Data Visualisation

301113 Programming for Data Science

WESTERN SYDNEY
UNIVERSITY

School of Computer, Data and Mathematical Sciences

Week 9 - Wickham and Grolemund, Ch 3 and 28

Outline

1. Grammar of Graphics
2. Geometric Objects
3. Labelling
4. Colouring plots
5. Scale and Legends
6. Coordinate Systems
7. Facets
8. Saving Plots
9. Tips for Good Plots

Motivation

While performing analysis, we need to look at the data to observe interesting features and identify the analysis that needs to be performed.

When presenting our results, we need to show what we found.

A picture is worth 1000 words.

Outline

1 Grammar of Graphics

2 Geometric Objects

3 Labelling

4 Colouring plots

5 Scale and Legends

6 Coordinate Systems

7 Facets

8 Saving Plots

9 Tips for Good Plots

Ggplot2

We have seen how to produce plots using the functions such as `plot`, `hist`, `barplot` and so on.

ggplot2 is an R library that attempts to bring all of the plotting components and styles together. It provides us with a Grammar of Graphics, where we compose the plots by adding layers such as the geometric features, the scales, labels an facets.

Ggplot2

We have seen how to produce plots using the functions such as `plot`, `hist`, `barplot` and so on.

ggplot2 is an R library that attempts to bring all of the plotting components and styles together. It provides us with a Grammar of Graphics, where we compose the plots by adding layers such as the geometric features, the scales, labels an facets.

To begin using ggplot2, it must be installed using the package manager.

```
install.packages(”ggplot2”)
```

Once the package is installed, the files are available on your hard drive.

To use ggplot in your script, it must be loaded into the workspace. Do so by placing the following at the top of the script.

```
library(”ggplot2”)
```

Programming Tips

Never call the function `install.packages` within a script. Packages only need to be installed once, so run the function from the console.

Creating ggplot objects

Rather than calling a single function to produce a plot, ggplot objects are built up using a set of functions whose results add to the plot.

Each plot description is stored in a variable, using this variable, we can:

- add to the plot
- view the plot
- save the plot

The function `ggplot` creates a plot variable.

```r
ggplot(data = mpg)
```

But since there is no description on how to produce the plot, it creates an empty plot. We need to build up the description.

Plot Layers

ggplot provides function to construct the layers:

- Geometric (showing the data)
- Labelling (axis labels)
- Scale (coordinate system)
- Legend (the position and style of the legend)
- Facets (mupltiple plots conditioned on a variable)

We will examine each of these layers to construct a plot.

Outline

1 Grammar of Graphics

2 Geometric Objects

3 Labelling

4 Colouring plots

5 Scale and Legends

6 Coordinate Systems

7 Facets

8 Saving Plots

9 Tips for Good Plots

Geometric Layer

The most important layer of a plot is the geometric layer. The geometric layer provides the type of the plot and the variables to be used in the plot.

The template for providing a plot is

```
ggplot(data = data frame) +
  geom_FUNCTION(mapping = aes(mapping data to plot))
```

Geometric Layer

The most important layer of a plot is the geometric layer. The geometric layer provides the type of the plot and the variables to be used in the plot.

The template for providing a plot is

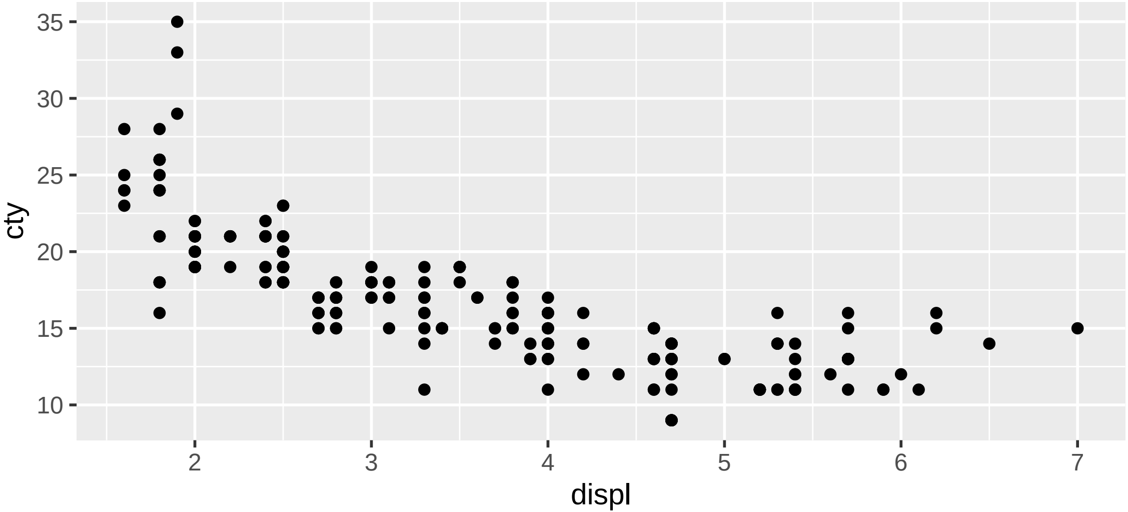
```
ggplot(data = data frame) +
  geom_FUNCTION(mapping = aes(mapping data to plot))
```

Common geometric layers are:

- geom_point (scatter plots)
- geom_histogram (histogram)
- geom_boxplot (box plots)
- geom_bar (bar plots)
- geom_smooth (smoothed line plot)

Scatter Plot

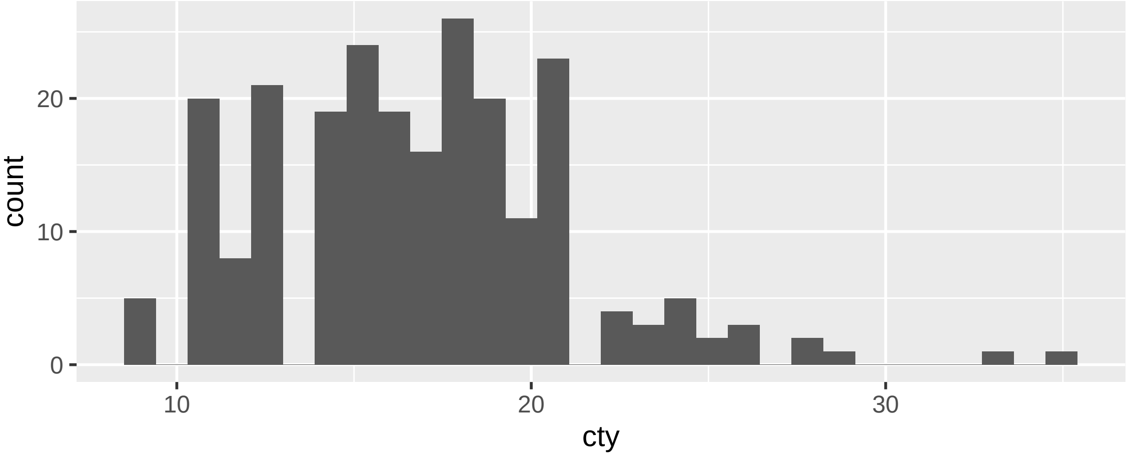
```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = cty))
```



Histogram

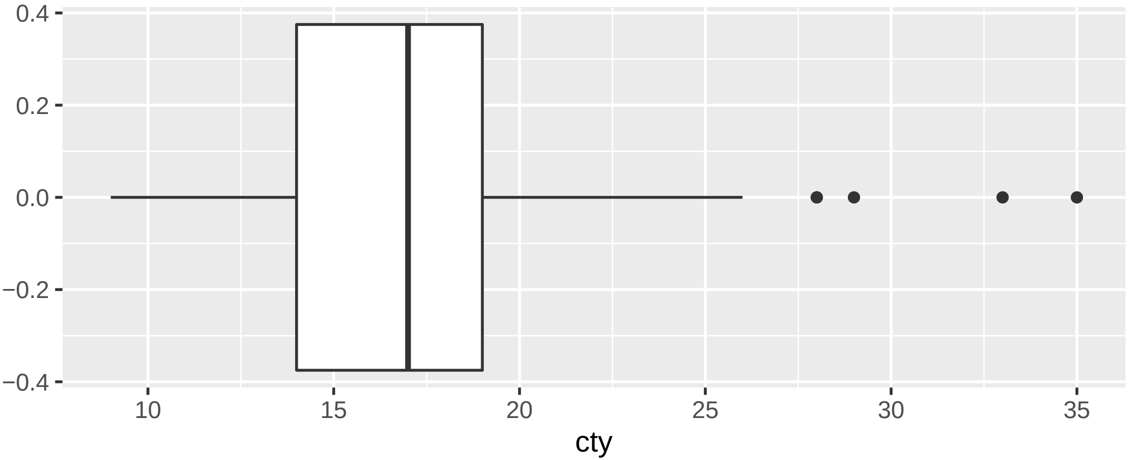
```
ggplot(data = mpg) +
  geom_histogram(mapping = aes(x = cty))
```

```
‘stat_bin()‘ using ‘bins = 30‘. Pick better value with ‘binwidth‘.
```



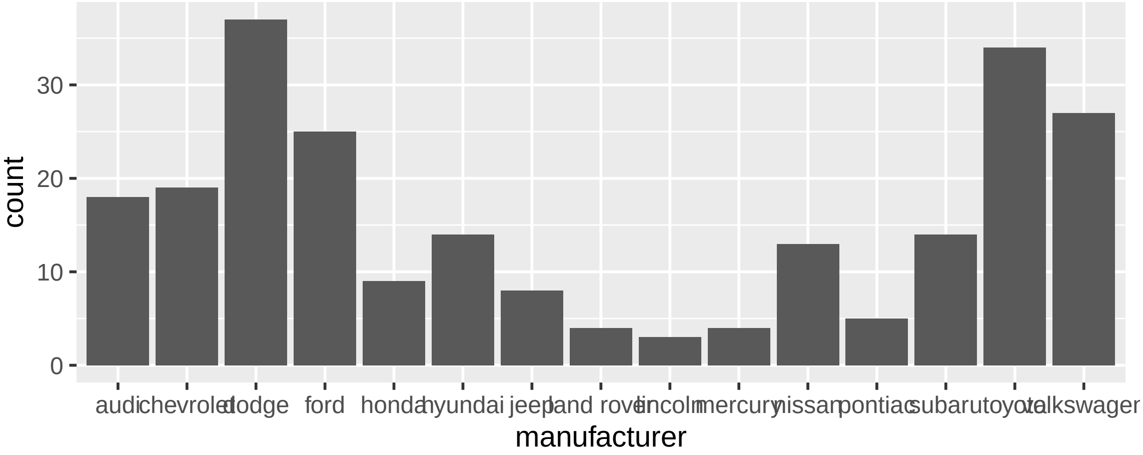
Box plot

```
ggplot(data = mpg) +
  geom_boxplot(mapping = aes(x = cty))
```



Bar plot

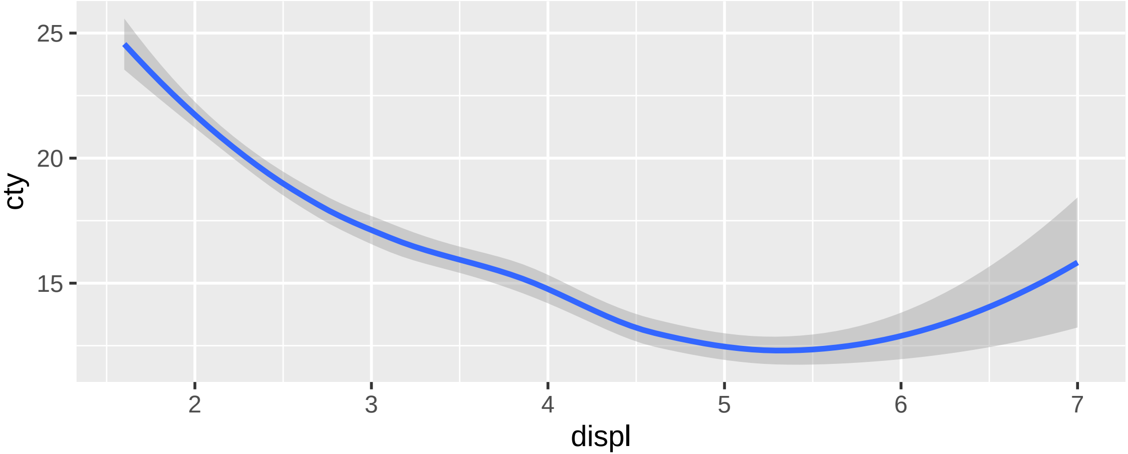
```
ggplot(data = mpg) +
  geom_bar(mapping = aes(x = manufacturer))
```



Smooth plot

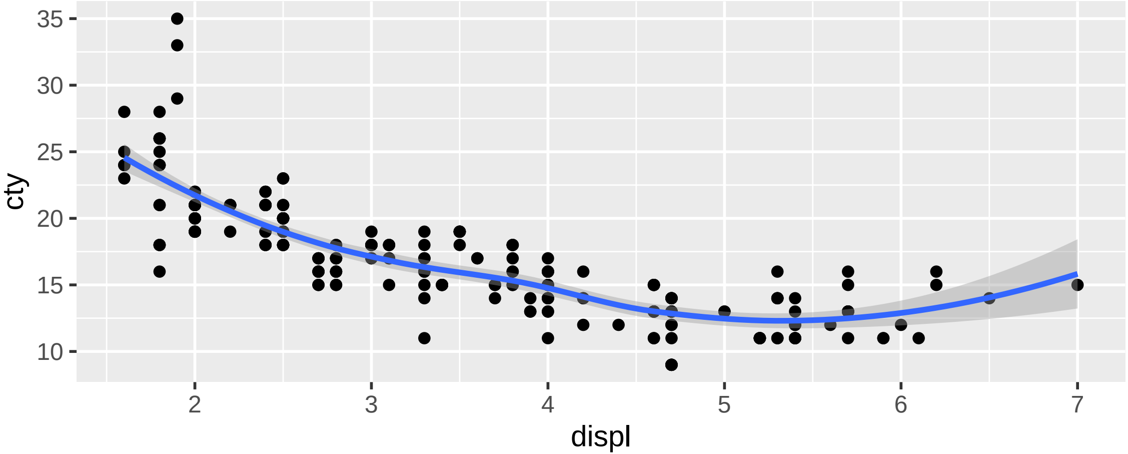
```r
ggplot(data = mpg) +
  geom_smooth(mapping = aes(x = displ, y = cty))
```

```
‘geom_smooth()‘ using method = ’loess’ and formula ’y ~ x’
```



Overlaying geometric layers

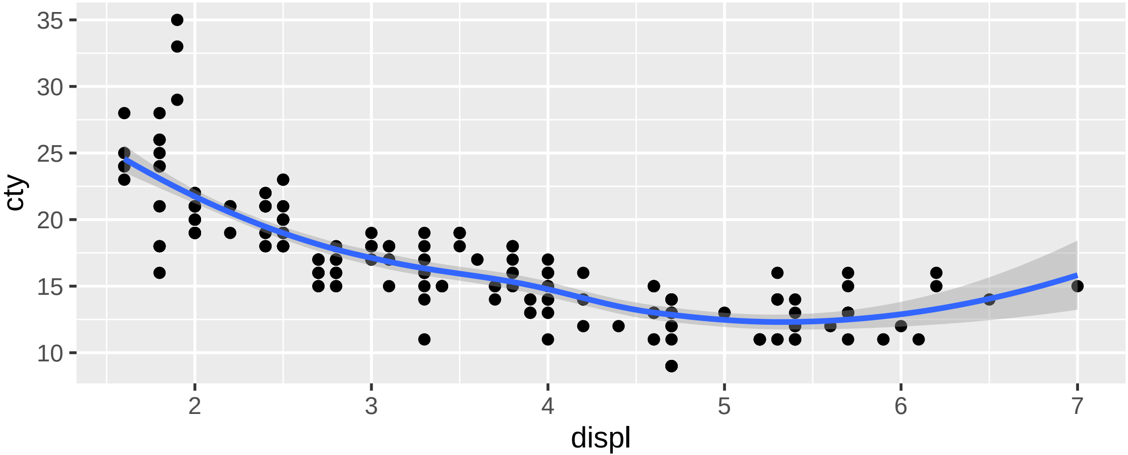
```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = cty)) +
  geom_smooth(mapping = aes(x = displ, y = cty))
```



Global mapping

If both layers use the same mapping, then we can set the mapping to be global.

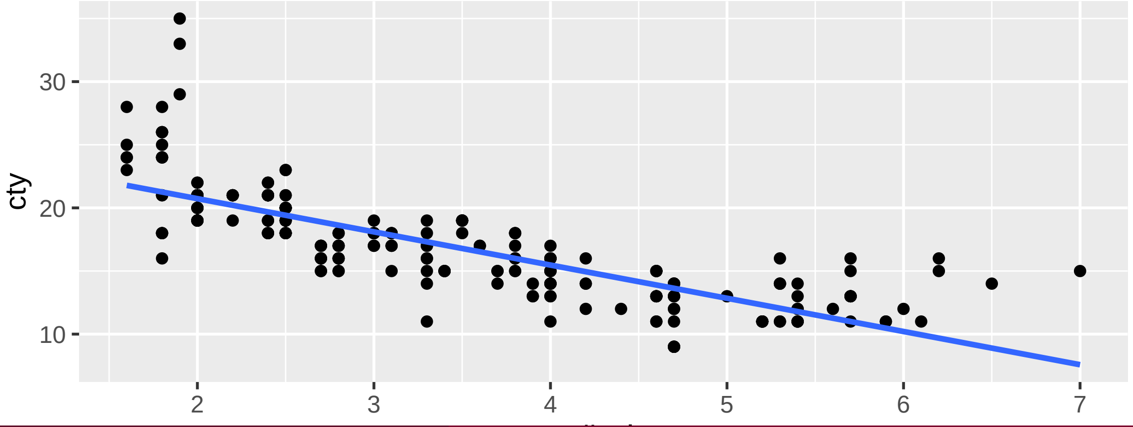
```r
ggplot(data = mpg, mapping = aes(x = displ, y = cty)) +
  geom_point() +
  geom_smooth()
```



Adding a fitted straight line

The default smoothing method is a local smoother. If we want a straight line, the smoothing method must be set to “lm”. Setting `se = FALSE` removes the envelope.

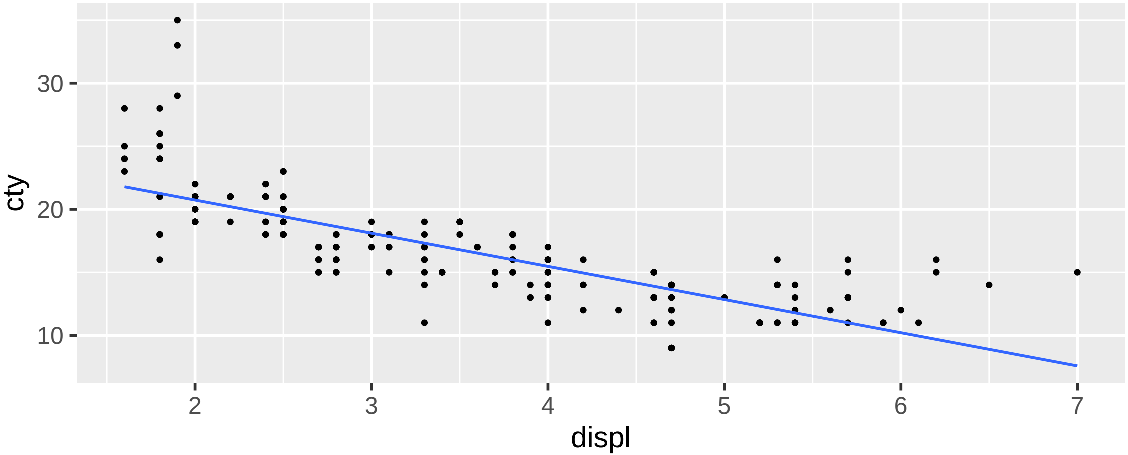
```
ggplot(data = mpg, mapping = aes(x = displ, y = cty)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```



Point size and line Thickness

The thickness of lines is controlled in the geom layer.

```
ggplot(data = mpg, mapping = aes(x = displ, y = cty)) +
  geom_point(size = 0.5) +
  geom_smooth(method = "lm", se = FALSE, size = 0.5)
```



Problem

Distribution of miles per gallon on highways

Use ggplot to visualise the distribution of the miles per gallon used for highway driving from the `mpg` data.

Outline

1 Grammar of Graphics

2 Geometric Objects

3 Labelling

4 Colouring plots

5 Scale and Legends

6 Coordinate Systems

7 Facets

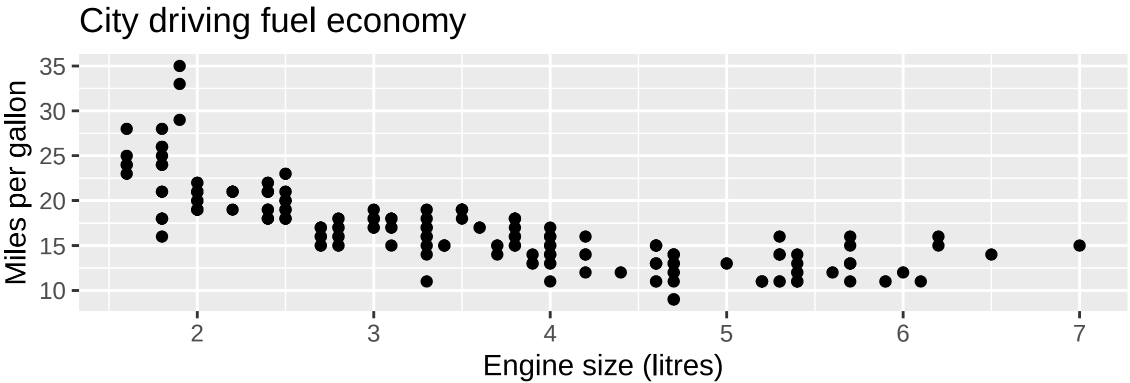
8 Saving Plots

9 Tips for Good Plots

Label layer

The label layer is provided by the function `labs`. The most common labels are `title`, `x` and `y`.

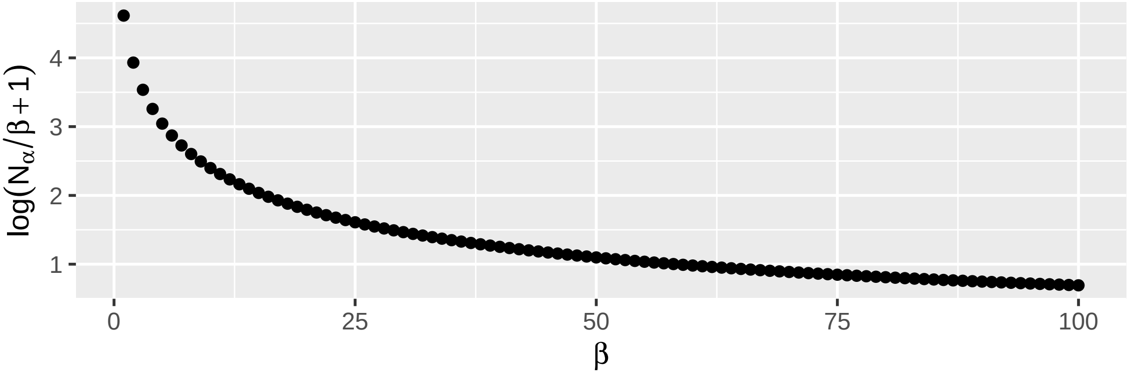
```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = cty)) +
  labs(title = "City driving fuel economy", x = "Engine size (litres)",
       y = "Miles per gallon")
```



Equations in labels

Equations in labels can be added using the function `quote` and the names of the mathematical symbols.

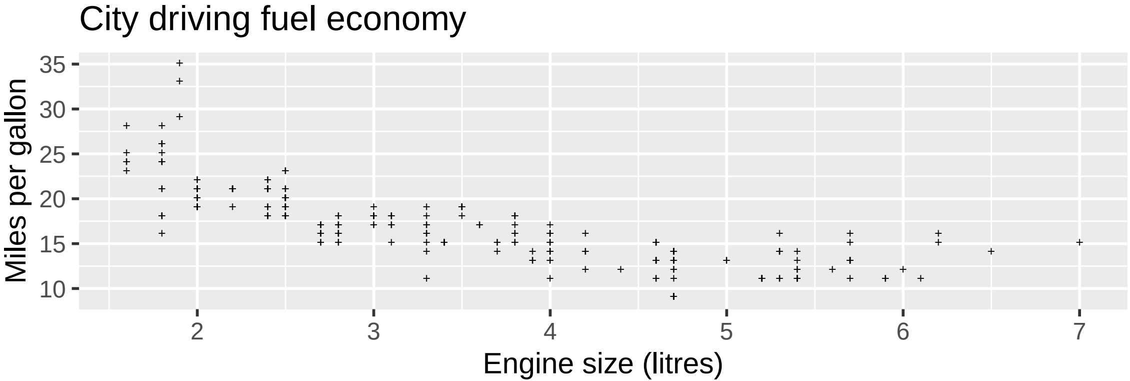
```r
ggplot(data = data.frame(x = 1:100, y = log(100/(1:100) + 1))) +
  geom_point(mapping = aes(x = x, y = y)) +
  labs(x = quote(beta), y = quote(log(N[alpha]/beta + 1)))
```



Point character

The point character by default is a small circle. It can be changed in the geom function using the argument pch.

```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = cty), pch = "+") +
  labs(title = "City driving fuel economy", x = "Engine size (litres)",
      y = "Miles per gallon")
```



Problem

WESTERN SYDNEY
UNIVERSITY

Distribution of miles per gallon on highways

Add descriptive axis labels to the scatter plot we previously produced.

Outline

1 Grammar of Graphics

2 Geometric Objects

3 Labelling

4 Colouring plots

5 Scale and Legends

6 Coordinate Systems

7 Facets

8 Saving Plots

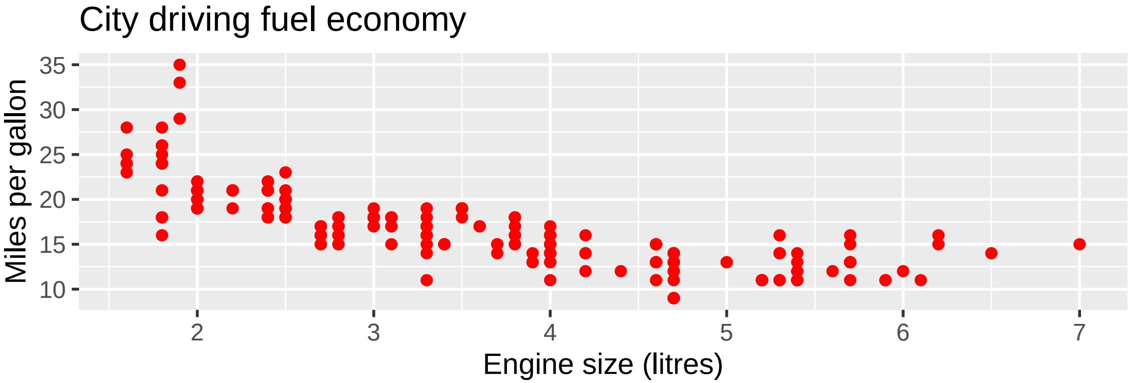
9 Tips for Good Plots

Colouring plots

The point colour can be set using the name of the colour, or a number, where each number represents a predefined colour.

Point colour

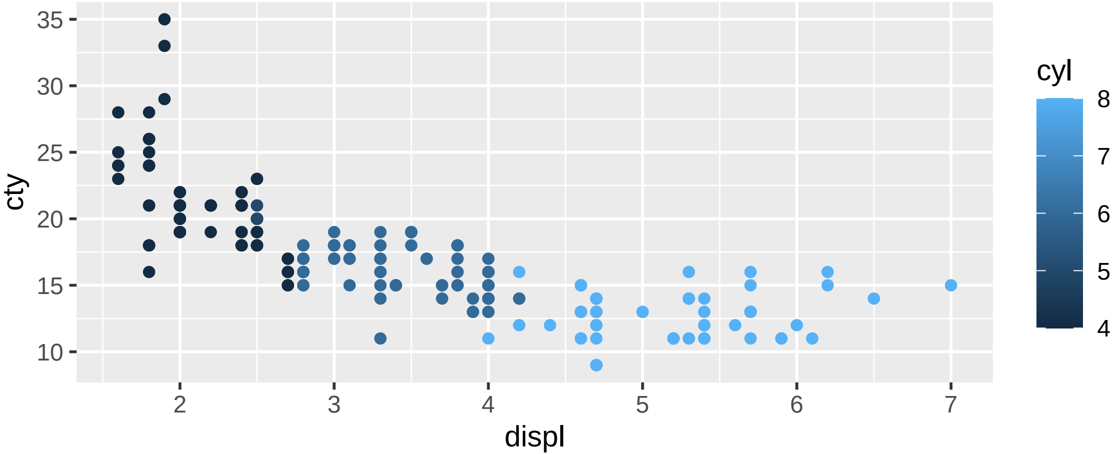
```r
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = cty), colour = ”red”) +
  labs(title = ”City driving fuel economy”, x = ”Engine size (litres)”,
      y = ”Miles per gallon”)
```



Colouring scatter plot continuous groups

If we move the colour into the aesthetics function, the colour can be dependent on a variable value. Here is an example of a continuous variable. A colour scale is provided.

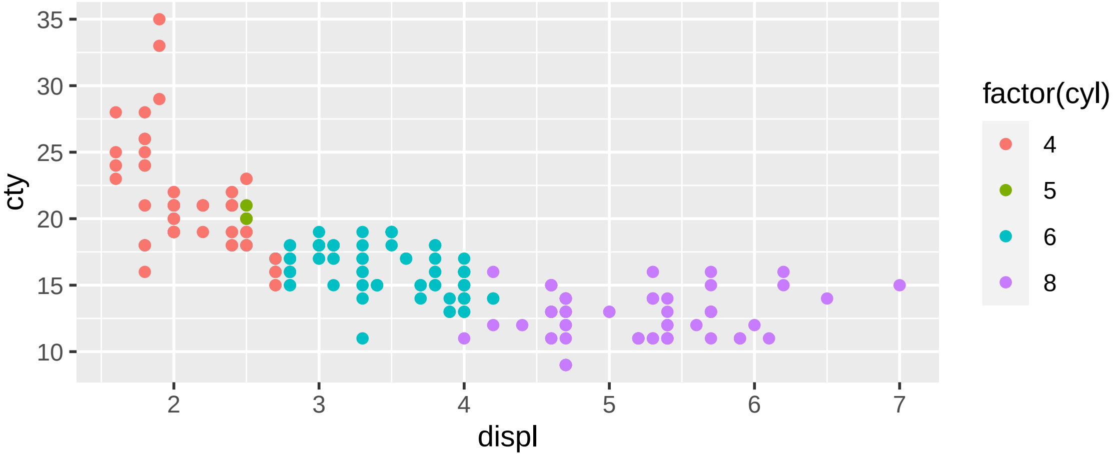
```
ggplot(data = mpg, mapping = aes(x = displ, y = cty)) +
  geom_point(mapping = aes(color = cyl))
```



Colouring scatter plot categorical groups

Here is an example of colouring based on a categorical variable. A legend is provided.

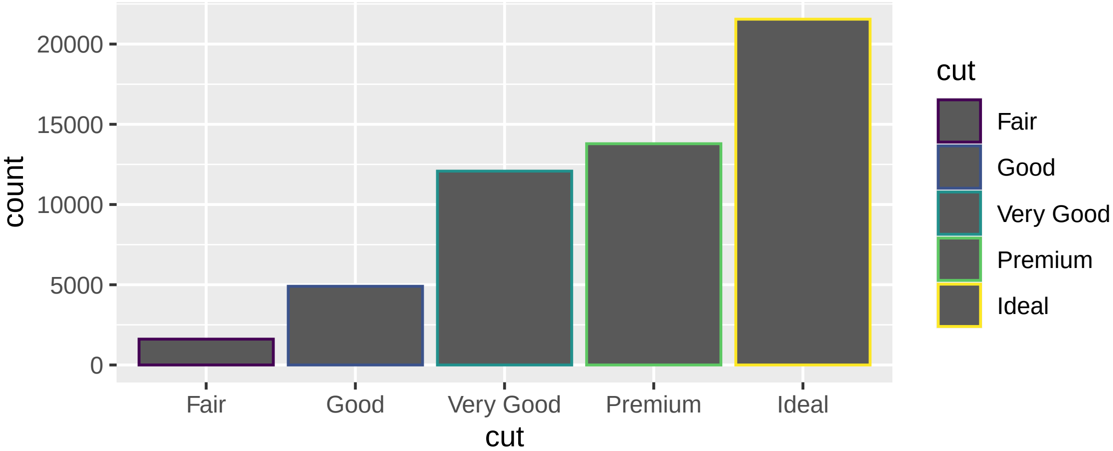
```
ggplot(data = mpg, mapping = aes(x = displ, y = cty)) +
  geom_point(mapping = aes(color = factor(cyl)))
```



Colouring bar plots

The colour argument is for colouring lines and points, so only outlines bar plots.

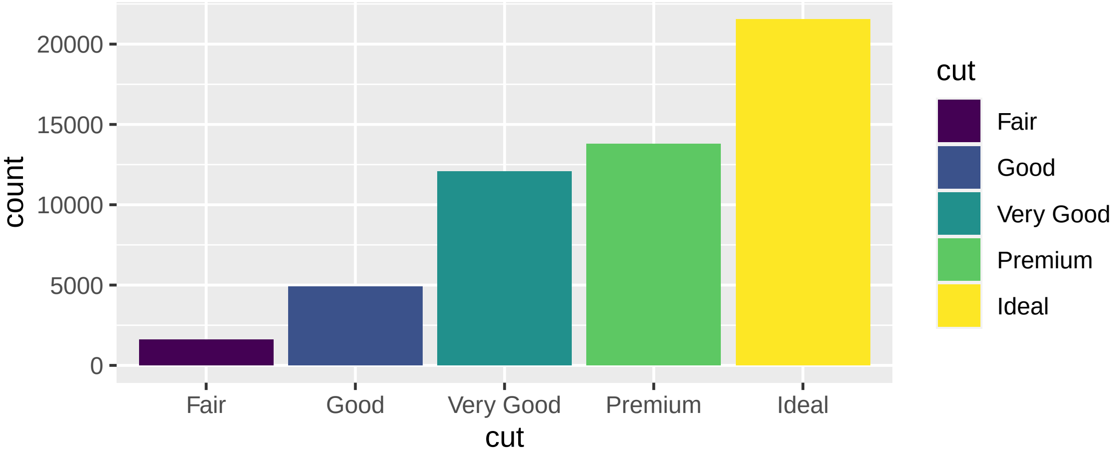
```
ggplot(data = diamonds) +
  geom_bar(mapping = aes(x = cut, colour = cut))
```



Filling bar plots

The argument `fill` can be used to colour blocks.

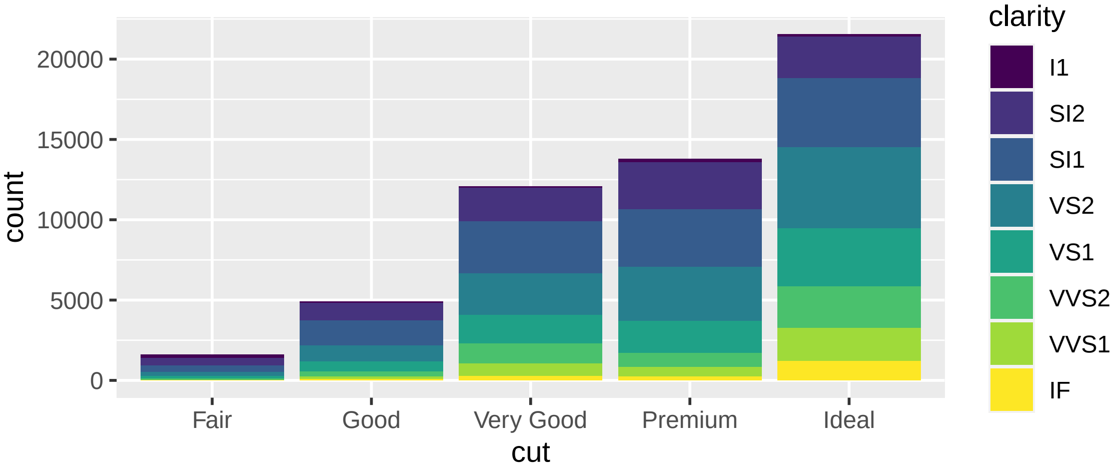
```r
ggplot(data = diamonds) +
  geom_bar(mapping = aes(x = cut, fill = cut))
```



Stacking bar plots using other variables

Stacked.

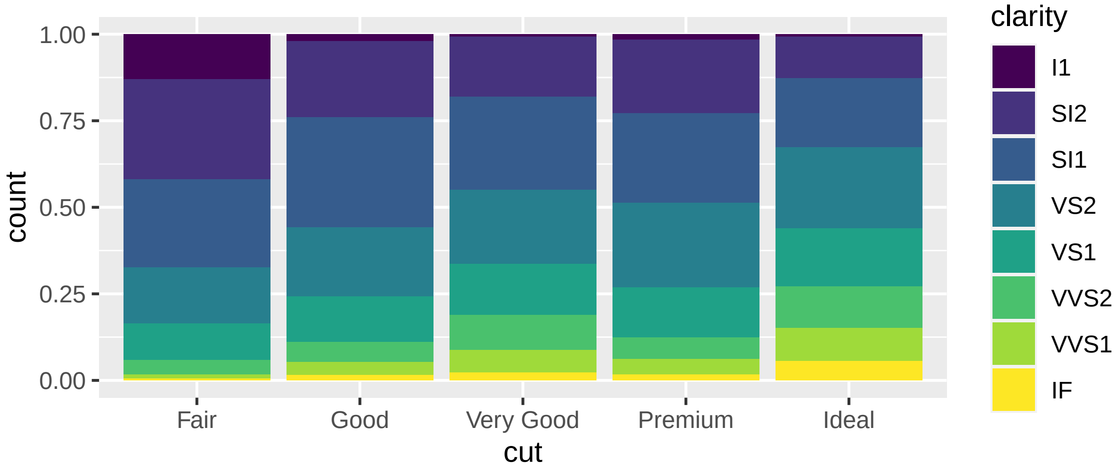
```r
ggplot(data = diamonds) +
  geom_bar(mapping = aes(x = cut, fill = clarity))
```



Filling bar plots using other variables

Filled

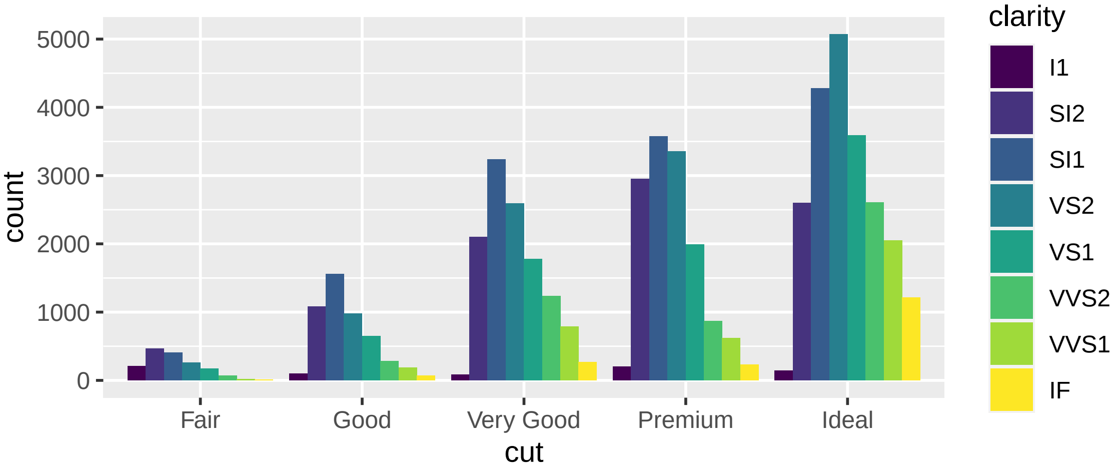
```
ggplot(data = diamonds) +
  geom_bar(mapping = aes(x = cut, fill = clarity), position = ”fill”)
```



Grouping bar plots using other variables

Grouped

```
ggplot(data = diamonds) +
  geom_bar(mapping = aes(x = cut, fill = clarity), position = ”dodge”)
```



Problem

WESTERN SYDNEY UNIVERSITY

Distribution of miles per gallon on highways

Adjust your plot to display the distributions based on their drive train type.

Outline

WESTERN SYDNEY
UNIVERSITY

1 Grammar of Graphics

2 Geometric Objects

3 Labelling

4 Colouring plots

5 **Scale and Legends**

6 Coordinate Systems

7 Facets

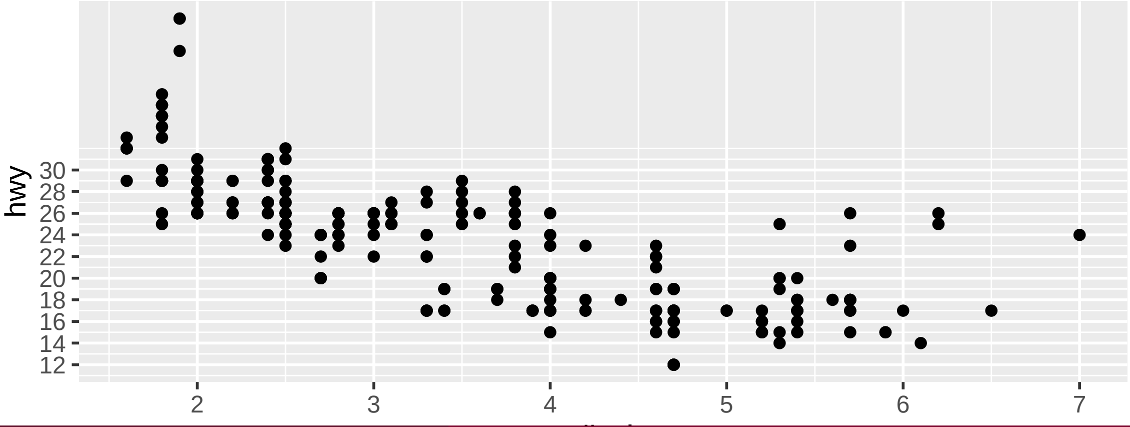
8 Saving Plots

9 Tips for Good Plots

Scale ticks

The functions `scale_x_continuous` and `scale_y_continuous` allow us to set the ticks of the x and y axis.

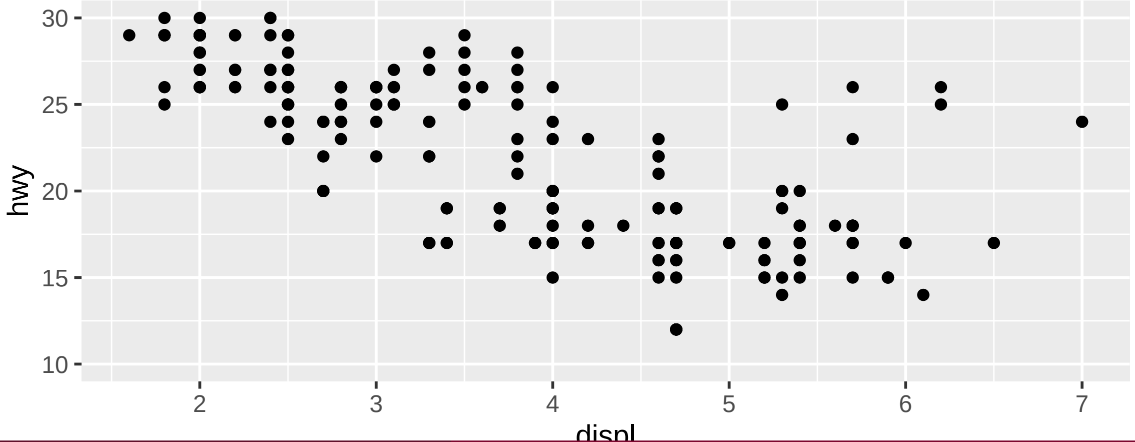
```
ggplot(mpg, aes(displ, hwy)) +
  geom_point() +
  scale_y_continuous(breaks = seq(10, 30, by = 2))
```



Scale limits

```
ggplot(mpg, aes(displ, hwy)) +
  geom_point() +
  scale_y_continuous(limits = c(10, 30))
```

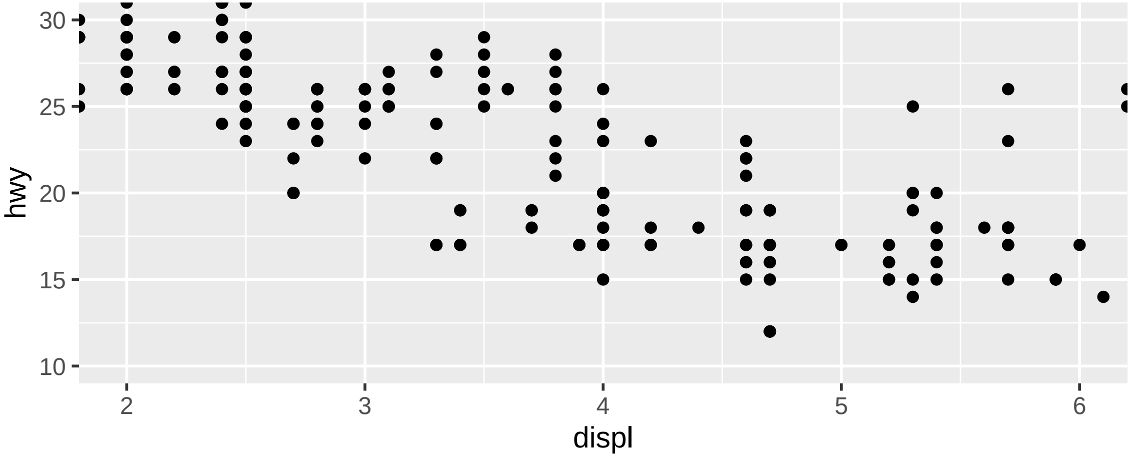
```
Warning: Removed 22 rows containing missing values (geom_point).
```



Scale limits again

The limits of both axes can be set using `coord_cartesian`.

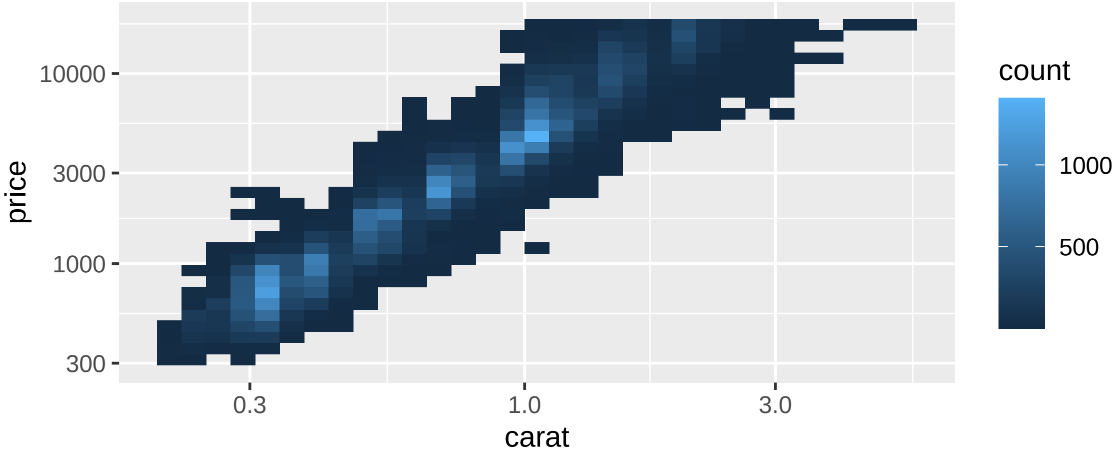
```r
ggplot(mpg, aes(displ, hwy)) +
  geom_point() +
  coord_cartesian(xlim = c(2, 6), ylim = c(10, 30))
```



Log scales

Log scales can be applied to either the x or y axis to visualise exponential scale data. Note that log of the data can be plotted, but the axes will show the logged values.

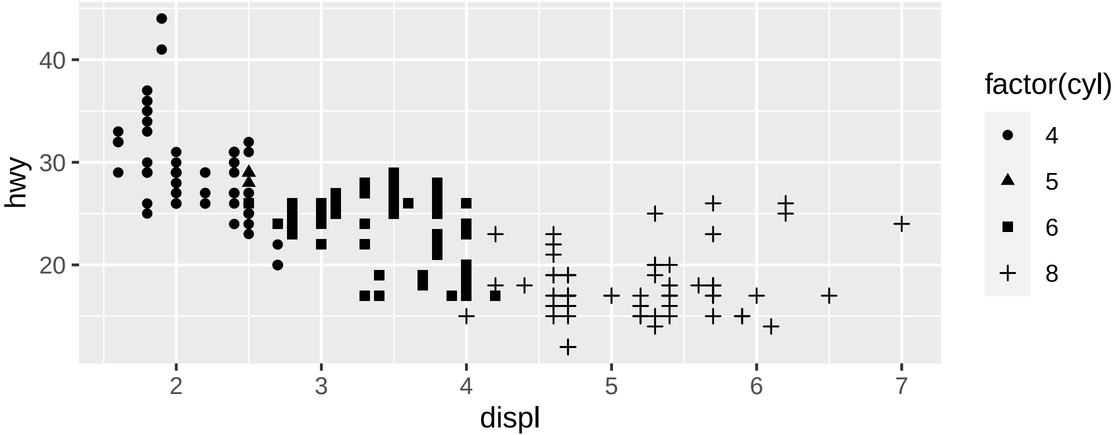
```
ggplot(diamonds, aes(carat, price)) +
  geom_bin2d() + scale_x_log10() + scale_y_log10()
```



Legends

When a reason for a legend is provided, a legend is added.

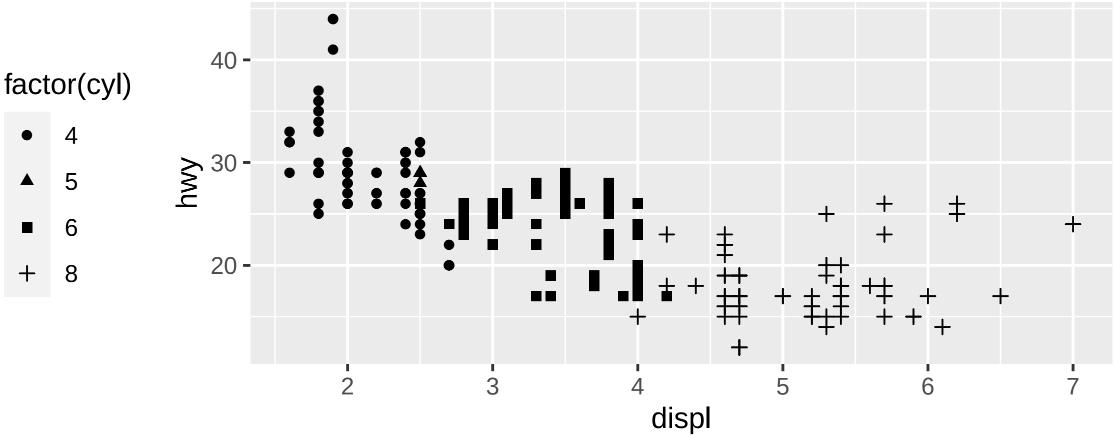
```r
ggplot(mpg, aes(displ, hwy)) +
  geom_point(aes(pch = factor(cyl))) # pch: point character
```



Legend position

The legend can be placed at the top, bottom, left or right side of the plot.

```
ggplot(mpg, aes(displ, hwy)) +
  geom_point(aes(pch = factor(cyl))) + theme(legend.position = "left")
```



Problem

Distribution of miles per gallon on highways

Adjust your plot to place the legend below the plot.

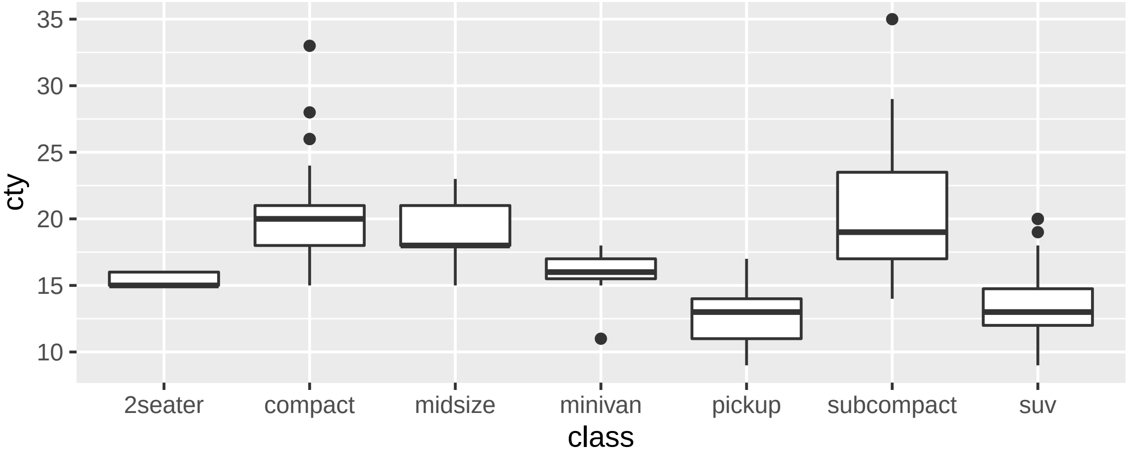
Outline

1. Grammar of Graphics
2. Geometric Objects
3. Labelling
4. Colouring plots
5. Scale and Legends
6. **Coordinate Systems**
7. Facets
8. Saving Plots
9. Tips for Good Plots

Vertical plots

When providing multiple box plots, the x and y assignments determine if the plot is vertical or horizontal. One variable must be numeric and the other a factor.

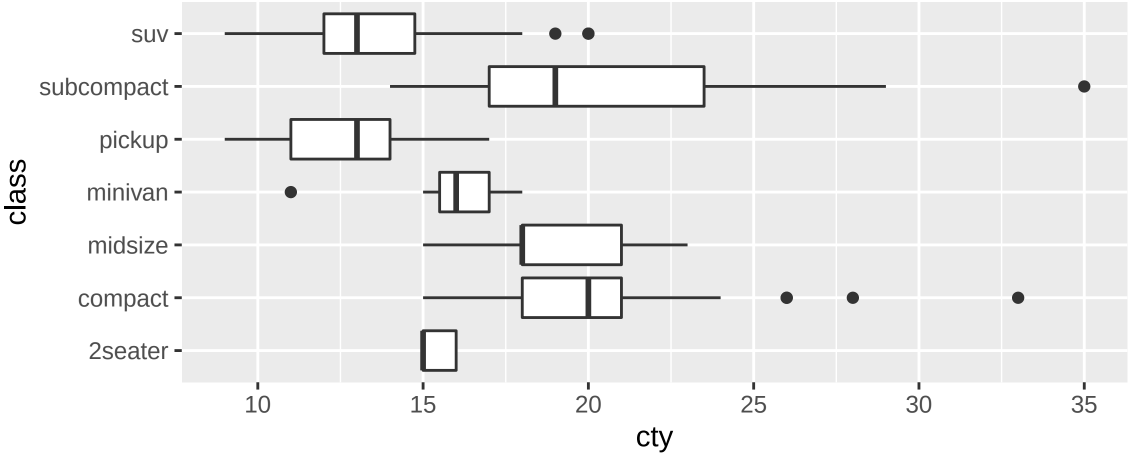
```r
ggplot(data = mpg, mapping = aes(x = class, y = cty)) +
  geom_boxplot()
```



Horizontal plots

In this case, x is numeric and y is a factor.

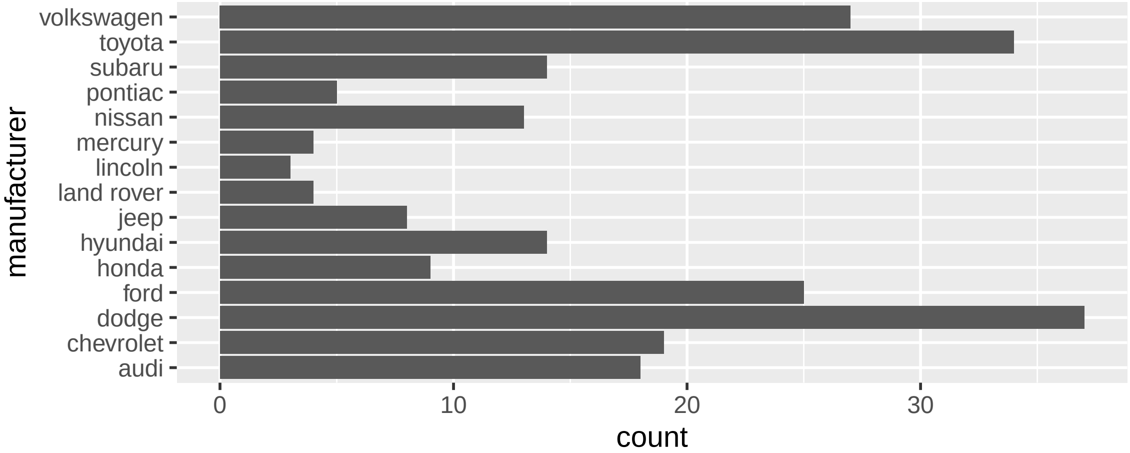
```
ggplot(data = mpg, mapping = aes(y = class, x = cty)) +
  geom_boxplot()
```



Flipping plots

Rather than rewriting the mapping, the coordinate system can be flipped.

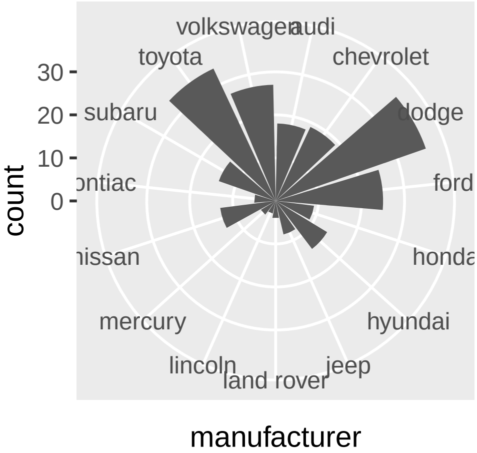
```
ggplot(data = mpg) +
    geom_bar(mapping = aes(x = manufacturer)) + coord_flip()
```



Polar coordinates

Polar coordinate can be used for cyclic variables.

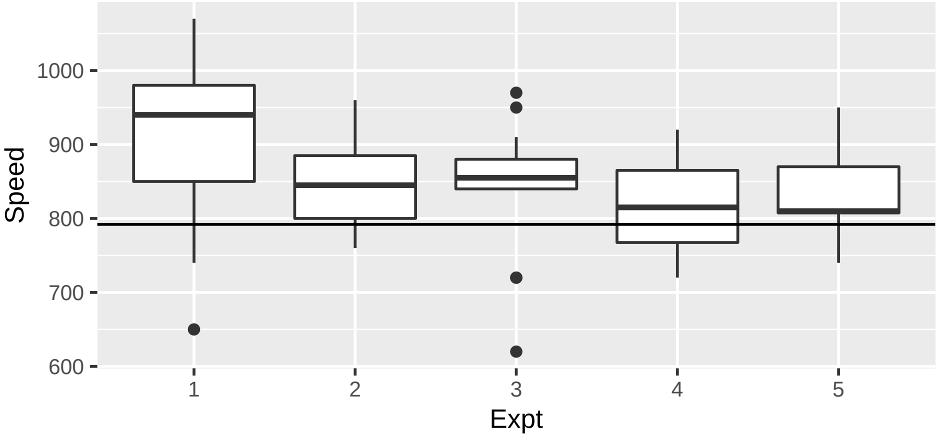
```
ggplot(data = mpg) +
  geom_bar(mapping = aes(x = manufacturer)) + coord_polar()
```



Adding lines

Straight lines can be added using `geom_hline`, `geom_vline`, or `geom_abline`.

```
ggplot(data = morley) +
    geom_boxplot(mapping = aes(y = Speed, x = Expt)) +
    geom_hline(yintercept = 792) + theme(text = element_text(size = 10))
```



Outline

WESTERN SYDNEY
UNIVERSITY

1 Grammar of Graphics

2 Geometric Objects

3 Labelling

4 Colouring plots

5 Scale and Legends

6 Coordinate Systems

7 Facets

8 Saving Plots

9 Tips for Good Plots

Visualising more variables

A facet is a face of a something that has many faces, such as a cut gem.

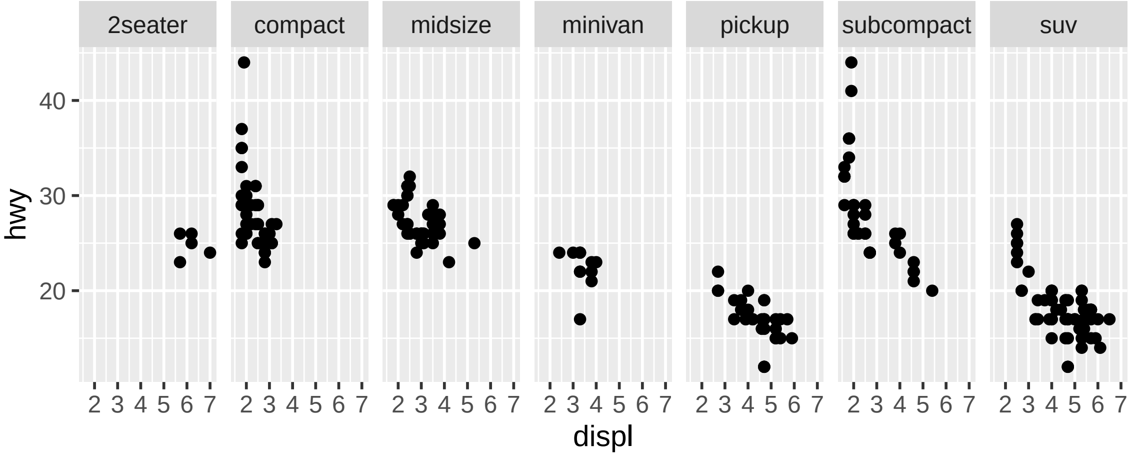
We can easily visualise the relationship between two variables using a scatter plot, but we need to add more dimensions to visualise more variables.

Rather than creating a high dimensional plot (which is hard to do in a 2d document), we can add more plots, where each plot is a category of a categorical variable.

The plots should be placed, so that we can visualise relationships.

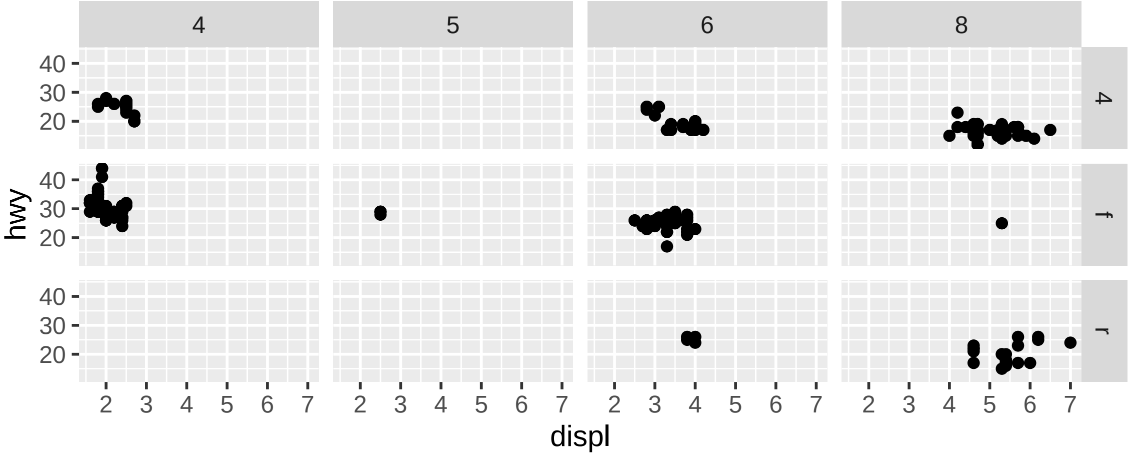
Splitting by one variable

```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = hwy)) +
  facet_wrap(~ class, nrow = 1)
```



Splitting by two variables

```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = hwy)) +
  facet_grid(drv ~ cyl)
```



Problem

Distribution of miles per gallon on highways

Adjust you plot to provide multiple facets, one for each cylinder count.

Outline

1. Grammar of Graphics
2. Geometric Objects
3. Labelling
4. Colouring plots
5. Scale and Legends
6. Coordinate Systems
7. Facets
8. **Saving Plots**
9. Tips for Good Plots

Saving Plots

Some plots need to be included in reports. This can be done by either exporting the plot to a PDF file and including the file in the report, or make the report an R Markdown file and embedding the code in the R Markdown file.

To save the plot to a file, use `ggsave`.

```r
ggsave("plotname.pdf", plot = p, width = 6, height = 4)
```

If the argument `plot` is not provided, the previously plotted plot will be saved. Otherwise, a variable containing a plot needs to be provided (e.g. `p <- ggplot()`).

Plots in R Markdown

Plots generated in R Markdown code chunks appear in the compiled document when the code is evaluated (`eval = TRUE`).

Code chunk options to control plots:

- fig.width: width of the generated image (in inches).
- fig.height: height of the generated image (in inches).
- fig.asp: aspect ratio of the generated figures (height/width).
- out.width: scaled image width in the document.
- out.height: scaled image height in the document.
- fig.align: horizontal alignment of image in the document.

Note that if the aspect ratio is provided, only one of the width of height needs to be provided.

Example code chunk options

Examples of code chunk options:

- No options: defaults are used
- fig.width = 6, fig.asp = 0.618: set image width to 6 inches and aspect ratio to 0.618 (giving a height of 3.708 inches). The default image scaling and alignment are used.
- fig.width = 10, fig.asp = 0.618, , out.width = “70%”, fig.align = “center”: set image width to 10 inches and aspect ratio to 0.618 (giving a height of 6.18 inches). The width of the image in the document is set to 70% of the line width and centre aligned.
- fig.width = 7, fig.height = 5, out.width = “100%”: set image width to 7 inches and the height to 5 inches. The width of the image in the document is set to 100% of the line width (alignment is not needed).

Note that if the width or height are set to large numbers, but the `out` values are small, the large image is scaled down by a large factor, so the font size of the plot in the document will be small.

Outline

1 Grammar of Graphics

2 Geometric Objects

3 Labelling

4 Colouring plots

5 Scale and Legends

6 Coordinate Systems

7 Facets

8 Saving Plots

9 Tips for Good Plots

Plot design

It is tempting to compose complicated plots to show how good we are at R programming, but remember that the purpose of a plot is to convey a message.

When including a plot in a document, ask the questions:

- What are we trying to show using this plot? E.g. a relationship exists between two variables.
- Is this plot showing what we want it to show?
- Are we are trying to show too many things in this plot?

Keep plots simple. Try to provide one plot for each hypothesis we are providing evidence for (if there are many hypotheses, provide a plot for each).

Plot design

It is tempting to compose complicated plots to show how good we are at R programming, but remember that the purpose of a plot is to convey a message.

When including a plot in a document, ask the questions:

- What are we trying to show using this plot? E.g. a relationship exists between two variables.
- Is this plot showing what we want it to show?
- Are we are trying to show too many things in this plot?

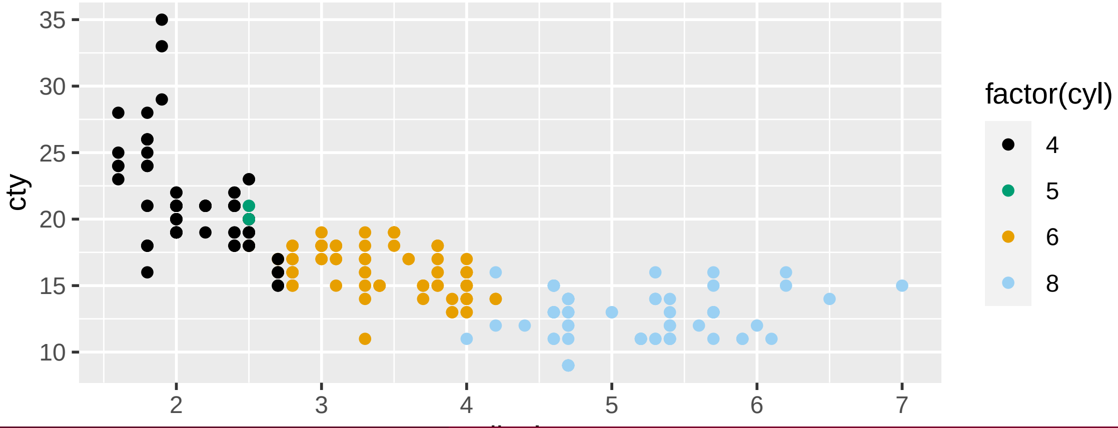
Keep plots simple. Try to provide one plot for each hypothesis we are providing evidence for (if there are many hypotheses, provide a plot for each).

Also consider the medium in which the plots will be presented. Will they be shown on the Web? Printed in black and white? Will the text be readable?

Plot colours

There are many sets of colours that can be used for plots. Sets designed to provide contrast to people with colourblindness are a good choice.

```
ggplot(data = mpg, mapping = aes(x = displ, y = cty)) +
  geom_point(mapping = aes(color = factor(cyl))) +
  scale_colour_manual(values = cbbPalette)
```



Summary

- Plots allow us to visualise data for our own exploration or for presenting results.
- A grammar of graphics allows us to construct plots by combining the layers of the plot.
- Ggplot2 provides us with a set of functions for combining presentation layers.
- Each plot must be presented for a reason, so we must make sure that the message we want to provide is shown in the plot.