Plotting in R

Oxford Biomedical Data Science Training Programme
University of Oxford
2020-05-26

Overview

- Plotting in base R
- Plotting with ggplot2 package

Plotting in base R

UNIVERSITY OF OXFORD

```
plot(x, y)          # scatter plot

hist(x)             # histogram

plot(x, y, type = "l")          # line graph

barplot(x, y)           # bar graph

boxplot(data) or boxplot(formula, data)          # boxplot
```

Plotting in base R

A number of parameters that modify plot appearance are shared between base plotting functions, including:

- `xlab`: x axis label
- `ylab`: y axis label
- `col`: colour
- `pch`: plotting symbol (default is open circle)
- `lty`: line type (default is solid line)
- `lwd`: line width

Plotting in base R

`par()` function is used to set global graphical parameters, such as:

- `mar`: margin size
- `oma`: outer margin size (default 0 for all sides)
- `bg`: background colour
- `las`: orientation of axis labels on the plot
- `mfrow`: number of plots per row and column (plots filled row-wise)
- `mfcol`: number of plots per row and column (plots filled column-wise)

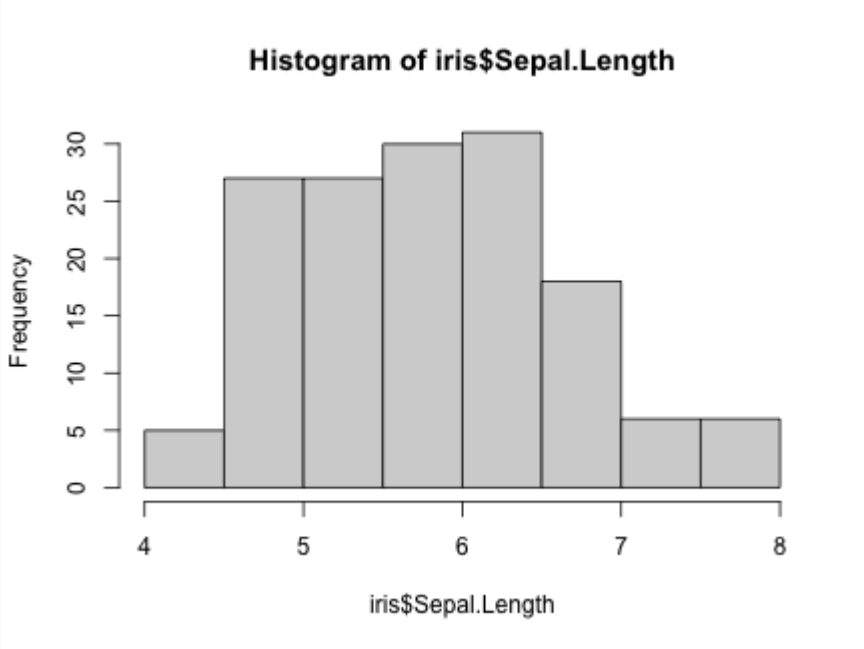
Check current parameters using e.g. `par()$bg`, `par()$mfrow`

Change parameters e.g.

```
par(bg = "red")          # change background colour to red
par(pch = 18)            # change plot symbol to filled diamond
```

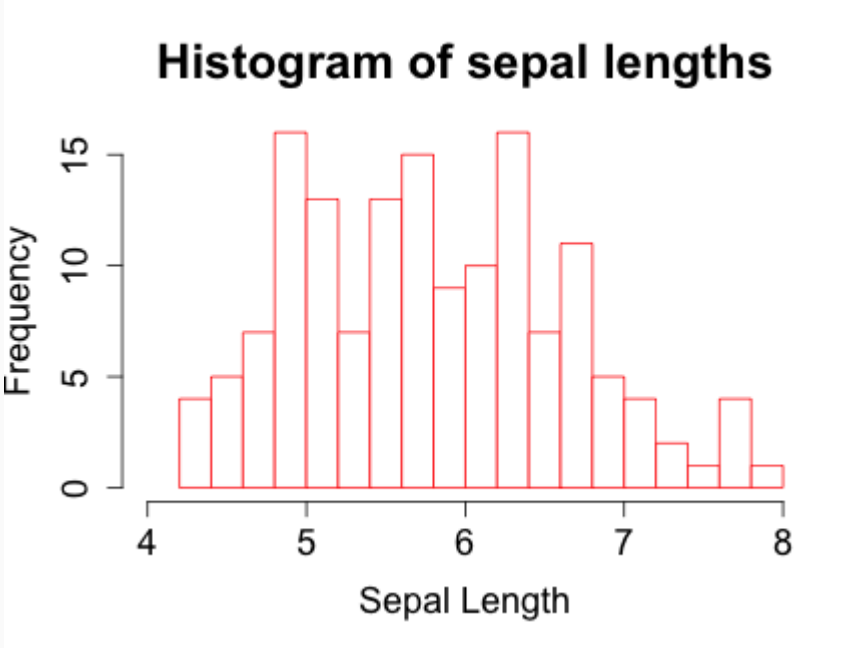
Plotting in base R

```
data(iris)
hist(iris$Sepal.Length)
```



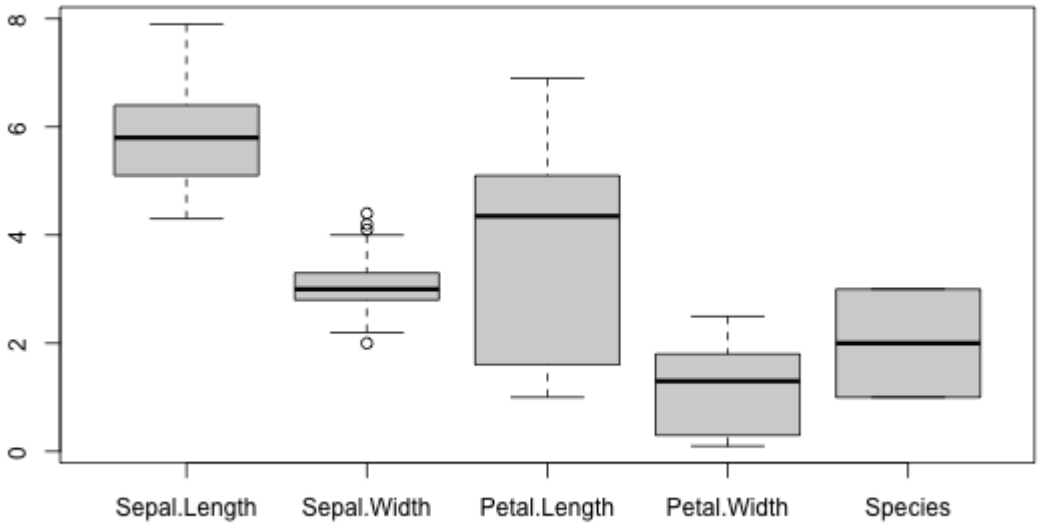
Plotting in base R

```
hist(iris$Sepal.Length, breaks = 25, xlab = "Sepal Length",
     main = "Histogram of sepal lengths", col = "white", border = "red",
     xlim = c(4, 8), cex.lab = 1.5, cex.axis = 1.5, cex.main = 2)
```



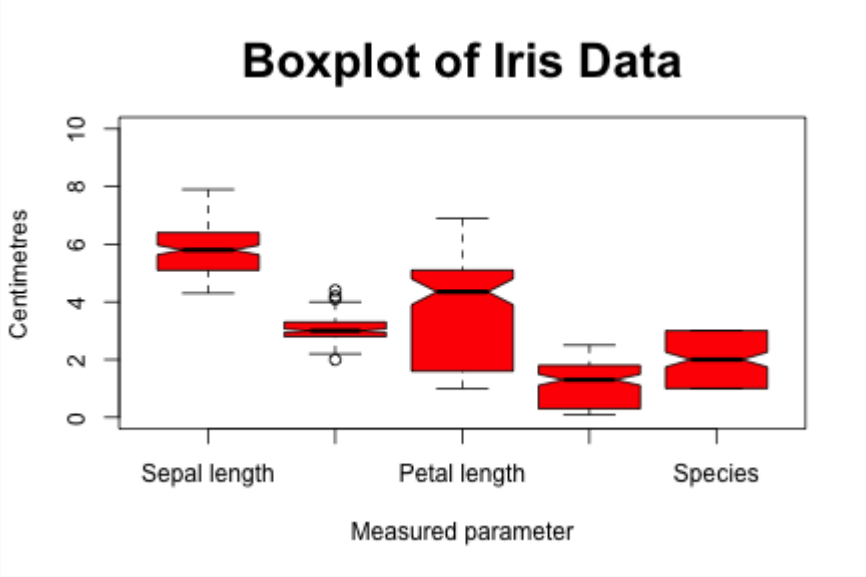
Plotting in base R

```
boxplot(iris)
```



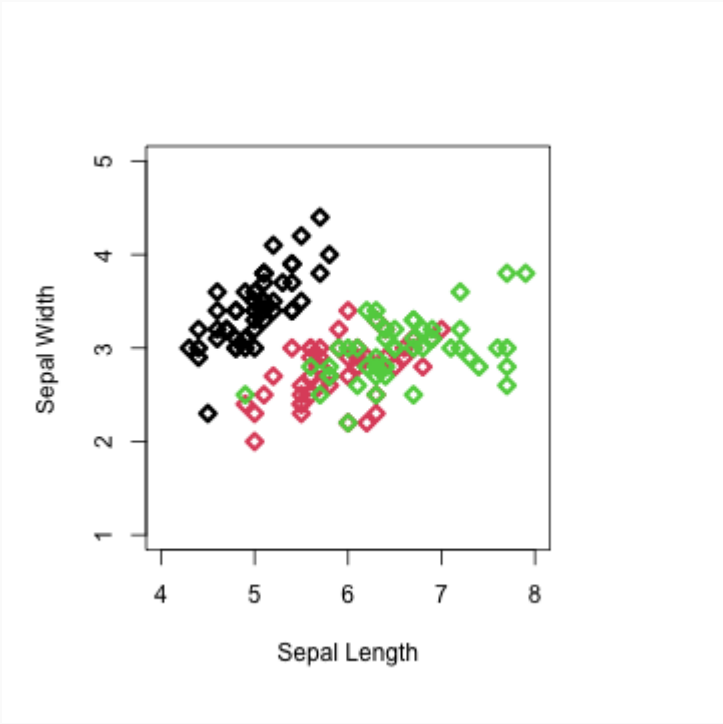
Plotting in base R

```r
boxplot(iris, xlab = "Measured parameter", ylab = "Centimetres",
        col = "red", border = "black", xaxt = "n", notch = TRUE,
        ylim = c(0, 10))

axis(1, at = 1:5, labels = c("Sepal length", "Sepal width",
                             "Petal length", "Petal width", "Species"))

title(main = "Boxplot of Iris Data", cex.main = 2)
```



Plotting in base R

```
par(mai = c(1.2, 1, 1, 1.2))

plot(iris$Sepal.Length, iris$Sepal.Width, xlab = "Sepal Length",
     ylab = "Sepal Width", xlim = c(4, 8), ylim = c(1, 5),
     col = iris$Species, lwd = 3, pch = 5)
```



Base R plotting exercises

Load built-in mtcars dataset

```
help(mtcars)        # information about the mtcars dataset
```

1. Draw a bar plot showing the number of cars with 3, 4 or 5 forward gears. Hint: use `table()` function to get data into the correct format.
    - Add x and y axis labels
    - Add a main title spread across two lines
    - Change the colour and width of the bars
    - Add a horizontal line at y = 6
    - Change plot margins to 6, 6, 5, 5

Base R plotting exercises

2. Generate a scatter plot of mpg vs. hp coloured by gear values
    - Change points to filled circles and increase their size
    - Add x and y axis titles and change size
    - Add a legend
    - Change colours to red, green and blue

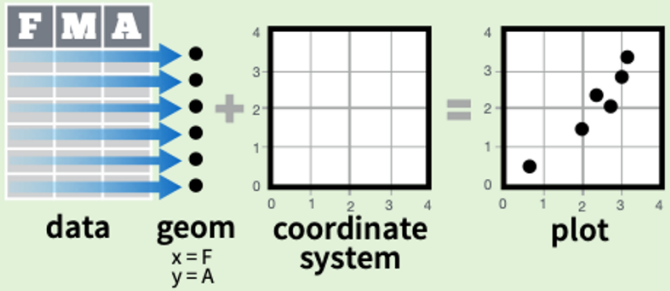
3. Plot the two plots that you have just made side-by-side by changing the global graphical parameters

ggplot2

- Popular plotting package in R
- Part of the Tidyverse - data needs to be in tidy format (long)
- Cheat sheet - https://github.com/rstudio/cheatsheets/blob/master/data-visualization-2.1.pdf

ggplot2

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.**



ggplot2 geoms

geoms (geometric objects) - actual marks on the plot, must have at least one

- `geom_point()` # scatter plot
- `geom_bar()` # specify data - bar chart
- `geom_histogram()` # histogram
- `geom_boxplot()` # box plot
- `geom_line()` # line diagram, observations connected by x value

ggplot2 examples using diamonds dataset

A dataset containing the price and other attributes of ~54,000 diamonds

```
help(diamonds)
```

Format

A data frame with 53940 rows and 10 variables:

price

price in US dollars (\$326--\$18,823)

carat

weight of the diamond (0.2--5.01)

cut

quality of the cut (Fair, Good, Very Good, Premium, Ideal)

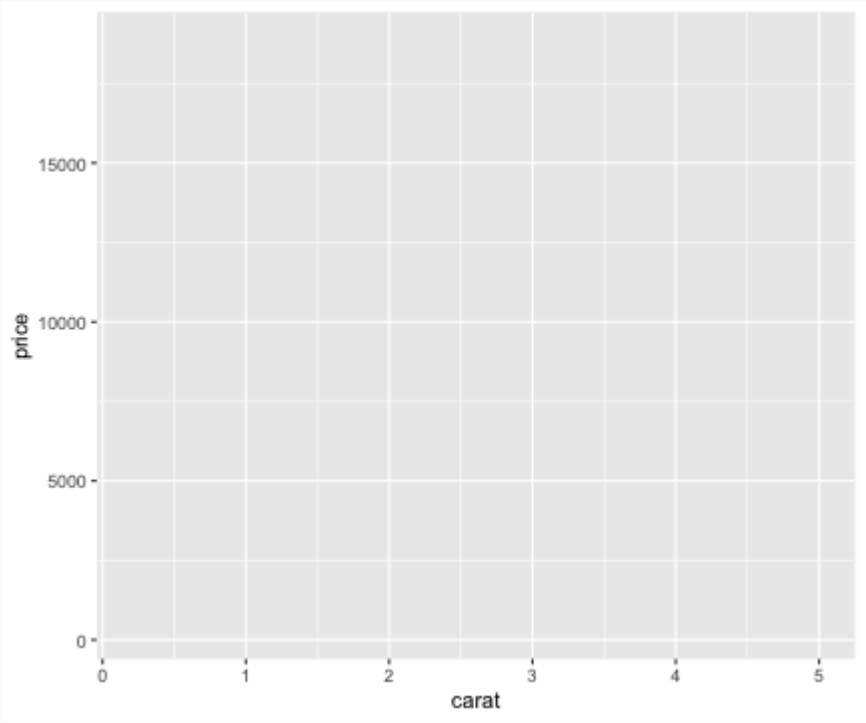
color

diamond colour, from D (best) to J (worst)

ggplot2 examples using diamonds dataset

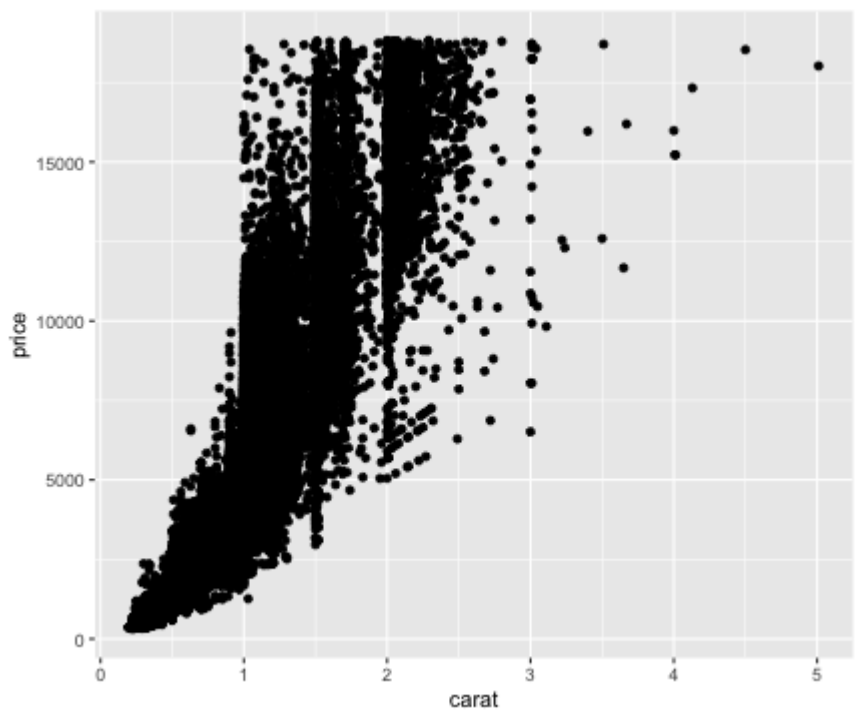
```
ggplot(diamonds)    # specify data, won't generate a plot
```

```
# Specify data and variables to put on the x and y axes
ggplot(diamonds, aes(x = carat, y = price))   # generates axes only
```



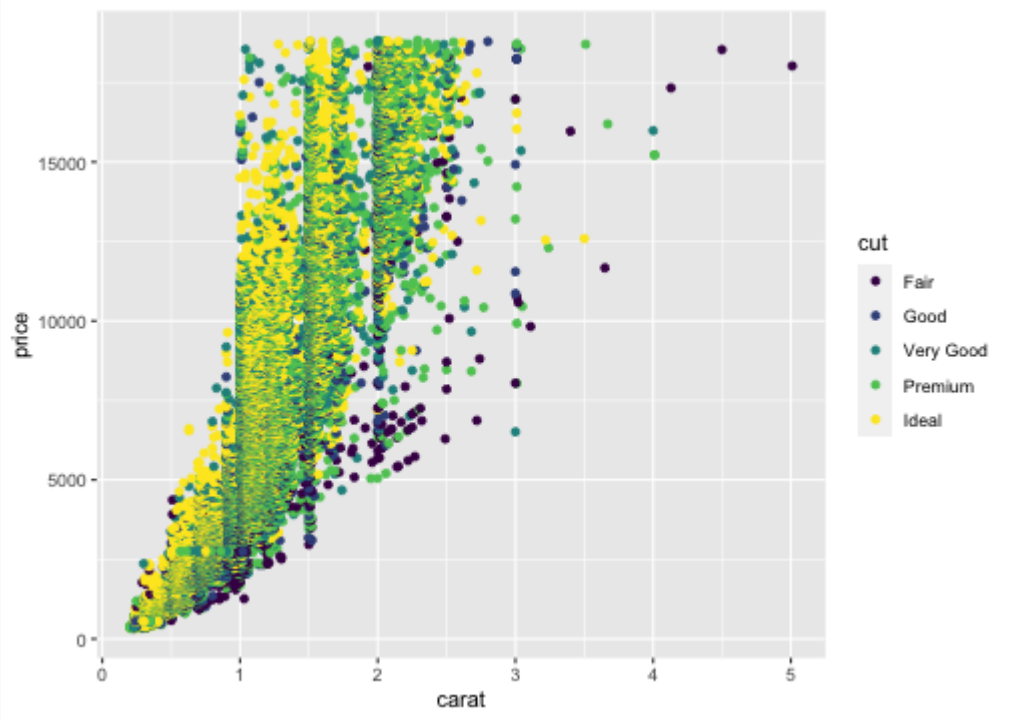
ggplot2 examples using diamonds dataset

```
ggplot(diamonds, aes(x = carat, y = price)) +
  geom_point()  # add geom_point() to specify that you want a scatter plot
```



ggplot2 examples using diamonds dataset

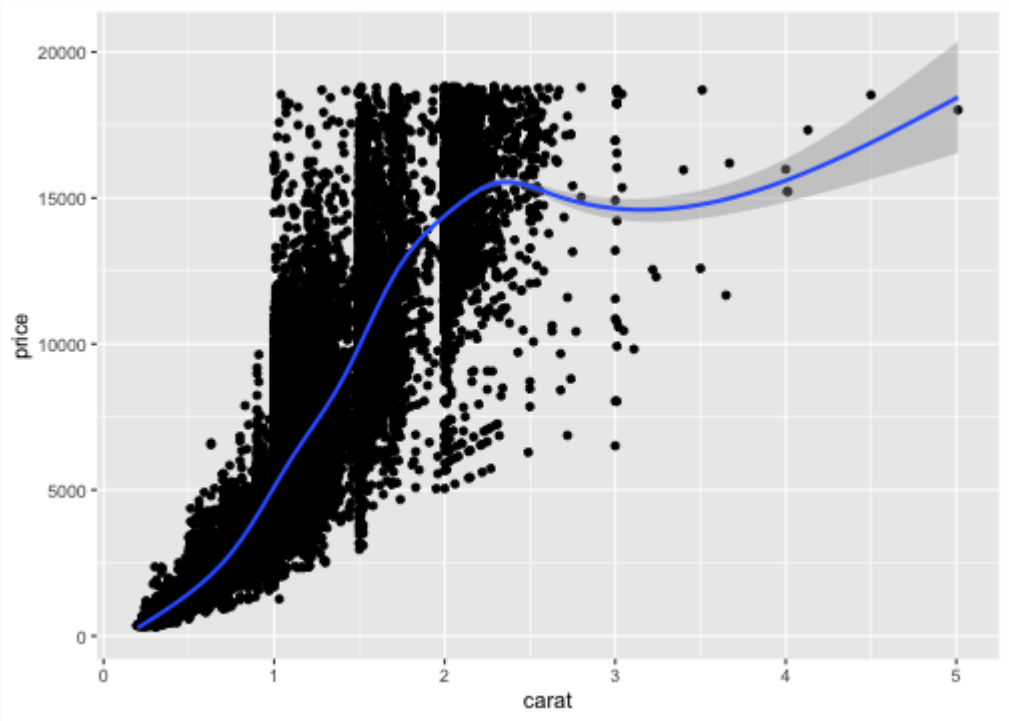
```
# Colour by diamond cut
ggplot(diamonds, aes(x = carat, y = price, colour = cut)) +
  geom_point()
```



ggplot2 examples using diamonds dataset

ggplot layers:

```
# Add a trend line with geom_smooth()
ggplot(diamonds, aes(x = carat, y = price)) +
  geom_point() +
  geom_smooth()
```



ggplot2 examples using diamonds dataset

Can anyone predict what the difference between these two plots will be?

Plot 1:

```
ggplot(diamonds, aes(x = carat, y = price, colour = cut)) +
  geom_point() +
  geom_smooth()
```

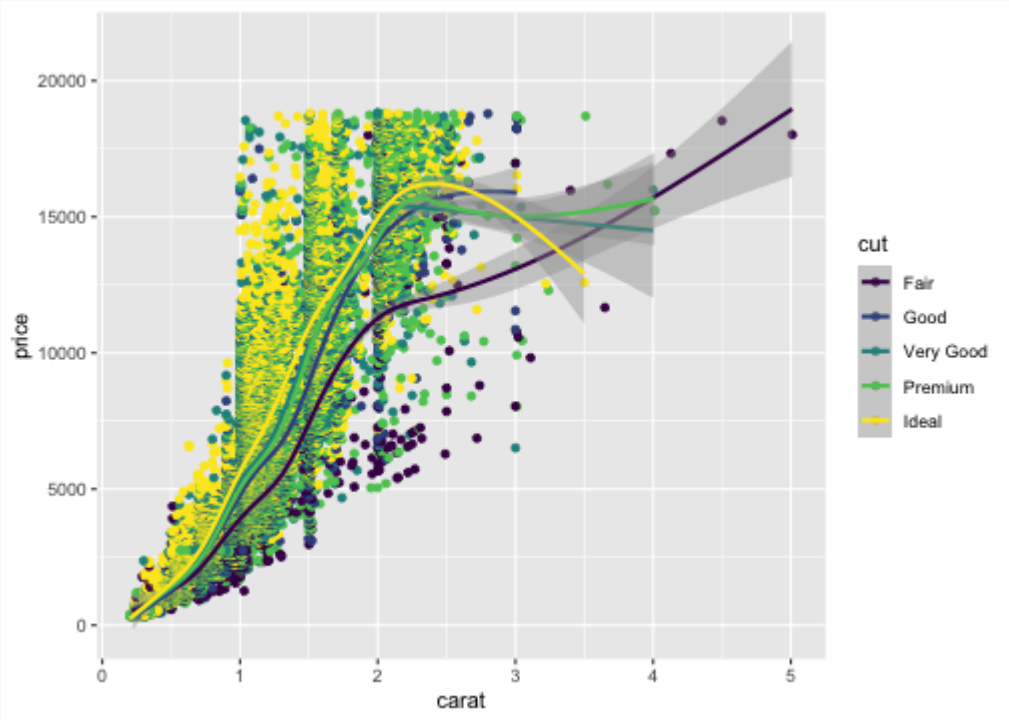
Plot 2:

```
ggplot(diamonds, aes(x = carat, y = price)) +
  geom_point(aes(colour = cut)) +
  geom_smooth()
```

ggplot2 examples using diamonds dataset

Plot 1:

```
ggplot(diamonds, aes(x = carat, y = price, colour = cut)) +
  geom_point() +
  geom_smooth()
```

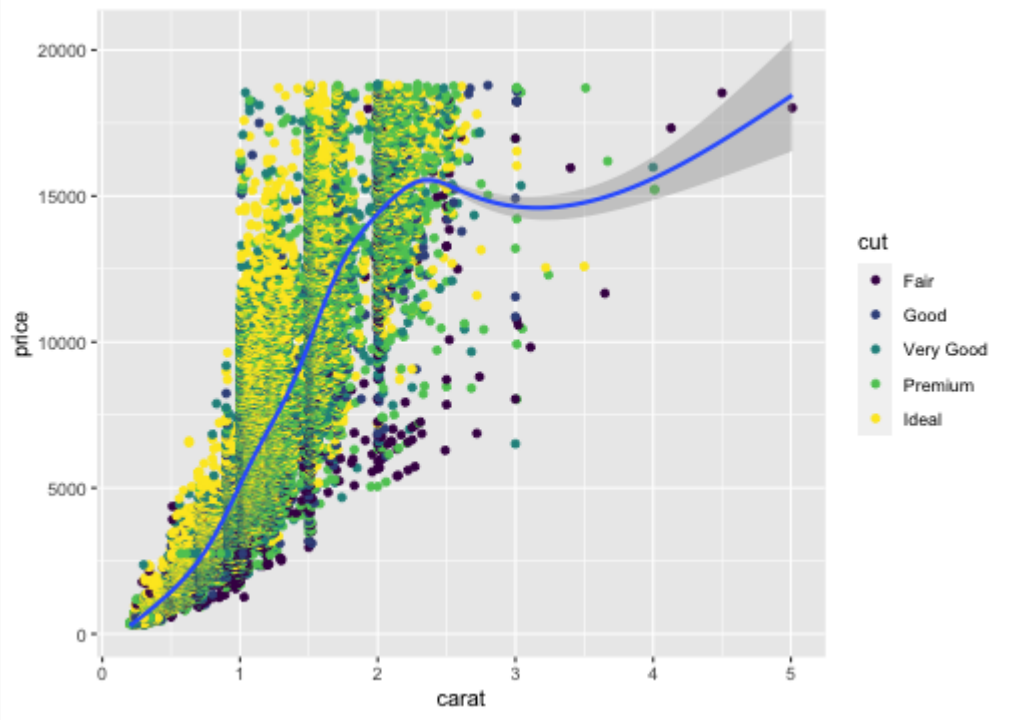


Colour aesthetic is passed onto `geom_point()` and `geom_smooth()`

ggplot2 examples using diamonds dataset

Plot 2:

```
ggplot(diamonds, aes(x = carat, y = price)) +
  geom_point(aes(colour = cut)) +
  geom_smooth()
```



Colour aesthetic is only applied to `geom_point()`

ggplot2 themes

UNIVERSITY OF OXFORD

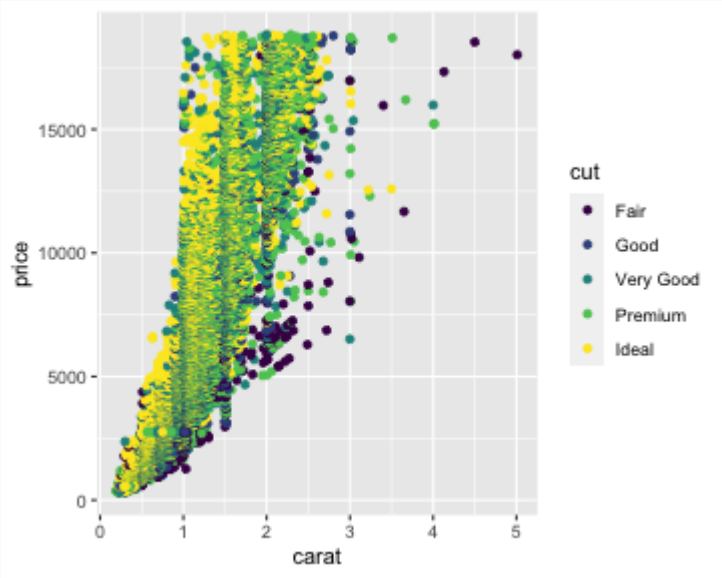
- `theme()` is used to modify the non-data aspects of a plot:
  - Line elements e.g. axis lines, plot panel border
  - Text elements e.g. plot title, axis titles, axis tick mark labels
  - Rectangle elements e.g. plot background, legend background
- `theme_get()` to get detailed information about the current theme
- Standard ggplot themes are available e.g. `theme_classic()`, `theme_minimal()`, `theme_bw()`
- Can also modify aspects of a theme individually e.g. axis text size

Default ggplot2 theme

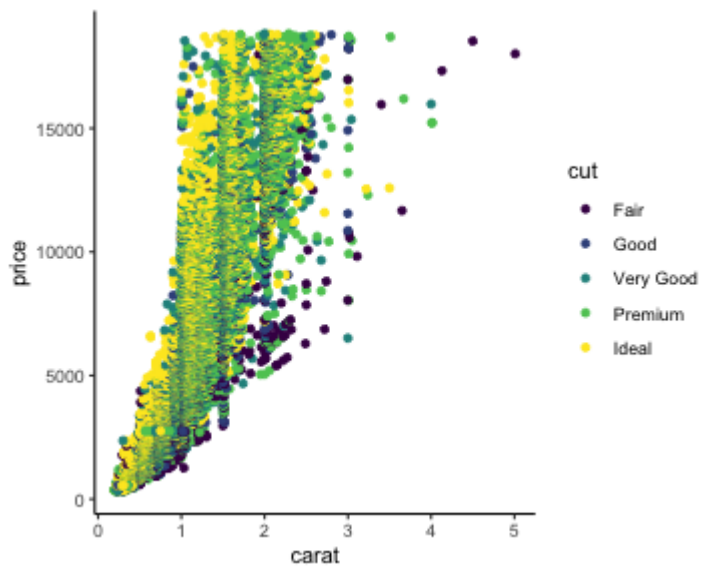
```
t <- theme(
  # Elements in this first block aren't used directly, but are inherited
  # by others
  line =                element_line(
                          colour = "black", size = base_line_size,
                          linetype = 1, lineend = "butt"
                        ),
  rect =                element_rect(
                          fill = "white", colour = "black",
                          size = base_rect_size, linetype = 1
                        ),
  text =                element_text(
                           family = base_family, face = "plain",
                           colour = "black", size = base_size,
                           lineheight = 0.9, hjust = 0.5, vjust = 0.5, angle = 0,
                           margin = margin(), debug = FALSE
                        ),

  axis.line =           element_blank(),
  axis.line.x =         NULL,
  axis.line.y =         NULL,
  axis.text =           element_text(size = rel(0.8), colour = "grey30"),
  axis.text.x =         element_text(margin = margin(t = 0.8 * half_line / 2), vjust = 1),
  axis.text.x.top =     element_text(margin = margin(b = 0.8 * half_line / 2), vjust = 0),
  axis.text.y =         element_text(margin = margin(r = 0.8 * half_line / 2), hjust = 1),
  axis.text.y.right =   element_text(margin = margin(l = 0.8 * half_line / 2), hjust = 0),
  axis.ticks =          element_line(colour = "grey20"),
  axis.ticks.length =   unit(half_line / 2, "pt"),
  axis.ticks.length.x = NULL,
  axis.ticks.length.x.top = NULL,
  axis.ticks.length.x.bottom = NULL,
  axis.ticks.length.y = NULL,
  axis.ticks.length.y.left = NULL,
  axis.ticks.length.y.right = NULL,
  axis.title.x =        element_text(
                          margin = margin(t = half_line / 2),
                          vjust = 1
                        ),
  axis.title.x.top =    element_text(
                          margin = margin(b = half_line / 2),
                          vjust = 0
                        ),
```

Standard ggplot2 themes

```
ggplot(diamonds,
       aes(x = carat,
           y = price,
           colour = cut)) +
      geom_point()
```

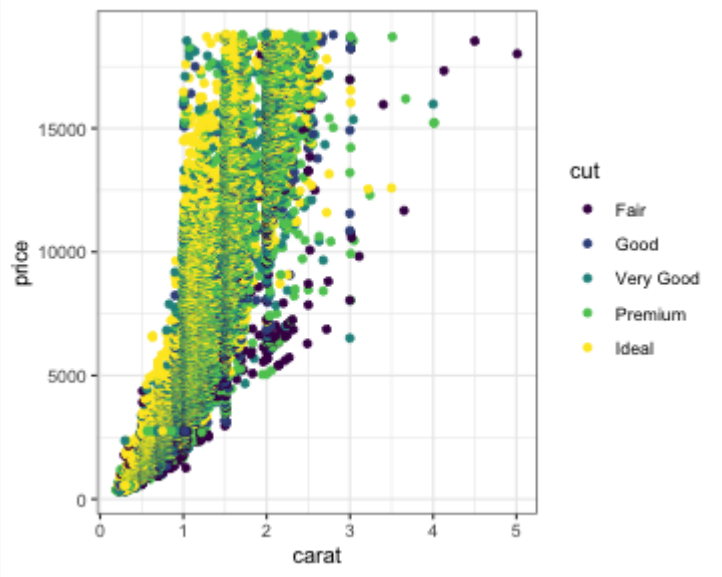



```
ggplot(diamonds,
       aes(x = carat,
           y = price,
           colour = cut)) +
      geom_point() +
      theme_classic()
```

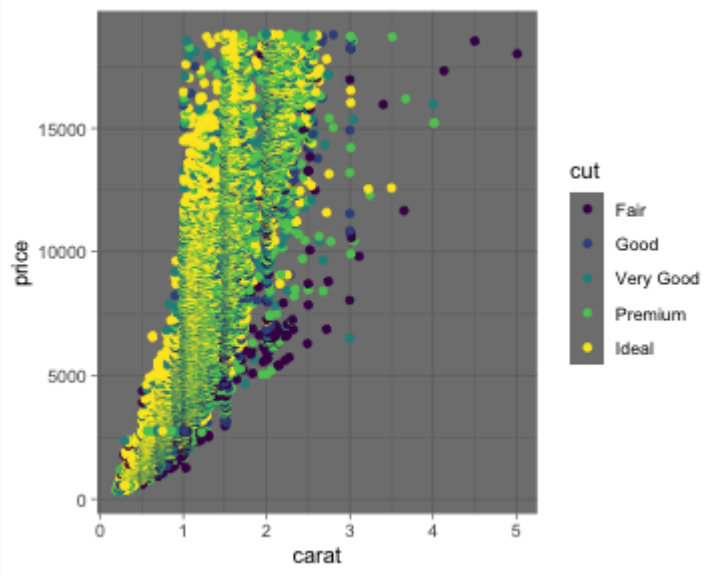


Standard ggplot2 themes

```
ggplot(diamonds,
       aes(x = carat,
           y = price,
           colour = cut)) +
      geom_point() +
      theme_bw()
```

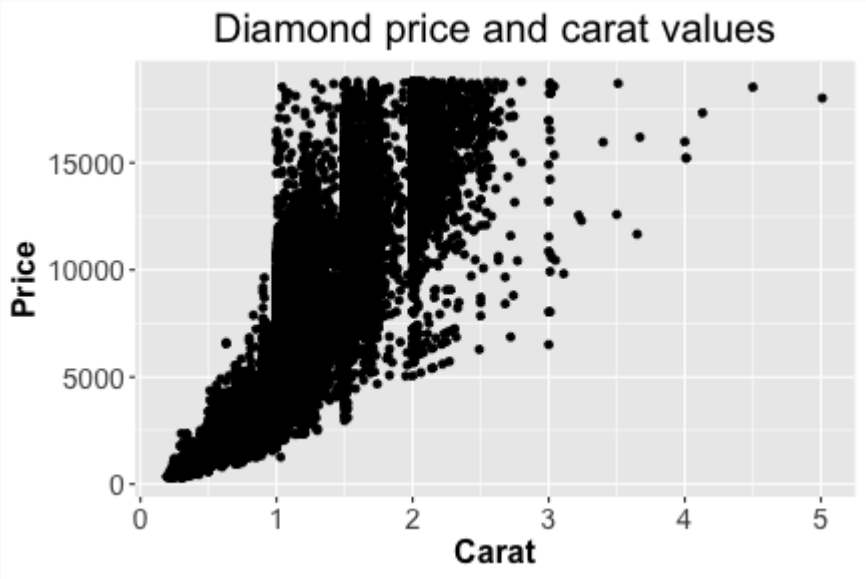



```
ggplot(diamonds,
       aes(x = carat,
           y = price,
           colour = cut)) +
      geom_point() +
      theme_dark()
```



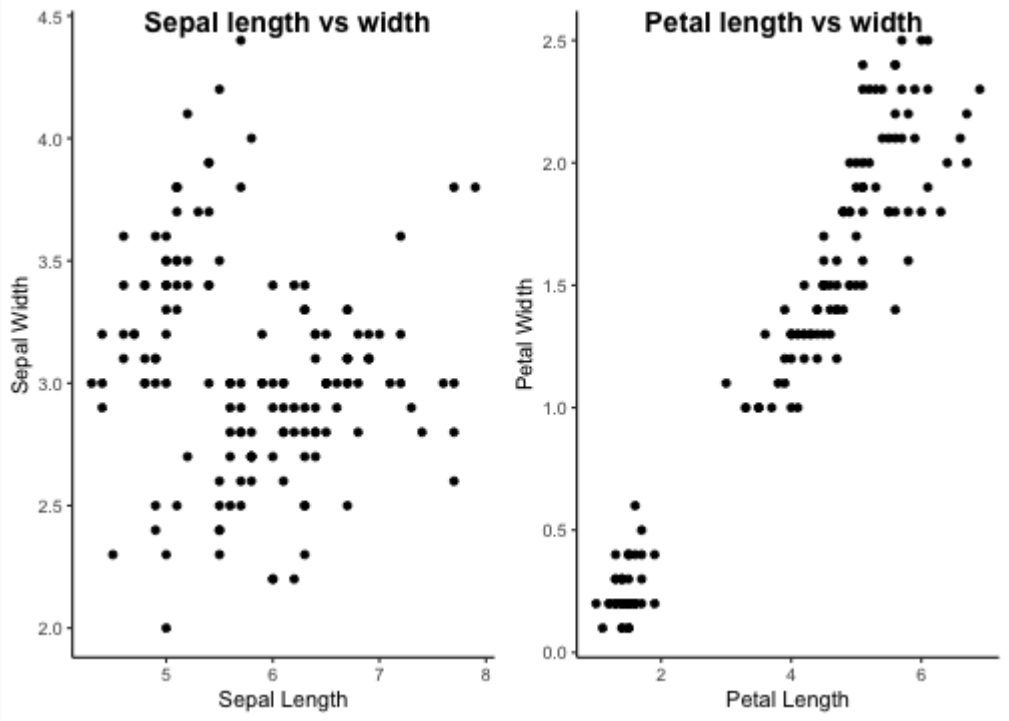
Modifying the ggplot2 theme

```
ggplot(diamonds, aes(x = carat, y = price)) +
    geom_point() +
    labs(title = "Diamond price and carat values",
         x = "Carat", y = "Price") +
    theme(axis.title = element_text(size = 16, face = "bold"),
          axis.text = element_text(size = 14),
          plot.title = element_text(hjust = 0.5, size = 20))
```



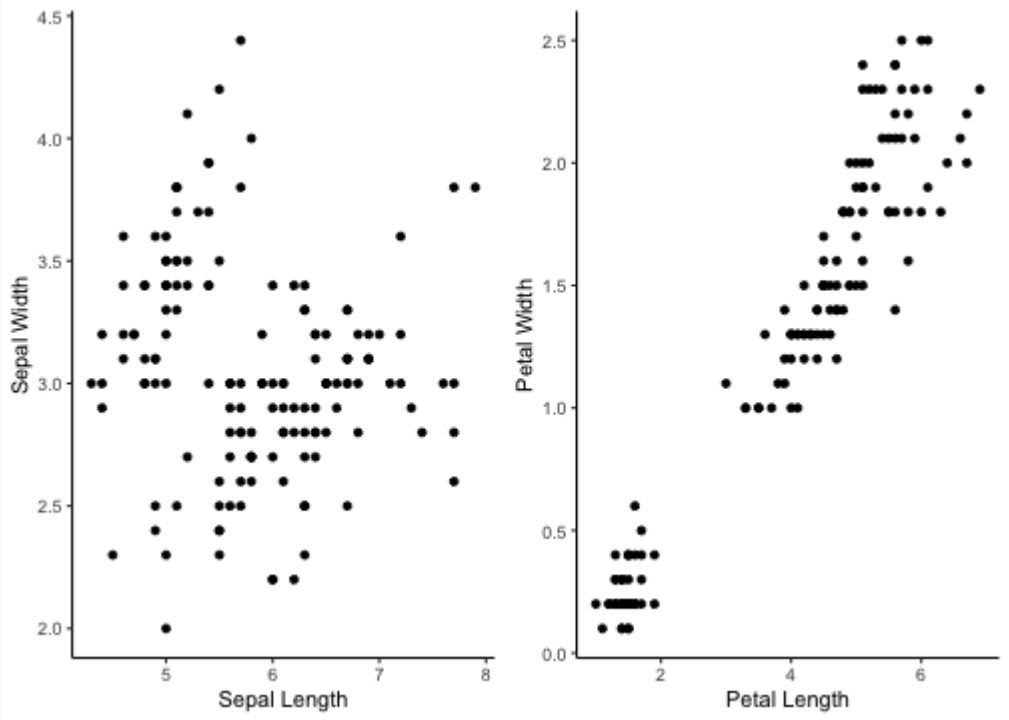
Arrange multiple plots into a grid

```
library(cowplot)
plot_grid(plot1, plot2,      # plot1 and plot2 are ggplot objects
          labels = c("Sepal length vs width", "Petal length vs width"),
          ncol = 2, nrow = 1)
```



Arrange multiple plots into a grid

```
library(gridExtra)
grid.arrange(plot1, plot2,      # plot1 and plot2 are ggplot objects
             ncol = 2, nrow = 1)
```



ggplot2 facets

Split data by one or more categorical variables

Plot subsets of data together in a grid

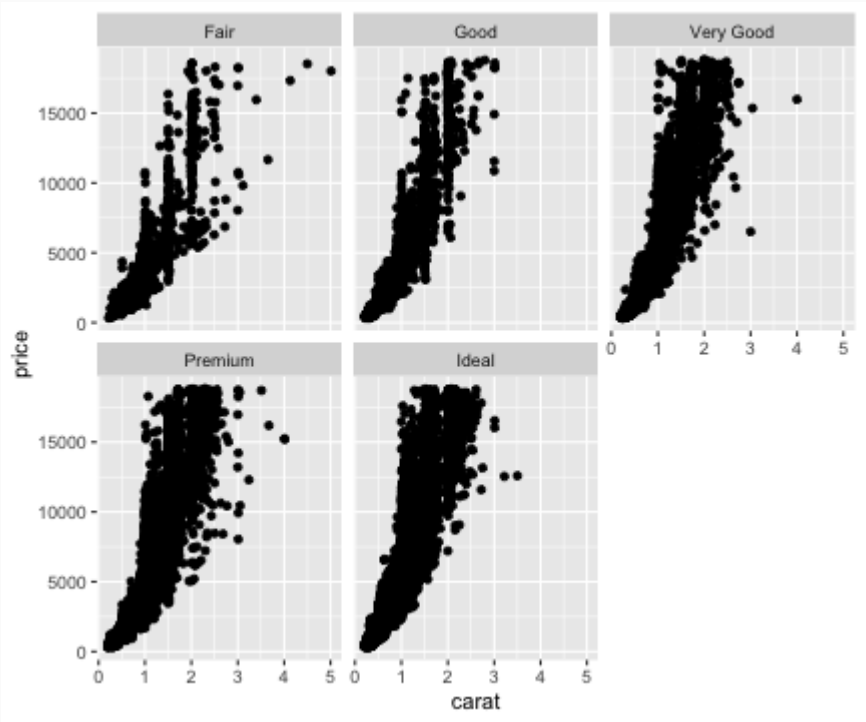
Two functions in ggplot:

- `facet_grid()`
- `facet_wrap()`

ggplot2 facets

Single variable using `facet_wrap()`

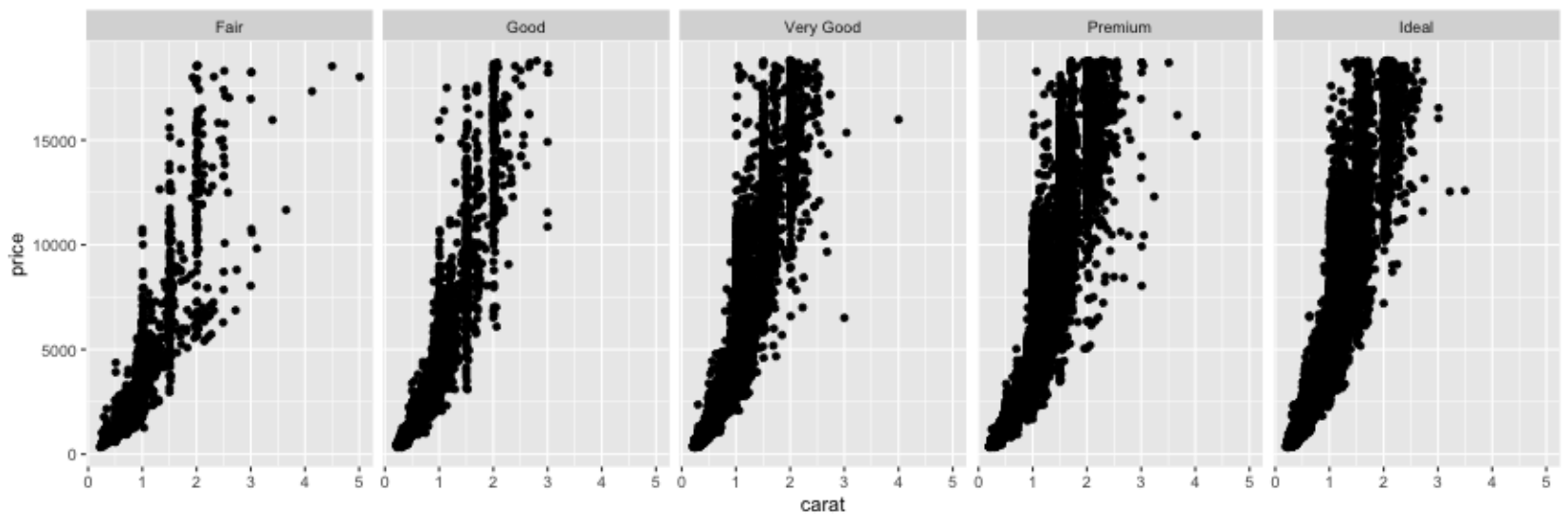
```
diamonds_plot <- ggplot(diamonds, aes(x = carat, y = price)) +
  geom_point()
diamonds_plot + facet_wrap(~ cut,         # variable to facet by
                           ncol = 3)      # number of columns
```



ggplot2 facets

Single variable using `facet_grid()`

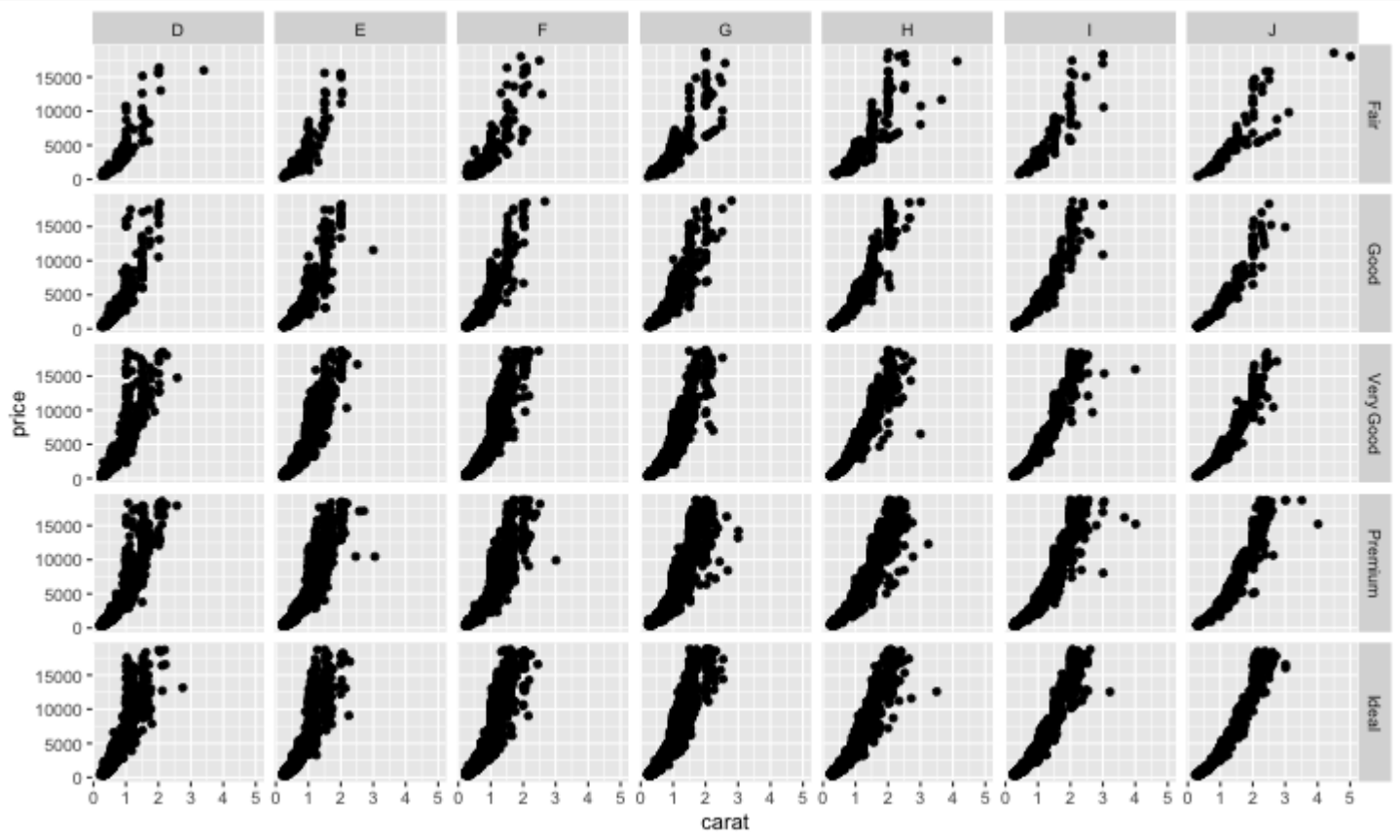
```
diamonds_plot + facet_grid(. ~ cut)      # vertical ~ horizonal
```



ggplot2 facets

Two variables using `facet_grid()`

```
diamonds_plot + facet_grid(cut ~ color)      # vertical ~ horizonal
```



ggplot2 exercises

UNIVERSITY OF OXFORD

1. Import coding_gene_region.bed file into R, add column names, and add a new column containing the length of each region (you should have done this in the base R practical)

2. Plot a histogram of the lengths using ggplot2:
    - Add a plot title
    - Change the x and y axis titles and sizes
    - Change the size and angle of the x tick labels
    - Change the x axis upper limit to 500,000
    - Change the number of bins or the bin width
    - Change the fill and border colour of the bars

ggplot exercises

3. Using the diamonds dataset, plot a scatter plot of diamond length by price, coloured by the diamond colour and sized according to diamond width:
    - Use one of the ggplot built-in themes to alter the plot appearance
    - Change the x axis upper limit to 12 and the intervals to 1.5
    - Add x and y axis titles and change their size
    - Plot the two plots that you have just made side-by-side using a ggplot2 function

Tutorials/extra info

https://ourcodingclub.github.io/tutorials/datavis

http://r-statistics.co/Complete-Ggplot2-Tutorial-Part1-With-R-Code.html

http://zevross.com/blog/2014/08/04/beautiful-plotting-in-r-a-ggplot2-cheatsheet-3/