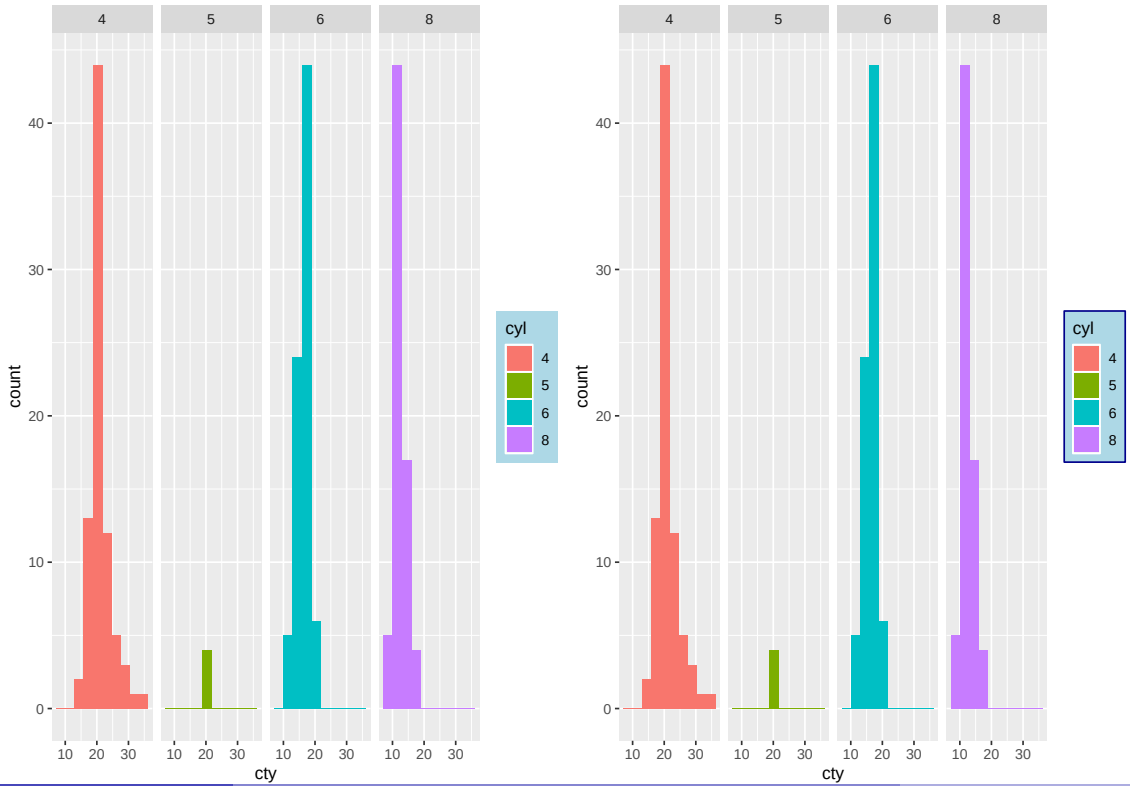
```
library(gridExtra);
p1= P + theme(legend.position="top");
p2= P + theme(legend.position="bottom");
grid.arrange(p1, p2, ncol=2);
```



Legend

- Change the background color of the legend box

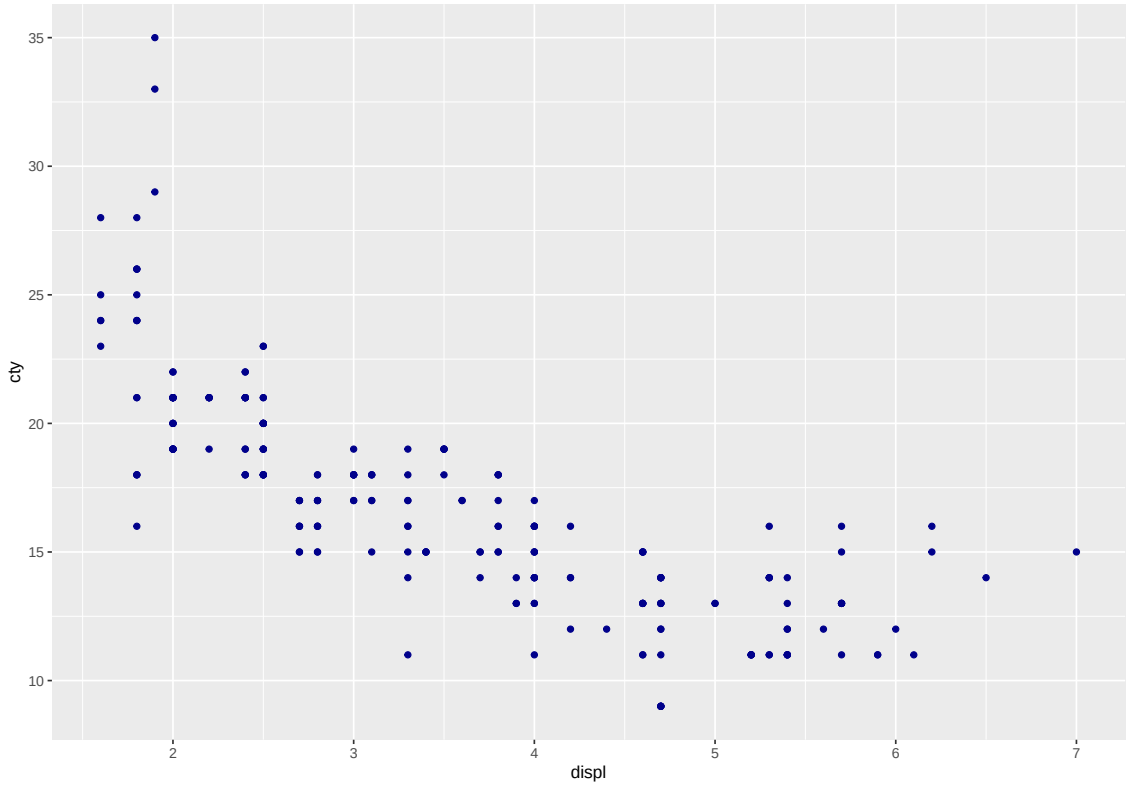
```
library(gridExtra);
p1= P + theme(legend.background=element_rect(fill="lightblue",
        size=0.5, linetype="solid"));
p2= P + theme(legend.background=element_rect(fill="lightblue",
  size=0.5, linetype="solid", color ="darkblue"))
grid.arrange(p1, p2, ncol=2);
```



Colors

- Change colors manually

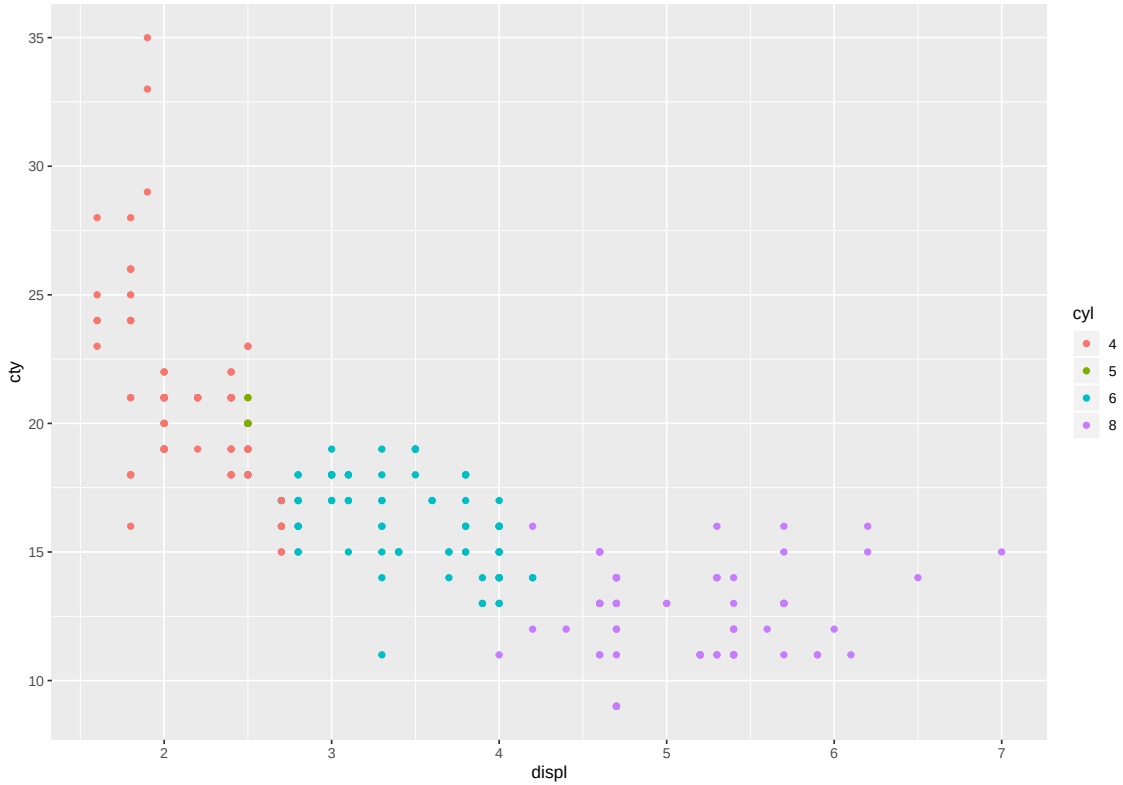
```
ggplot(mpg, aes(x=displ, y=cty)) +
  geom_point(color='darkblue');
```



Colors

- Default colors by groups

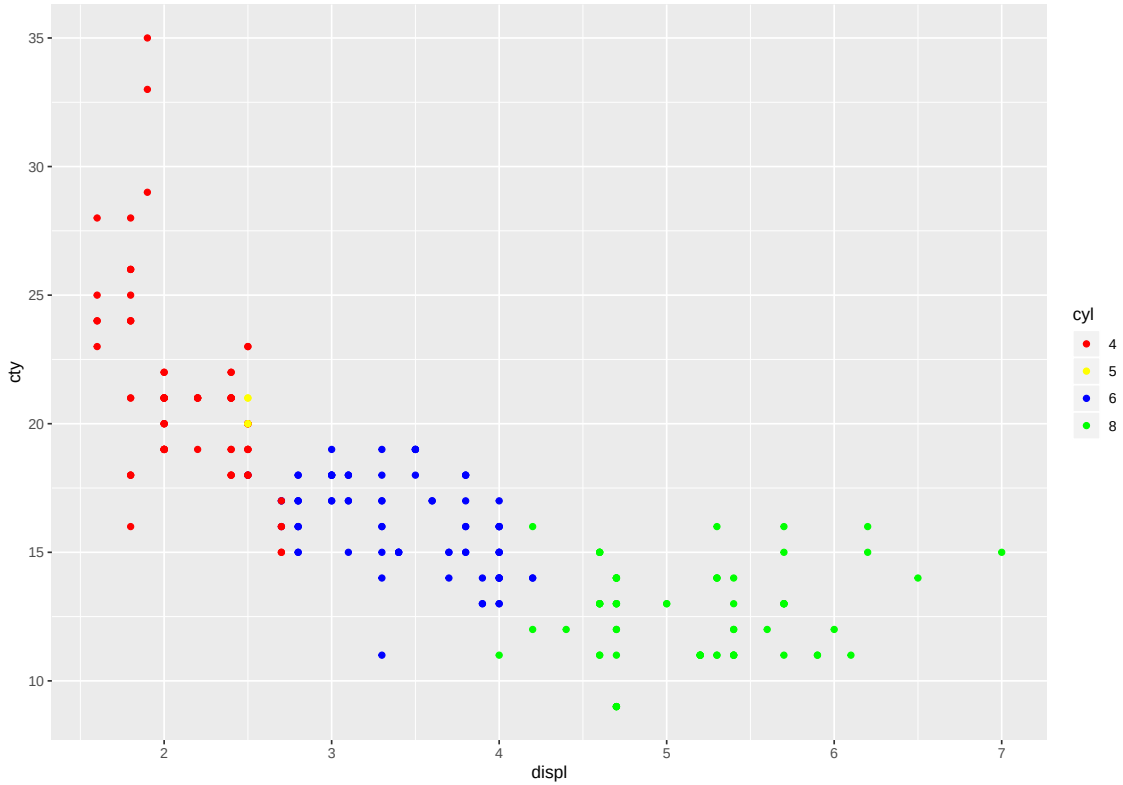
```
ggplot(mpg, aes(x=displ, y=cty,color=cyl)) +
  geom_point();
```



Colors

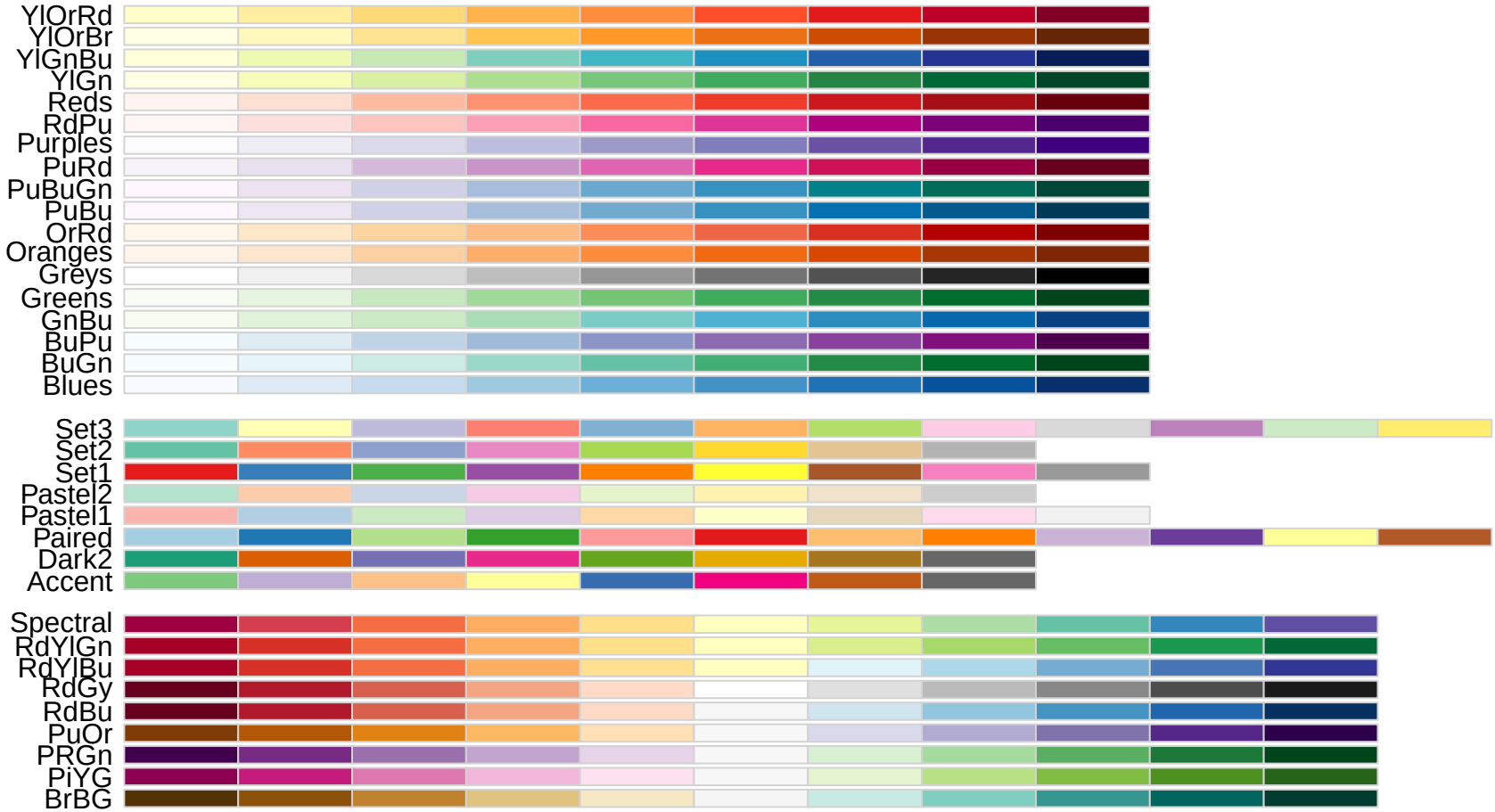
- Change colors by groups

```
ggplot(mpg, aes(x=displ, y=cty,color=cyl)) +
  geom_point() +
  scale_color_manual(breaks = c("4", "5", "6","8"),
  values=c("red", "yellow","blue", "green"));
```



Colors

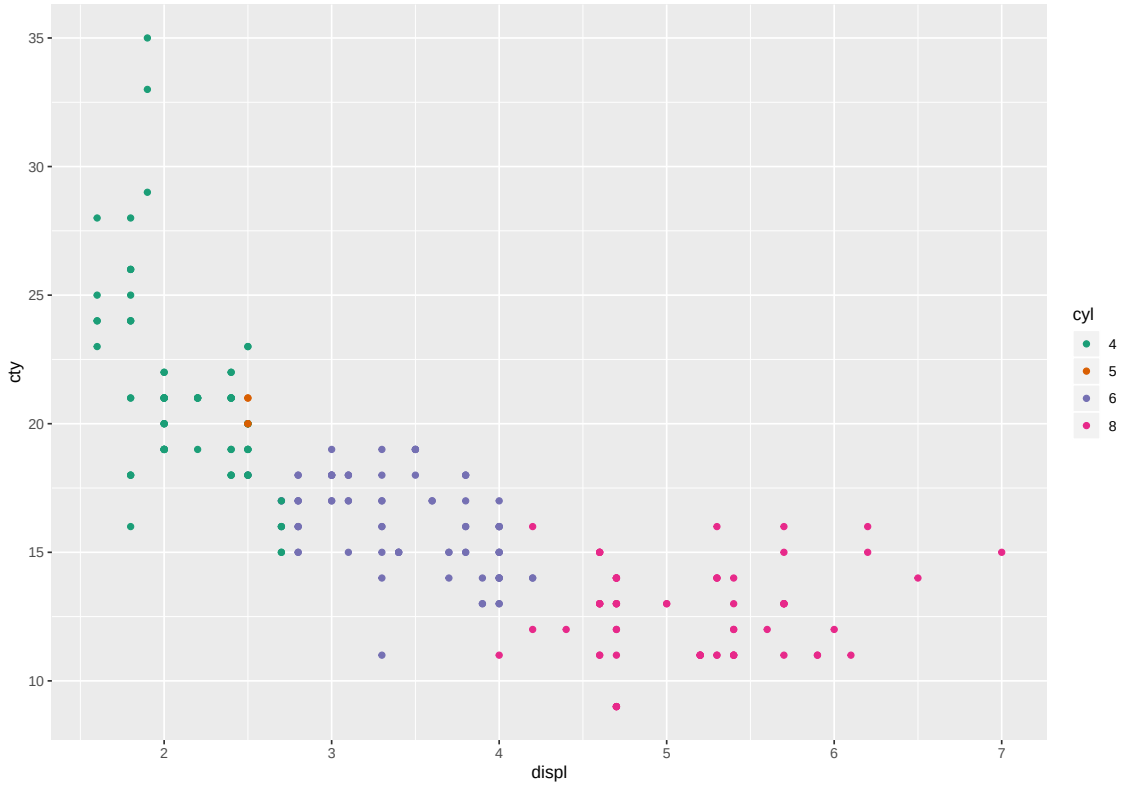
- Use RColorBrewer palettes
- The color palettes available in the **RColorBrewer** package are described here : color in R.
- The available color palettes in the RColorBrewer package are:



Colors

- Use RColorBrewer palettes

```
library(RColorBrewer);
ggplot(mpg, aes(x=displ, y=cty,color=cyl)) +
  geom_point() +
  scale_color_brewer(palette="Dark2");
```



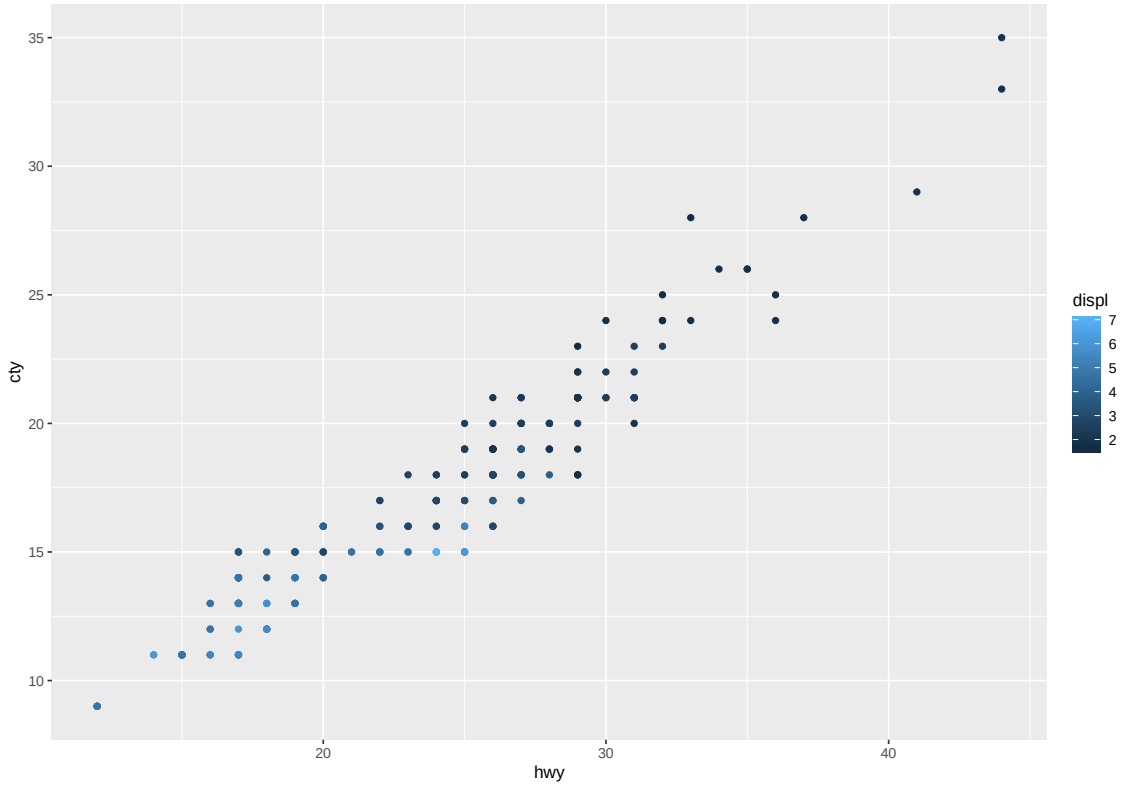
Colors

- Continuous colors: he graph can be colored according to the values of a continuous variable using the functions:
  - scale_color_gradient(), scale_fill_gradient() for sequential gradients between two colors
  - scale_color_gradient2(), scale_fill_gradient2() for diverging gradients
  - scale_color_gradientn(), scale_fill_gradientn() for gradient between n colors

Colors

- Continuous colors

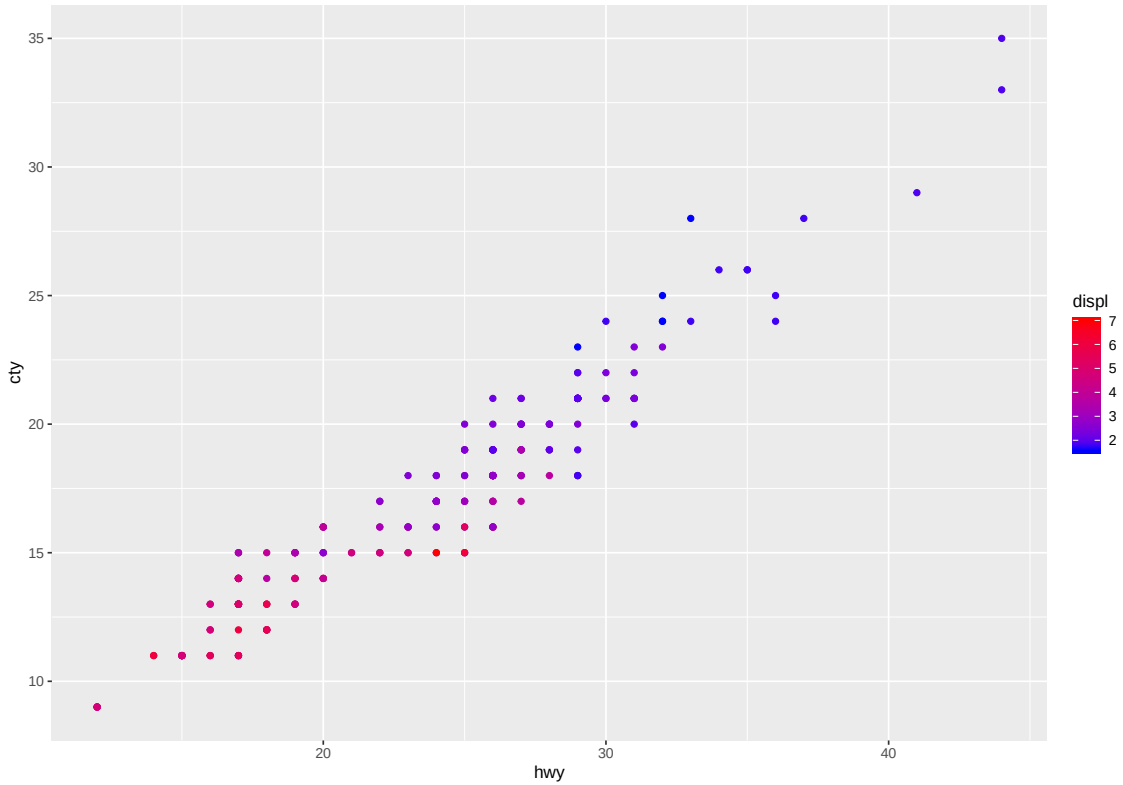
```
ggplot(mpg, aes(x=hwy, y=cty, color=displ)) +
  geom_point();
```



Colors

- Continuous colors

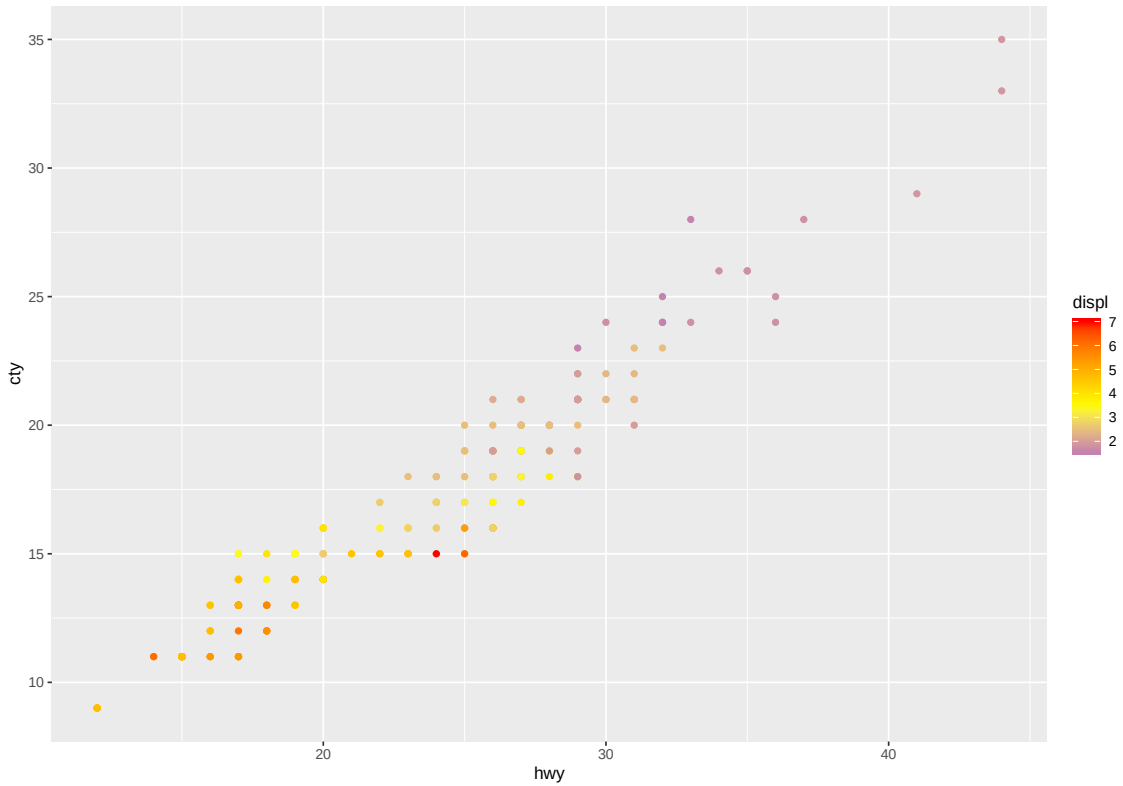
```
# Change the low and high colors
# Sequential color scheme
ggplot(mpg, aes(x=hwy, y=cty, color=displ)) +
  geom_point() +
  scale_color_gradient(low="blue", high="red");
```



Colors

- Continuous colors

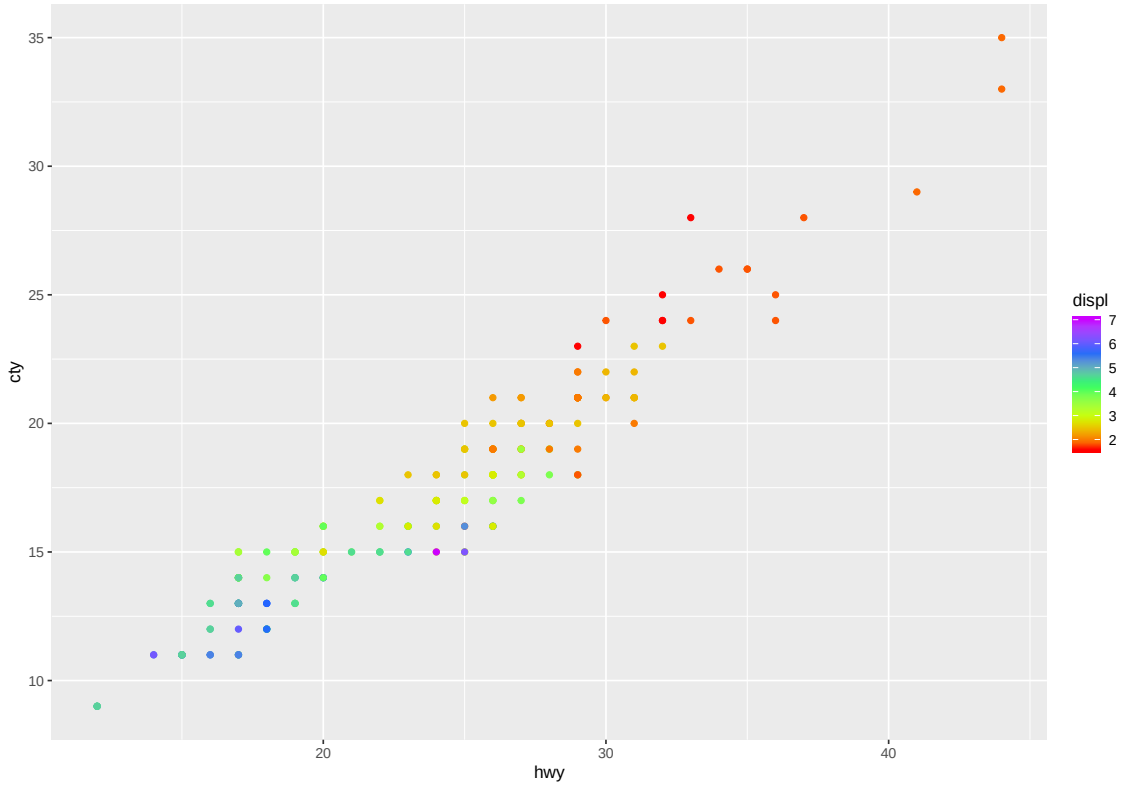
```
# Diverging color scheme
mid=mean(displ);  #average value of displ
ggplot(mpg, aes(x=hwy, y=cty, color=displ)) +
  geom_point() +
  scale_color_gradient2(midpoint=mid, low="blue",
  mid="yellow", high="red", space ="Lab");
```



Colors

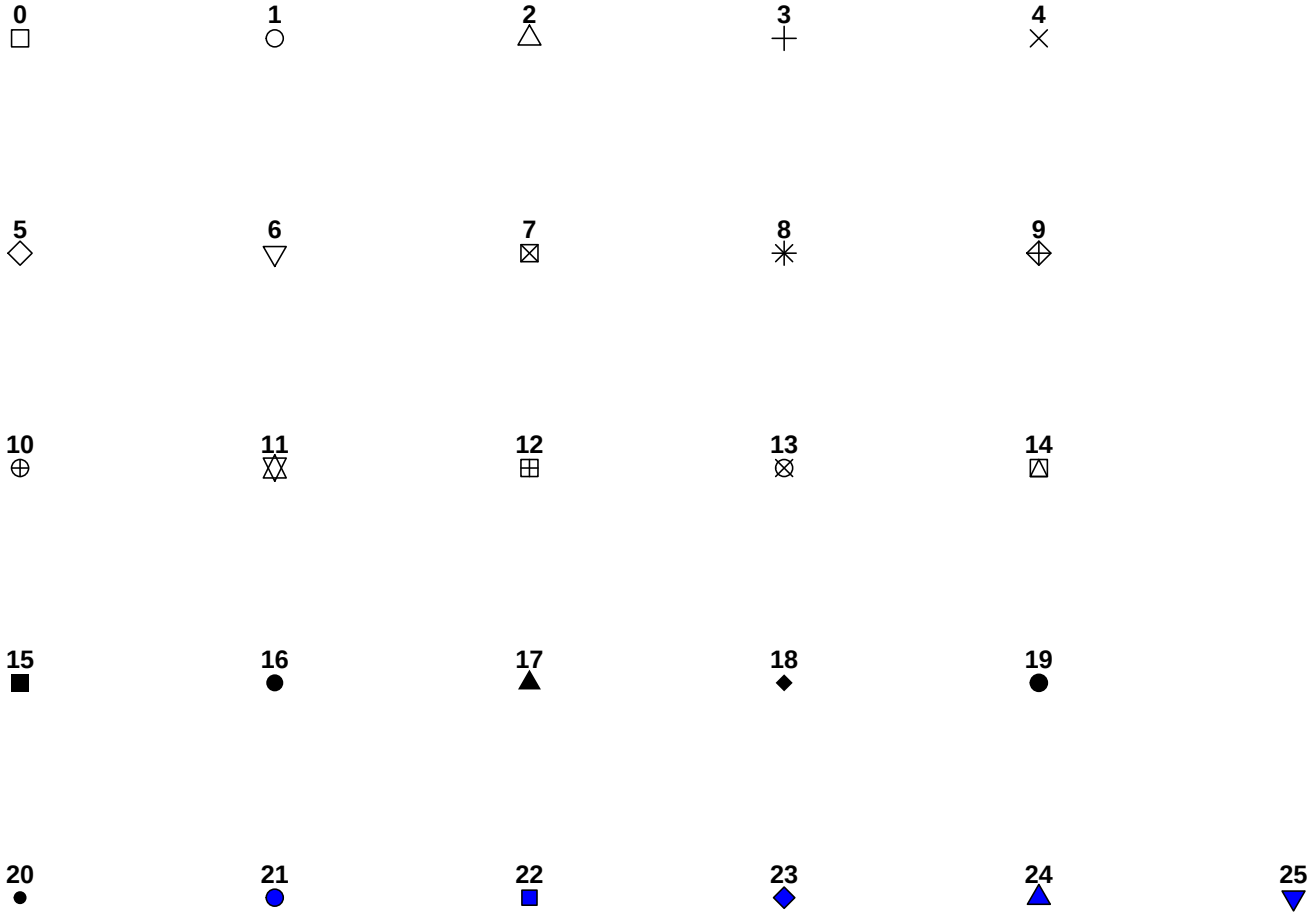
- Gradient between n colors

```
# Gradient between n colors
ggplot(mpg, aes(x=hwy, y=cty, color=displ)) +
  geom_point() +
scale_color_gradientn(colours = rainbow(5));
```



Points

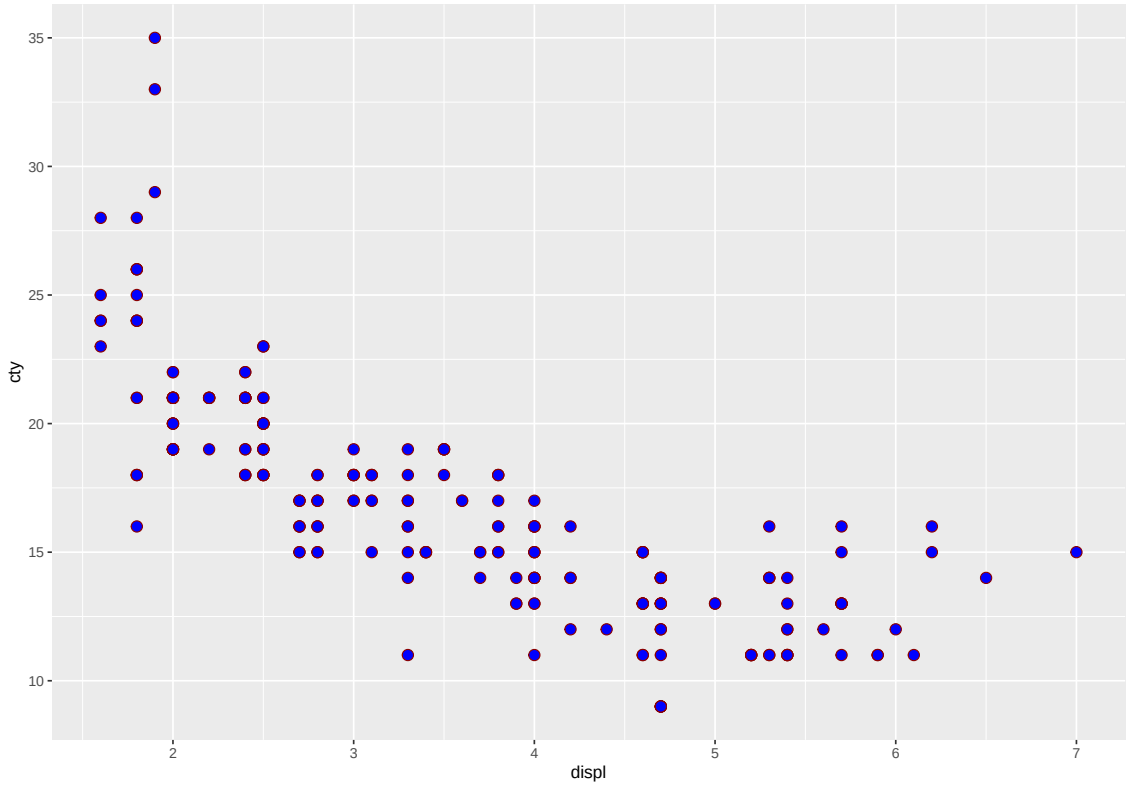
- Points shapes available in R:



Points

- Change the point shapes, colors and sizes automatically

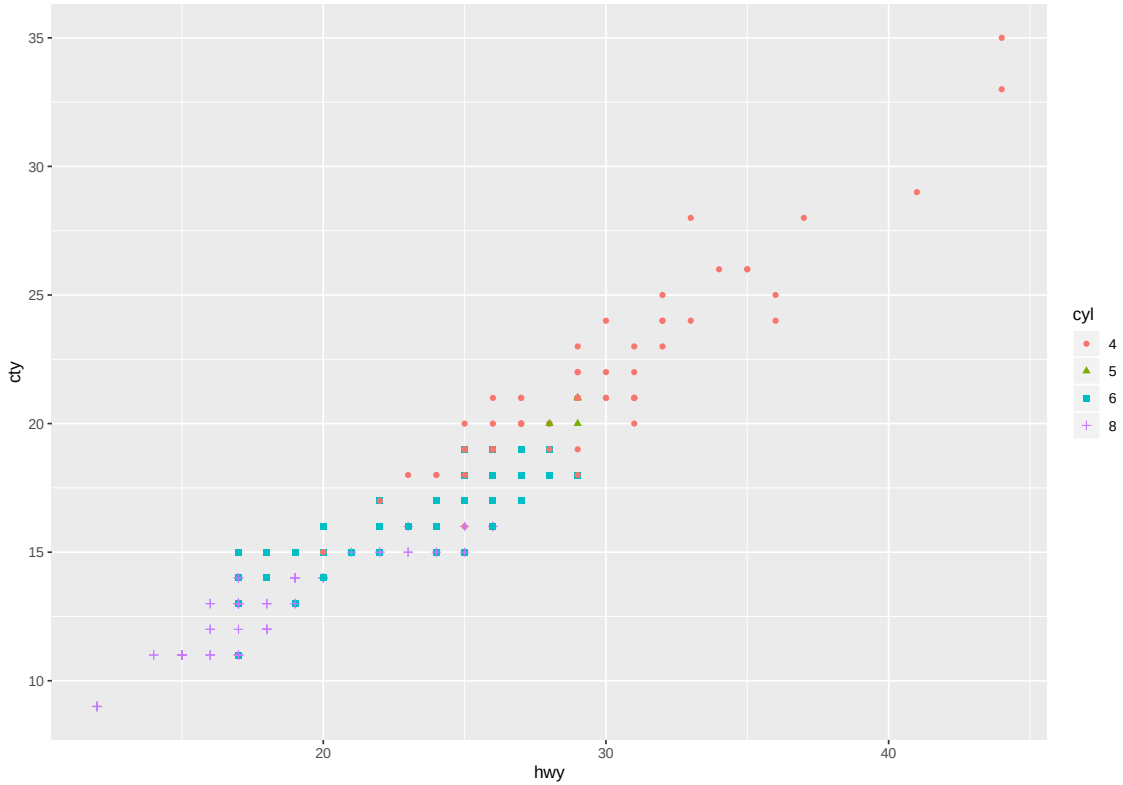
```
ggplot(mpg, aes(x=displ, y=cty)) +
  geom_point(shape=21, fill="blue", color="darkred", size=3);
```



Points

- Change the point shapes, colors and sizes automatically

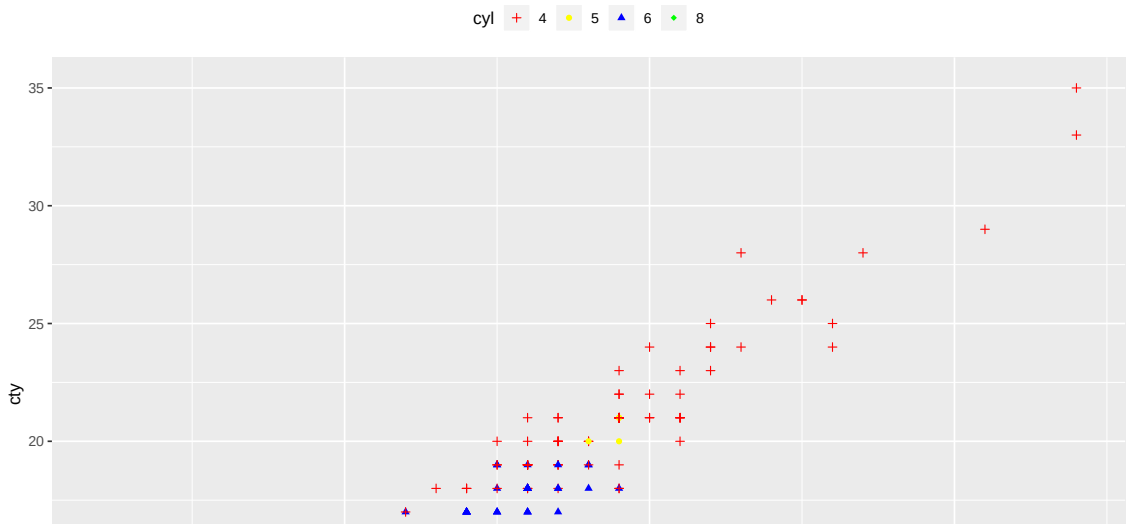
```
ggplot(mpg, aes(x=hwy, y=cty, group=cyl)) +
  geom_point(aes(shape=cyl, color=cyl));
```



Points

- Change point shapes, colors and sizes manually. The functions below can be used:
  - scale_shape_manual() : to change point shapes
  - scale_color_manual() : to change point colors
  - scale_size_manual() : to change the size of points

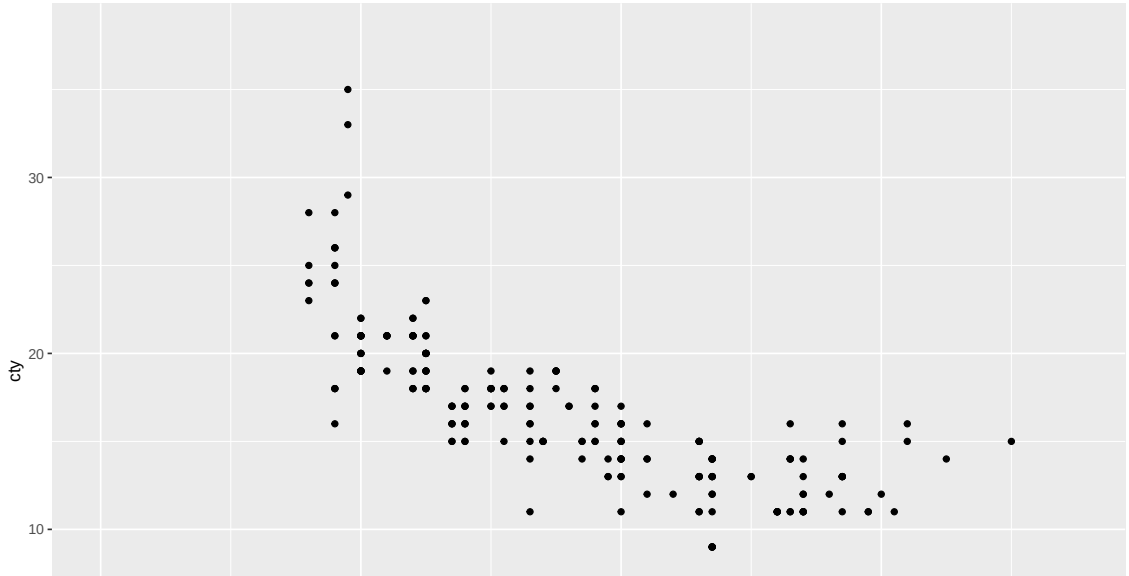
```
ggplot(mpg, aes(x=hwy, y=cty, group=cyl)) +
  geom_point(aes(shape=cyl, color=cyl)) +
  scale_shape_manual(values=c(3, 16, 17, 18))+
  scale_color_manual(values=c("red", "yellow","blue", "green"))+
  scale_size_manual(values=c(2,3,4,5))+
  theme(legend.position="top");
```



Axis scales

- Change x and y axis limits. There are different functions to set axis limits:
  - xlim() and ylim()
  - expand_limits()
  - scale_x_continuous() and scale_y_continuous()
- Use xlim() and ylim() functions
  - xlim(min, max) # x axis limits
  - ylim(min, max) # y axis limits

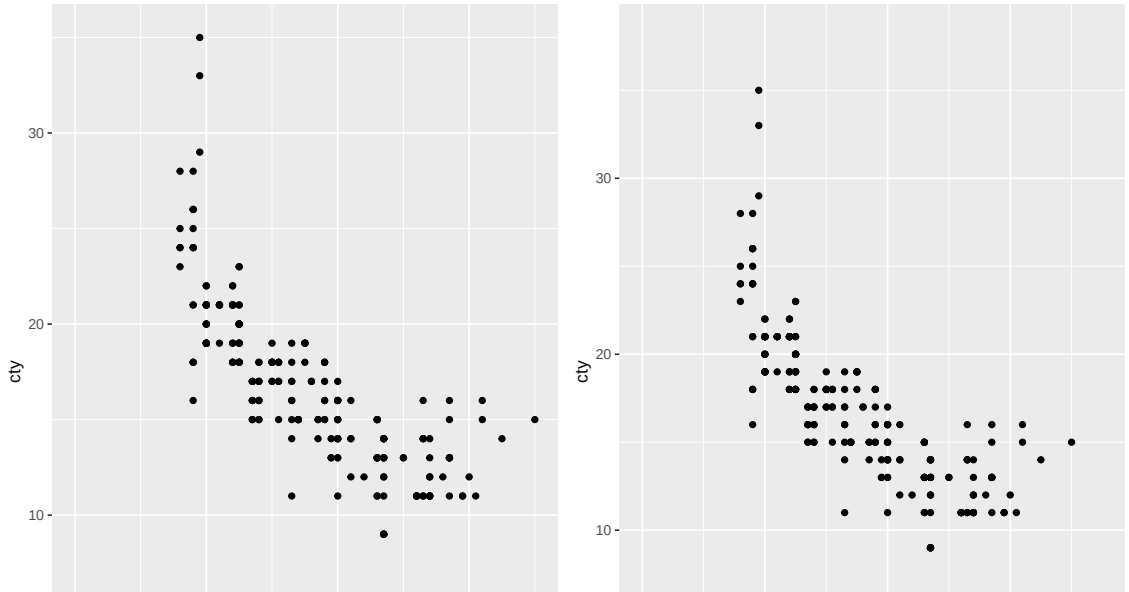
```
ggplot(mpg, aes(x=displ, y=cty)) + geom_point() +
  xlim(0, 7.5)+ylim(0, 38);
```



Axis scales

- Change x and y axis limits using `expand_limts()` function. The function `expand_limits()` can be used to:
  - quickly set the intercept of x and y axes at (0,0)
  - change the limits of x and y axes

```
library(gridExtra);
p1=ggplot(mpg, aes(x=displ, y=cty)) + geom_point() +
  expand_limits(x=0, y=0);
p2=ggplot(mpg, aes(x=displ, y=cty)) + geom_point() +
  expand_limits(x=c(0,7.5), y=c(0, 38));
grid.arrange(p1, p2, ncol=2);
```

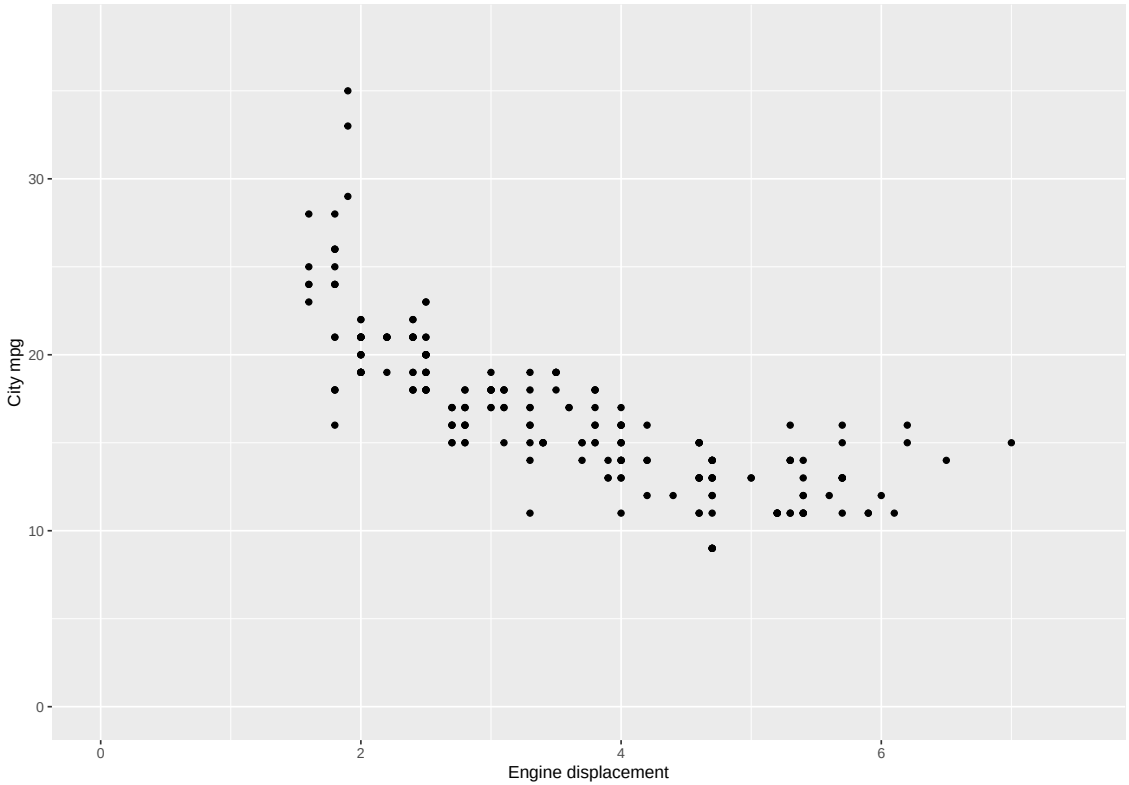


Axis scales

- It is also possible to use the functions `scale_x_continuous()` and `scale_y_continuous()` to change x and y axis limits, respectively.
  - scale_x_continuous(name, breaks, labels, limits, trans)
  - scale_y_continuous(name, breaks, labels, limits, trans)
- The arguments are
  - name : x or y axis labels
  - breaks : to control the breaks in the guide (axis ticks, grid lines, .). Among the possible values, there are :
    - ★ NULL : hide all breaks
    - ★ waiver() : the default break computation
    - ★ a character or numeric vector specifying the breaks to display
  - labels : labels of axis tick marks. Allowed values are :
    - ★ NULL for no labels
    - ★ waiver() for the default labels
    - ★ character vector to be used for break labels
  - limits : a numeric vector specifying x or y axis limits (min, max)
  - trans for axis transformations. Possible values are "log2", "log10", ...

Axis scales

```
ggplot(mpg, aes(x=displ, y=cty)) + geom_point() +
scale_x_continuous(name="Engine displacement", limits=c(0, 7.5)) +
scale_y_continuous(name="City mpg", limits=c(0, 38))
```



Questions?