

# FIT3152 Data analytics – Lecture 3

---

## Graphics...

- Quick follow up of last week's lecture

## R Tips

- R Markdown, Scripts, User-defined functions

## Assignment 1

## Data Manipulation

- Making tables and summaries, Working with factors
- Transforming data, *dplyr* package,
- Working with dates and times.

# Consultations on Zoom

---

Clayton consultations have commenced:

- Any student can attend any consultation.
- Schedule on Moodle, <https://lms.monash.edu/>
- Current days/times:
- Monday 9:30-10:30AM, 2:00-3:00PM, 6:00-7:00PM,
- Tuesday 9:00-10:00AM, 12:00PM-1:00PM,
- Wednesday 10:00AM-11:00, 11:00-12:00PM,
- Thursday 1:00PM-02:00PM, 6:00PM-7:00PM.
- Please check the schedule for any changes.

# Week-by-week

---

Week Starting	Lecture	Topic	Tutorial	A1	A2
28/2/22	1	Intro to Data Science, review of basic statistics using R	...		
7/3/22	2	Exploring data using graphics in R	T1		
14/3/22	3	Data manipulation in R	T2	Released	
21/3/22	4	Data Science methodologies, dirty/clean/tidy data, data manipulation	T3		
28/3/22	5	Network analysis	T4		
4/4/22	6	Regression modelling	T5		
11/4/22	7	Classification using decision trees	T6		
		Mid-semester Break		Submitted	
25/4/22	8	Naïve Bayes, evaluating classifiers	T7		Released
2/5/22	9	Ensemble methods, artificial neural networks	T8		
9/5/22	10	Clustering	T9		
16/5/22	11	Text analysis	T10		Submitted
23/5/22	12	Review of course, Exam preparation	T11		

---

## Brief review of Lecture 2...

---

## Visualising data

- The number of dimensions in a data set
- Major families of graph types: time series, statistical distributions, maps, hierarchies, networks.
- Plotting the Iris data: basic scatterplot and increasing the number of dimensions presented.
- lattice and ggplot2 packages, and the Grammar of Graphics approach.

# R for Data Science

---

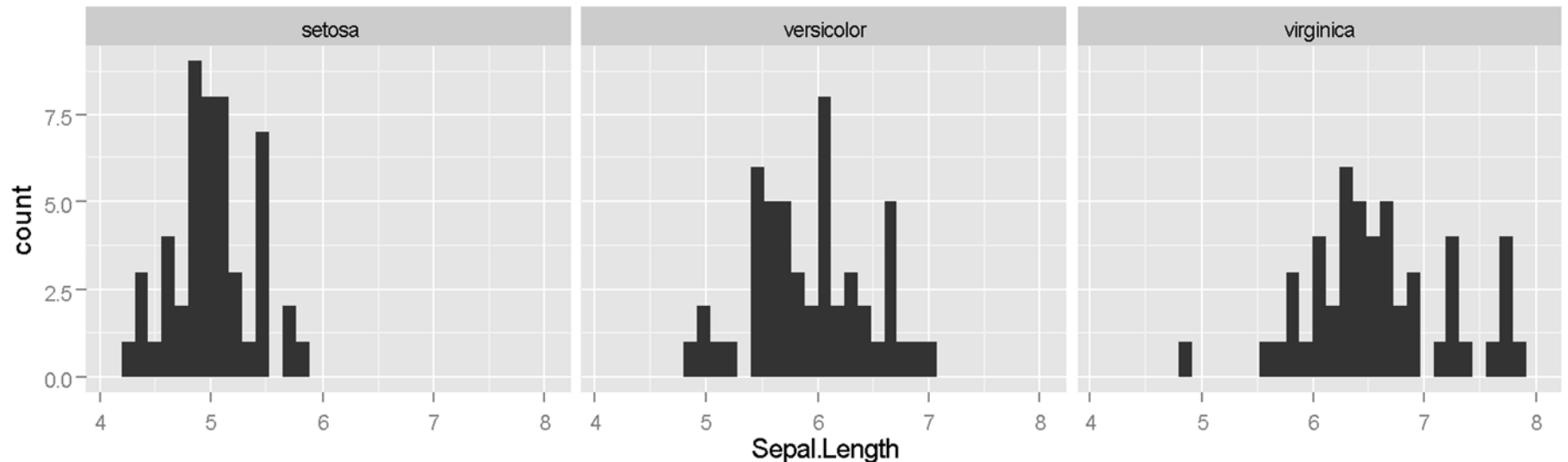
- A physical and web-based book by the author of ggplot2, Hadley Wickham, and Garrett Grolemund:  
<http://r4ds.had.co.nz/>
- The book takes you through all aspects of the data science workflow (more later)
- A good chapter on ggplot2, including the syntax underpinning all ggplots, for example:  

```
> ggplot(data = <DATA>) +  
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```

# Syntax: histogram + facet\_wrap

---

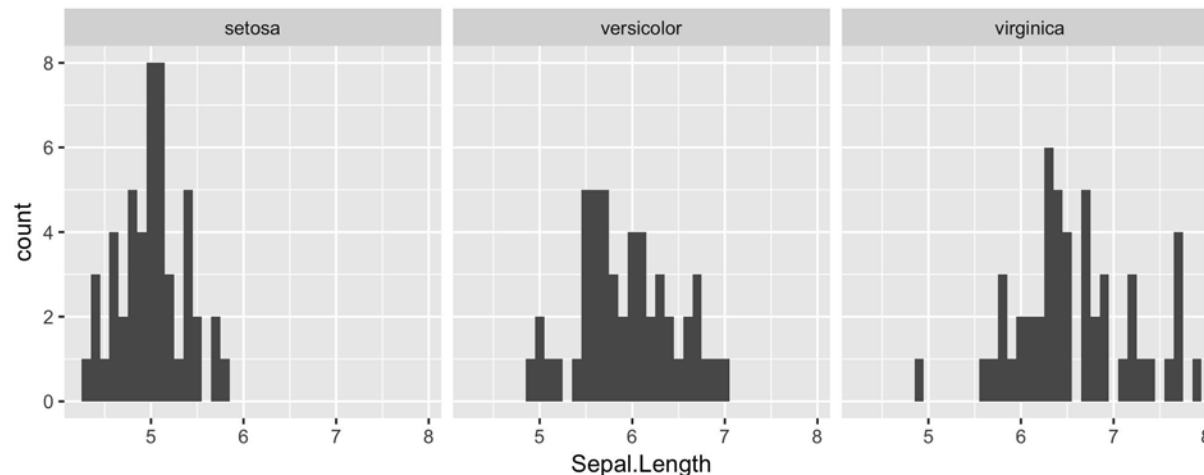
```
> qplot(Sepal.Length, data = iris, geom = "histogram",  
  facets = Species ~ .) + facet_wrap(~ Species, ncol = 3)
```



# Using the grammar approach...

---

- > `m = ggplot(iris, aes(x = Sepal.Length)) #data`
- > `m = m + geom_histogram(binwidth = 0.1) #graph type`
- > `m = m + facet_wrap(~Species, ncol = 3) #grouping var`
- > `ggsave("irissepallen.jpg", m, width = 20, height = 8, units = "cm")`





# MPG example

---

Recall the mpg data set.

```
> head(mpg)
# A tibble: 6 x 11
  manufacturer model displ  year   cyl    trans   drv   cty   hwy   fl   class
    <chr>    <chr> <dbl> <int> <int>   <chr> <chr> <int> <int> <chr>  <chr>
1     audi     a4   1.8  1999     4 auto(l5)   f    18    29   p compact
2     audi     a4   1.8  1999     4 manual(m5)  f    21    29   p compact
3     audi     a4   2.0  2008     4 manual(m6)  f    20    31   p compact
4     audi     a4   2.0  2008     4 auto(av)    f    21    30   p compact
5     audi     a4   2.8  1999     6 auto(l5)   f    16    26   p compact
6     audi     a4   2.8  1999     6 manual(m5)  f    18    26   p compact
```

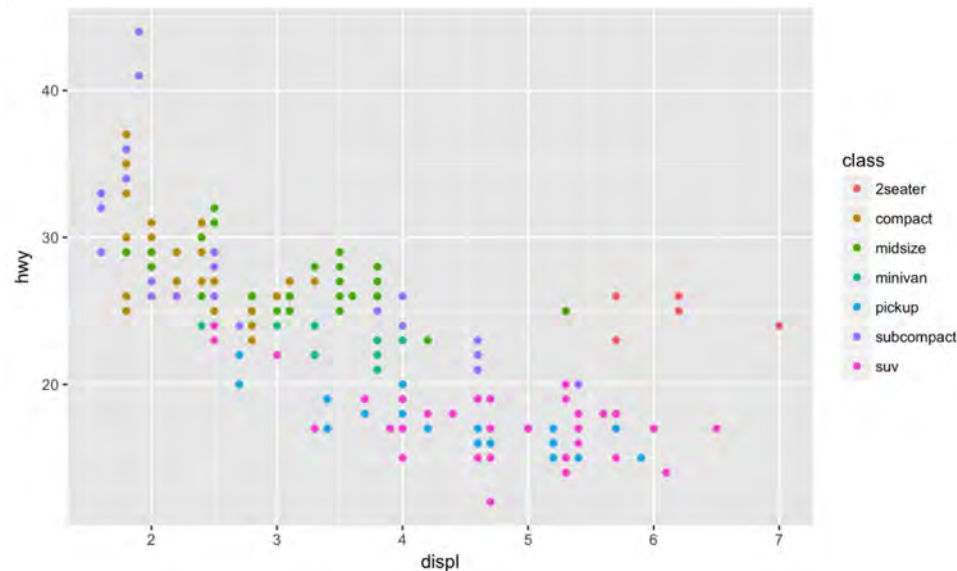
Investigate the relationship between fuel consumption and engine displacement.

# Basic plot

---

Using a grammar of graphics approach (from R4DS)

- > `g = ggplot(data = mpg)`
- > `g = g + geom_point(mapping = aes(x = displ, y = hwy, color = class))`
- > `g`



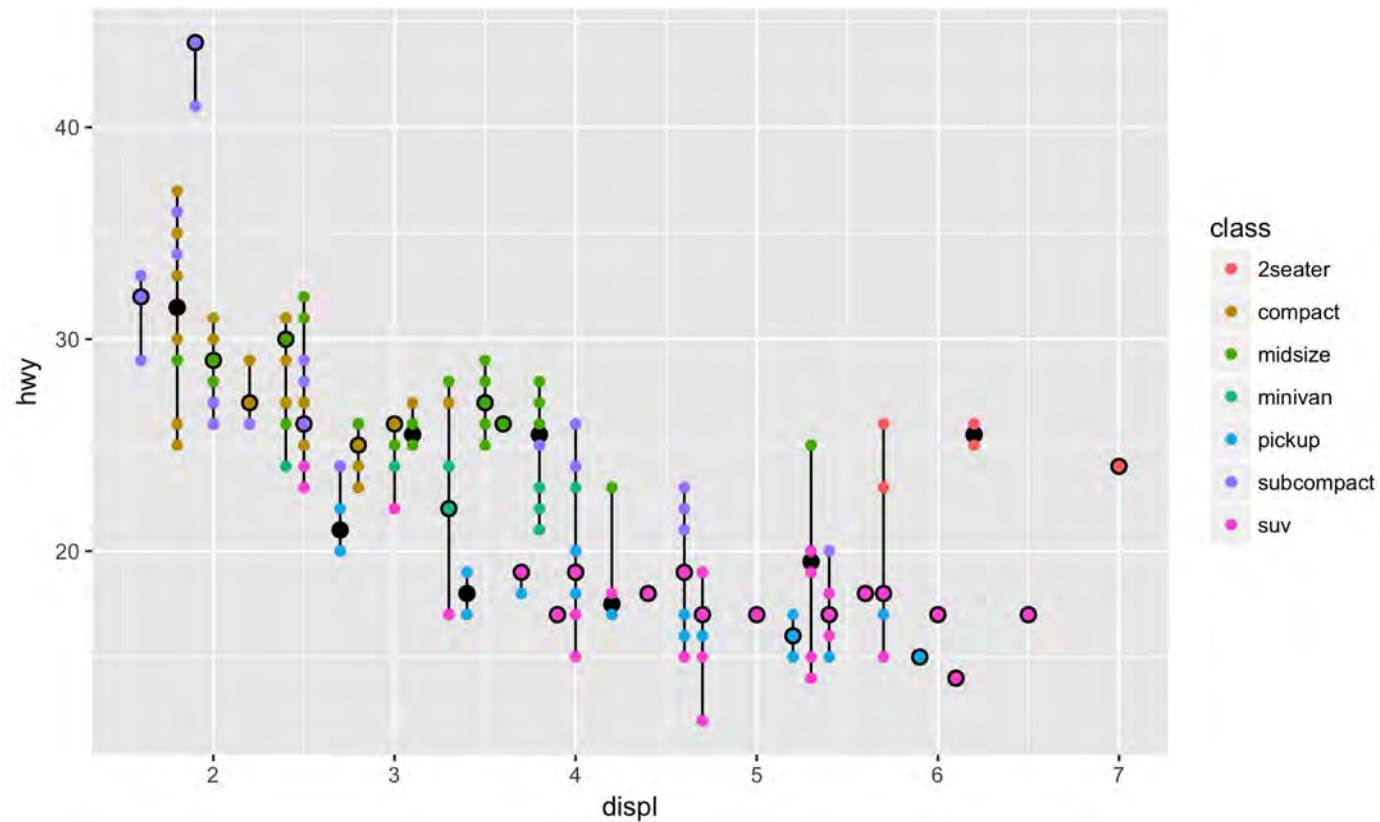
# Underplotting (min, median, max)

---

- > d <- ggplot(mpg, aes(displ, hwy, color = class)) +  
geom\_point()
- > d = d + stat\_summary(mapping = aes(x = displ, y =  
hwy), fun.min = min, fun.max = max, fun = median,  
orientation = "x", colour = "black")
- > d = d + geom\_point(mapping = aes(x = displ, y = hwy,  
color = class)) **# overplots original points**
- > d
- > ggsave("hwyvdispl.jpg", d, width = 20, height = 12,  
units = "cm")

---

# The plot. What improvements would you make?

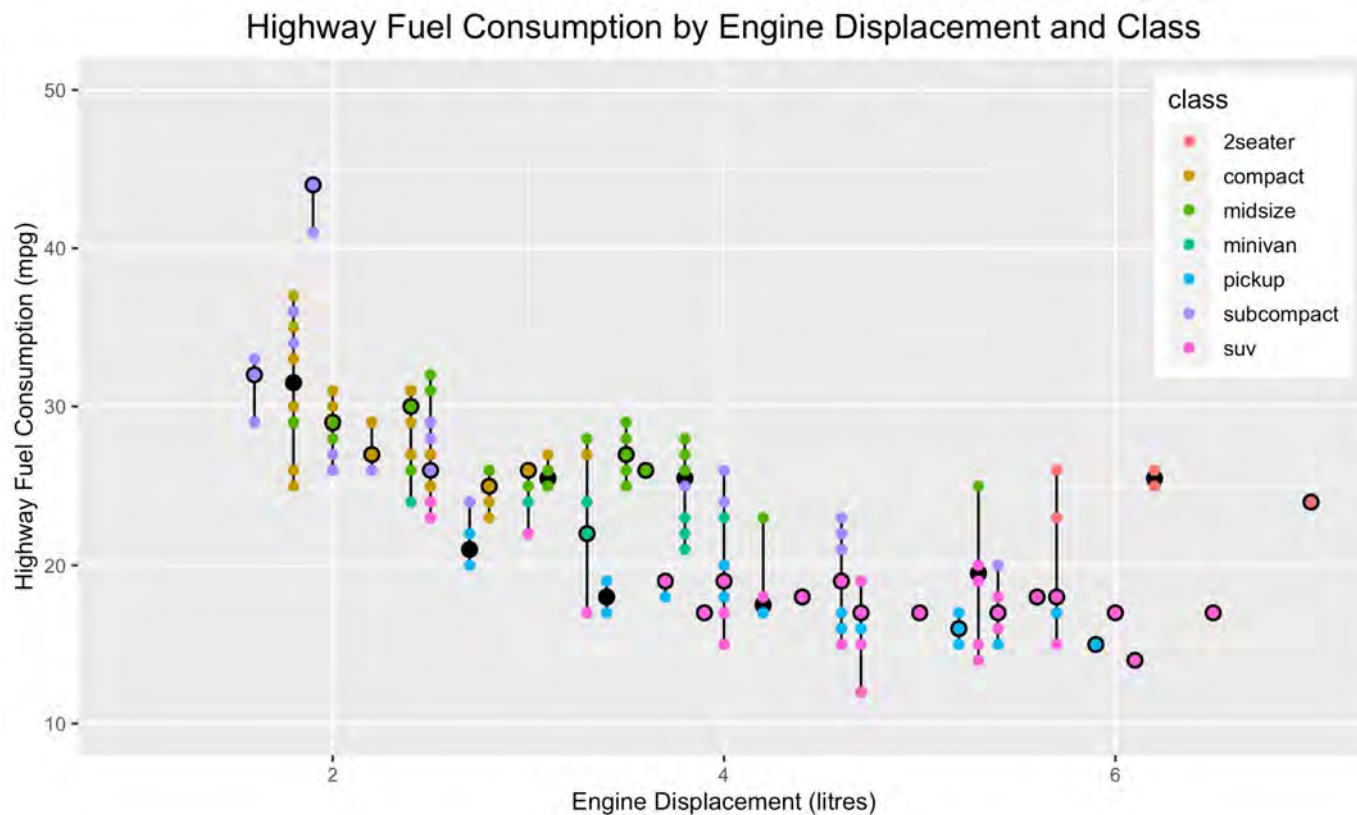


# Improving: axes, title, legend

---

```
> d = d + theme(axis.text = element_text(size = 8))
> d = d + theme(axis.title = element_text(size = 10))
> # could by axis.text.x or .y etc. to adjust separately
> d = d + xlab("Engine Displacement (litres)")
> d = d + ylab("Highway Fuel Consumption (mpg)")
> d = d + ylim(10,50) + xlim(1,7)
> d = d + theme(plot.title = element_text(size = 14))
> d = d + theme(plot.title = element_text(hjust = 0.5))
> d = d + ggtitle("Highway ... and Class")
> d = d + theme(legend.position = c(0.91, 0.76),
  legend.key.height= unit(0.5, 'cm'))
```

# Making incremental improvements by trial and error

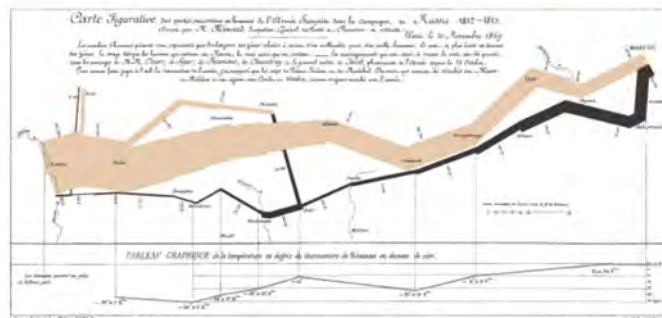


# Better graphics

---

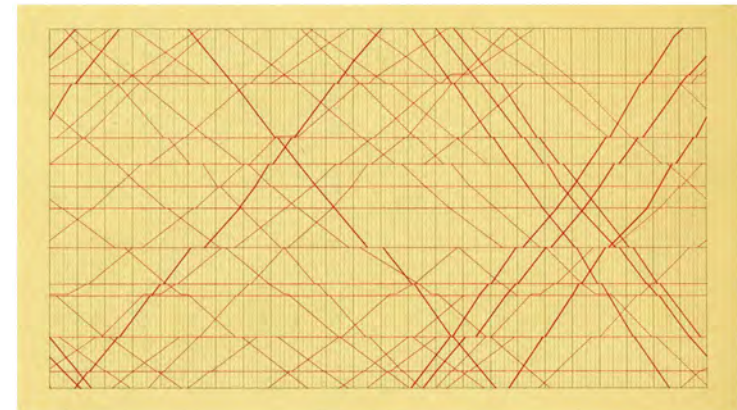
One source of inspiration is Edward Tufte:

- Read: Tufte, E. The visual display of quantitative information, Graphics Press (via Monash Library).
- A strong advocate for good information design.



Napoleon's March to Moscow The War of 1812

This figure is a historical map showing the route of Napoleon's army during the War of 1812. The map is titled 'Carte d'Aspern' and shows the route of the army from Aspern to Moscow. The map is a historical map and is not a modern map. The map is a historical map and is not a modern map.



<https://www.edwardtufte.com/tufte/>

<https://medium.com/>

# Review questions

---

- Answer in the Zoom chat if you like.



# Question 1

---

The figure is from the \_\_\_\_\_ graph family?

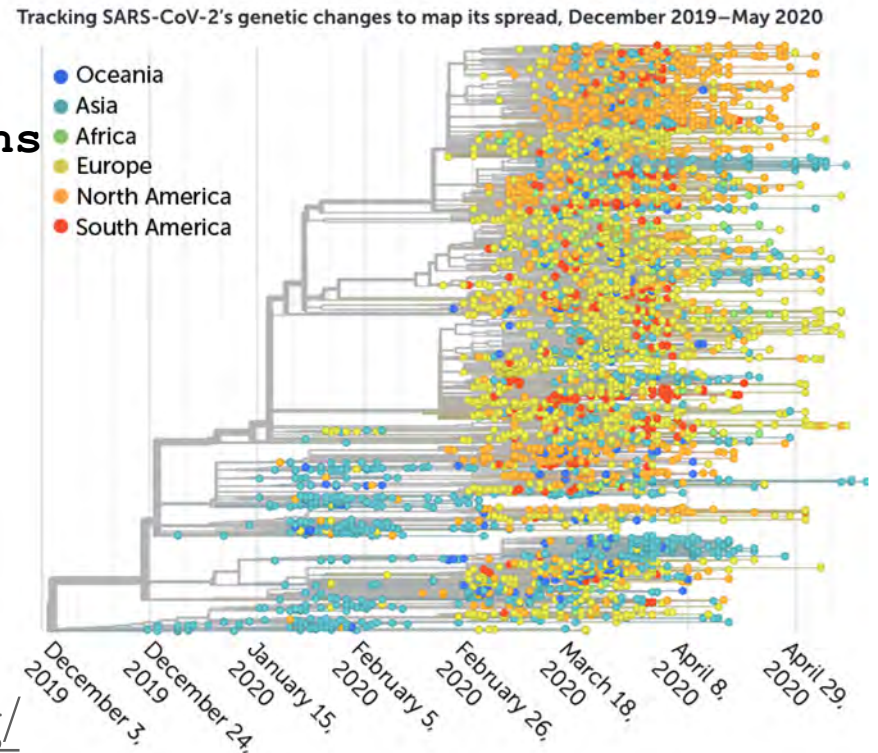
**A. Time Series**

B. Statistical Distributions

C. Maps

D. Hierarchies

E. Networks



Source: <https://www.sciencenews.org/>

# Question 2

The figure is from the \_\_\_\_\_ graph family?

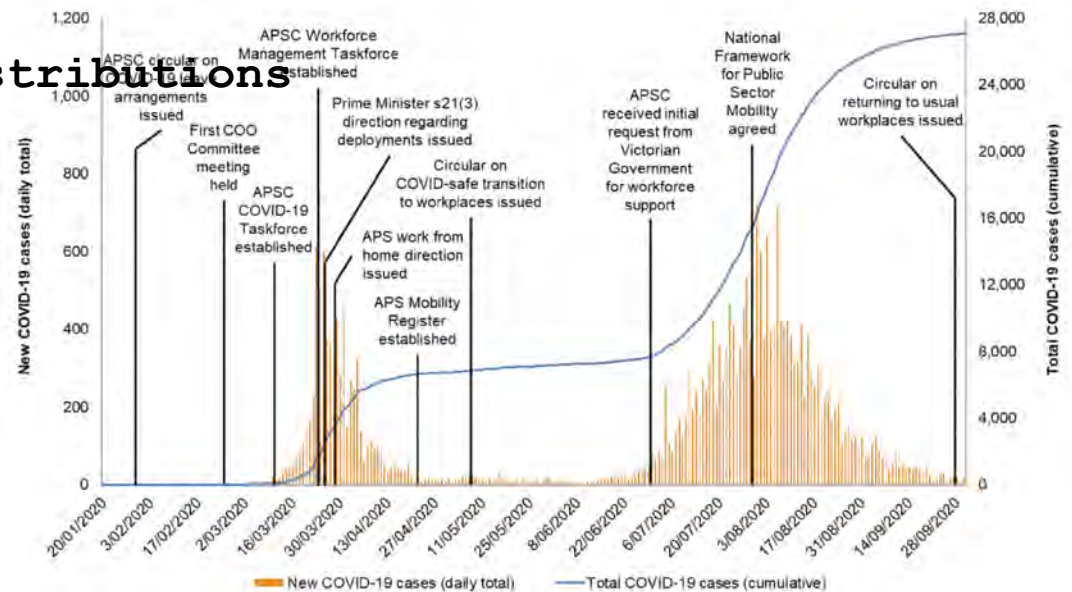
**A. Time Series**

**B. Statistical Distributions**

**C. Maps**

**D. Hierarchies**

**E. Networks**



Source: <https://www.anao.gov.au/>

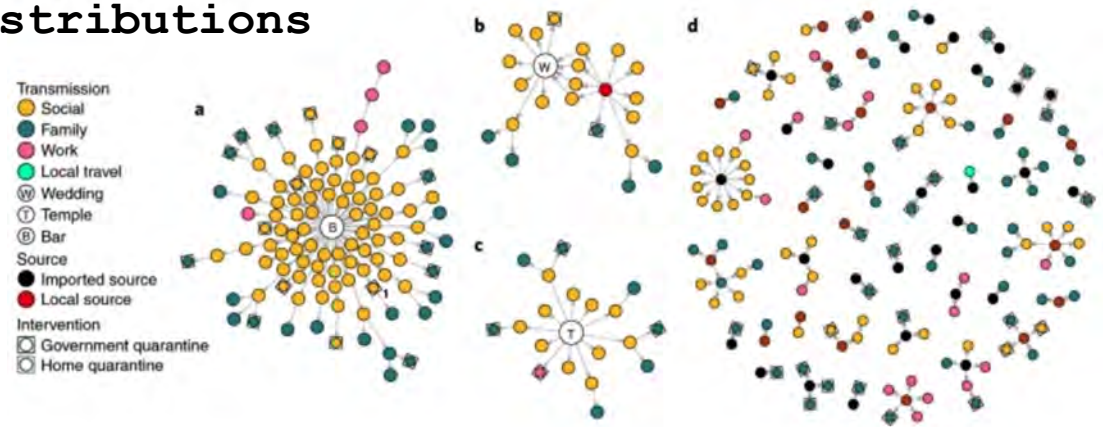
# Question 3

---

The figure is from the \_\_\_\_\_ graph family?

- A. Time Series
- B. Statistical Distributions
- C. Maps
- D. Hierarchies
- E. Networks**

**Fig. 2: Chains of SARS-CoV-2 transmission in Hong Kong initiated by local or imported cases.**



Source: <https://www.nature.com/>

# Question 4

The figure is from the \_\_\_\_\_ graph family?

A. Time Series

B. Statistical Distributions

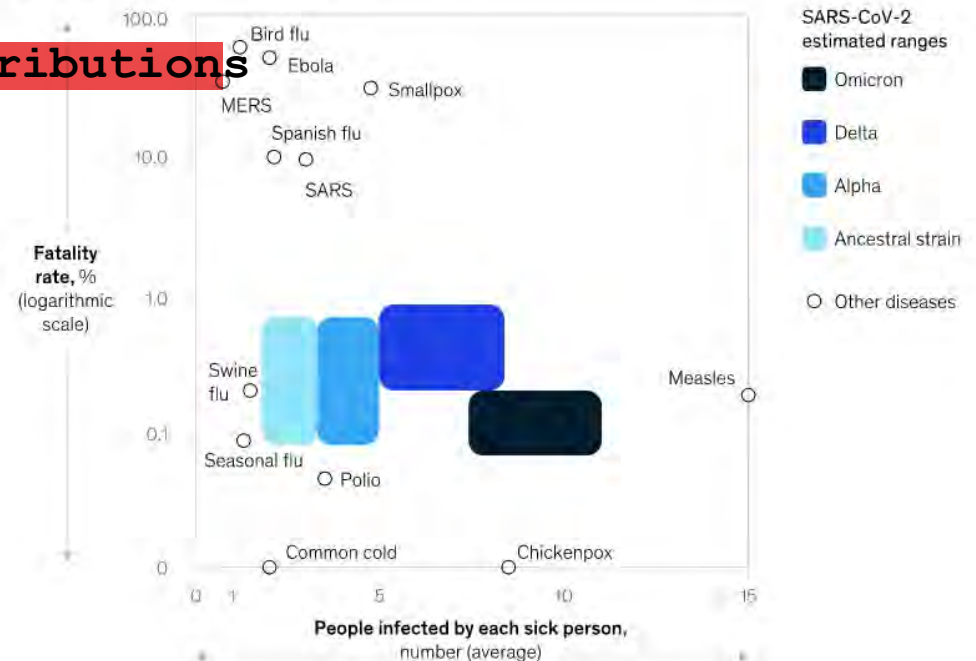
C. Maps

D. Hierarchies

E. Networks

Omicron is more infectious than other common viruses, and less fatal than Delta.

Disease fatality and infection rates<sup>1</sup>



<https://www.mckinsey.com/>

# Some R tips

---

## Scripts:

- Very important, learn how to use these now if you've not done so already.

## RMarkdown:

- Useful if you're doing a job that requires a lot of routine reporting, but not essential.

## User-Defined Functions

- Useful, and they improve your R code, but not essential. We will learn functions defined on the fly.

# Scripts

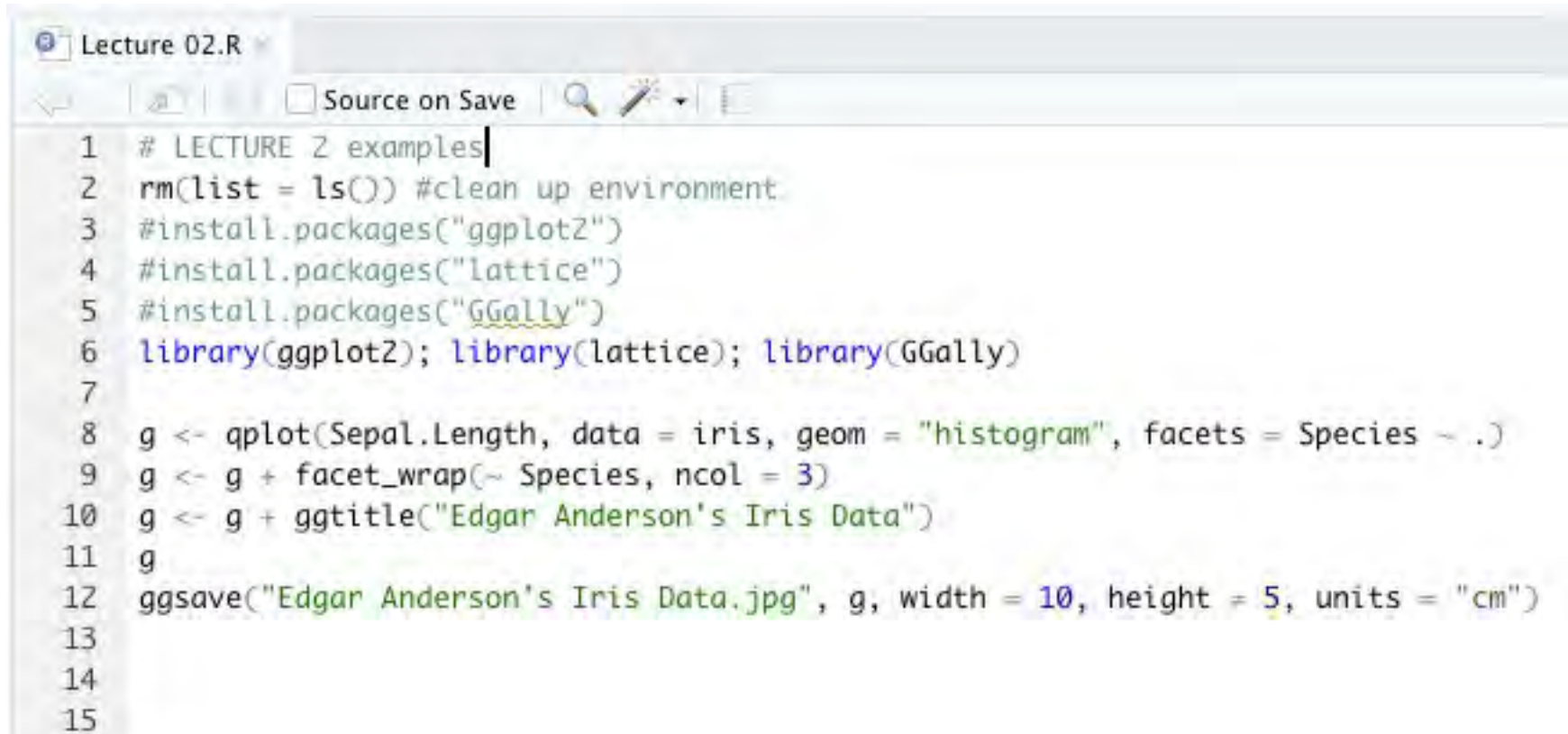
---

Scripts allow you to save your working from session to session.

- Use them to automate environment settings etc.
- Create a new script: File > New File > R Script
- Save with a filename
- Use “Source” to evaluate on the fly
- Note: # comments, pre-emptive text
- Next slide shows example from last lecture as a script...

# Scripts

---



```
Lecture 02.R
Source on Save

1 # LECTURE 2 examples
2 rm(list = ls()) #clean up environment
3 #install.packages("ggplot2")
4 #install.packages("lattice")
5 #install.packages("GGally")
6 library(ggplot2); library(lattice); library(GGally)
7
8 g <- qplot(Sepal.Length, data = iris, geom = "histogram", facets = Species ~ .)
9 g <- g + facet_wrap(~ Species, ncol = 3)
10 g <- g + ggtitle("Edgar Anderson's Iris Data")
11 g
12 ggsave("Edgar Anderson's Iris Data.jpg", g, width = 10, height = 5, units = "cm")
13
14
15
```

# R Markdown

---

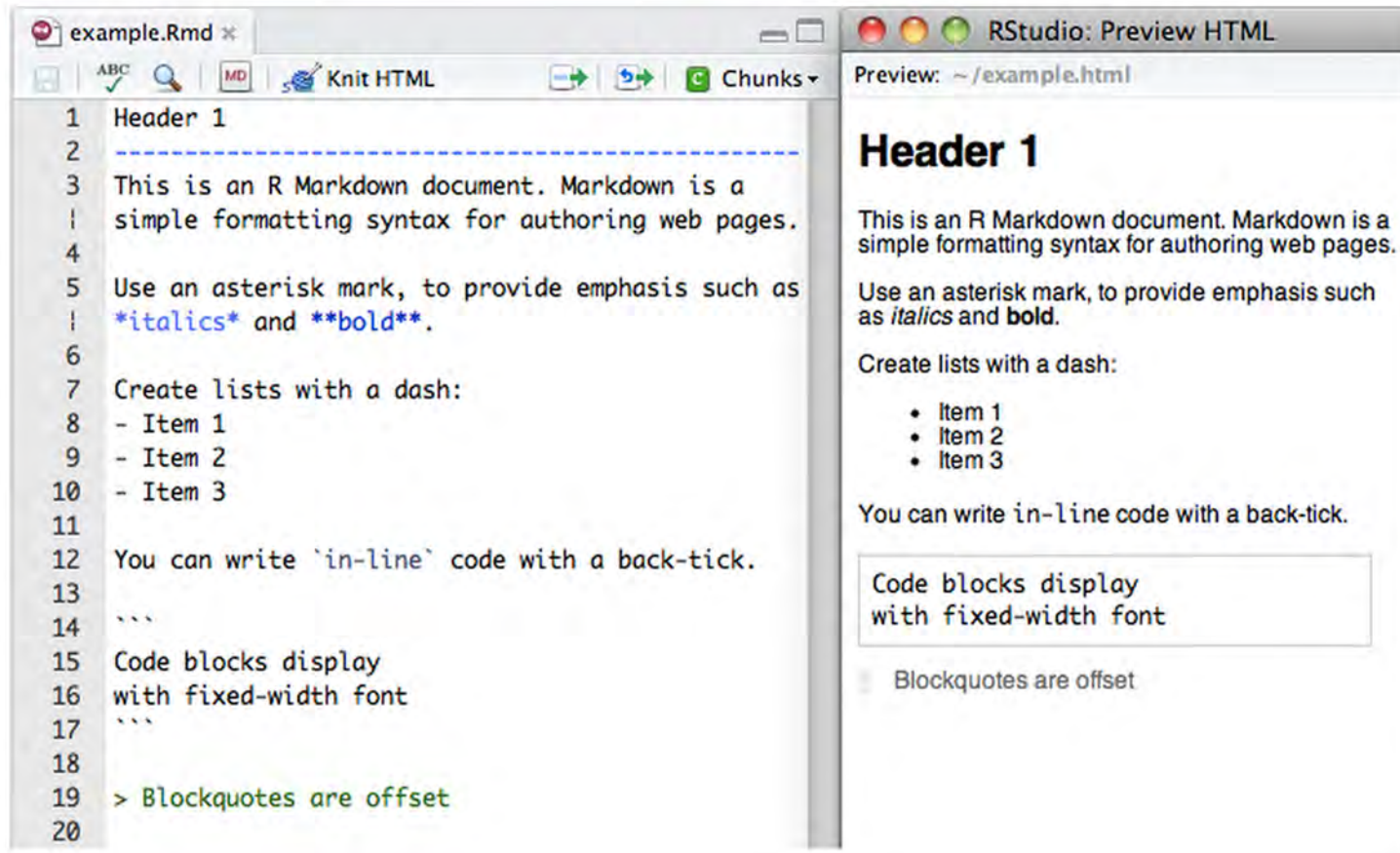
Is a package that enables the creation of HTML and PDF documents etc. based on your R session. You may choose to use it, but it is optional.

- You can embed R code and graphics.
- You can get started with R Markdown by creating a new R Markdown file in R Studio (the required files will be automatically installed).
- <http://rmarkdown.rstudio.com/>



# R Markdown

---



- <http://rmarkdown.rstudio.com/>

# Creating user-defined functions

---

It is possible to create named, user-defined, functions that can be saved between sessions using a script (see ATHR pp. 40 – 41).

Syntax:

```
> my_function <- function(arg1, arg2, ...) {  
>   object <- Calculations(arg1, arg2, ...)  
>   Return(object)  
> }
```

# Creating user-defined functions

---

Example:

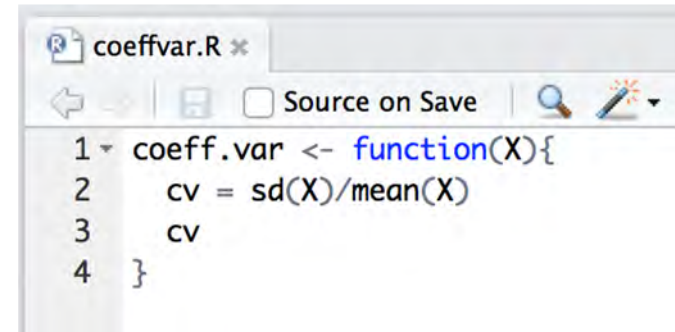
```
> coeff.var <- function(X){  
>   cv = sd(X)/mean(X)  
>   cv}  
  
> Y = c(1, 2, 3, 4, 5, 6)  
> coeff.var(Y)  
[1] 0.5345225
```

# Saving and re-using functions

---

In Rstudio:

- Create a new R script,
- Write function in script editor,
- Save as (filename.R)



To run function in a new session of R studio:

- Open and run script: code > source file (filename.R)

# Assignment 1

---

# Assignment 1: Summary

---

## FIT3152 Data analytics – 2022: Assignment 1

<b>Your task</b>	<ul style="list-style-type: none"><li>• Analyse the activity, language use and social interactions of an on-line community using metadata and linguistic summary from a real on-line forum and submit a report of your findings.</li><li>• This is an individual assignment.</li></ul>
<b>Value</b>	<ul style="list-style-type: none"><li>• This assignment is worth <b>20%</b> of your total marks for the unit.</li><li>• It has 30 marks in total.</li></ul>
<b>Suggested Length</b>	<ul style="list-style-type: none"><li>• 6 – 8 A4 pages (for your report) + extra pages as appendix (for your code)</li><li>• Font size 11 or 12pt, single spacing</li></ul>
<b>Due Date</b>	<b>11.55pm Friday 22<sup>nd</sup> April 2022</b>
<b>Submission</b>	<ul style="list-style-type: none"><li>• PDF file only. Naming convention: <i>FirstnameSecondnameID.pdf</i></li><li>• Via Moodle Assignment Submission.</li><li>• Turnitin will be used for similarity checking of all submissions.</li></ul>
<b>Late Penalties</b>	<ul style="list-style-type: none"><li>• 10% (3 mark) deduction per calendar day for up to one week.</li><li>• Submissions more than 7 calendar days after the due date will receive a mark of zero (0) and no assessment feedback will be provided.</li></ul>



# Assignment 1: Instructions

---

## Instructions

Submit the results of your analysis, answering the research questions and report anything else you discover of relevance. If you choose to analyse only a subset of your data, you should explain why.

You are expected to include at least one multivariate graphic summarising key results. You may also include simpler graphs and tables. Report any assumptions you've made in modelling, and include your R code as an appendix.

There are two options for compiling your report:

- (1) You can submit a single pdf with R code pasted in as machine-readable text as an appendix, or
- (2) As an R Markup document that contains the R code with the discussion/text interleaved. Render this as an HTML file and print off as a pdf and submit.

Regardless of which method you choose, you will submit a single pdf, and your R code will be machine readable text. We need to conform to this format as the university now requires all student submission to be processed by plagiarism detection software.

Submit your report as a single PDF with the file name ***FirstnameSecondnameID.pdf*** on Moodle.

# Assignment 1: Software

---

## Software

It is expected that you will use R for your data analysis and graphics and tables. You are free to use any R packages you need but please document these in your report and include in your R code.



# Assignment 1: Questions a & b

---

## Questions

Activity, language use and social interactions in an on-line community. Analyse the metadata and linguistic summary from a real on-line forum and submit a report of your findings. Do the following:

- (a) Analyse activity and language on the forum over time:
  - 1. How active are participants over the longer term (that is, over months and/or years)? Are there periods where activity increases or decreases? Is there a trend over time? **(3 Marks)**
  - 2. Looking at the linguistic variables, do the levels of these change over the duration of the forum? Is there a relationship between linguistic variables over the longer term? **(3 Marks)**
  
- (b) Analyse the language used by threads:

We can think of threads as groups of participants posting on the same topic.

  - 1. Using the relevant linguistic variables, is it possible to see whether or not particular threads are happier or more optimistic than other threads, or the forum in general, at different periods in time. **(3 Marks)**

# Assignment 1: Question C

---

(c) Analyse social networks online:

We can think of authors posting to the same thread at similar times (for example during the same month) as having a connection to each other, forming a social network. This is called a two-mode network. When an author posts to more than one network during the same time period their social network extends to include authors from both networks, and so on. We will cover social network analysis in Lecture 5.

1. Create a non-trivial social network of all authors who are posting over a particular time period. For example, over one month. To create this, your social network should include at least 30 authors, some of whom will have posted to multiple (2 or more) threads during this period. Your social network should be connected, although some authors may be disconnected from the main group. Present your result as a network graph. **(3 Marks)**
2. Identify the most important author in the social network you created. Looking at the language they use, can you observe any difference between them and other members of their social network? **(3 Marks)**

# Assignment 1: Overall considerations

---

(d) Overall considerations:

- The quality and clarity of your reasoning and assumptions. **(3 Marks)**
- The strength of support for your findings. **(3 Marks)**
- The quality of your writing in general and communication of results. **(3 Marks)**
- The quality of your graphics throughout, including at least one high-quality multivariate graphic. **(3 Marks)**
- The quality of your R coding. **(3 Marks)**



# Assignment 1: Data generation

---

## Data

The data is contained in the file `webforum.csv` and consists of the metadata and linguistic analysis of posts over the years 2002 to 2011. You will each work with 20,000 posts, randomly selected from the original file. The linguistic analysis was conducted using Linguistic Inquiry and Word Count (LIWC), which assesses the prevalence of certain thoughts, feelings and motivations by calculating the proportion of key words used in communication. See <http://liwc.wpengine.com/> for more information, including the language manual [http://liwc.wpengine.com/wp-content/uploads/2015/11/LIWC2015\\_LanguageManual.pdf](http://liwc.wpengine.com/wp-content/uploads/2015/11/LIWC2015_LanguageManual.pdf)

Create your individual data as follows:

```
rm(list = ls())
set.seed(XXXXXXXX) # XXXXXXXX = your student ID
webforum <- read.csv("webforum.csv")
webforum <- webforum [sample(nrow(webforum), 20000), ] # 20000 rows
```

# Assignment 1: Data fields

---

Data fields given. (see the language manual for more detail and examples):

Column	Brief Descriptor	Column	Brief Descriptor
ThreadID	Unique ID for each thread	we	"We, us, our" words
AuthorID	Unique ID for each author	you	"You" words
Date	Date	shehe	"She, her "him words
Time	Time	they	"They" words
WC	Word count of the text of the post	posemo	Expressing positive emotions
Analytic	Summary: Analytical thinking	negemo	Expressing negative emotions
Clout	Summary: Power, force, impact	anx	Indicating anxiety
Authentic	Summary: Authentic tone of voice	anger	Indicating anger
Tone	Summary: Emotional tone	sad	Indicating sadness
ppron	"I, we, you" words	focuspast	Expressing a focus on the past
i	"I, me, mine" words	focuspresent	Expressing a focus on the present
focusfuture	Expressing a focus on the future	focusfuture	Expressing a focus on the future

# Assignment 1: Data extract

---

ThreadID	AuthorID	Date	Time	WC	Analytic	Clout	Authentic	Tone	ppron	i	we	you	shehe	they	...
144564	41084	9/8/04	4:46	134	55.23	69.94	63.91	68.05	7.46	2.99	2.24	1.49	0	0.75	...
404119	128515	21/7/07	22:27	12	1	79.76	74.76	25.77	33.33	8.33	0	0	0	25	...
395992	93243	19/6/07	1:02	28	13.85	76.25	1.06	99	7.14	3.57	0	3.57	0	0	...
405421	99958	24/7/07	1:40	16	84.57	89.42	35.37	1	6.25	0	0	6.25	0	0	...
662470	185647	5/12/09	16:05	37	32.06	79.13	21.26	75.85	18.92	8.11	0	0	5.41	5.41	...
420058	53655	13/9/07	22:59	17	26.21	3.89	99	1	11.76	5.88	0	0	0	5.88	...
13933	1740	9/3/02	2:01	61	22.35	37.15	72.51	25.77	11.48	6.56	1.64	0	0	3.28	...
245087	80190	9/11/05	15:06	94	82.45	66.48	44.79	25.77	4.26	2.13	1.06	0	0	1.06	...
442550	47686	6/12/07	5:06	80	61.95	54.96	59.88	96.76	7.5	5	0	1.25	0	1.25	...
352716	26979	5/1/07	21:33	10	8.19	84.14	1	25.77	0	0	0	0	0	0	...
463617	104430	29/2/08	8:02	249	98.57	78.92	15.3	83.06	3.61	0.8	1.61	0	0.8	0.4	...
363541	-1	15/2/07	11:30	26	53.63	87.57	38.39	99	11.54	3.85	0	7.69	0	0	...
258941	44297	1/1/06	13:47	59	94.34	91.23	10.76	6.73	8.47	1.69	1.69	5.08	0	0	...
765163	54960	17/12/10	21:06	139	26.01	58.53	13.52	66.61	7.91	1.44	0.72	2.88	0	2.88	...
263152	79878	18/1/06	7:34	114	48.42	73.03	9.58	1	10.53	4.39	0	2.63	0	3.51	...
228773	166362	6/9/09	4:52	14	13.85	98.33	89.63	25.77	14.29	0	0	14.29	0	0	...
254482	83344	6/1/06	0:17	107	80.6	77.26	24.3	1	2.8	0.93	0	0.93	0	0.93	...
255544	81721	17/12/05	21:46	166	98.84	45.21	34.91	17.07	1.2	0	0.6	0.6	0	0	...
218880	22130	18/7/05	5:07	11	12.85	81.84	99	1	18.18	9.09	0	9.09	0	0	...
244912	41084	8/11/05	2:46	35	99	38.74	13.15	98.56	0	0	0	0	0	0	...
273089	-1	25/2/06	4:22	92	90.46	58.59	68.63	11.64	8.7	2.17	1.09	0	5.43	0	...
265715	38794	2/2/06	0:57	275	81.4	69.47	29.78	20.28	6.55	2.91	0.73	0.73	1.09	1.09	...
198321	21367	17/4/05	22:23	110	54.02	89.83	14.1	94.75	10.91	5.45	0	1.82	0.91	2.73	...
45244	13359	21/12/02	18:01	45	92.84	81.29	10.08	67.75	8.89	4.44	0	0	0	4.44	...
233103	70832	1/10/05	9:19	77	95.05	69.84	65.41	97.38	2.6	0	0	1.3	0	1.3	...
566748	109818	25/3/09	5:25	77	89.94	74.2	9.09	99	2.6	0	1.3	0	0	1.3	...
146671	116703	24/1/07	7:25	38	33.88	1.81	98.54	74.74	7.89	7.89	0	0	0	0	...
745917	105443	1/11/10	6:46	242	27.37	38.61	93.65	6.99	12.81	8.26	1.24	2.48	0	0.83	...
618782	165386	11/7/09	2:46	119	55.71	50	10.42	1	3.36	0.84	0	0	0.84	1.68	...
55689	19796	10/2/03	2:07	12	1	20.24	98.01	25.77	16.67	16.67	0	0	0	0	...
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

# Data manipulation

---

# Summarizing data by groups

---

Data grouped by factors:

- Applying a function to a single column
- Applying a function to a group of columns

Why do we need to do this?

- To simplify the data, making comparisons easier
- Reduce data complexity, enabling further analysis



# Edgar Anderson's Iris data

---

50 samples from 3 species:

- Iris setosa, – virginica, – versicolor

Four features measured:

- Sepal width and length
- Petal width and length

Is it possible to distinguish species using physical measurements?

- Data is packaged with R: “iris”

[http://en.wikipedia.org/wiki/Iris\\_flower\\_data\\_set](http://en.wikipedia.org/wiki/Iris_flower_data_set)



# Print

---

```
> iris # = print(iris)
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa
11	5.4	3.7	1.5	0.2	setosa
12	4.8	3.4	1.6	0.2	setosa
13	4.8	3.0	1.4	0.1	setosa
14	4.3	3.0	1.1	0.1	setosa
15	5.8	4.0	1.2	0.2	setosa
...					

# Two challenges

---

## (1) Easy!

- Create a table of column means grouped by species.

## (2) Harder!

- Create a CSV file containing the correlation between sepal length and sepal width as well as petal length and petal width for each species.

# High level view

---

Data analysis is easier if you have a high-level view of the data:

- 4 columns + 1 factor (Species)
- Two pairs of related columns: sepals & petals

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Setosa
Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Virginica
Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Versicolor

# Challenge 1. Function: aggregate

---

The 'aggregate' function creates a table by applying a function to data in individual columns grouped by a factor (or factors). To calculate averages:

- Note: columns referred to by their index (number) [ ] for compactness

```
> aggregate(iris[1:4], iris[5], mean)
```

	Species	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
1	setosa	5.01	3.43	1.46	0.246
2	versicolor	5.94	2.77	4.26	1.326
3	virginica	6.59	2.97	5.55	2.026

# ?aggregate

---

- Description

`aggregate(x, ...)` : Splits the data into subsets, computes summary statistics for each, and returns the result in a convenient form.

- Usage

`aggregate(x, by, FUN, ..., simplify = TRUE)`

- Arguments

**X** : An R object.

**By** : List of grouping elements

**FUN** : Function to compute the summary statistics

**Simplify** : Indicates whether results should be simplified to a vector or matrix if possible.

# Challenge 2. Function: by

---

‘by’ enables a function to be applied across **individual or multiple columns** of a data frame grouped by a factor or factors.

- To calculate the correlation of sepal length and width

```
> by(iris, iris[5], function(df) cor(df$Sepal.Length,
df$Sepal.Width))
Species: setosa
[1] 0.743
Species: versicolor
[1] 0.526
Species: virginica
[1] 0.457
```

# ?by

---

- Description

Apply a Function to a Data Frame Split by Factors

- Usage

`by(data, INDICES, FUN, ..., simplify = TRUE)`

- Arguments

**Data** : an R object, normally a data frame, possibly a matrix.

**INDICES** : a factor or a list of factors, each of length `nrow(data)`.

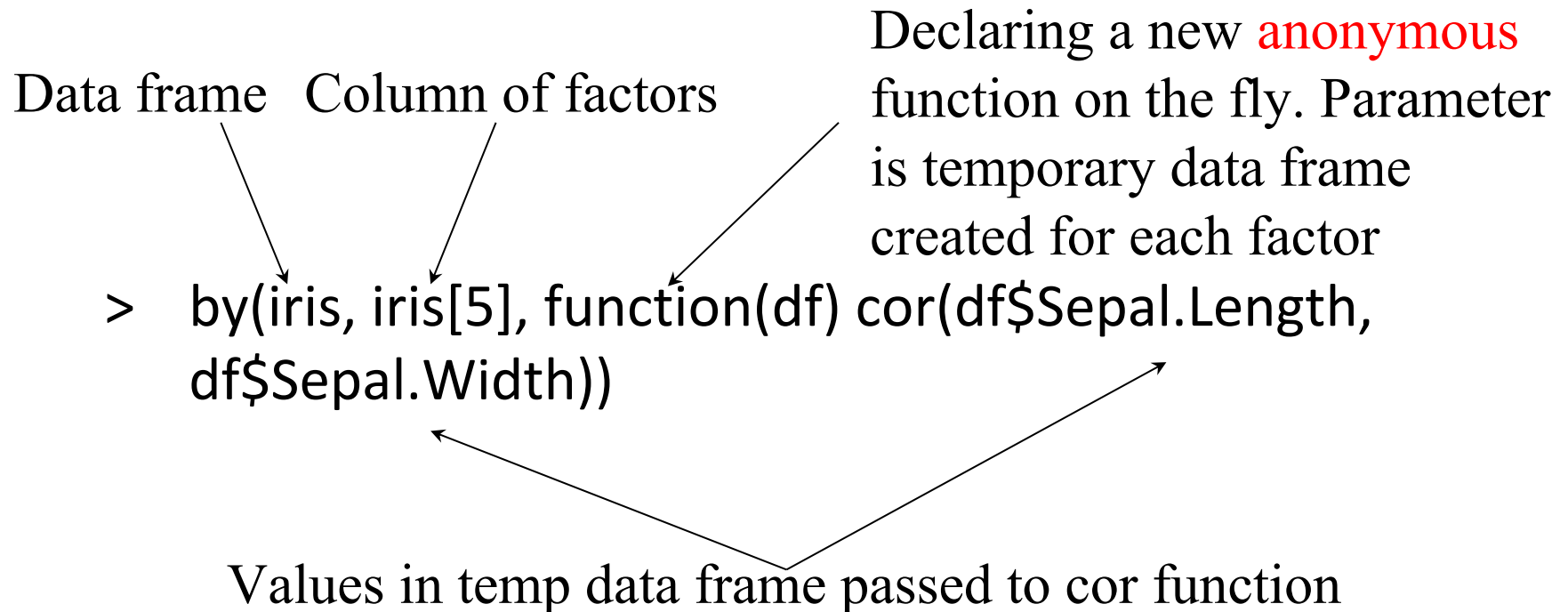
**FUN** : a function to be applied to data frame subsets of data...



# ?by: applying the cor function

---

Looking more closely at the way correlation is calculated:



# Anonymous functions

---

If a function is only to be used once, it can be defined when it is used. These are anonymous functions (having no name) see ATHR p.41.

Example (sum the sepal length by species)

```
> by(iris, iris[5], function(df) sum(df$Sepal.Length))  
Species: setosa  
[1] 250.3  
Species: versicolor  
[1] 296.8  
Species: virginica  
[1] 329.4
```

...

---

Changing earlier example to a more compact notation, using column indexes.

From:

```
> by(iris, iris[5], function(df) cor(df$Sepal.Length,  
  df$Sepal.Width))
```

To:

```
> by(iris, iris[5], function(df) cor(df[1], df[2]))
```

# Function: as.table

---

This function converts the output format of a function from a list to a table

```
> as.table(by(iris, iris[5], function(df) cor(df[1], df[2])))
```

**Species**

<b>setosa</b>	<b>versicolor</b>	<b>virginica</b>
0.743	0.526	0.457

# Function: as.data.frame

---

This function converts “coerces” the output of a table into a data frame

```
> Sepal.cor <- as.data.frame(as.table(by(iris, iris[5],  
function(df) cor(df[1], df[2]))))
```

```
> Sepal.cor
```

	Species	Freq
1	setosa	0.743
2	versicolor	0.526
3	virginica	0.457

# Function: colnames

---

This function assigns new column names to a data frame.

```
> colnames(Sepal.cor) <- c("Species", "Sepal.cor")
```

```
> Sepal.cor
```

	<b>Species</b>	<b>Sepal.cor</b>
1	<b>setosa</b>	<b>0.743</b>
2	<b>versicolor</b>	<b>0.526</b>
3	<b>virginica</b>	<b>0.457</b>

# Now for petals...

---

Repeating the previous code for petals...

```
> Petal.cor <- as.data.frame(as.table(by(iris, iris[5],  
  function(df) cor(df[3], df[4]))))  
> colnames(Petal.cor) <- c("Species", "Petal.cor")  
> Petal.cor
```

	Species	Petal.cor
1	setosa	0.332
2	versicolor	0.787
3	virginica	0.322

# Merging data frames (and saving)

---

Using a common column – “Species” – and rounding data. *Note: we could have used cbind – since the two data frames are aligned.*

```
> iris.cor <- merge(Sepal.cor, Petal.cor, by = "Species")
> iris.cor[,2] = round(iris.cor[,2], digits = 3)
> iris.cor[,3] = round(iris.cor[,3], digits = 3)
> write.csv(iris.cor, file = "Iris.cor.csv",
  row.names=FALSE)
```



# The saved file

---

SepalPetalcor.csv

Species	Sepal.cor	Petal.cor
setosa	0.743	0.332
versicolor	0.526	0.787
virginica	0.457	0.322

This gives a much more compact presentation of the main correlations compared to the table created last lecture using:

```
> by(iris[1:4], factor(iris$Species), cor)
```

# Correlation matrix – by... factor

---

From last week: Pairwise correlation by species

```
> by(iris[1:4], factor(iris$Species), cor)
```

```
factor(iris$Species): setosa
      Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length      1.0000000    0.7425467    0.2671758    0.2780984
Sepal.Width        0.7425467    1.0000000    0.1777000    0.2327520
Petal.Length       0.2671758    0.1777000    1.0000000    0.3316300
Petal.Width        0.2780984    0.2327520    0.3316300    1.0000000
-----
factor(iris$Species): versicolor
      Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length      1.0000000    0.5259107    0.7540490    0.5464611
Sepal.Width        0.5259107    1.0000000    0.5605221    0.6639987
Petal.Length       0.7540490    0.5605221    1.0000000    0.7866681
Petal.Width        0.5464611    0.6639987    0.7866681    1.0000000
-----
factor(iris$Species): virginica
      Sepal.Length Sepal.Width Petal.Length Petal.Width
Sepal.Length      1.0000000    0.4572278    0.8642247    0.2811077
Sepal.Width        0.4572278    1.0000000    0.4010446    0.5377280
Petal.Length       0.8642247    0.4010446    1.0000000    0.3221082
Petal.Width        0.2811077    0.5377280    0.3221082    1.0000000
```

# dplyr



If some of the manipulation we've done so far looks a bit intimidating, you might want to try the 'dplyr' package. It:

- is a *Grammar of Data* manipulation,
- provides a consistent set of verbs to simplify the most common data manipulation challenges.
- <https://dplyr.tidyverse.org/>
- See Chapter 5, Data Transformation in R for Data Science <https://r4ds.had.co.nz/>

# dplyr



dplyr is a grammar of data manipulation, providing a consistent set of verbs that help you solve the most common data manipulation challenges:

- `mutate()` adds new variables that are functions of existing variables
- `select()` picks variables based on their names.
- `filter()` picks cases based on their values.
- `summarise()` reduces multiple values down to a single summary.
- `arrange()` changes the ordering of the rows.

These all combine naturally with `group_by()` which allows you to perform any operation “by group”. You can learn more about them in `vignette("dplyr")`. As well as these single-table verbs, dplyr also provides a variety of two-table verbs, which you can learn about in `vignette("two-table")`.

From: <https://dplyr.tidyverse.org/>

# dplyr



## Quick start:

- Use pipes, `%>%`, to connect data to a grouping variable, and then apply a function.
- For example, to find average sepal length by species::  

```
> iris %>% group_by(Species) %>%  
  summarise(Ave.Sepal.len = mean(Sepal.Length))
```

```
# A tibble: 3 × 2  
  Species Ave.Sepal.len  
  <fct>      <dbl>  
1 setosa      5.01  
2 versicolor  5.94  
3 virginica   6.59
```

# dplyr



## Tibbles...

- Dplyr creates tibbles instead of data frames. To get an overview of the difference between these, see:
- <https://r4ds.had.co.nz/tibbles.html>
- You can convert a tibble to a data frame if preferred using:
  - > `NewDataFrame = as.data.frame(TibbleName)`

# dplyr (Challenge 1)



- For Challenge 1, find column means by species:  
> iris %>% group\_by(Species) %>% summarise(ASL = mean(Sepal.Length), ASW = mean(Sepal.Width), APL = mean(Petal.Length), APW = mean(Petal.Width))

```
# A tibble: 3 × 5
  Species      ASL      ASW      APL      APW
  <fct>      <dbl> <dbl> <dbl> <dbl>
1 setosa      5.01   3.43   1.46  0.246
2 versicolor  5.94   2.77   4.26  1.33
3 virginica   6.59   2.97   5.55  2.03
```

# dplyr (Challenge 2)



- For Challenge 2, find the correlation between sepal length and width, and petal length and width by species – first step is shown below:  
> iris %>% group\_by(Species) %>% summarise(Sepal.cor = cor(Sepal.Length, Sepal.Width))

```
# A tibble: 3 × 2
  Species      Sepal.cor
  <fct>      <dbl>
1 setosa      0.743
2 versicolor 0.526
3 virginica   0.457
```



# Two more challenges

---

## (3) Easy!

- Examine the difference between the aspect ratios (Length / Width) for sepals and petals between the different species.

## (4) Harder!

- Report the data for the flower having the longest petal in each species.

# Challenge 3: Add/remove columns

---

By default, R will add a new column to a data frame if the output of a column operation is specified as a new column.

Alternatively, the `cbind` function can be used to append a vector or data frame by columns.

This lets us store the results of row operations, including factor generation.

# Making new columns

---

Add two columns containing the aspect ratio (length/width) for sepals and petals:

- > `niris <- iris # creating a new data frame`
- > `niris$Sepal.ar <- niris$Sepal.Length/niris$Sepal.Width`  
`# add new column`
- > `niris$Petal.ar <- niris$Petal.Length/niris$Petal.Width #`  
`add new column`
- > `head(niris)`

---

## The augmented data frame: niris

```
> head(niris)
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species	Sepal.ar	Petal.ar
1	5.1	3.5	1.4	0.2	setosa	1.46	7.00
2	4.9	3.0	1.4	0.2	setosa	1.63	7.00
3	4.7	3.2	1.3	0.2	setosa	1.47	6.50
4	4.6	3.1	1.5	0.2	setosa	1.48	7.50
5	5.0	3.6	1.4	0.2	setosa	1.39	7.00
6	5.4	3.9	1.7	0.4	setosa	1.38	4.25

# Deleting columns

---

This is easy – but cannot be undone!

To remove a single column, do it by name.

To remove the first column:

```
> niris$Sepal.Length <- NULL
```

Tedious for multiple columns. A quicker but potentially dangerous way to remove first 4 columns:

```
> niris <- niris[,c(5:7)] # reassign cols 5:7 on to self!
```

---

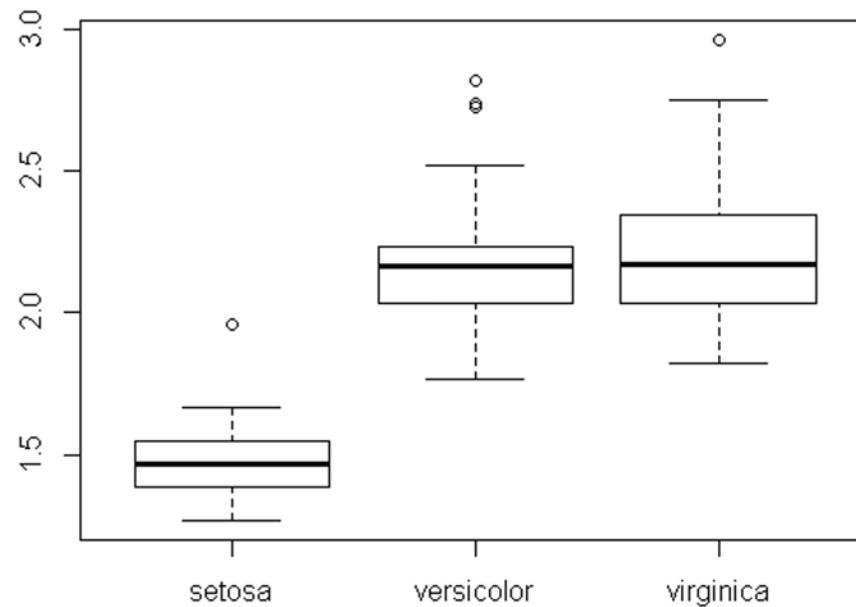
## After removing columns:

```
> head(niris)
```

	Species	Sepal.ar	Petal.ar
1	setosa	1.46	7.00
2	setosa	1.63	7.00
3	setosa	1.47	6.50
4	setosa	1.48	7.50
5	setosa	1.39	7.00
6	setosa	1.38	4.25

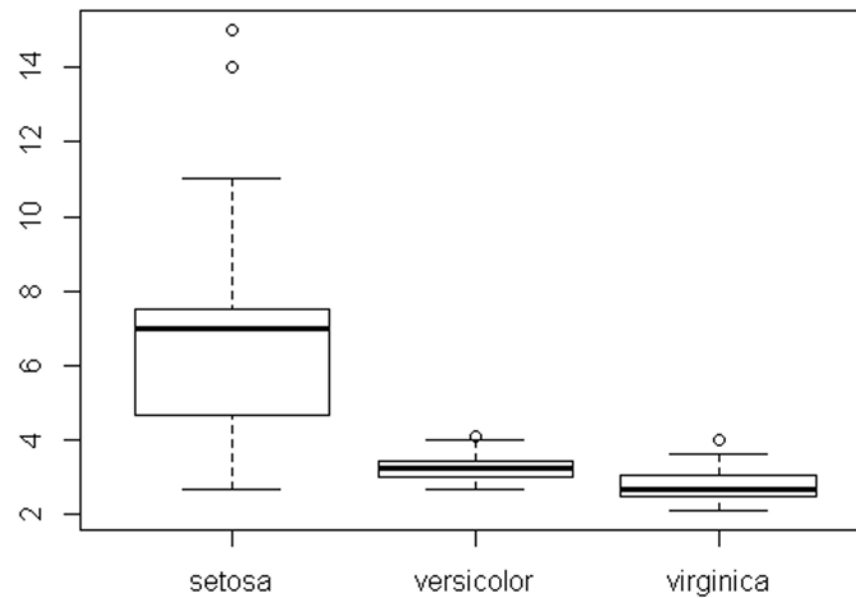
---

> `boxplot(Sepal.ar~Species, data = niris)`



---

> `boxplot(Petal.ar~Species, data = niris)`

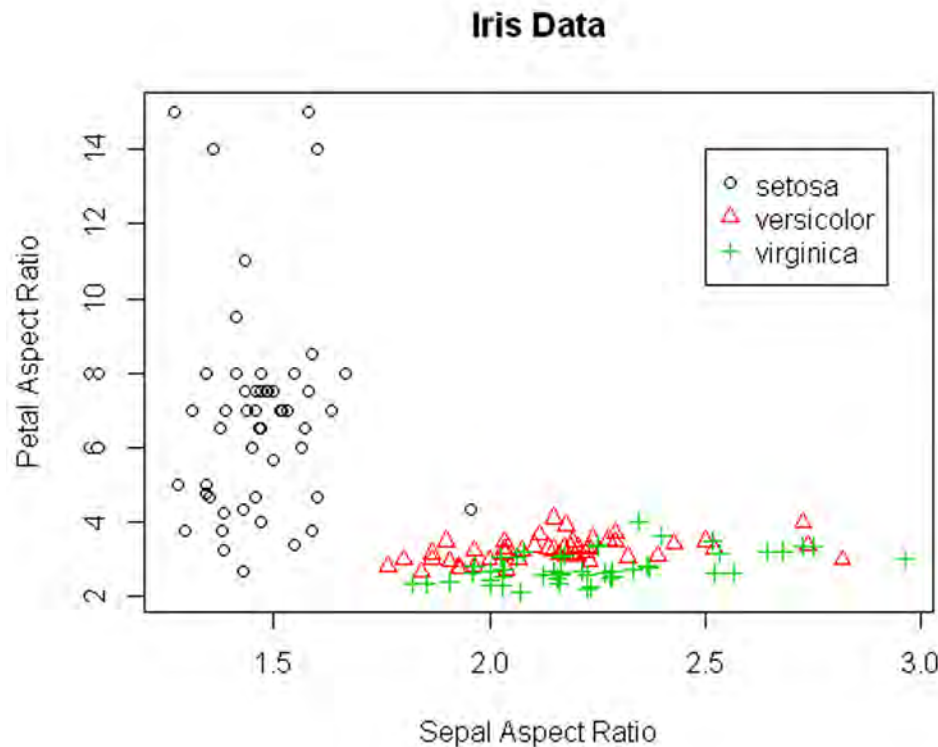




# Scatterplot

---

## Petal vs Sepal aspect ratio (Length / Width)



---

## Code for scatterplot on previous slide:

- > with(niris, plot(Sepal.ar, Petal.ar, col = Species, pch=as.numeric(Species), main = ("Iris Data"), xlab = "Sepal Aspect Ratio", ylab = ("Petal Aspect Ratio")))
- > with(niris, legend(2.5, 14, as.vector(unique(Species)), pch=unique(Species), col = unique(Species)))

# Challenge 4: using dplyr



This task shows off how useful dplyr is:

To find the flower having the longest petal in each species:

```
> iris %>% group_by(Species) %>% top_n(1, Petal.Length)
```

```
# A tibble: 4 × 5
```

```
# Groups:   Species [3]
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
	<dbl>	<dbl>	<dbl>	<dbl>	<fct>
1	4.8	3.4	1.9	0.2	setosa
2	5.1	3.8	1.9	0.4	setosa
3	6	2.7	5.1	1.6	versicolor
4	7.7	2.6	6.9	2.3	virginica

Results show two setosa flowers having equally long petals.

# Challenge 4: using base R functions

---

The following slides show how a similar outcome is achieved using the base R functions and introduces important functions such as `max`, `which.max` and `rbind`.

Please read and work through the example in your own time.

# Challenge 4: using base R functions

---

One way to find the longest petal in each species is to apply the ‘aggregate’ function to the Petal.Length column of the dataframe:

```
> aggregate(iris[3], iris[5], max)
```

	Species	Petal.Length
1	setosa	1.9
2	versicolor	5.1
3	virginica	6.9

Suppose we want find the flower having the longest petal and report all its measurements.

# ?which.max

---

- Description

Determines the location, i.e., index of the (first) minimum or maximum of a numeric vector.

- Usage

`which.min(x)`

`which.max(x)`

- Arguments

**x** : numeric (integer or double) vector, whose min or max is searched for.

# Printing the row with the longest petal

---

To find the row containing the longest petal (ignoring species), use:

```
> which.max(iris[,3])  
[1] 119
```

To print this row use:

```
> iris[which.max(iris[,3]),]
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
119	7.7	2.6	6.9	2.3	virginica

To find the maximum for each species use ‘by’ function and a temporary data frame.

# Longest petal in each group

---

Putting together gives:

```
> by(iris, iris[5], function(df) df[which.max(df[,3]),])
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
25	4.8	3.4	1.9	0.2	setosa
-----					
	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
84	6	2.7	5.1	1.6	versicolor
-----					
	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
119	7.7	2.6	6.9	2.3	virginica

We will now tidy up the output...



# ?do.call

---

- Description

`do.call` constructs and executes a function call from a name or a function and a list of arguments to be passed to it.

- Usage

```
do.call(what, args, quote = FALSE, envir =  
parent.frame())
```

- Arguments

`what` : function or string naming the function

`args` : a list of arguments to the function call.

`quote` : indicating whether to quote the arguments.

`envir` : an environment within to evaluate the call.

# ?rbind

---

- Description

Take a sequence of vector, matrix or data frames arguments and combine by columns or rows.

- Usage

```
cbind(..., deparse.level = 1)
```

```
rbind(..., deparse.level = 1)
```

- Arguments

`...` : vectors or matrices. These can be given as named arguments.

`deparse.level` : (ignore at this stage)

# Tidying the output

---

First, assign a variable name (for clarity):

```
> max.type <- by(iris, iris[5], function(df)  
  df[which.max(df[,3]),])
```

```
> do.call(rbind, max.type)
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
setosa	4.8	3.4	1.9	0.2	setosa
versicolor	6.0	2.7	5.1	1.6	versicolor
virginica	7.7	2.6	6.9	2.3	virginica

This can be converted to a data frame using:

```
> XX <- as.data.frame(do.call(rbind, max.type))
```

# Working with dates and times

---

To work with dates in R, you first need to convert the character representation of date into a ‘date’ object using the ‘as.Date’ function so that data is read and interpreted correctly.

- We will do this before applying the ‘which.min’ and ‘which.max’ functions to a date.
- The Dunnhumby data (Tutorial 2) records the sale date and amount spent by 20 customers.
- Find the earliest sale date for each customer.

# Dunnhumby : data

---

customer_id	visit_date	visit_delta	visit_spend
40	4/04/10	NA	44.83
40	6/04/10	2	69.68
40	19/04/10	13	44.61
40	1/05/10	12	30.39
40	2/05/10	1	60.73
40	12/05/10	10	50
40	15/05/10	3	3
40	18/05/10	3	36.89
40	19/05/10	1	9.07
40	23/05/10	4	14.01
40	26/05/10	3	16.97
40	31/05/10	5	8.69
...	...	...	...

# Without date conversion

---

Calculating minimums without date conversion:

```
> min.type <- by(DH, DH[1], function(df)
  df[which.min(df[,2]),])
> do.call(rbind,min.type)
```

.	customer_id	visit_date	visit_delta	visit_spend
40	40	01-05-10	12	30.39
79	79	01-01-11	9	81.70
119	119	01-03-11	3	10.69
123	123	01-02-11	4	35.20
134	134	01-02-11	1	54.77

# With date conversion

---

## Calculating minimums with date conversion:

```
> min.type <- by(DH, DH[1], function(df)
  df[which.min(as.Date(df[,2], "%d-%m-%y")),])
> do.call(rbind, min.type)
```

.	customer_id	visit_date	visit_delta	visit_spend
40	40	04-04-10	NA	44.83
79	79	07-04-10	NA	150.87
119	119	01-04-10	NA	20.00
123	123	02-04-10	NA	66.94
134	134	01-04-10	NA	50.32

# Summary

---

## Summarizing data using factors using

- Base functions: `aggregate`, `by`
- dplyr package functions: `group_by`, `summarise`.

## Creating and removing columns

## Searching, indexing and combining rows

- dplyr functions: `group_by`, `top_n`.
- Base functions: `as.table`, `as.data.frame`, `colnames`, `which.max`, `do.call`, `rbind`, `cbind`.



# Answers to the review questions

---

1. D: Hierarchy (Phylogenetic tree)
2. A: Time Series (Daily infections)
3. E: Network (Transmission network)
4. B: Statistical (Scatter (Bubble) plot)

# References

---

## Books – online from the Monash Library

- Spector, P., Data manipulation with R. (pp 113 – 118 used as a reference for last part of today's lecture)
- Wickham, H., ggplot2: elegant graphics for data analysis.

dplyr Cheat Sheet <https://github.com/rstudio/cheatsheets>

R Reference card (Tom Short) available from contributed documentation on CRAN site.

<http://cran.r-project.org/>