Data Visualisation

Chapter 5: Grammar and Vocabulary

Dr James Baglin

How to use these slides

Viewing slides…

- Press ‘f’ enable fullscreen mode
- Press ‘o’ or ‘Esc’ to enable overview mode
- Pressing ‘Esc’ exits all of these modes.
- Hold down ‘alt’ and click on any element to zoom in. ‘Alt’ + click anywhere to zoom back out.
- Use the Search box (top right) to search keywords in presentation

Printing slides…

- Click here to open a printable version of these slides.
- Right click and print from browser or save as PDF (e.g. Chrome)

A Layered Grammar of Graphics

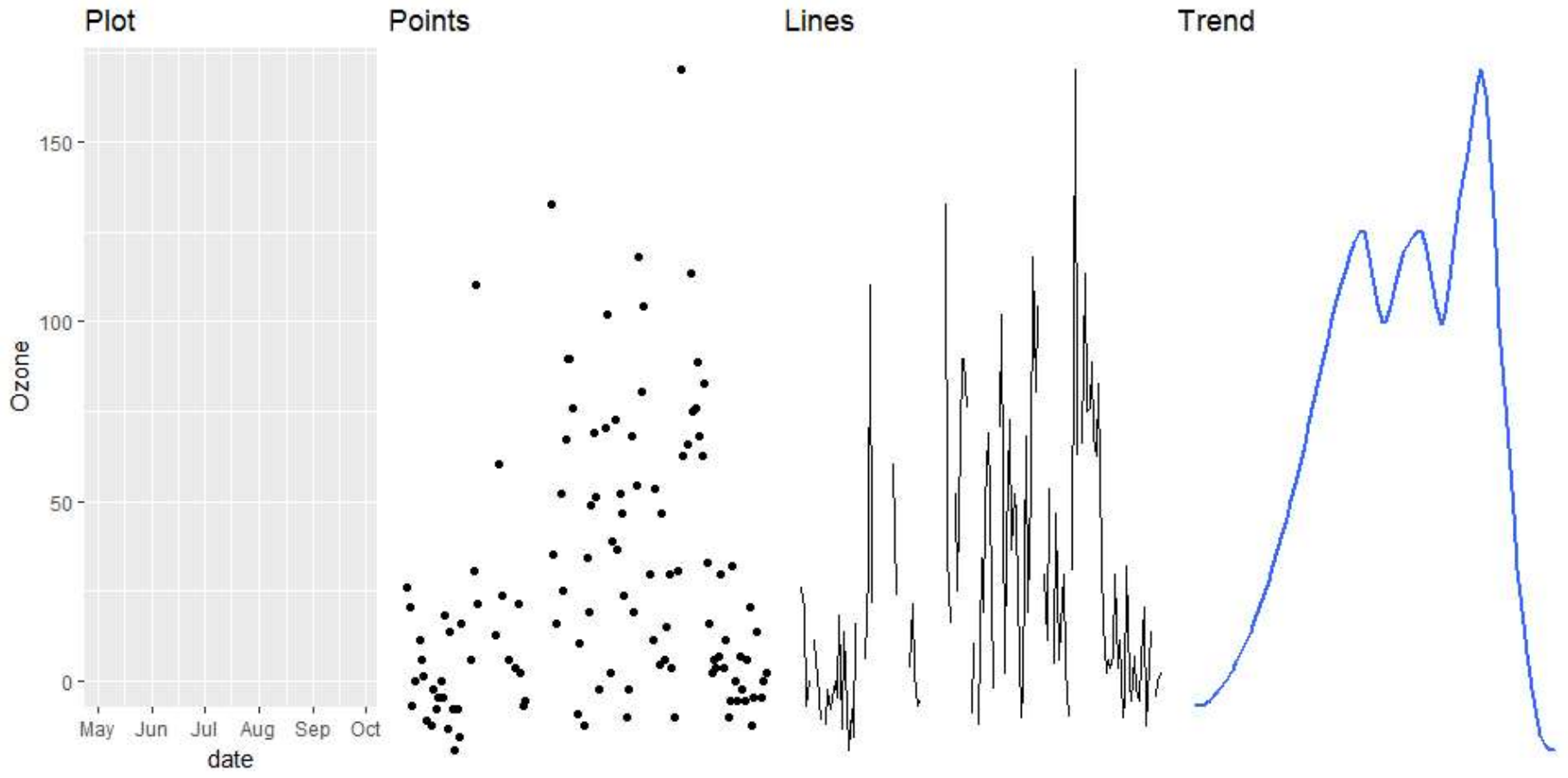
- Wickham (2010) proposed the Layered Grammar of Graphics, which built upon the original Grammar of Graphics first proposed by Wilkinson (2005).
- The idea was to build a grammar that could describe any data visualisation as succinctly as possible.
- This is a big idea because it allows us to move away from a narrow list of methods, into unlimited possibilities

A Layered Grammar of Graphics

- Wickham proposed that a graphic is a series of layers consisting of…
  - a default dataset and set of mappings from variables to aesthetics,
  - one or more layers, with each layer having
    - one geometric object
    - one statistical transformation,
    - one position adjustment,
    - and optionally, one dataset and set of aesthetic mappings
    - one scale for each aesthetic mapping used
  - a coordinate system
  - an optional facet specification.

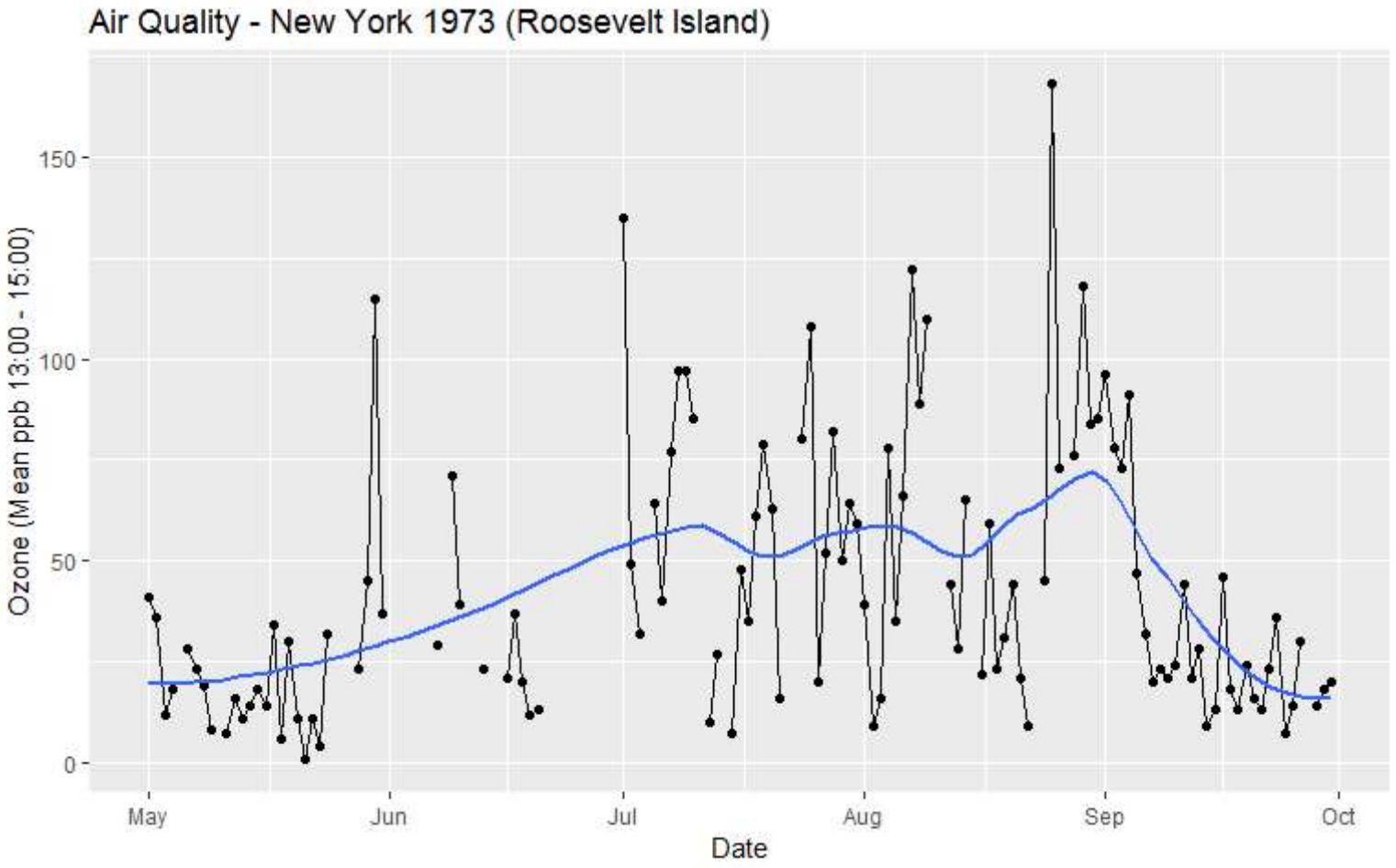
Layers

- Any graphic can be thought of as a series of layers…



- Put them together and we create a graph…

Layers Cont.


Air Quality - New York 1973 (Roosevelt Island)
Ozone (Mean ppb 13:00 - 15:00)
150
100
50
0
May
Jun
Jul
Aug
Sep
Oct
Date

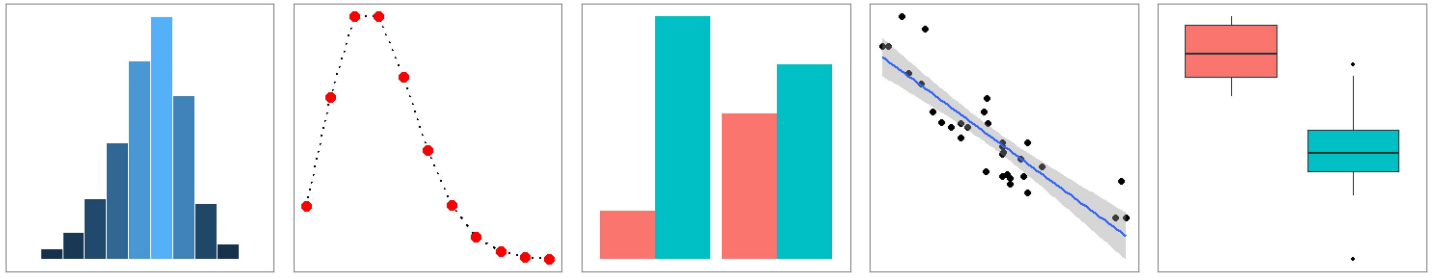
Data

- Layers are composed of data, aesthetic mappings, statistical transformations, geometric objects and optional position adjustments.
- Data are obvious…

```
##     Ozone Solar.R Wind Temp Month Day       date
## 1      41     190  7.4   67     5   1 1973-05-01
## 2      36     118  8.0   72     5   2 1973-05-02
## 3      12     149 12.6   74     5   3 1973-05-03
## 4      18     313 11.5   62     5   4 1973-05-04
## 5      NA      NA 14.3   56     5   5 1973-05-05
## 6      28      NA 14.9   66     5   6 1973-05-06
## 7      23     299  8.6   65     5   7 1973-05-07
## 8      19      99 13.8   59     5   8 1973-05-08
## 9       8      19 20.1   61     5   9 1973-05-09
## 10     NA     194  8.6   69     5  10 1973-05-10
## 11      7      NA  6.9   74     5  11 1973-05-11
## 12     16     256  9.7   69     5  12 1973-05-12
## 13     11     290  9.2   66     5  13 1973-05-13
## 14     14     274 10.9   68     5  14 1973-05-14
## 15     18      65 13.2   58     5  15 1973-05-15
## 16     14     334 11.5   64     5  16 1973-05-16
## 17     34     307 12.0   66     5  17 1973-05-17
```

Geometric Objects

- **Geometric objects** are use to represent data or statistical transformations of the data.
- We are already familiar with many common geometric objects.
  - Boxes used in boxplots
  - Bins used in histograms
  - Bars used in barchart
  - Points in a scatter plot
  - Lines in a line chart



Mapping Aesthetics

- Properties of geometric objects, such as points, lines, colours, and shapes, are referred to as **aesthetics**
- The process of assigning variables from a dataset to aesthetics is known as **mapping**.
- For example in the air quality example x = Date and y = Ozone.
- Points are geometric objects and the position they are drawn on the plot is determined by the mappings.

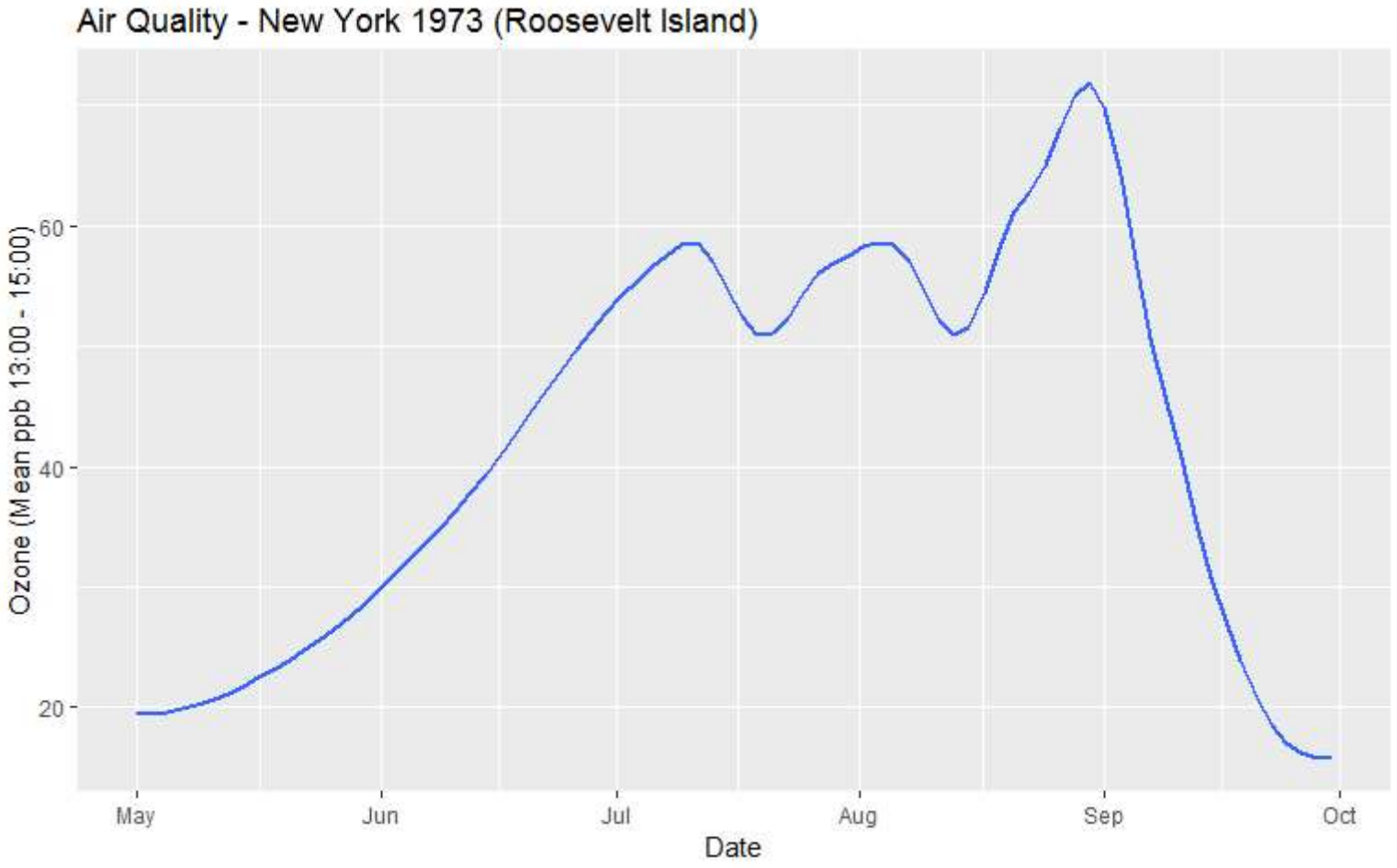
geoms vs aes

Statistical Transformations

- Many visualisations use statistical summaries of the raw data.
- Examples of `stats` transformations include the following:
  - quartiles of box plots
  - means
  - error bar/confidence intervals
  - binning in histograms and dot plots
  - tallies, counts, proportions, percentages in bar charts
  - lines of best fit for linear regression.
- Statistical transformations are the reason why statistics is so important for data visualisation.

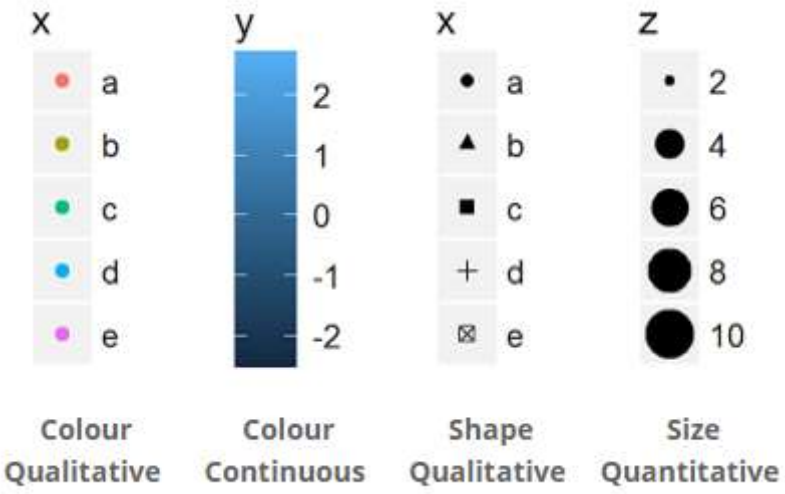
Statistical Transformations Cont.

- The smoothed trend line in the Air Quality visualisation was estimated using a non-parametric, locally weighted regression model



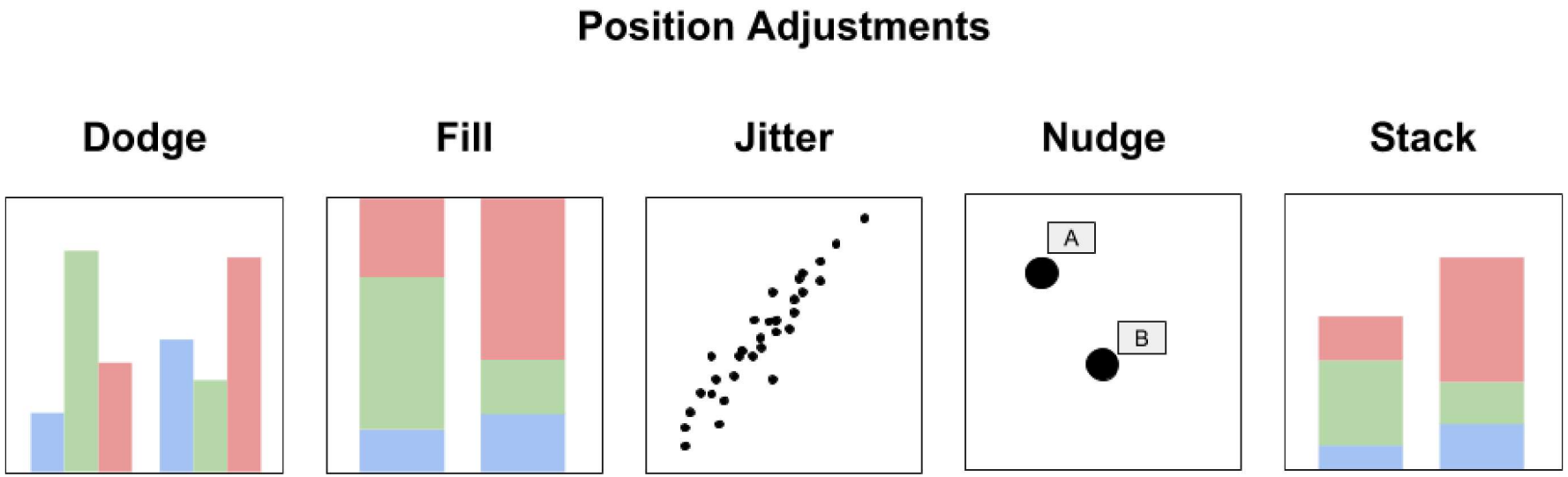
Scales

- Scales are used to control the mapping between a variable and an aesthetic.
- For example, changing the range of the x axis, or changing the colour of a colour scale.



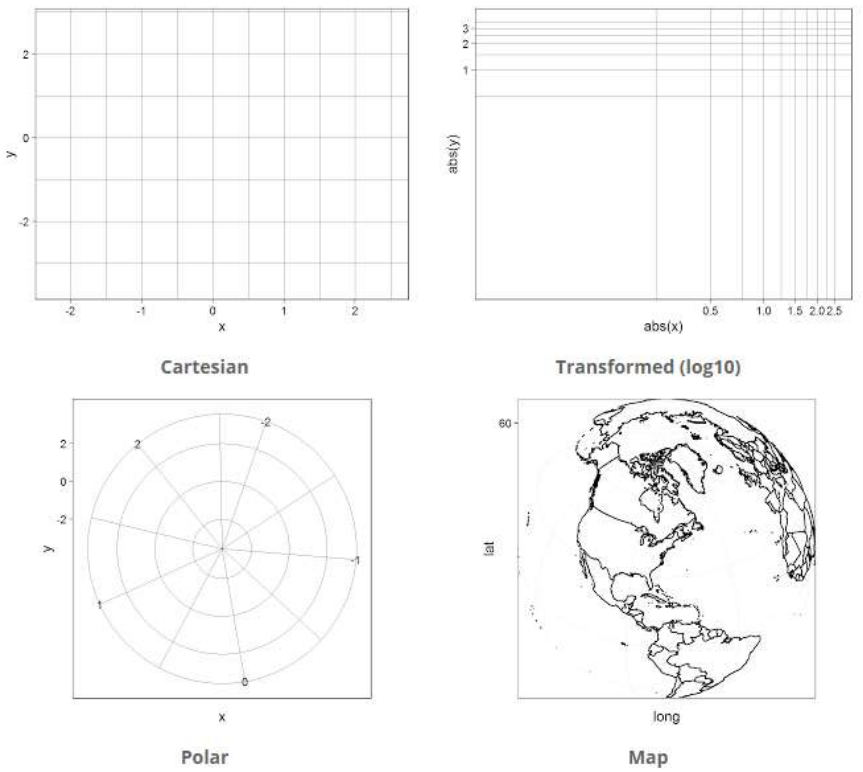
Position Adjustment

- Position adjustments aim to avoid overlapping elements by either dodging, filling, jittering, nudging or stacking. Layers can incorporate multiple position adjustments.



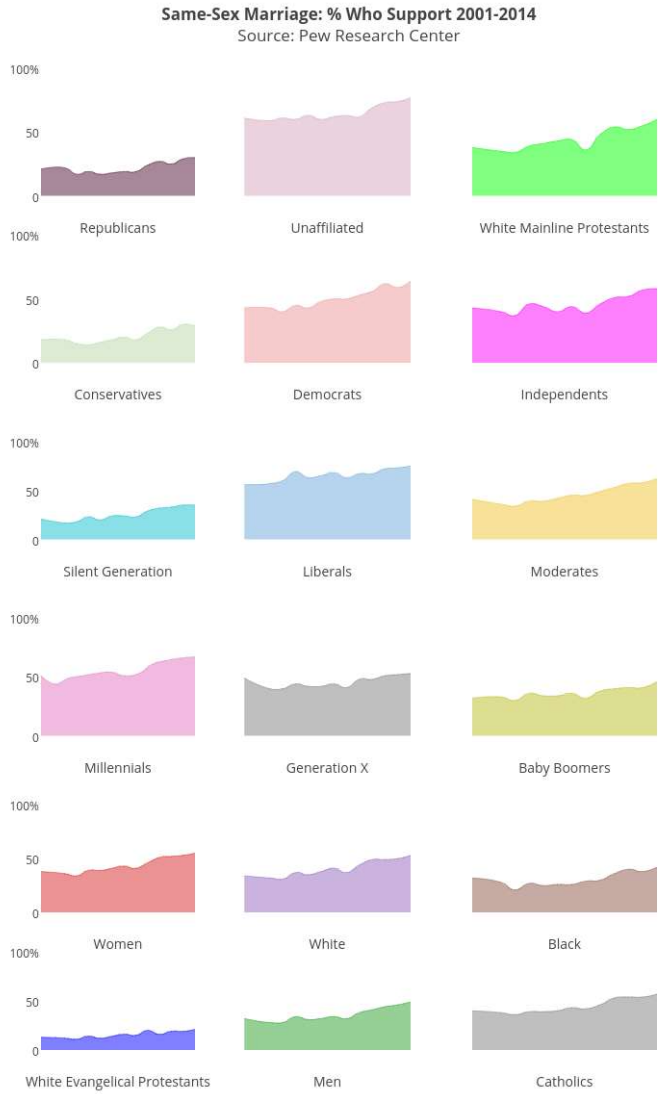
Coordinate System

- **Coordinate systems** are used to determine the placement of geometric objects within a plot:
  - Cartesian
  - Transformed
  - Polar
  - Map



Faceting

- **Faceting** is a powerful way to break a data visualisation into small multiples. This process is also known as **latticing** or **trellising**.



`ggplot2`

- `ggplot2` is a high-level R package, developed by Hadley Wickham based on Layered Grammar of Graphics
- Why is it import to learn ggplot2?
  - Develop a basic but powerful grammar for building visualisations
  - Learn how to build visualisations layer by layer
  - Helps promote good practice in data visualisation - sensible defaults and colour use
  - Powerful and fully customisable

gglot2 - A Verbose Example

- Let's take a look at how `ggplot2` builds a visualisation.

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

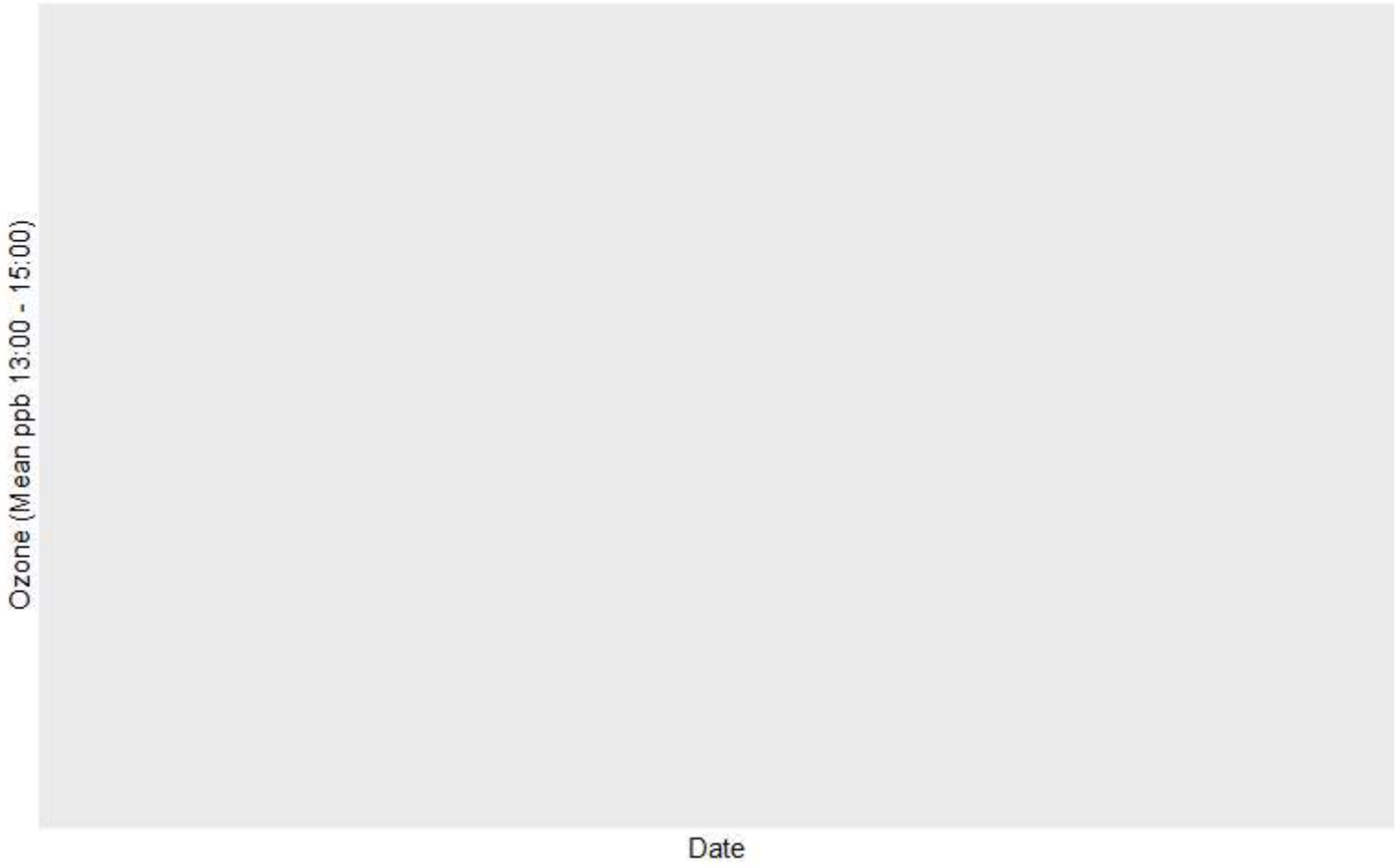
- We begin by creating a `ggplot2` object, defining a coordinate system and selecting our scales.

```
ggplot() +
  coord_cartesian() +
  scale_x_date(name = "Date") +
  scale_y_continuous(name = "Ozone (Mean ppb 13:00 - 15:00)")
```

- And the result...

gglot2 - A Verbose Example Cont.

- Nothing because we have not defined any layers.



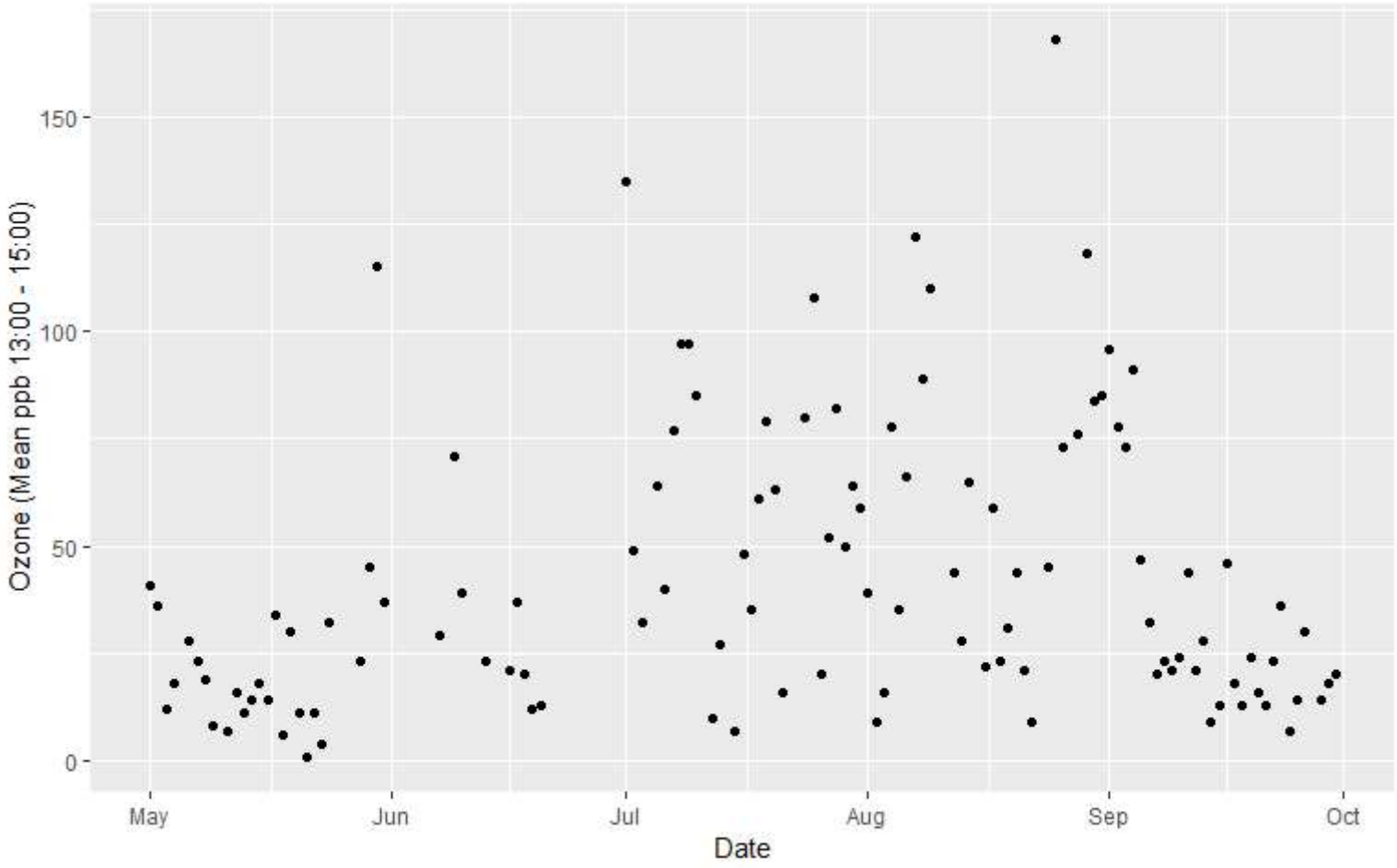
ggplot2 - A Verbose Example Cont. 2

- Let's add a layer for points...

```
ggplot() +
  coord_cartesian() +
  scale_x_date(name = "Date") +
  scale_y_continuous(name = "Ozone (Mean ppb 13:00 - 15:00)") +
  layer(
    data = airquality,
    mapping = aes(x = date, y = Ozone),
    stat = "identity",
    geom = "point",
    position = position_identity()
  )
```

gglot2 - A Verbose Example Cont. 3

- Now we are getting somewhere…

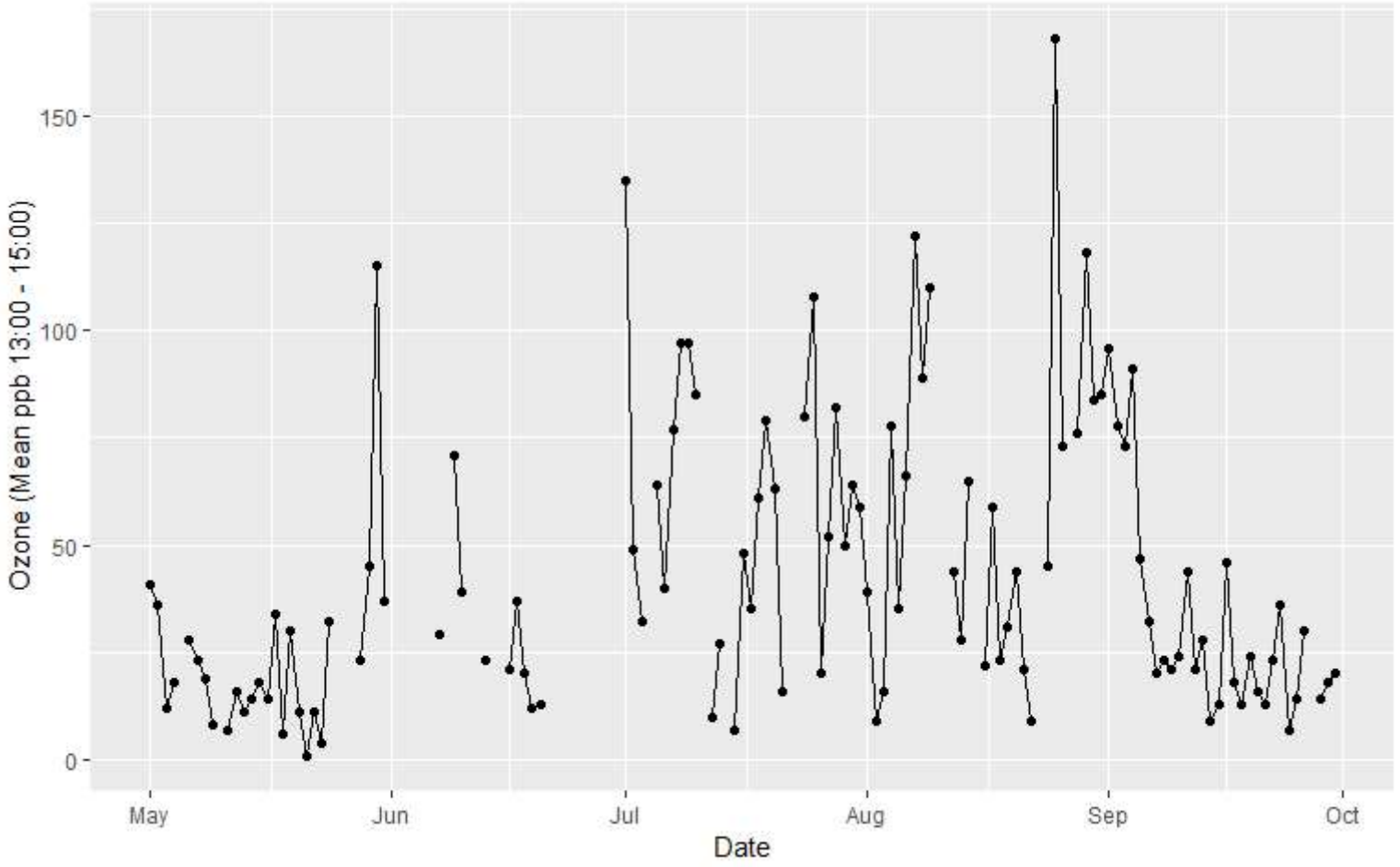


ggplot2 - A Verbose Example Cont. 4

- Now for the lines...

```
ggplot() +
  coord_cartesian() +
  scale_x_date(name = "Date") +
  scale_y_continuous(name = "Ozone (Mean ppb 13:00 - 15:00)") +
  layer(
    data = airquality,
    mapping = aes(x = date, y = Ozone),
    stat = "identity",
    geom = "point",
    position = position_identity()
  ) +
  layer(
    data = airquality,
    mapping = aes(x = date, y = Ozone),
    stat = "identity",
    geom = "line",
    position = position_identity()
  )
```

gglot2 - A Verbose Example Cont. 5

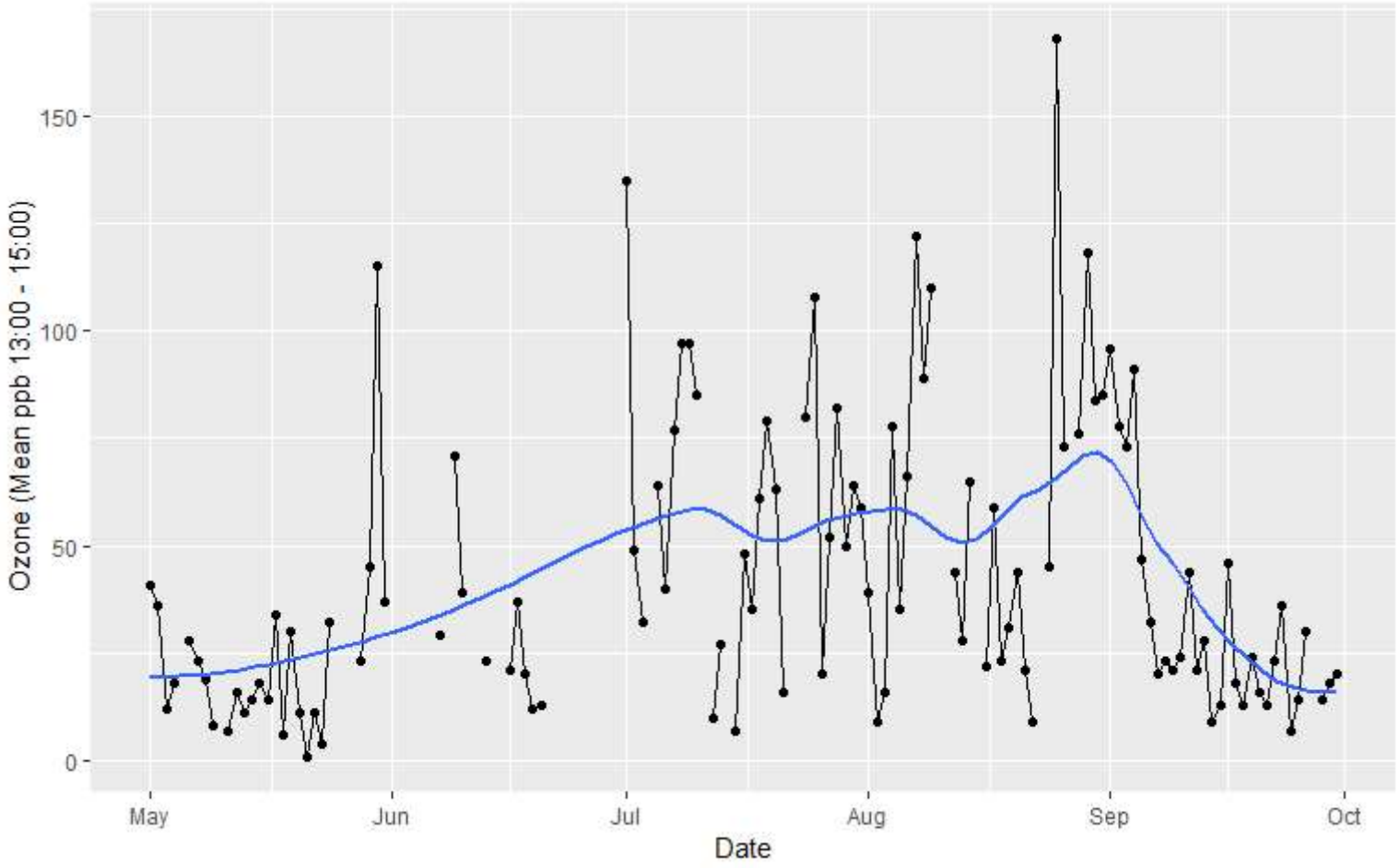


gglot2 - A Verbose Example Cont. 6

- And the trend line...

```
ggplot() +
  coord_cartesian() +
  scale_x_date(name = "Date") +
  scale_y_continuous(name = "Ozone (Mean ppb 13:00 - 15:00)") +
  layer(
    data = airquality,
    mapping = aes(x = date, y = Ozone),
    stat = "identity",
    geom = "point",
    position = position_identity()
  ) +
  layer(
    data = airquality,
    mapping = aes(x = date, y = Ozone),
    stat = "identity",
    geom = "line",
    position = position_identity()
  ) +
```

gglot2 - A Verbose Example Cont. 7



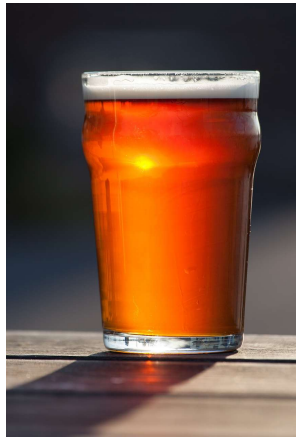
gglot2 in Practice

- That was a lot of code!
- In practice, `ggplot2` is far more succinct.
- The following code will reproduce the same visualisation

```
p <- ggplot(data = airquality, aes(x = date, y = Ozone))
p + geom_point() +
  geom_line(aes(group = 1)) +
  geom_smooth(se = FALSE, span = 0.4) +
  labs(
    title = "Air Quality - New York 1973 (Roosevelt Island)",
    x = "Date",
    y = "Ozone (Mean ppb 13:00 - 15:00)"
    )
```

- Let's take a closer look at more efficient ways to code `ggplot2`.

Demo Data - Student Alcohol Survey



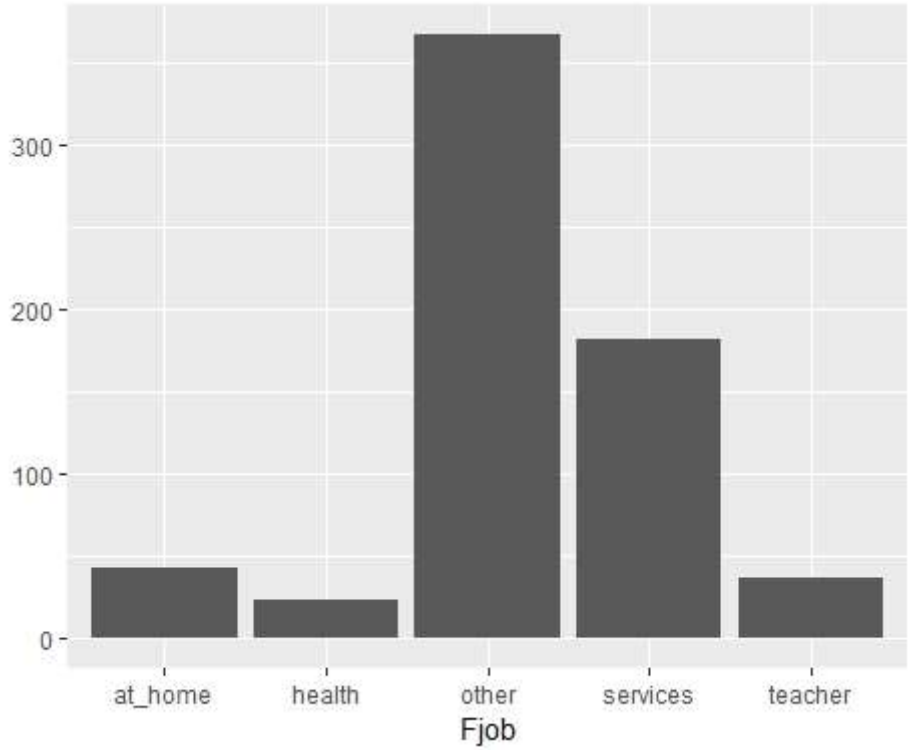
- A survey of 649 Portuguese students aged 15 to 22.
- Questions related to alcohol consumption, demographics, family background, academic and social factors.
- A clean copy of the data can be downloaded here.
- The original data was downloaded from the UCI Machine Learning Repository

```
student <- read.csv("../data/Student_alcohol_survey.csv")
```

qplot()

- The `qplot()` function in `ggplot2` is a quick method to develop basic data visualisations with sensible defaults.

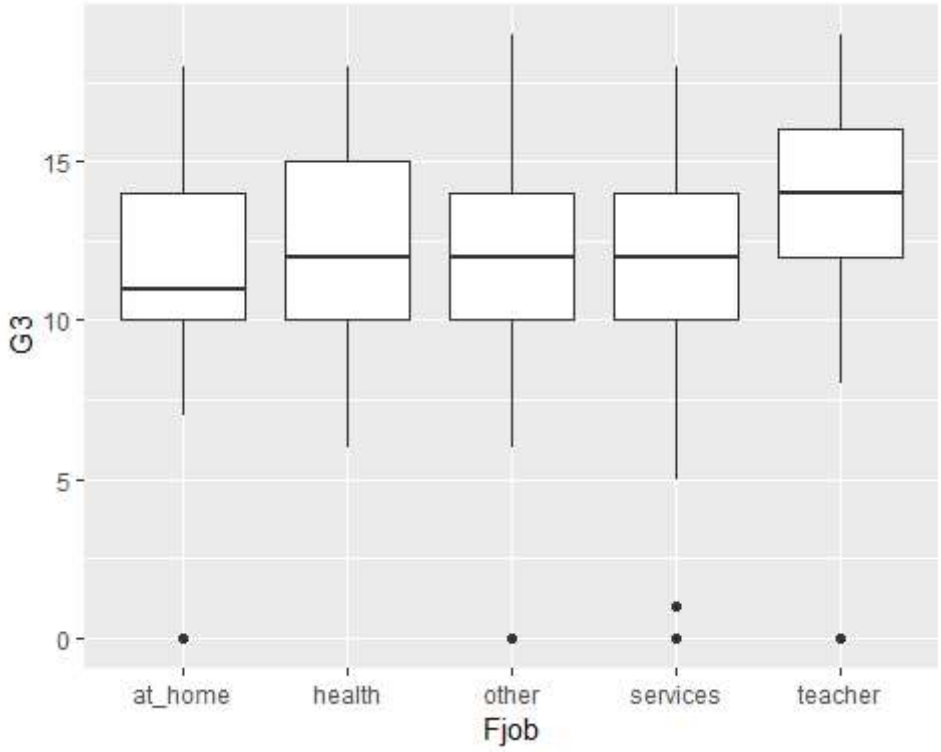
```
qplot(x = Fjob, data = student, geom = "bar")
```



`qplot()` cont.

- Box plot:

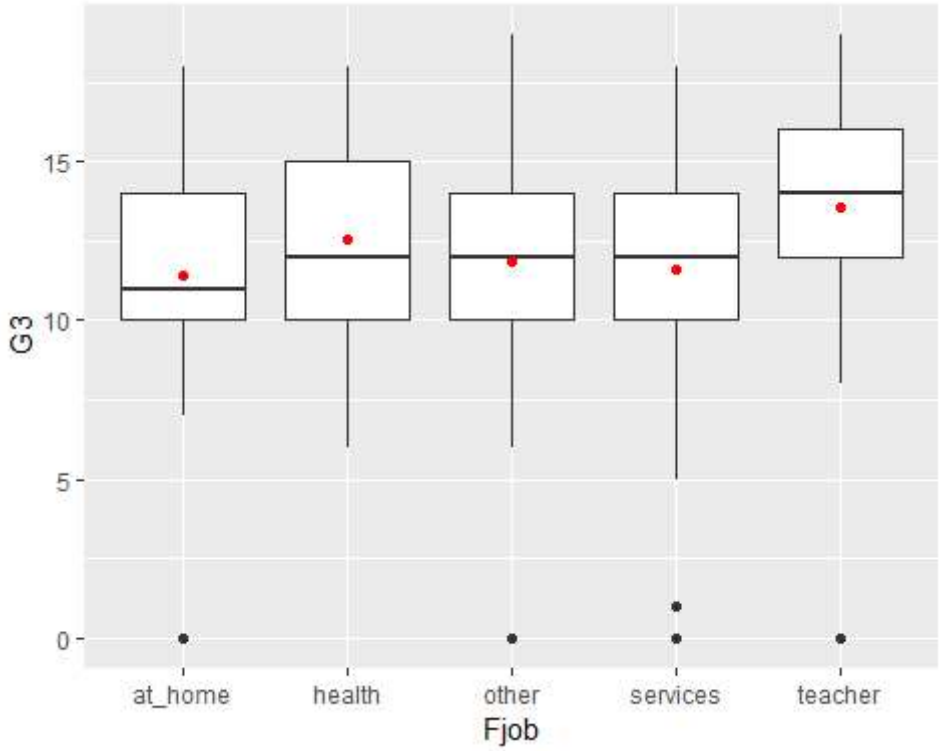
```
qplot(x = Fjob, y = G3, data = student, geom = "boxplot")
```



`qplot()` cont. 2

- Add some markers for the mean:

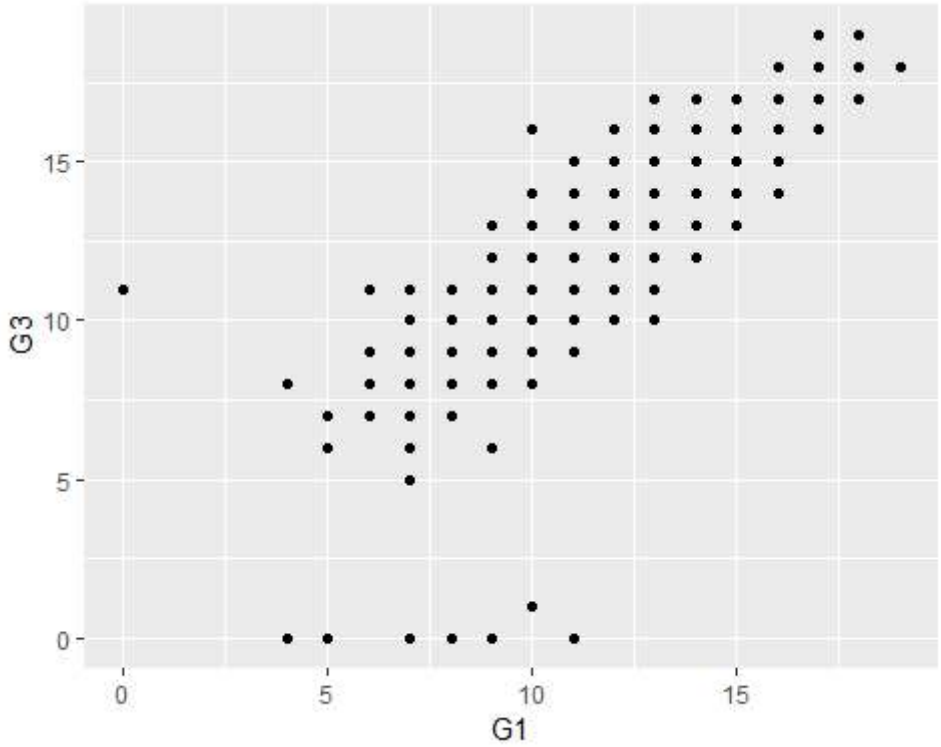
```
qplot(x = Fjob, y = G3, data = student ,geom = "boxplot") +
  stat_summary(fun.y = mean, colour = "red", geom = "point")
```



`qplot()` cont. 3

- A basic scatter plot:

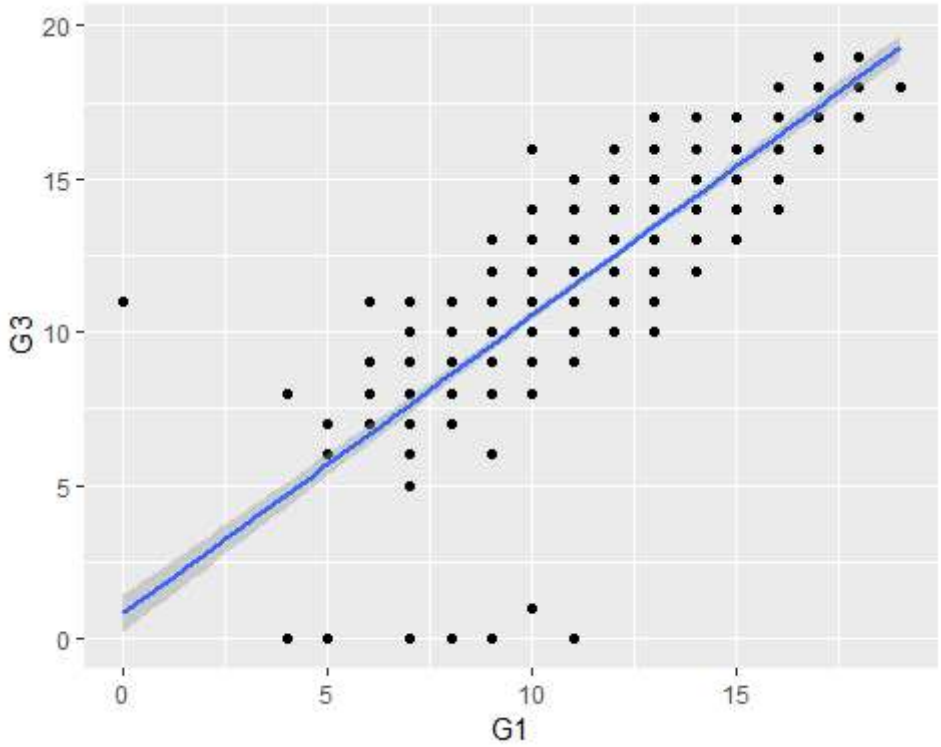
```
qplot(x = G1, y = G3, data = student, geom = "point")
```



`qplot()` cont. 4

- Add a line of best fit based on a linear model:

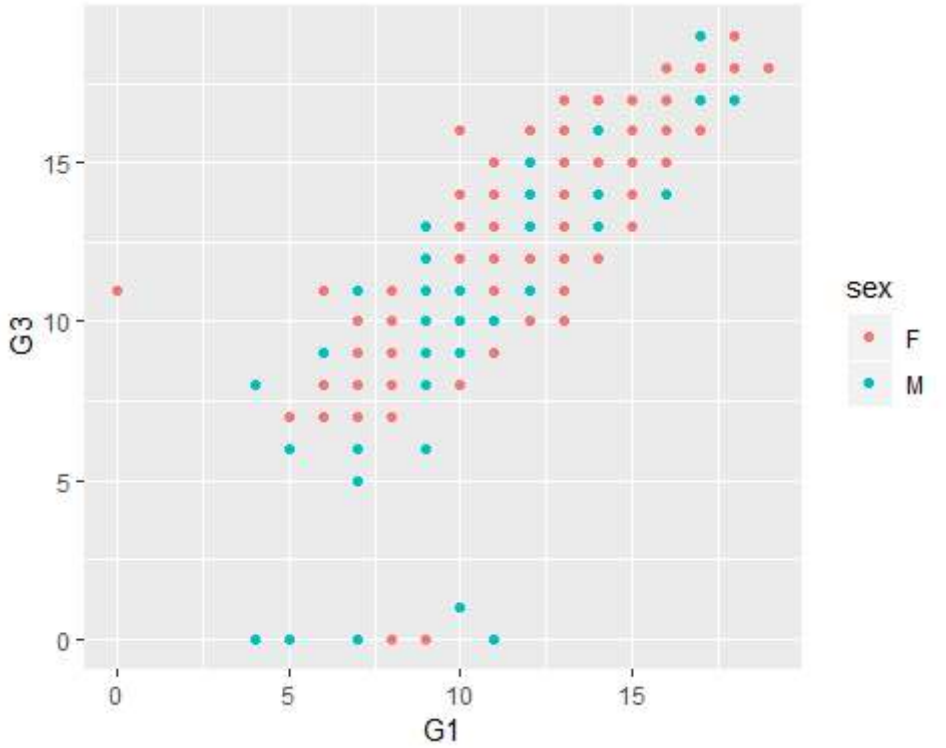
```
qplot(x = G1, y = G3, data = student, geom = "point") +
  geom_smooth(method = "lm")
```



`qplot()` cont. 5

- A scatter plot with a colour mapped to sex:

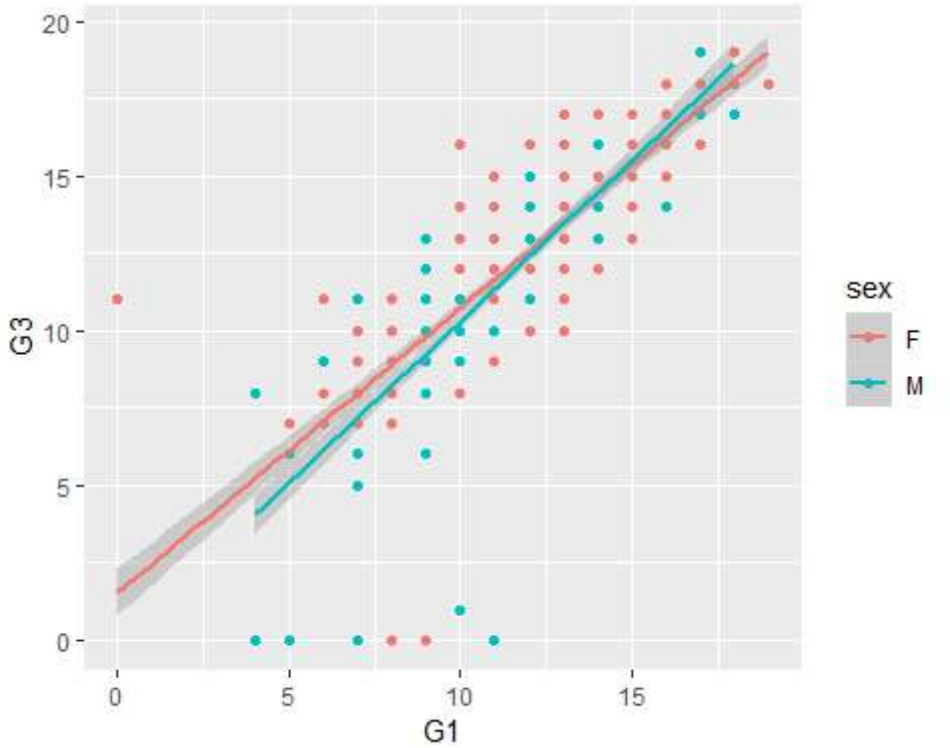
```
qplot(x = G1, y = G3, data = student, colour = sex, geom = "point")
```



`qplot()` cont. 6

- Add individual lines of best fit for each factor level mapped to the colour aesthetic:

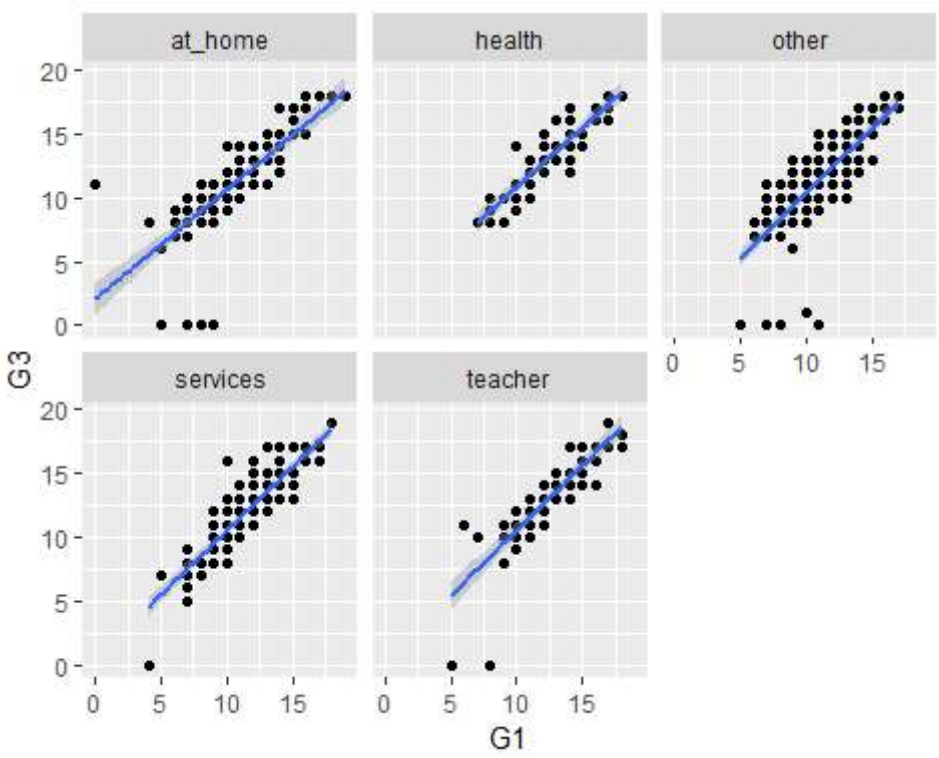
```
qplot(x = G1, y = G3, data = student, colour = sex, geom = "point") +
  geom_smooth(method = "lm")
```



`qplot()` cont. 7

- Facet scatter plots by mother's job:

```
qplot(x = G1,y = G3, data = student, geom = "point") +
  geom_smooth(method="lm") + facet_wrap(~ Mjob)
```



`ggplot()`

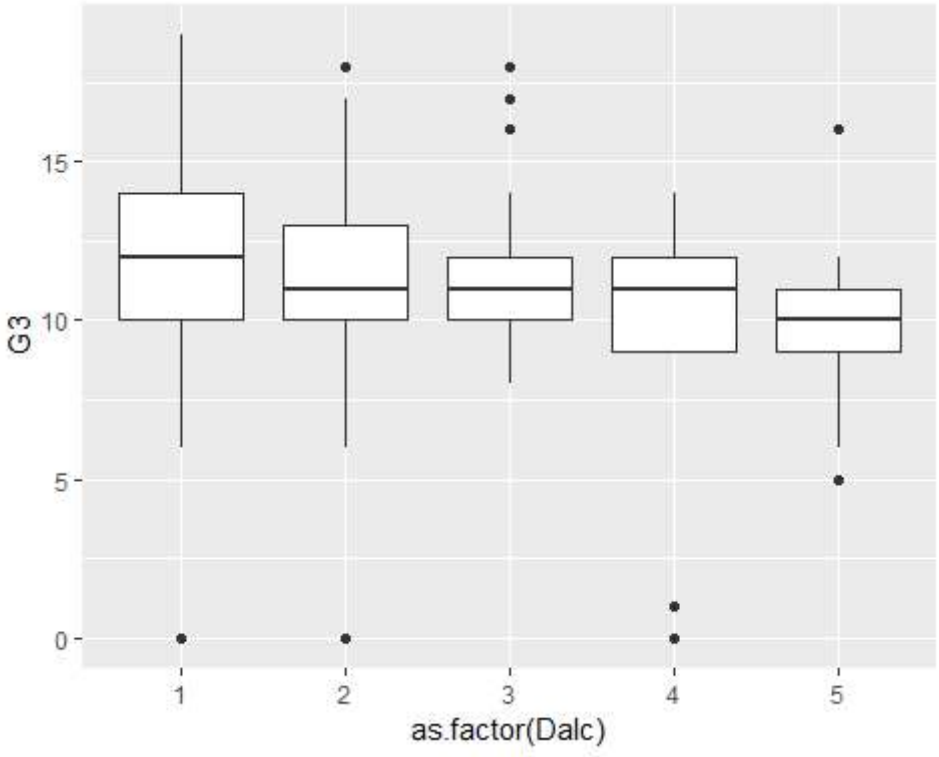
- `ggplot()` is a more powerful, layered approach to building a visualisation.
- Let's create a simple box plot comparing final grades by alcohol use ratings (1 - 5)
- First we define a `ggplot` object, `p1`

```
p1 <- ggplot(data = student, mapping = aes(x = as.factor(Dalc), y = G3))
```

- `aes` are the aesthetic mappings. Any layers added will map the data to the corresponding aesthetics in the `geom`. For example, a box plot.

`ggplot()` cont.

```
p1 + geom_boxplot()
```

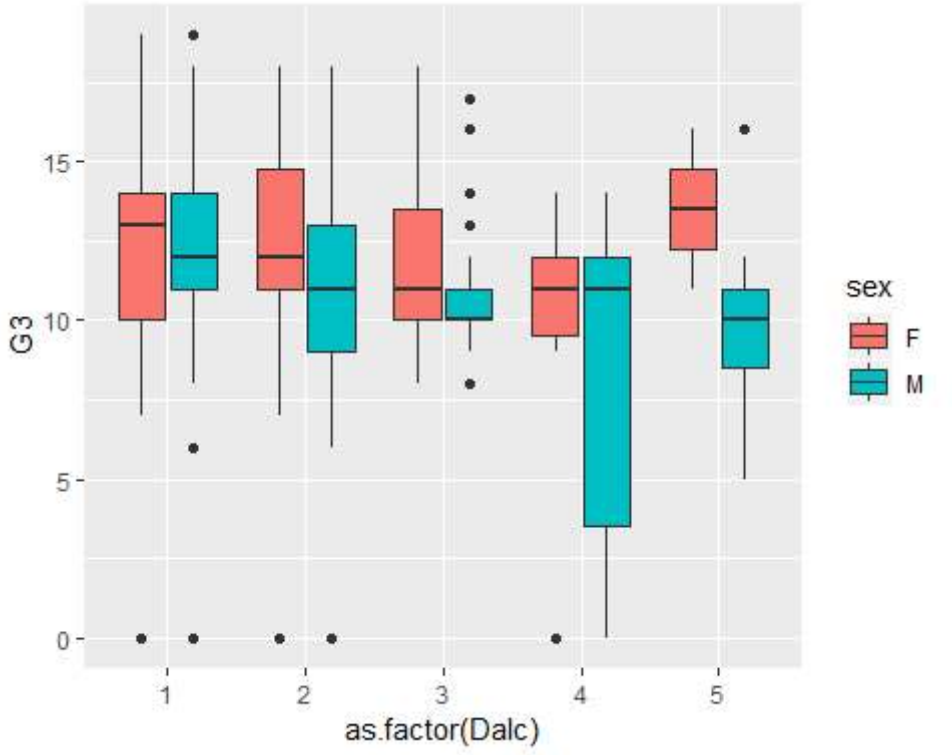


- Now we can continue to use `p1` to add more layers or change the visualisation completely.

`ggplot()` cont. 2

- We can map additional variables to other scales…
- Note how we mapped a `fill` aesthetic to `sex`.

```
p2 <- ggplot(student, aes(x = as.factor(Dalc), y = G3, fill = sex))
p2 + geom_boxplot()
```



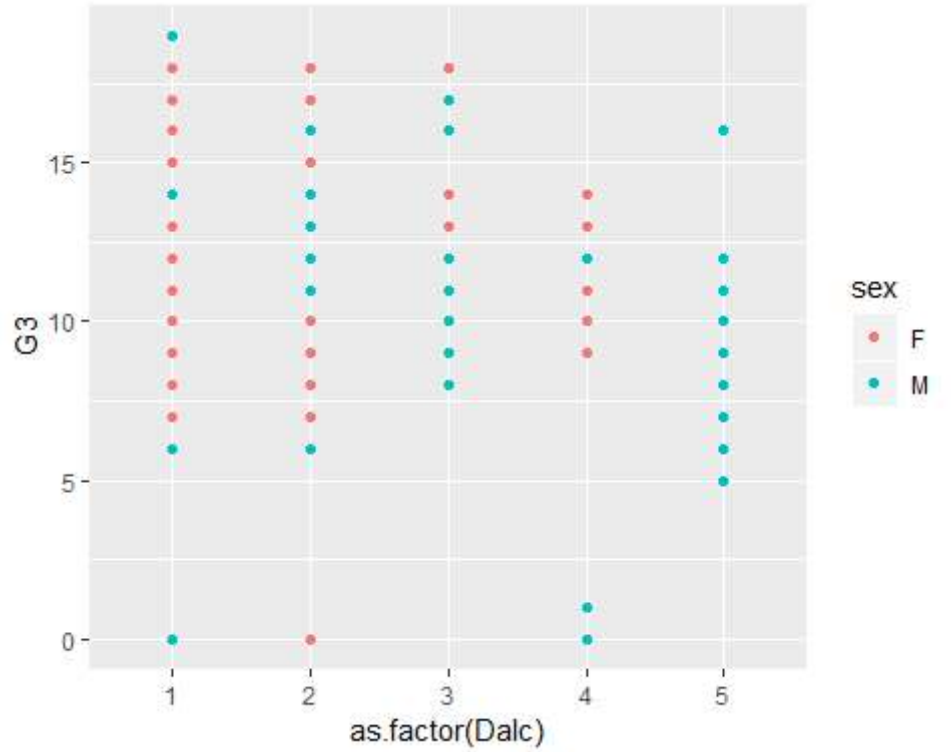
`ggplot()` cont. 3

- Box plots hide sample size. Use a different `geom` that conveys sample size…

```
p3 <- ggplot(student, aes(x = as.factor(Dalc), y = G3, colour = sex))
p3 + geom_point()
```

`ggplot()` cont. 4

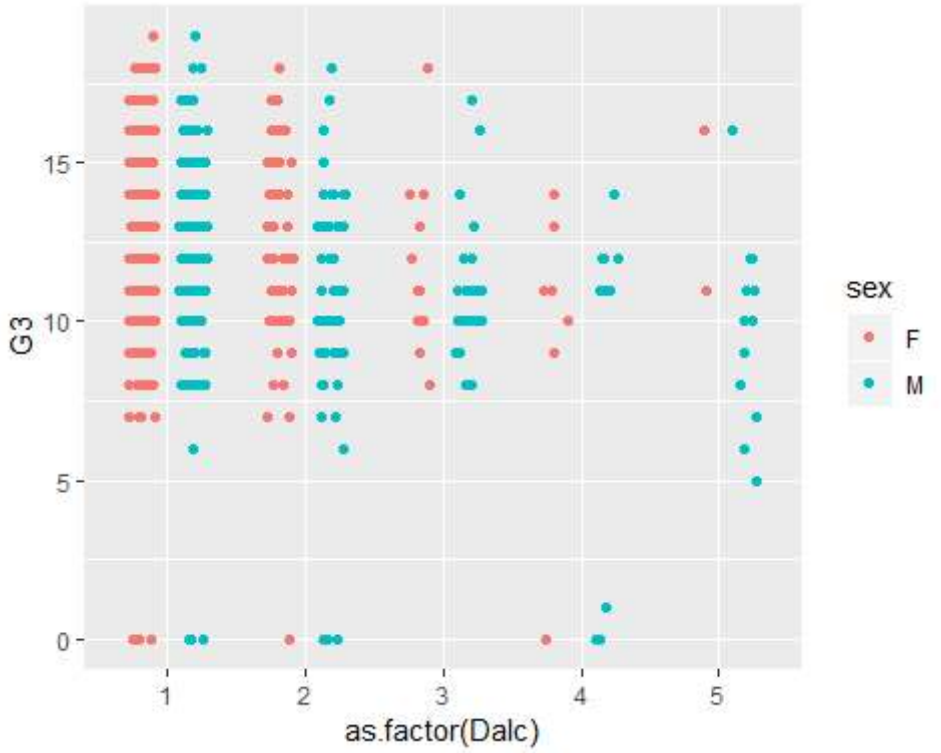
- Note `colour` refers to the outline or solid colour of an object. `fill` refers to the inside colour of a large object, e.g. bar or box.



`ggplot()` Jitter

- Points and categories overlap!
- Use position adjustments to avoid over-plotting

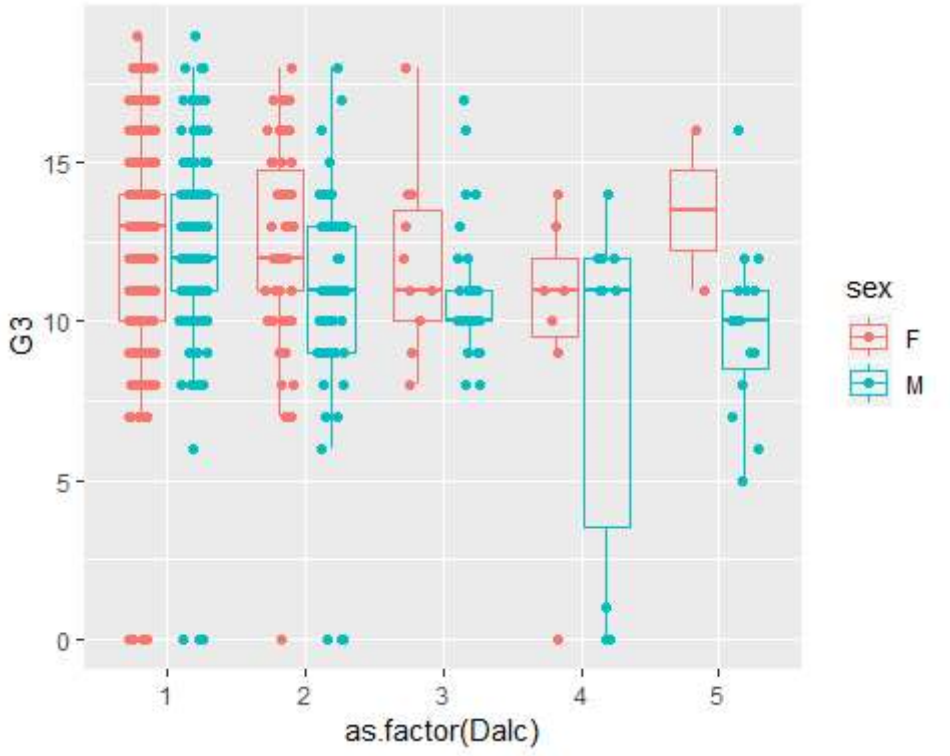
```
p3 + geom_jitter(position = position_jitterdodge())
```



`ggplot()` Adding Layers

- Overlaying additional `geoms` is easy…
- Note how we use `outlier.shape = NA` to suppress the outliers plotted for the boxplot

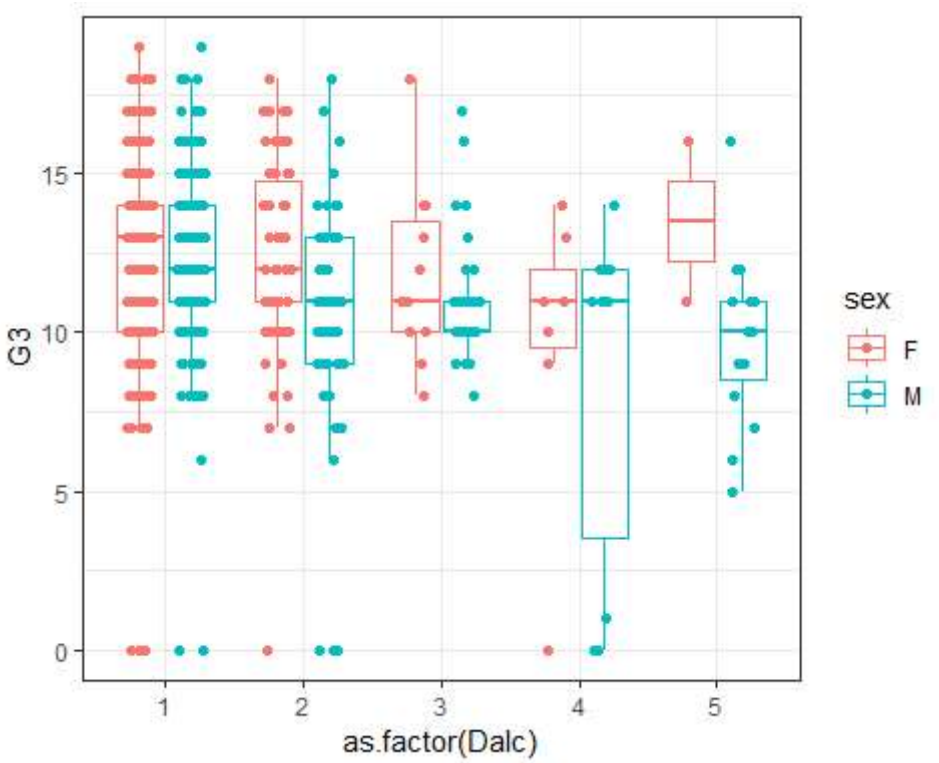
```
p3 + geom_jitter(position = position_jitterdodge()) +
  geom_boxplot(fill = NA, outlier.shape = NA)
```



`ggplot()` Themes

- You can change default themes:

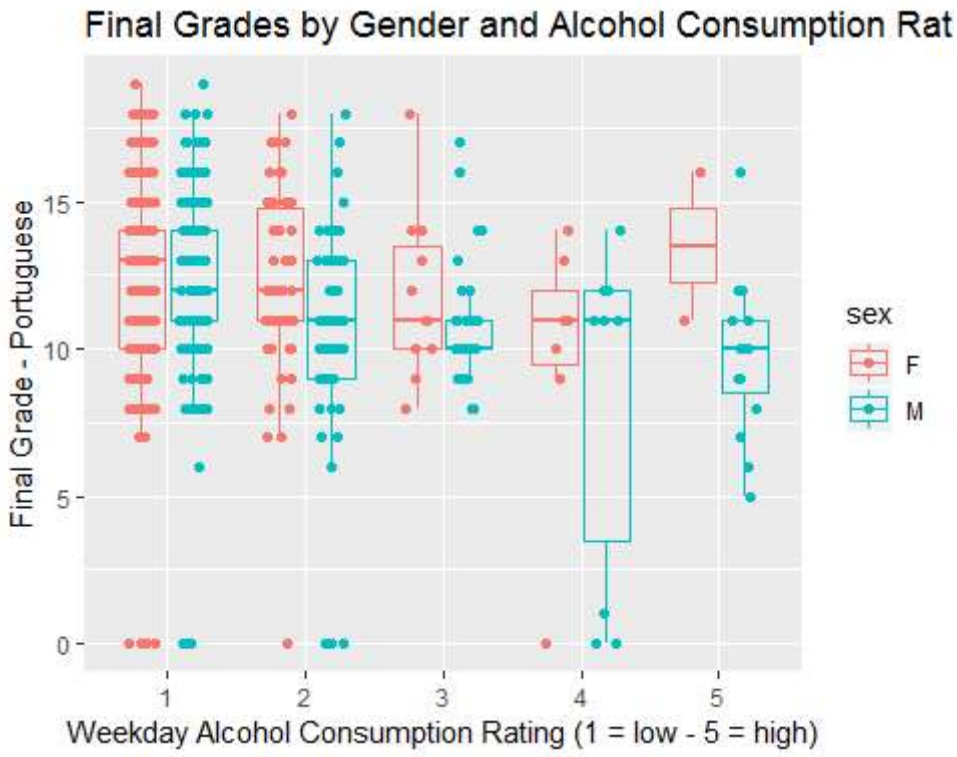
```
p3 + geom_jitter(position = position_jitterdodge()) +
  geom_boxplot(fill = NA, outlier.shape = NA) + theme_bw()
```



`ggplot()` Adding Titles and Labels

- You can define descriptive labels:

```
p3 + geom_jitter(position = position_jitterdodge()) +
  geom_boxplot(fill = NA, outlier.shape = NA) +
  labs(title = "Final Grades by Gender and Alcohol Consumption Ratings",
       x = "Weekday Alcohol Consumption Rating (1 = low - 5 = high)",
       y = "Final Grade - Portuguese")
```



`ggplot()` Saving

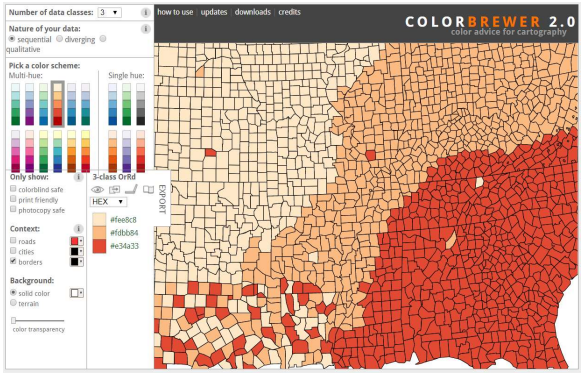
- You can use RStudio to save and export your data visualisation as an image (PNG, JPG, TIFF, BMP, Metafile, SVG and EPS), PDF or to the clipboard.
- You can also use the nifty `ggsave()` function which saves an image, using sensible defaults, to your working directory.

```
ggsave("Box_plot_Grades_Gender_Consumption_01.png",
       width = 18, height = 12, units = "cm")
```

- You might have to play around with the width and height. Measurements are in inches by default. You need to specify `"cm"` if you want metric.

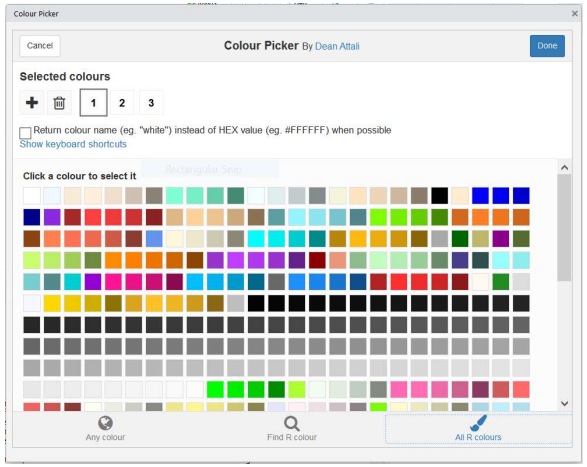
Colour in R

- R allows colour assignment using a few different methods:
  - Colour code models: For example, hexadecimal codes as well as others including HSV, HSL, RGB and CMYK.
  - Inbuilt R colour names (657 to choose from)
  - Packages such as `RColorBrewer`



colourpicker

- The `colourpicker` package provides `ggplot2` with a very useful interface for colour picking and exploration.



- First, install the package.

```
install.packages("colourpicker")
```

- Now, load the package.

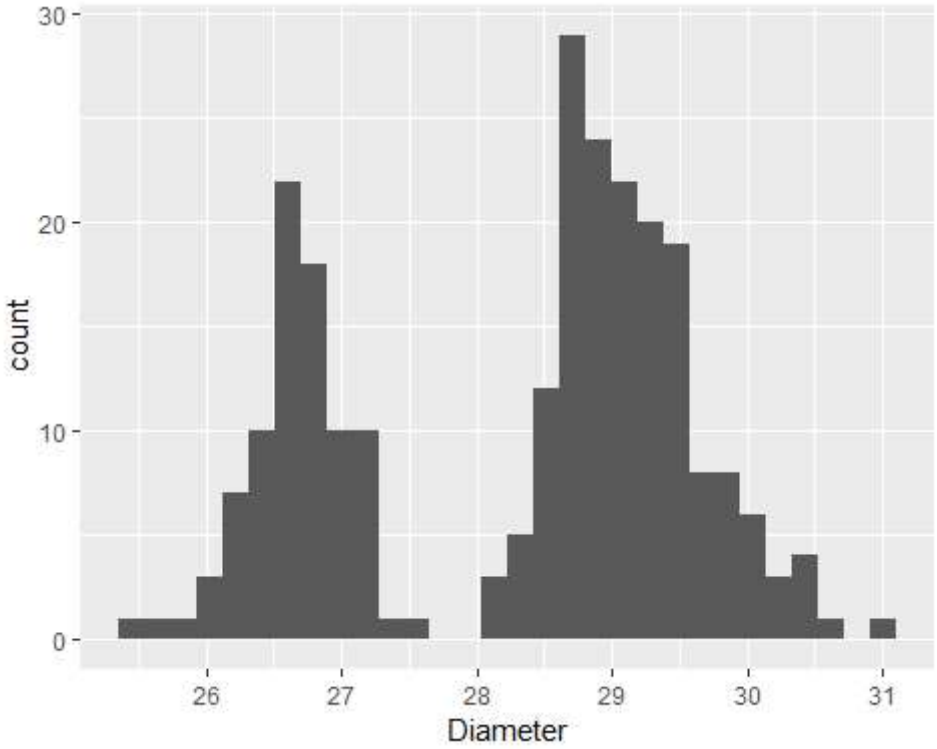
```
library(colourpicker)
```

- You can access the Colour Picker through the **Addins** menu in RStudio

Basic Colour Assignment

- Produce a histogram showing the distribution of pizza diameter.

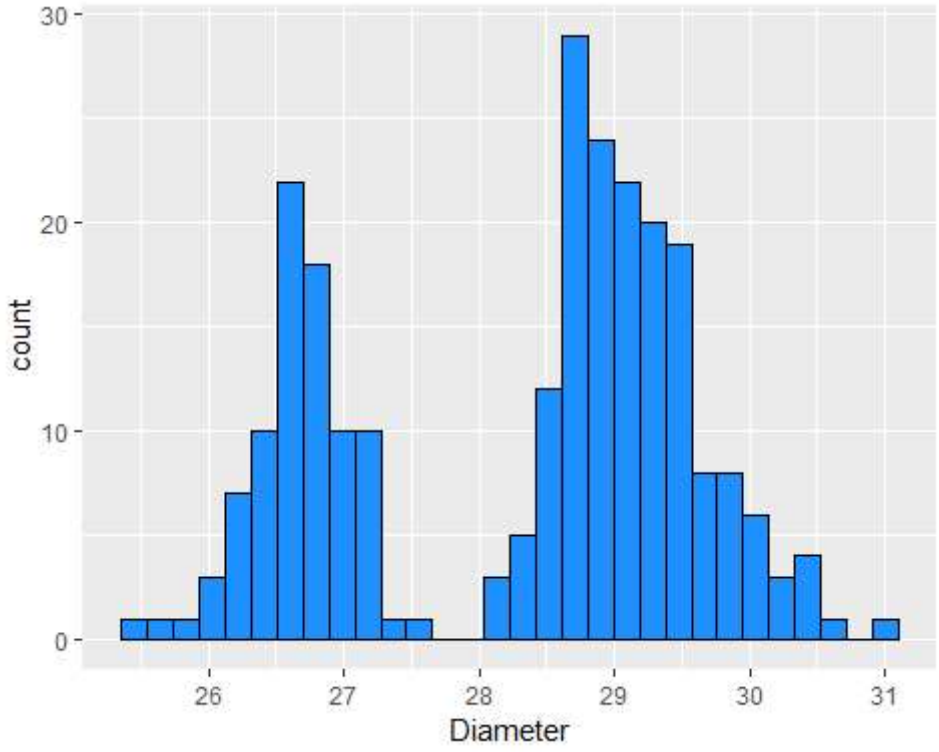
```
Pizza <- read.csv("../data/Pizza.csv")
p1 <- ggplot(data = Pizza, aes(x = Diameter))
p1 + geom_histogram()
```



Basic Colour Assignment Cont.

- Change colour using R colour names

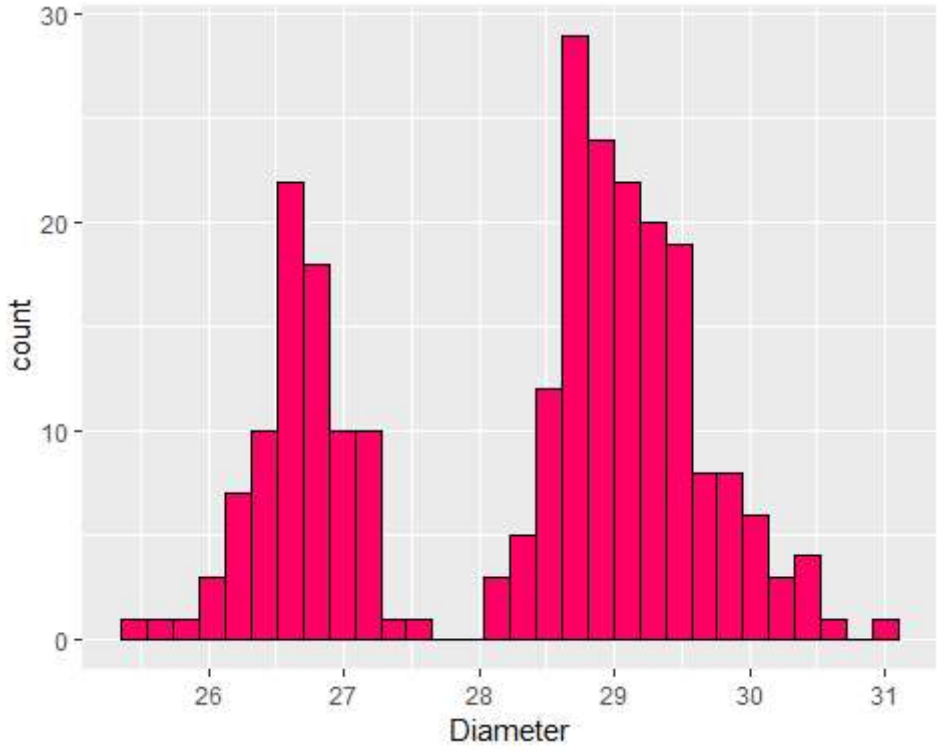
```
p1 + geom_histogram(fill = "dodgerblue", colour = "black")
```



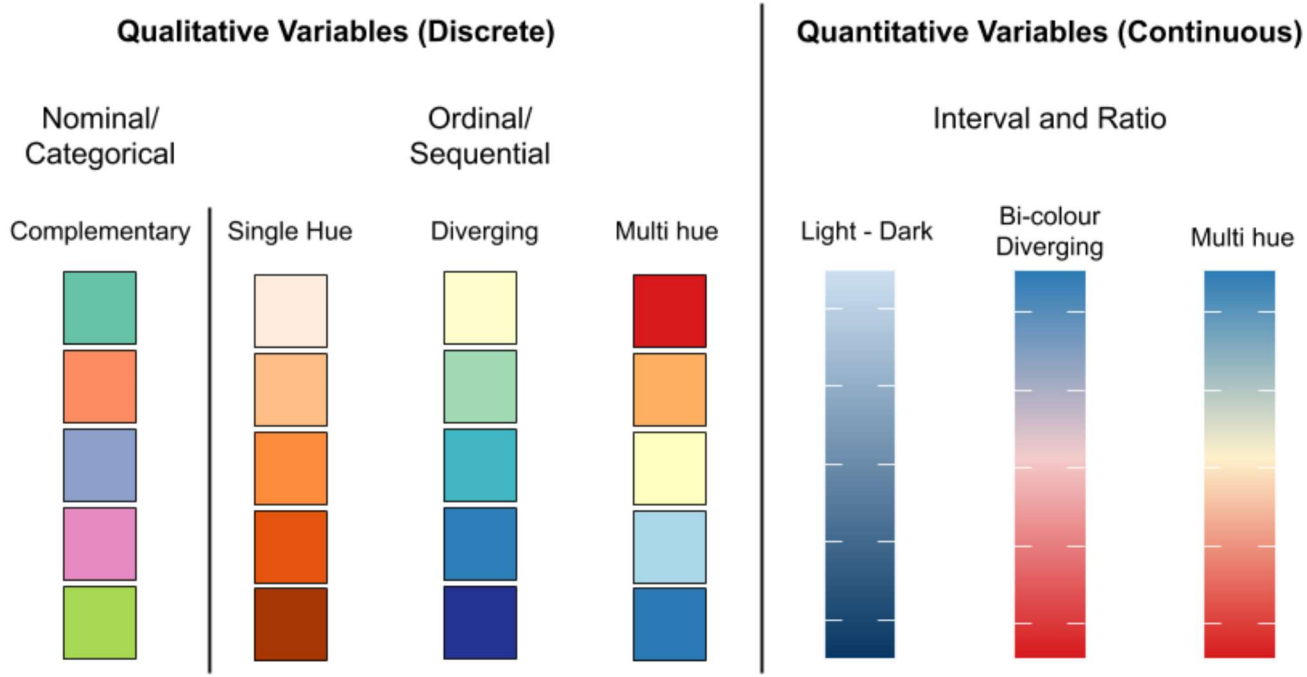
Basic Colour Assignment Cont. 2

- Change colour using hex codes

```
p1 + geom_histogram(fill = "#ff0066", colour = "#000000")
```



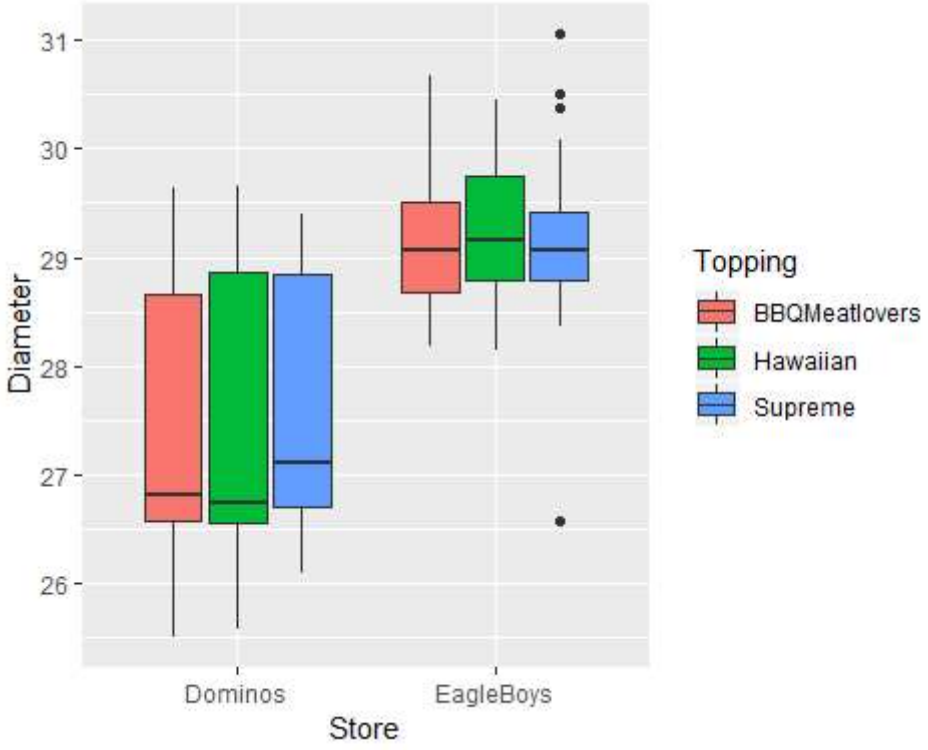
Colour Scales



Nominal Colour Scale Example

- Side-by-side box plot comparing pizza diameter by Store and Topping

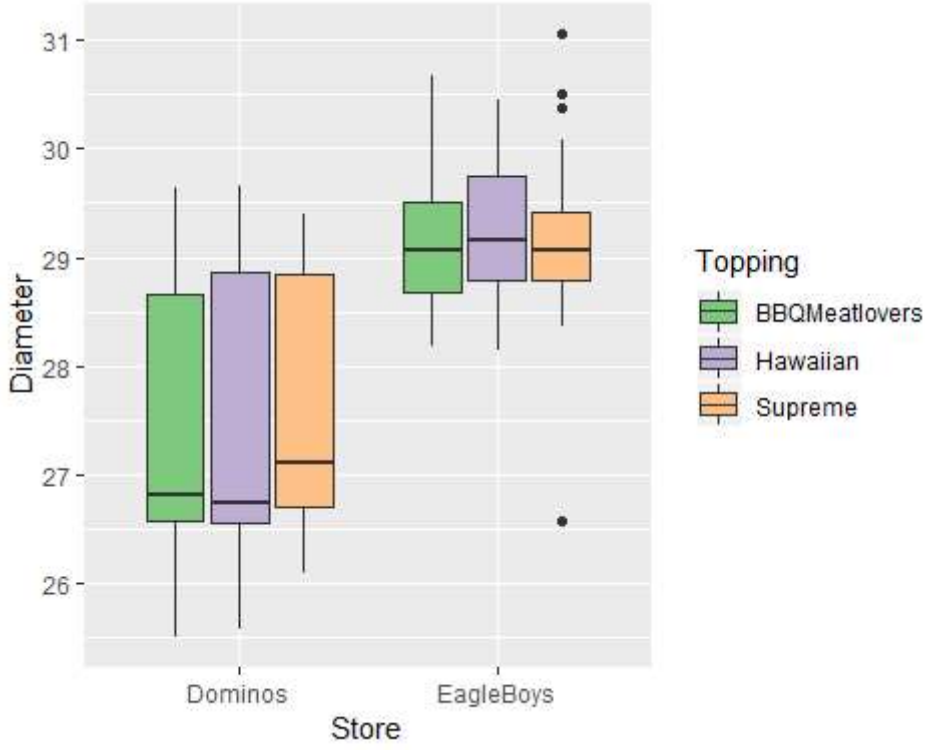
```
p2 <-ggplot(data = Pizza, aes(x = Store, y = Diameter, fill = Topping))
p2 + geom_boxplot()
```



Nominal Colour Scale Example Cont.

- Change ColourBrewer palette…

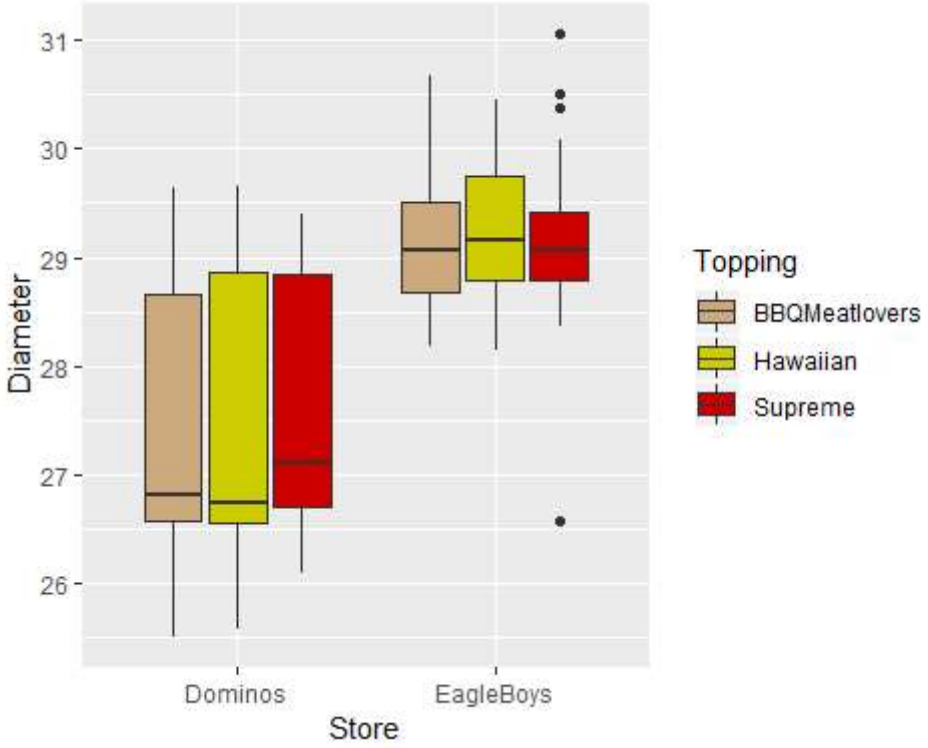
```
p2 + geom_boxplot() + scale_fill_brewer(palette = "Accent")
```



Nominal Colour Scale Example Cont. 2

- Set manual colour scale…

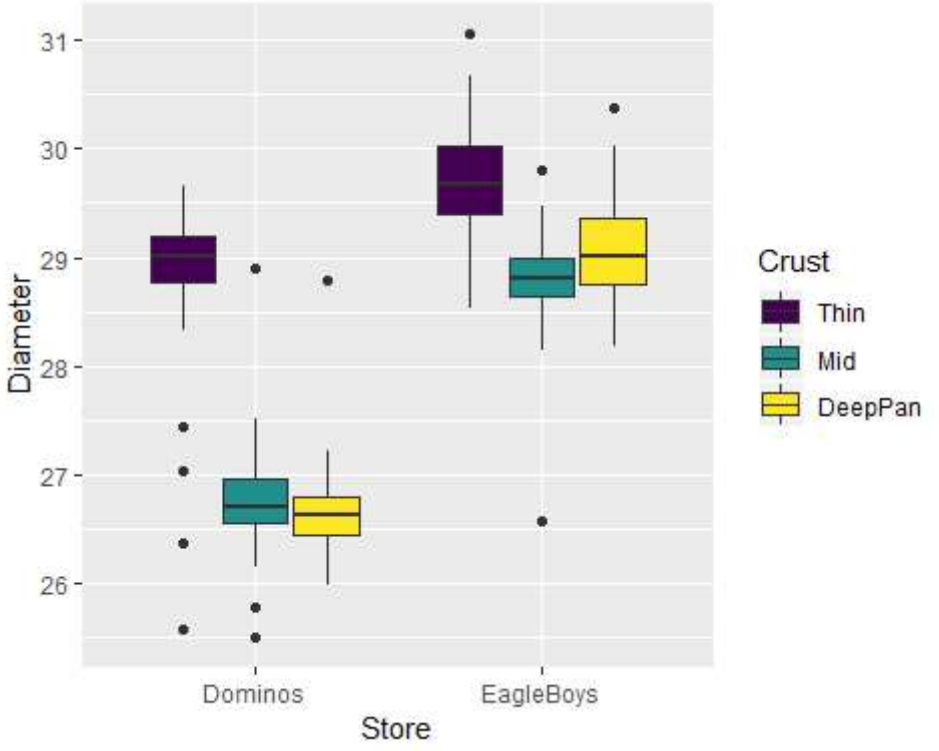
```
p2 + geom_boxplot() + scale_fill_manual(
  values = c("burlywood3","yellow3","red3"))
```



Ordinal Colour Scale Example

- Boxplot comparing pizza diameter by Store and Crust.

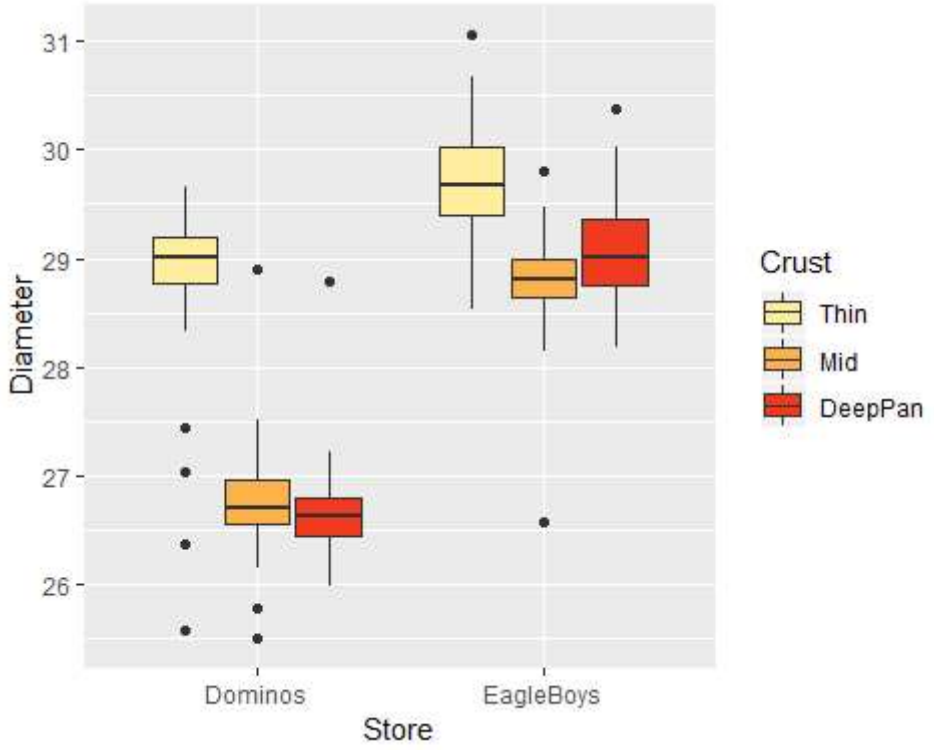
```
Pizza$Crust<-factor(Pizza$Crust, levels = c("Thin","Mid","DeepPan"),
                    ordered = T)
p3 <-ggplot(data = Pizza, aes(x = Store, y = Diameter, fill = Crust))
p3 + geom_boxplot()
```



Ordinal Colour Scale Example Cont.

- Change `palette()`...

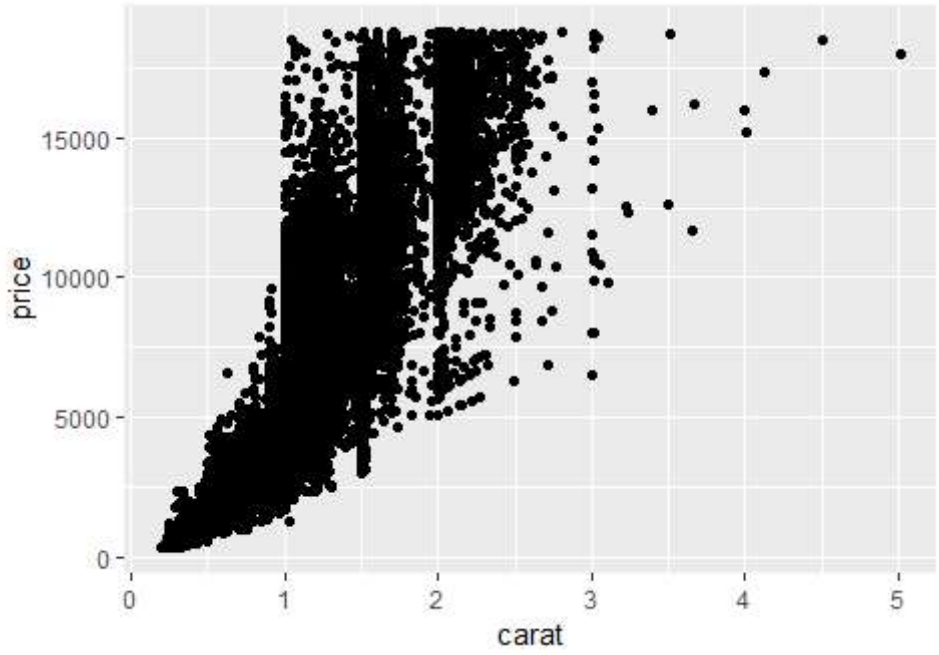
```
p3 + geom_boxplot() + scale_fill_brewer(palette = "YlOrRd")
```



Continuous Colour Scale

- Explore the relationship between diamond carat and price

```
Diamonds <- read.csv("../data/Diamonds.csv")
p4 <- ggplot(data = Diamonds, aes(x = carat, y = price))
p4 + geom_point()
```

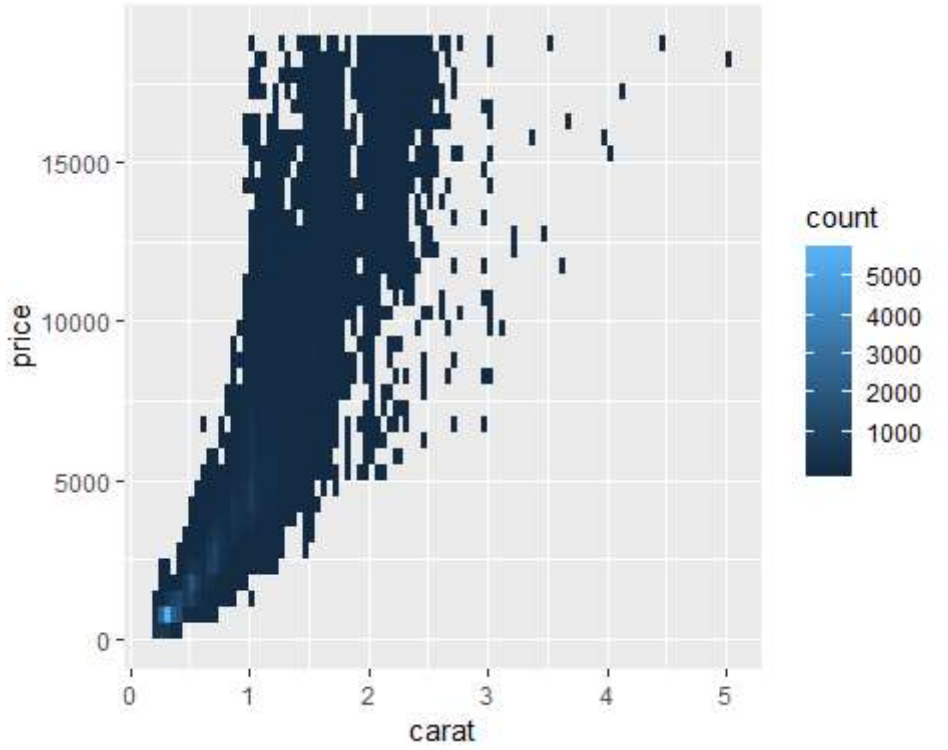


- Hard to see data density as $n = 53940$.

Continuous Colour Scale Cont.

- Use a continuous colour scale to represent data density in a 2d histogram

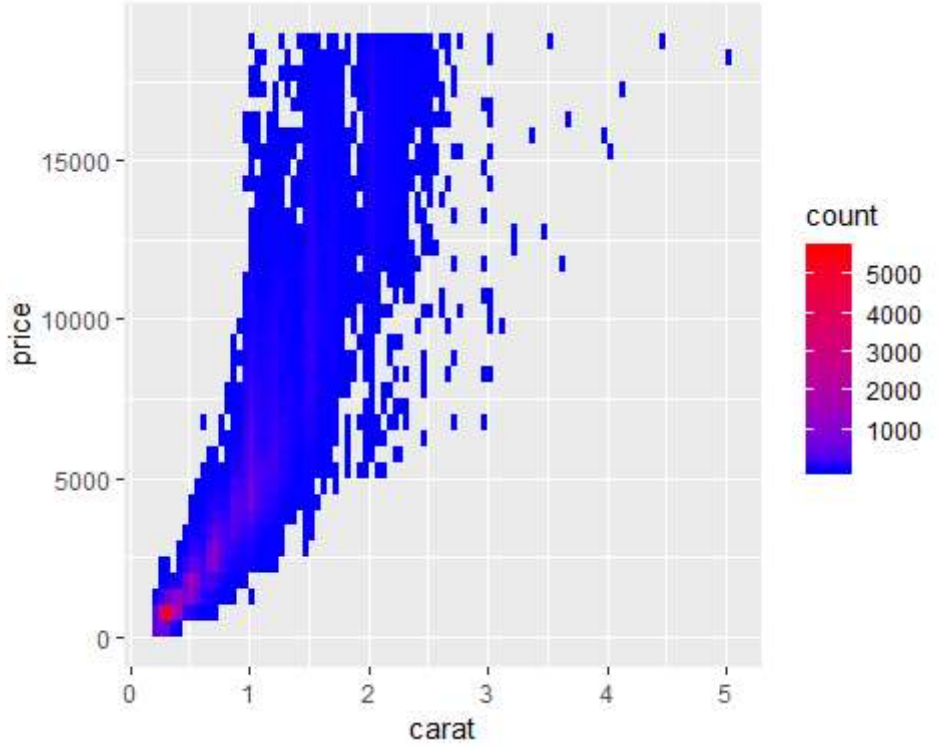
```
p4 + geom_bin2d(binwidth = c(0.05, 500))
```



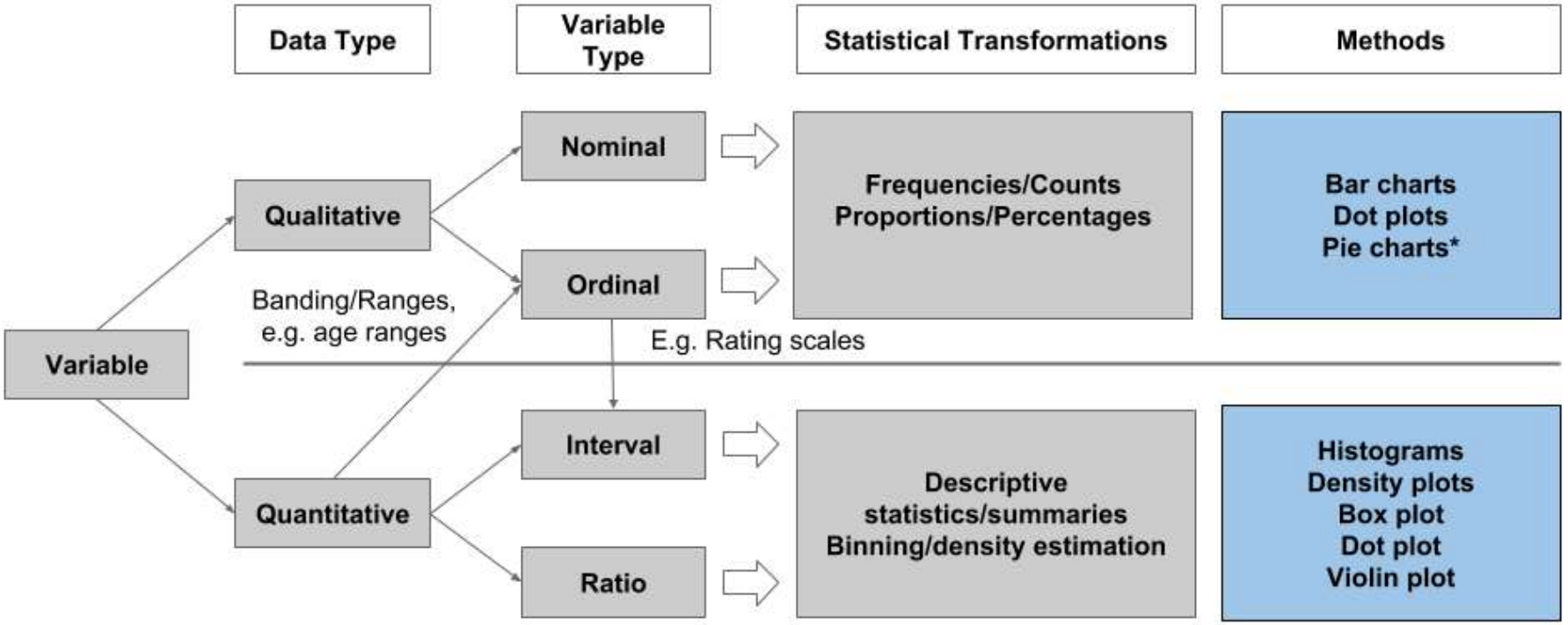
Continuous Colour Scale Cont. 2

- Change colour gradient…

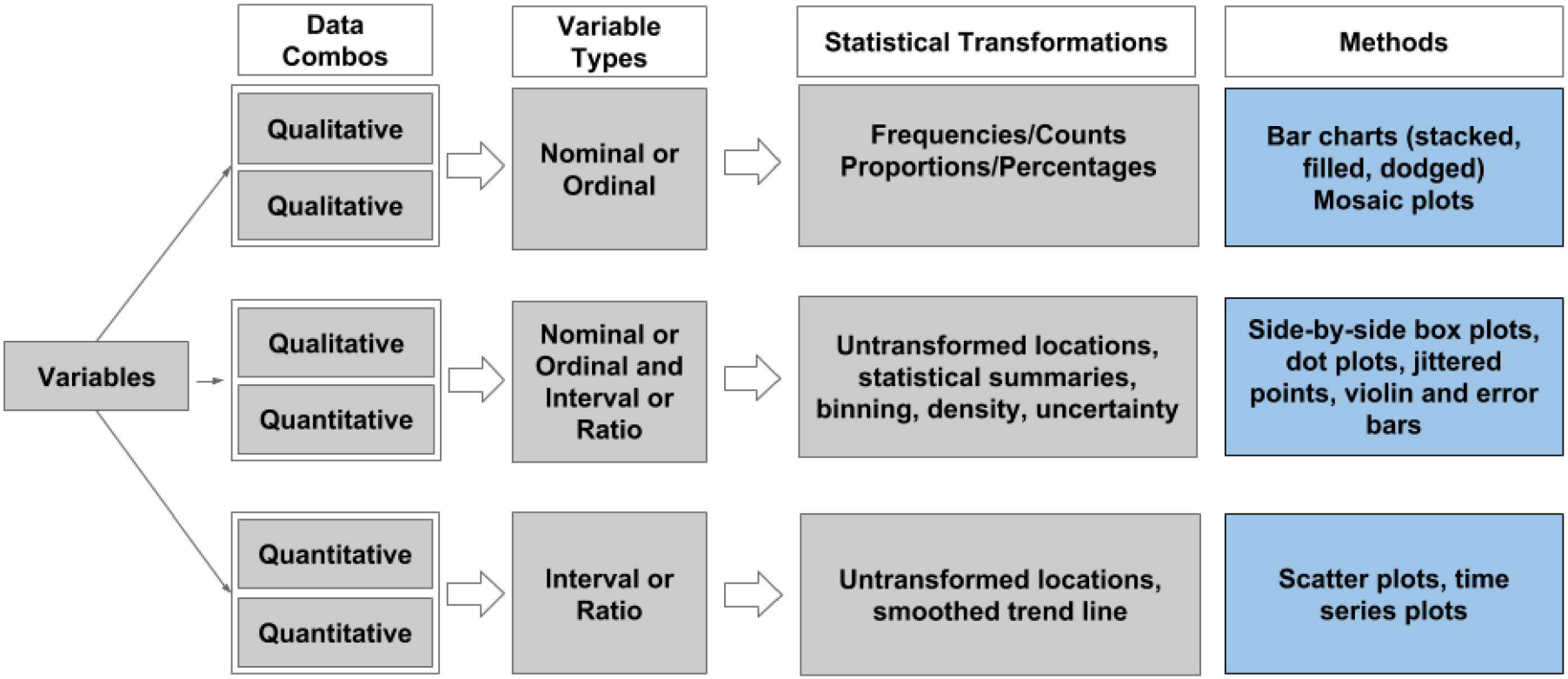
```
p4 + geom_bin2d(binwidth = c(0.05, 500)) +
  scale_fill_gradient(low="blue", high="red")
```



Overview - Common Univariate Methods



Overview - Common Bivariate Methods



`ggplot2` Resources

- Recommended resources:
  - `ggplot2` Documentation
  - `ggplot2`: Elegant Graphics for Data Analysis
  - Data Visualisation with `ggplot2` Cheat Sheet
  - Read Hadley's Wickham's paper - A Layered Grammar of Graphics

References

Wickham, H. 2010. “A layered grammar of graphics.” *Journal of Computational and Graphical Statistics* 19 (1): 3–28. doi:10.1198/jcgs.2009.07098.

Wilkinson, L. 2005. *The Grammar of Graphics*. Statistics and Computing. New York: Springer-Verlag. doi:10.1007/0-387-28695-0.