FIT3152 Data analytics – Lecture 2

Visualisation of data

- Recent examples
- Common themes

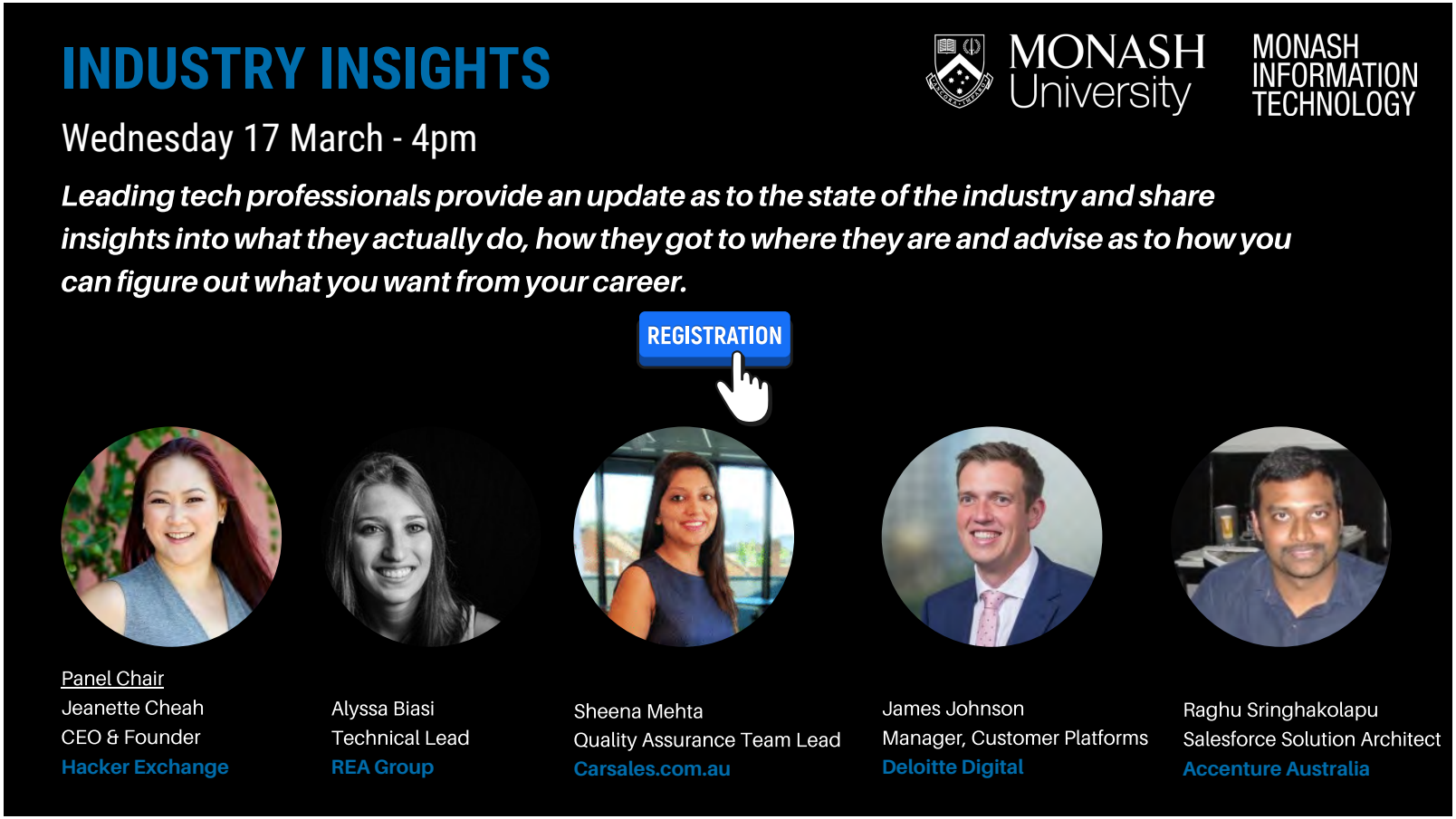
Visualisation using R

- First steps: looking at the data
- Visualization for analysis
- Presentation quality graphics

Recruiter - ask me anything event

https://forms.gle/3u8cuE5koDVCH7Rb9

Industry insights panel event



https://forms.gle/36Wqt2hdYPQkxbw59

Quick review of last week:

What is data science?

Overview of the unit

Using R

A few quick review questions:

You can answer in the Zoom chat if you want

Question 1

Predict the output from the following commands:

```
> X <- c(9, 16)
> sqrt(X)
```

A. 3

B. 3, 4

C. 4

D. 7

Question 2

Predict the output from the following commands:

```
> X <- c(1, 2)
> Y <- c(3, 4)
> X + Y
```

A. 4, 6
B. 3, 7
C. 10
D. 1, 2, 3, 4

Question 3

Predict the output from the following commands:

```
> X <- c(1, 2)
> Y <- c(3, 4)
> X * Y
```

A. 3, 8
B. 2, 12
C. 14
D. 24

Question 4

Predict the output from the following commands:

```
> X <- c(9, 16)
> class(X)
```

A. numeric

B. character

Question 5

Predict the output from the following commands:

```
> X <- c(9, 16, "monkey")
> class(X)
```

A. numeric

B. character

C. numeric, character

Question 6

Predict the output from the following commands:

```
> X <- c(1, 2)
> Y <- c("cat", "dog")
> Z <- c(X, Y)
> class(Z)
```

A. numeric
B. character
C. numeric, character

Week-by-week

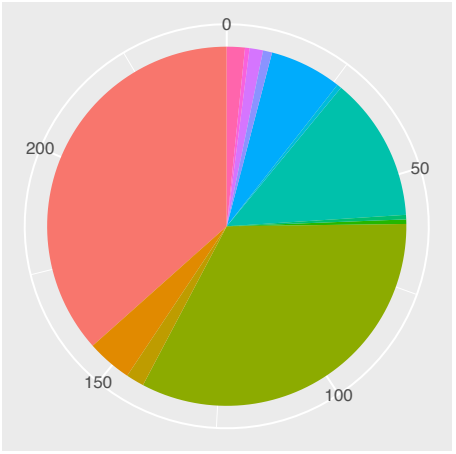
Week Starting	Lecture	Topic	Tutorial	A1	A2
2/3/21	1	Intro to Data Science, review of basic statistics using R	...		
9/3/21	2	Exploring data using graphics in R	T1		
16/3/21	3	Data manipulation in R	T2	Released	
23/3/21	4	Data Science methodologies, dirty/clean/tidy data, data manipulation	T3		
30/3/21	5	Network analysis	T4		
6/4/21		Mid-semester Break			
13/4/21	6	Regression modelling	T5		
20/4/21	7	Classification using decision trees	T6	Submitted	
27/4/21	8	Naïve Bayes, evaluating classifiers	T7		Released
4/5/21	9	Ensemble methods, artificial neural networks	T8		
11/5/21	10	Clustering	T9		
18/5/21	11	Text analysis	T10		Submitted
25/5/21	12	Review of course, Exam preparation	T11		

Visualizing data

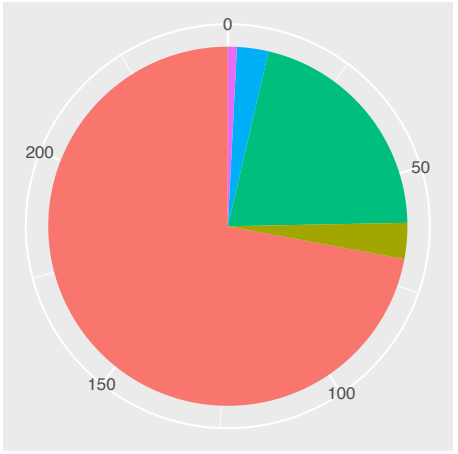
Some examples of data graphics follow. For each image think about:

- What is being conveyed
- How it is being conveyed, what is the main device: size, shape, colour, position...
- The number of dimensions represented. *That is, how many variables are associated with each data point?*
- How is space used

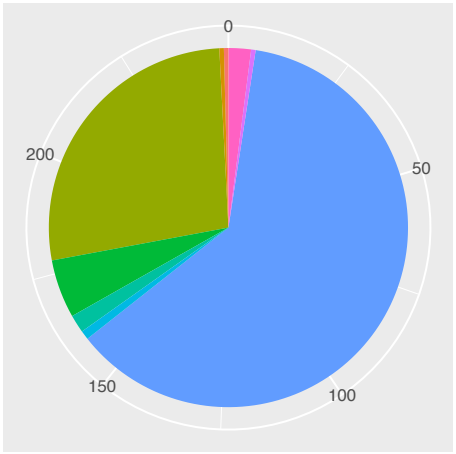
Class survey results (Pie Chart)



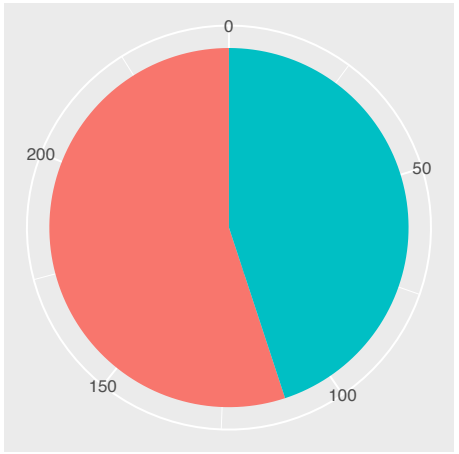
Course Studied
- BCS
- BCS/BusEco
- BCS/Sci
- BInfoTech
- BIT/BSC
- BIT/BSE
- BIT/BusEco
- BIT/Hum
- BSE
- BSE/Sci
- BusEco
- BusEco/Sci
- Science



Programming Skill
- Basic Skill Level
- Basic/Competent
- Competent
- Never
- Never/Basic



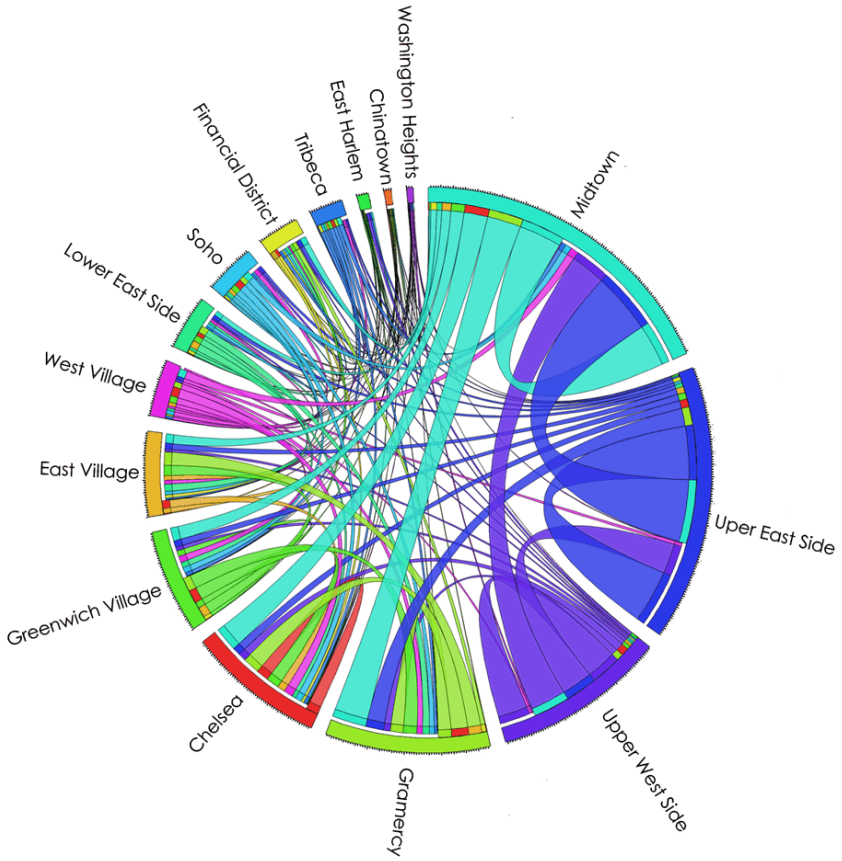
OS Used
- Earlier Windows
- Linux
- OSX
- OSX/Windows 10
- OSX/Windows 10/Linux
- Other
- Windows 10
- Windows 10/Earlier Windows
- Windows 10/Linux



Used R
- No
- Yes

New York taxi trips by neighbourhood

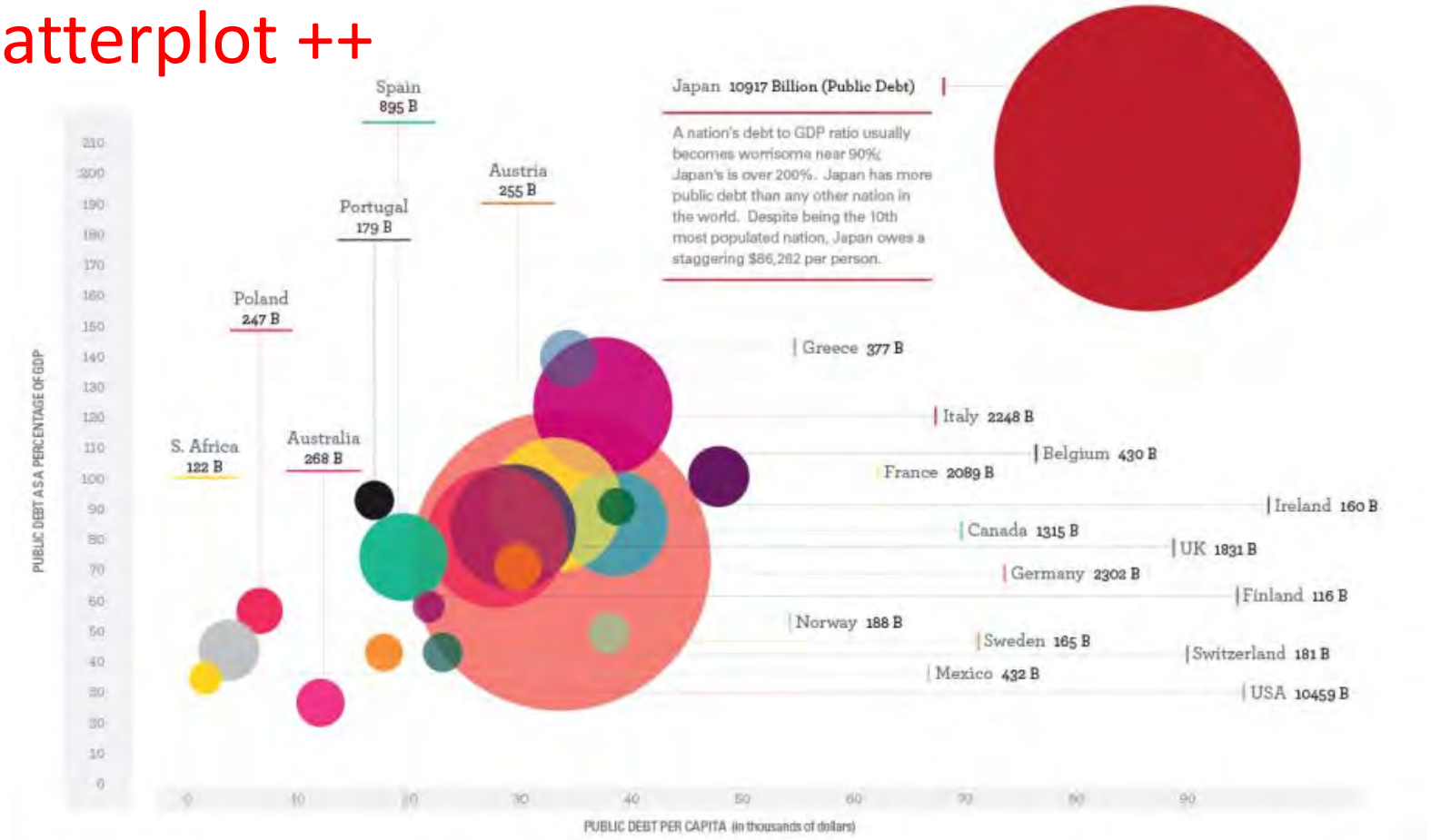
Network



http://www.binaryspark.com/heytaxi/

Debt crisis: Japan

Scatterplot ++



http://www.infogra.ph/ (accessed 2015)

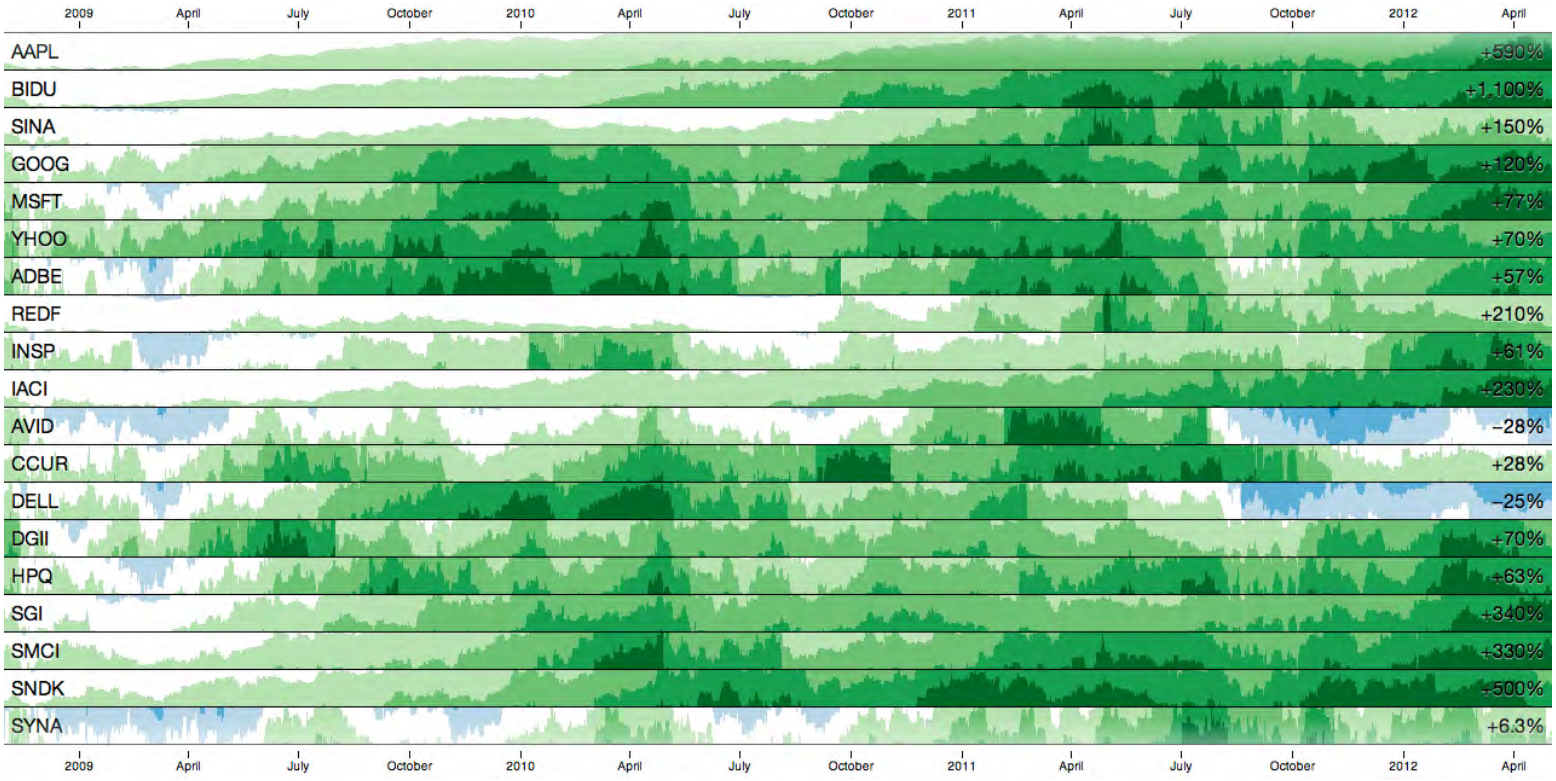
Security visualization

Mosaic

https://secviz.org/content/applied-security-visualization (accessed 2020)

Share prices

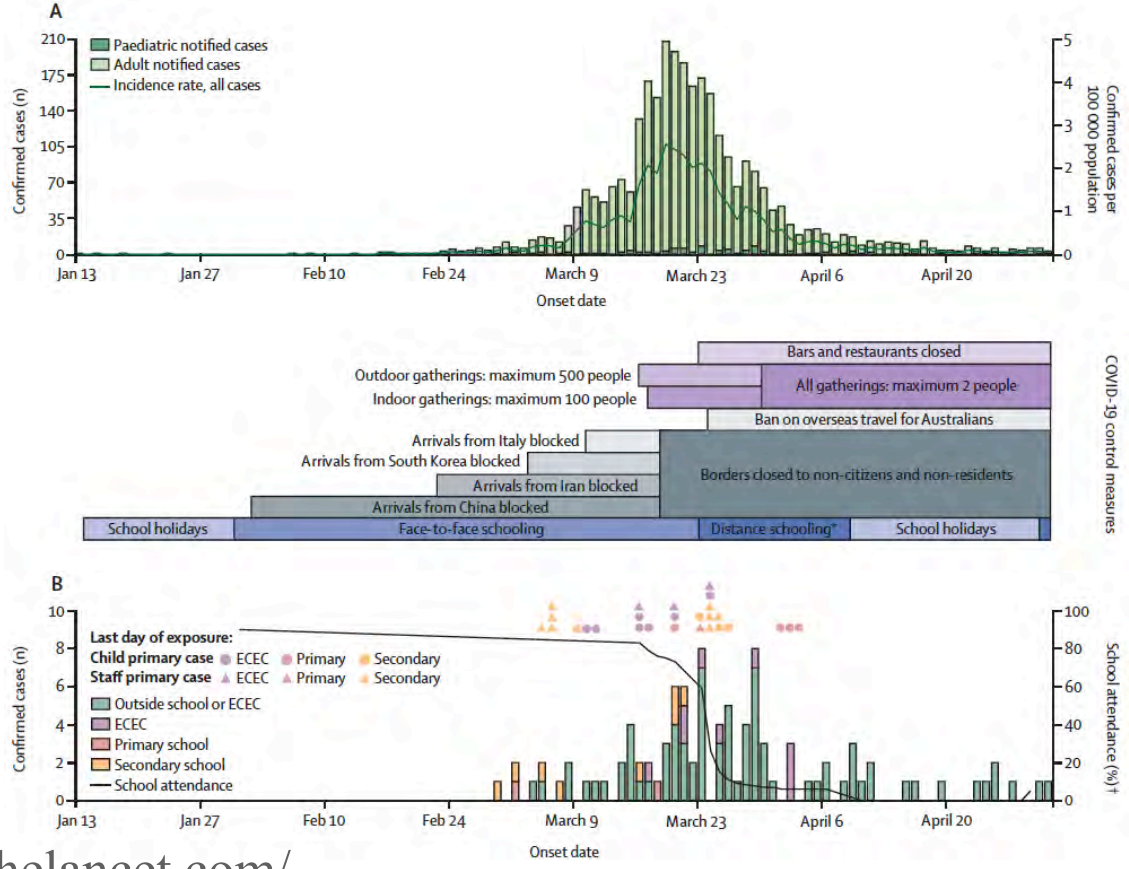
Horizon



https://bost.ocks.org/mike/cubism/intro/#12

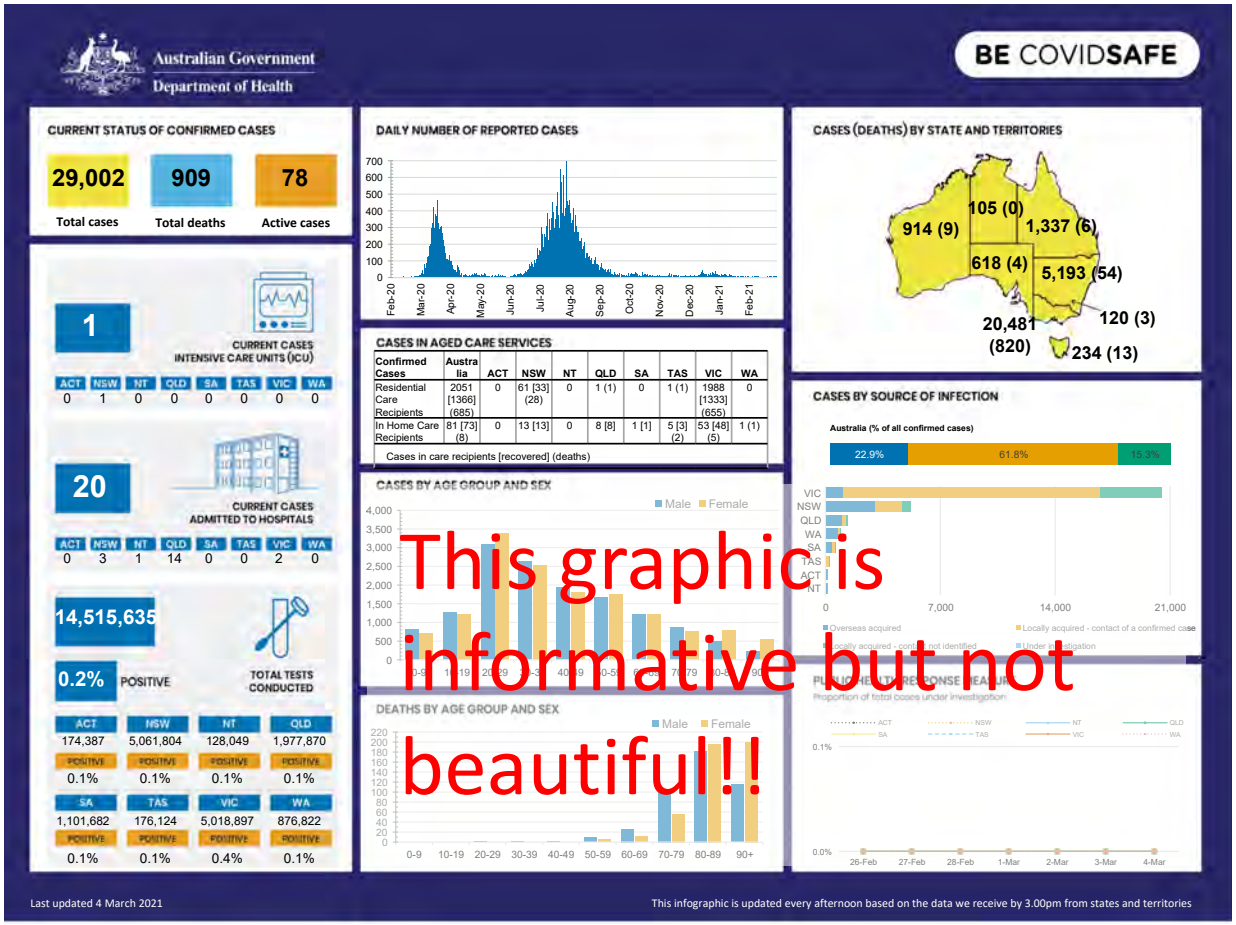
Controlling the COVID-19 pandemic

Composite



https://www.thelancet.com/

COVID-19 stats 4th March 2021



https://www.health.gov.au/

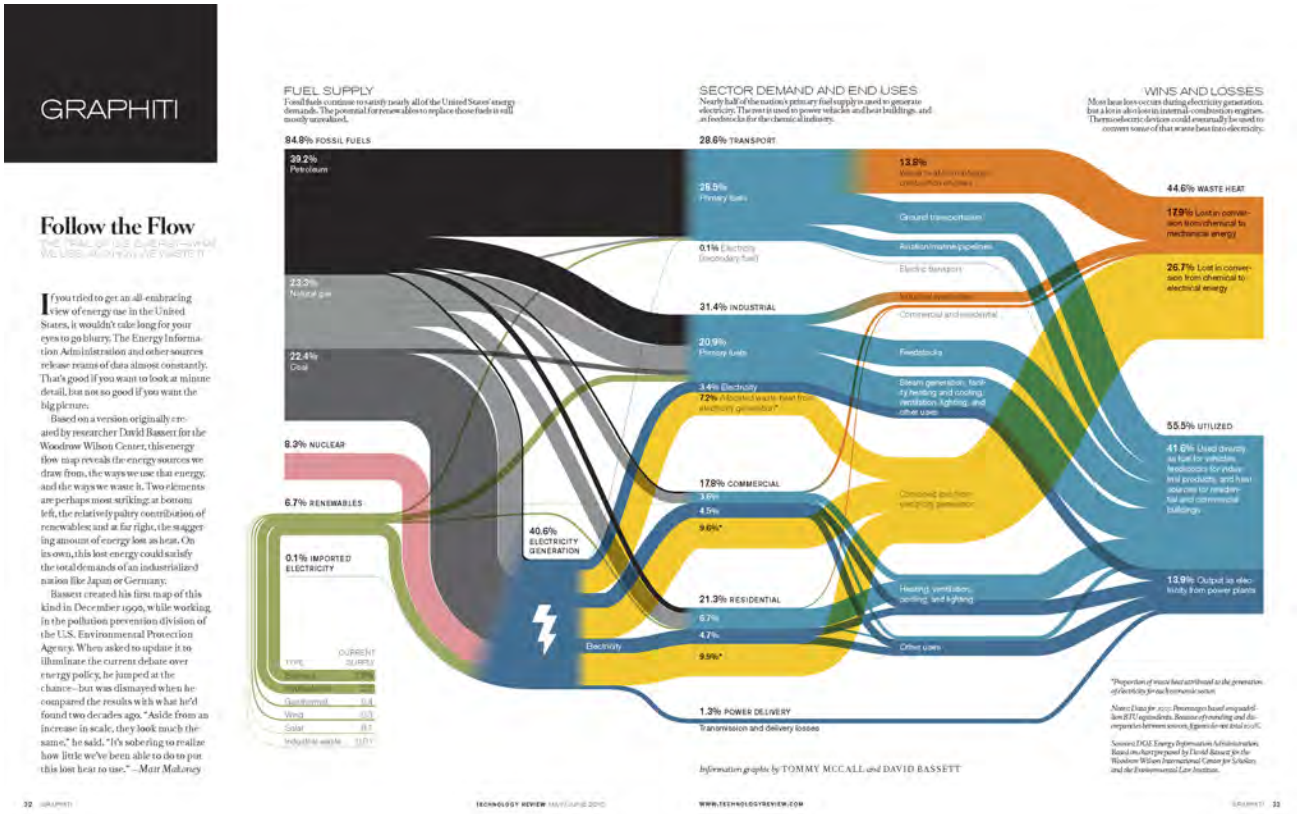
Coronavirus Graphics: best/worst

- The MIT Technology Review has collected together the best and worst (in their view) of the current Coronavirus dashboards

https://www.technologyreview.com/s/615330/best-worst-coronavirus-dashboards/

US energy production/consumption

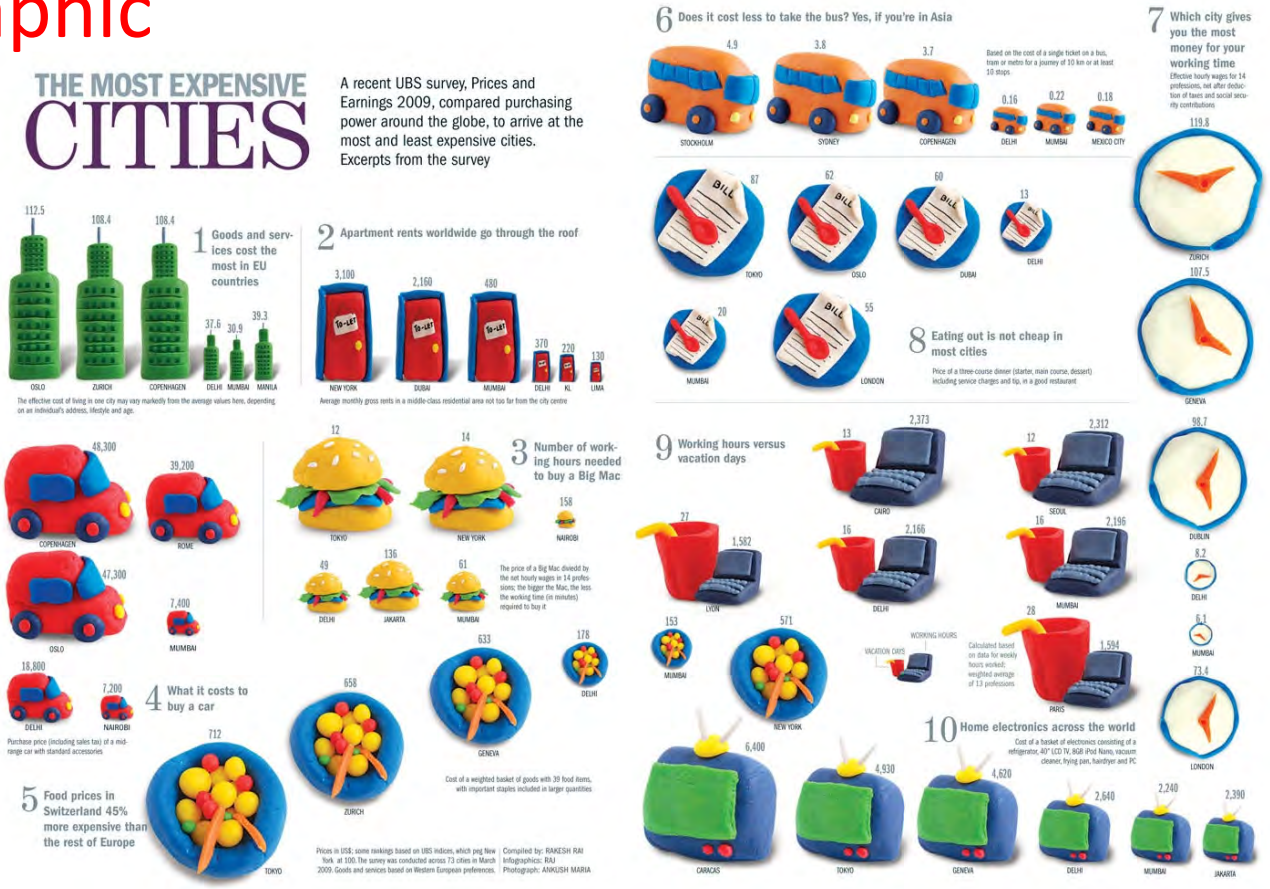
Sankey Diagram



https://visual.ly/community/Infographics/economy/follow-flow

Most expensive cities

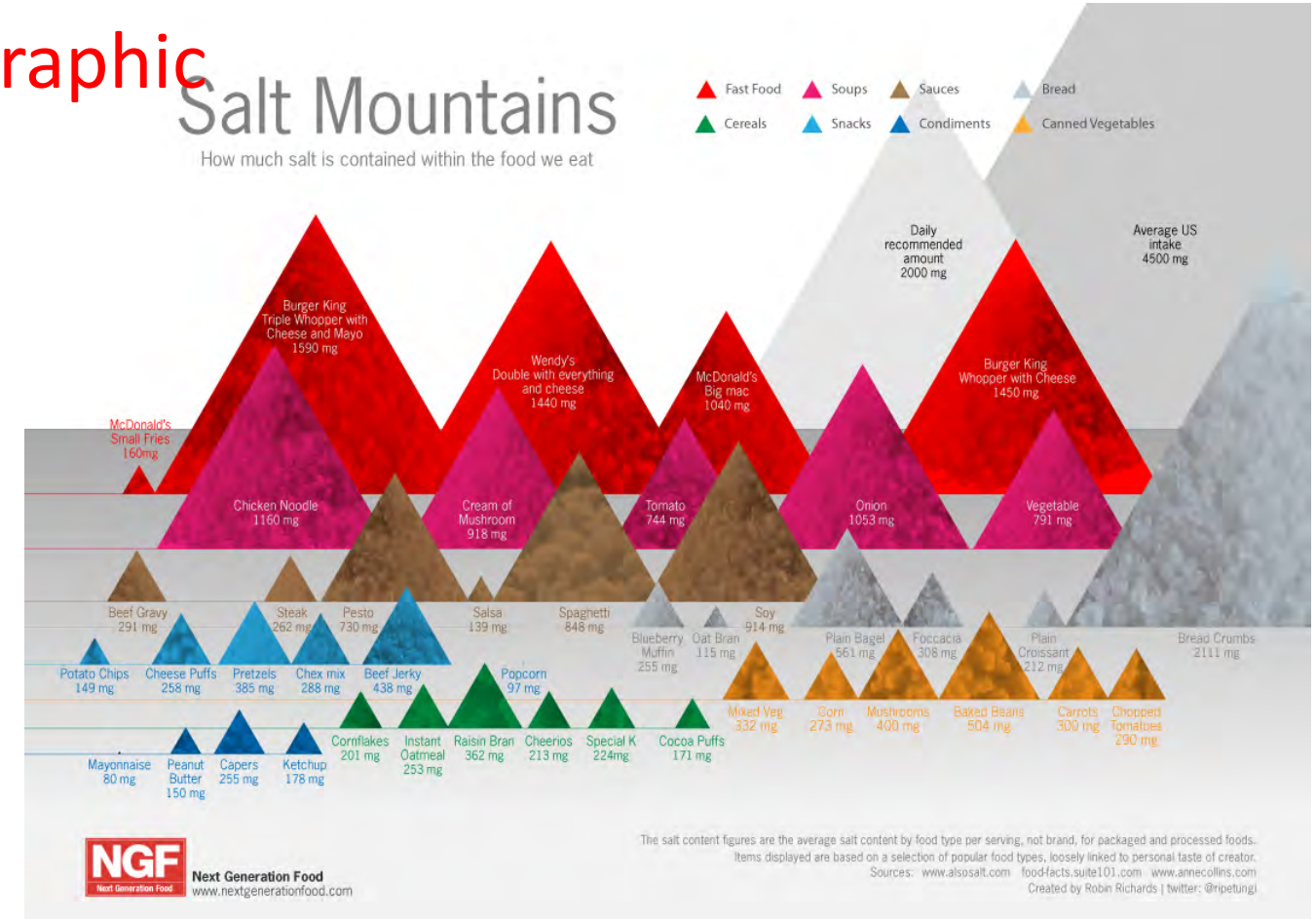
Infographic



https://infographiclist.com/the-most-expensive-cities-infographic/

Salt in food

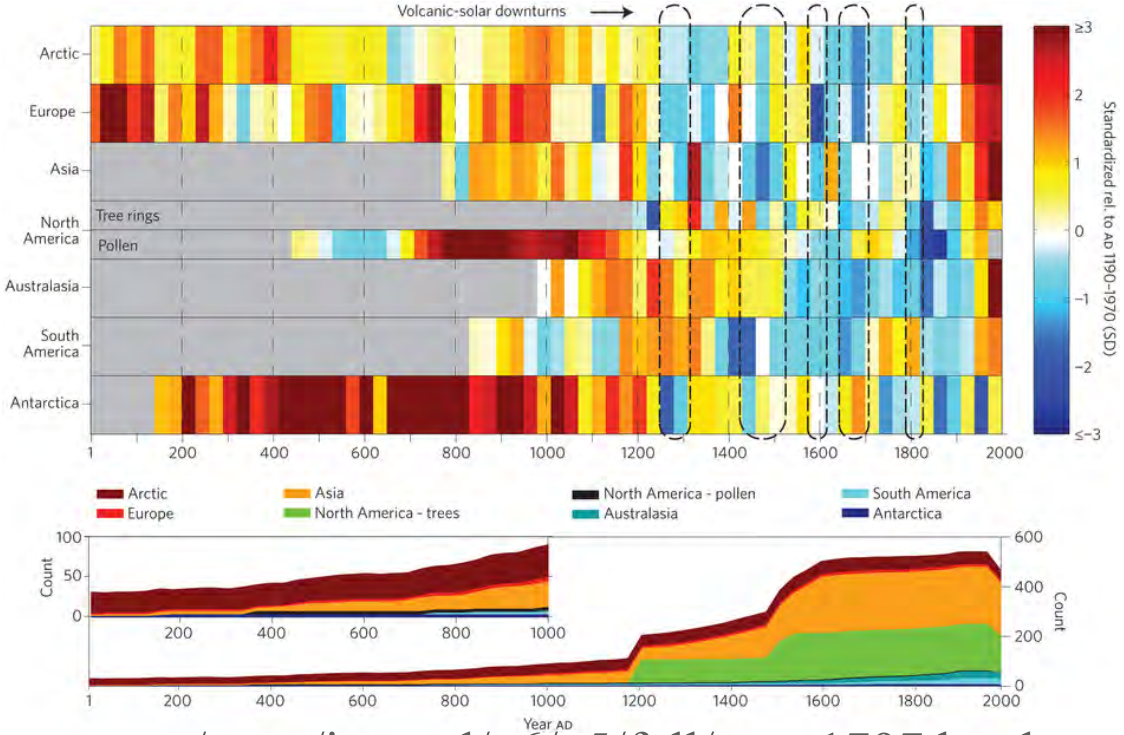
Infographic



http://thumbnails.visually.netdna-cdn.com/salt-mountains_50290aaeb00da.png

Climate change

Continental-scale temperature variability during the past two millennia



http://www.nature.com/ngeo/journal/v6/n5/full/ngeo1797.html

Question 8

How many dimensions does the figure show?

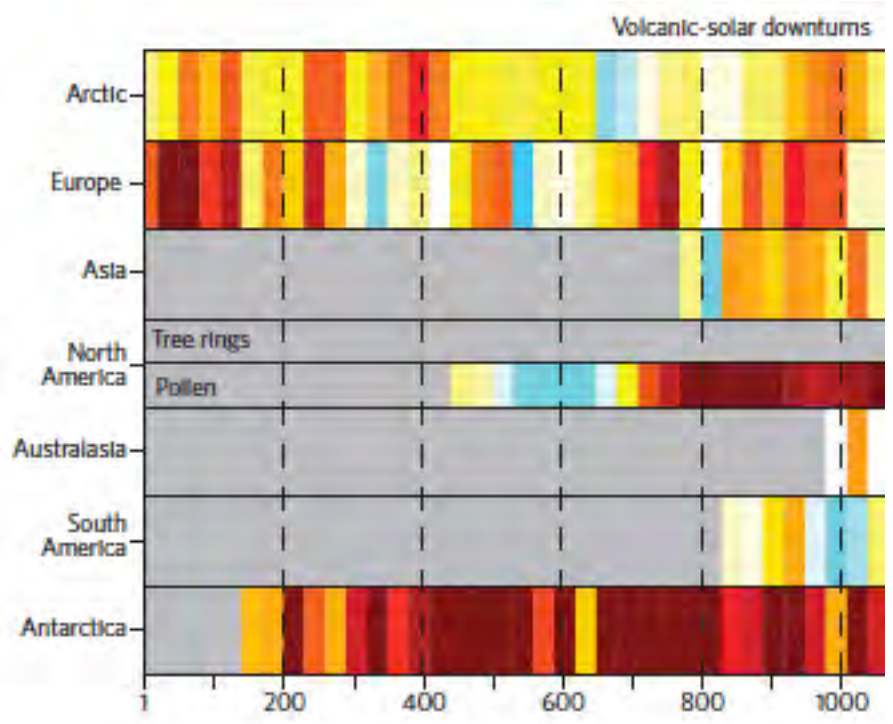
A. 1

B. 2

C. 3

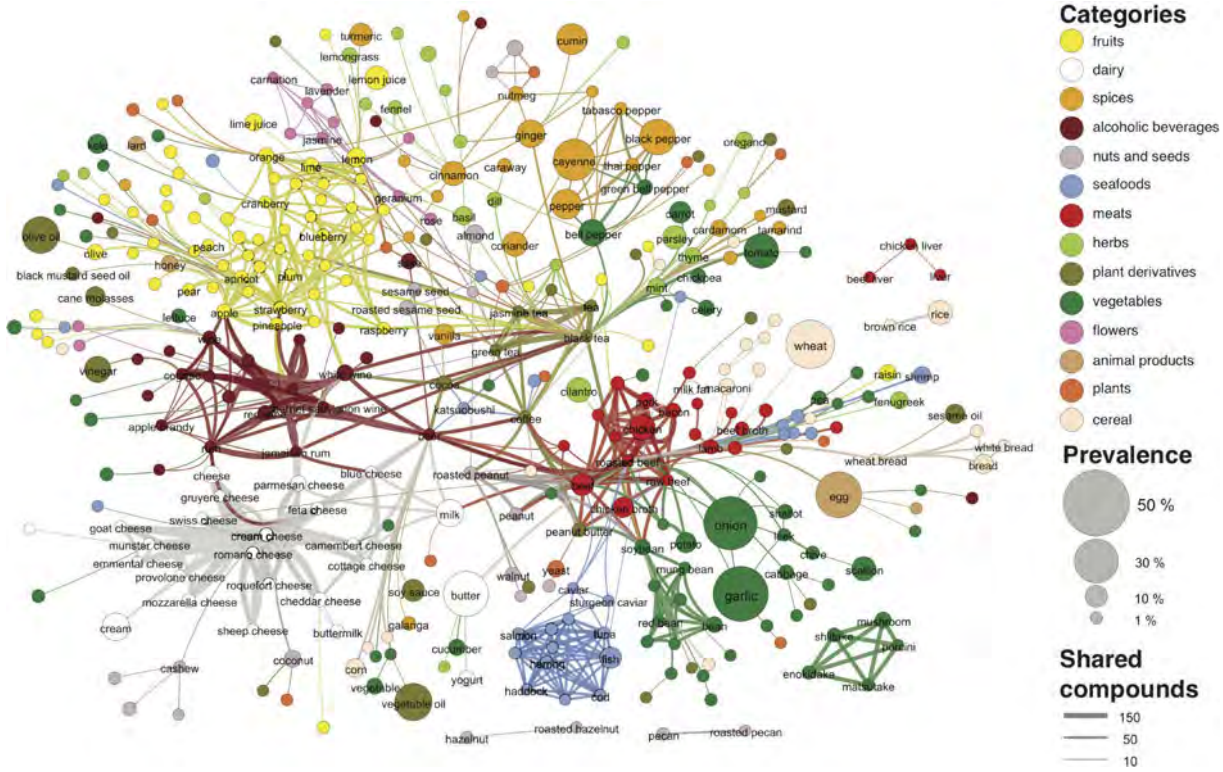
D. 4

E. More than 4



Food networks

Flavor network and the principles of food pairing



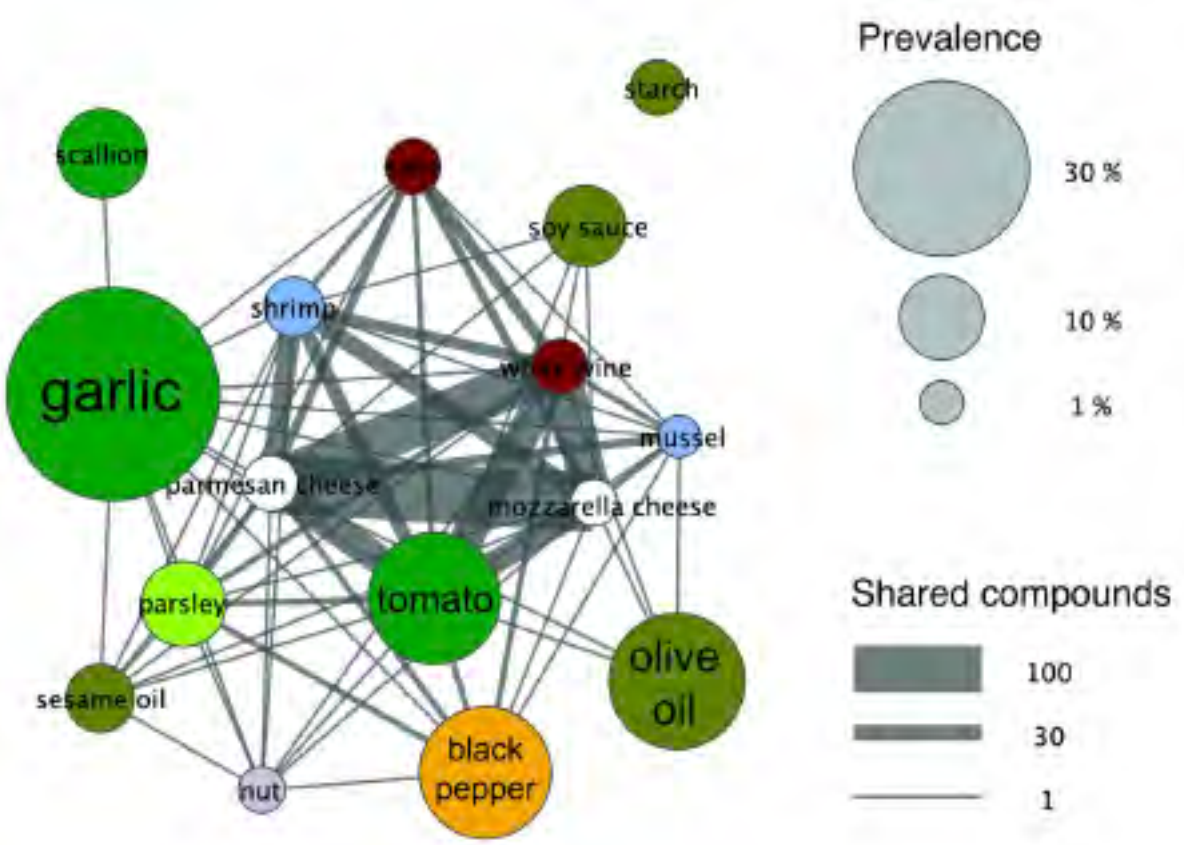
http://www.nature.com/srep/2011/111215/srep00196/full/srep00196.html

Question 7

How many dimensions does the figure show?

A. 1
B. 2
C. 3
D. 4
E. More than 4

Think about how many variables are associated with each data point...



Inspiration:

What type of graphic do you want to create?

What data do you have, and what do you want the graphic to say?

Some starting points:

The Visualization Zoo…

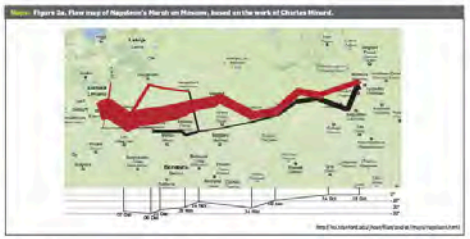
A tour through the visualization zoo

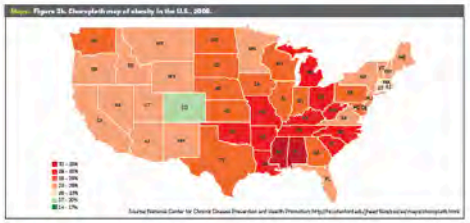
http://dl.acm.org/citation.cfm?id=1743567

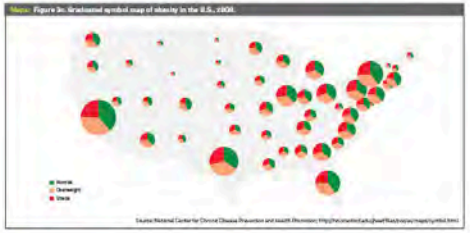
Identifies the major graphic types and their subtypes.

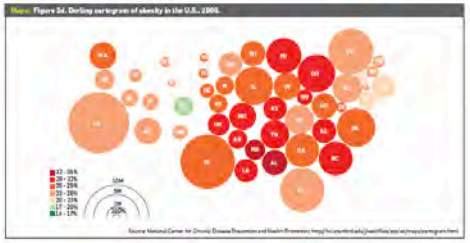
- Time Series
- Statistical distributions
- Maps
- Hierarchies
- Networks

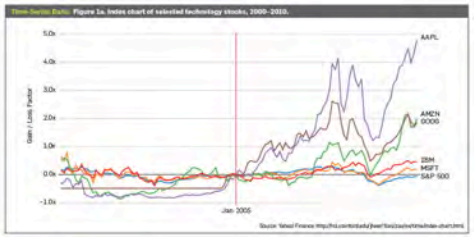
The Visualization Zoo…

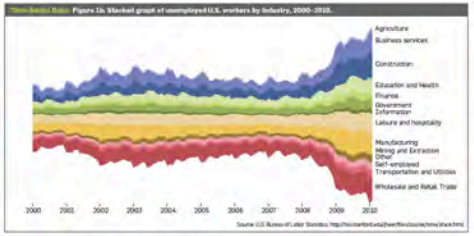


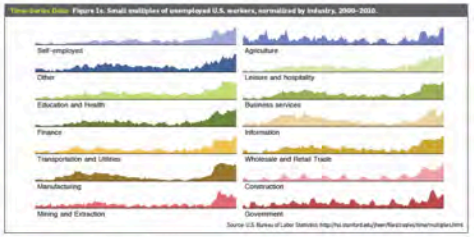


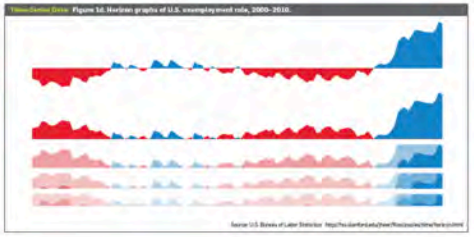


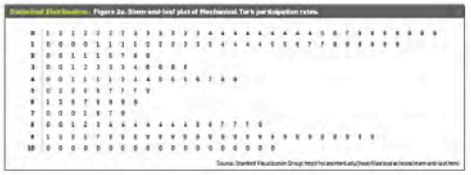


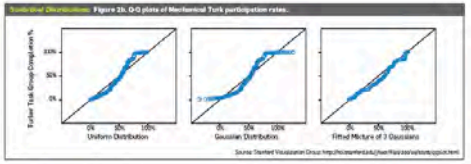


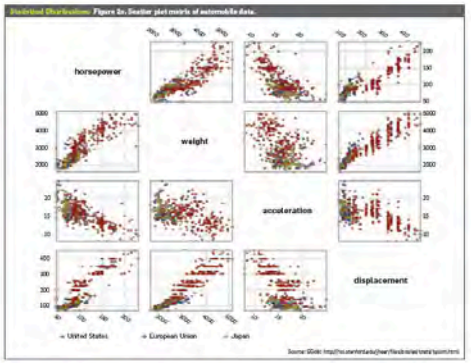


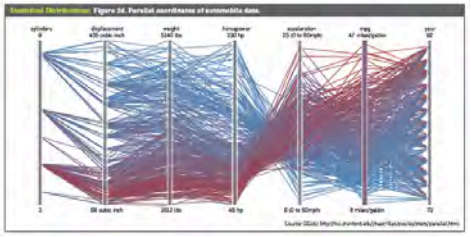




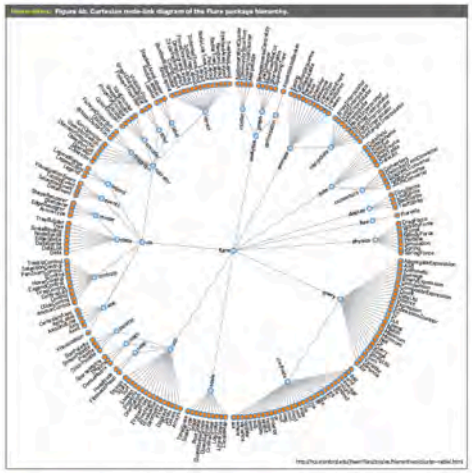


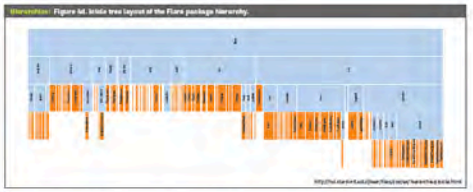


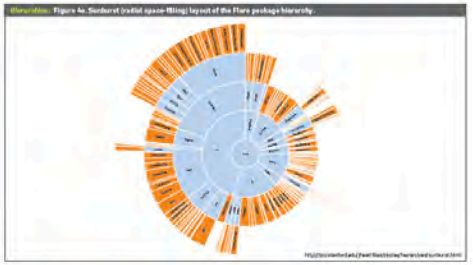


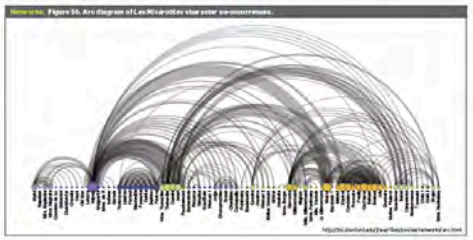


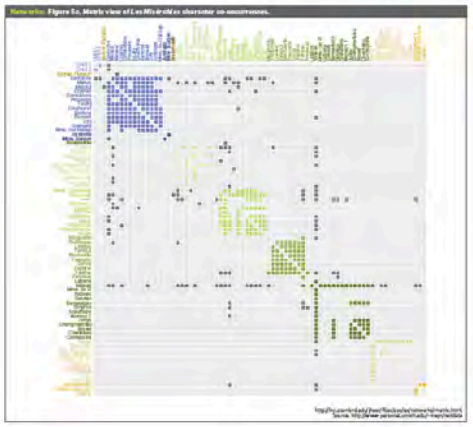
The Visualization Zoo…



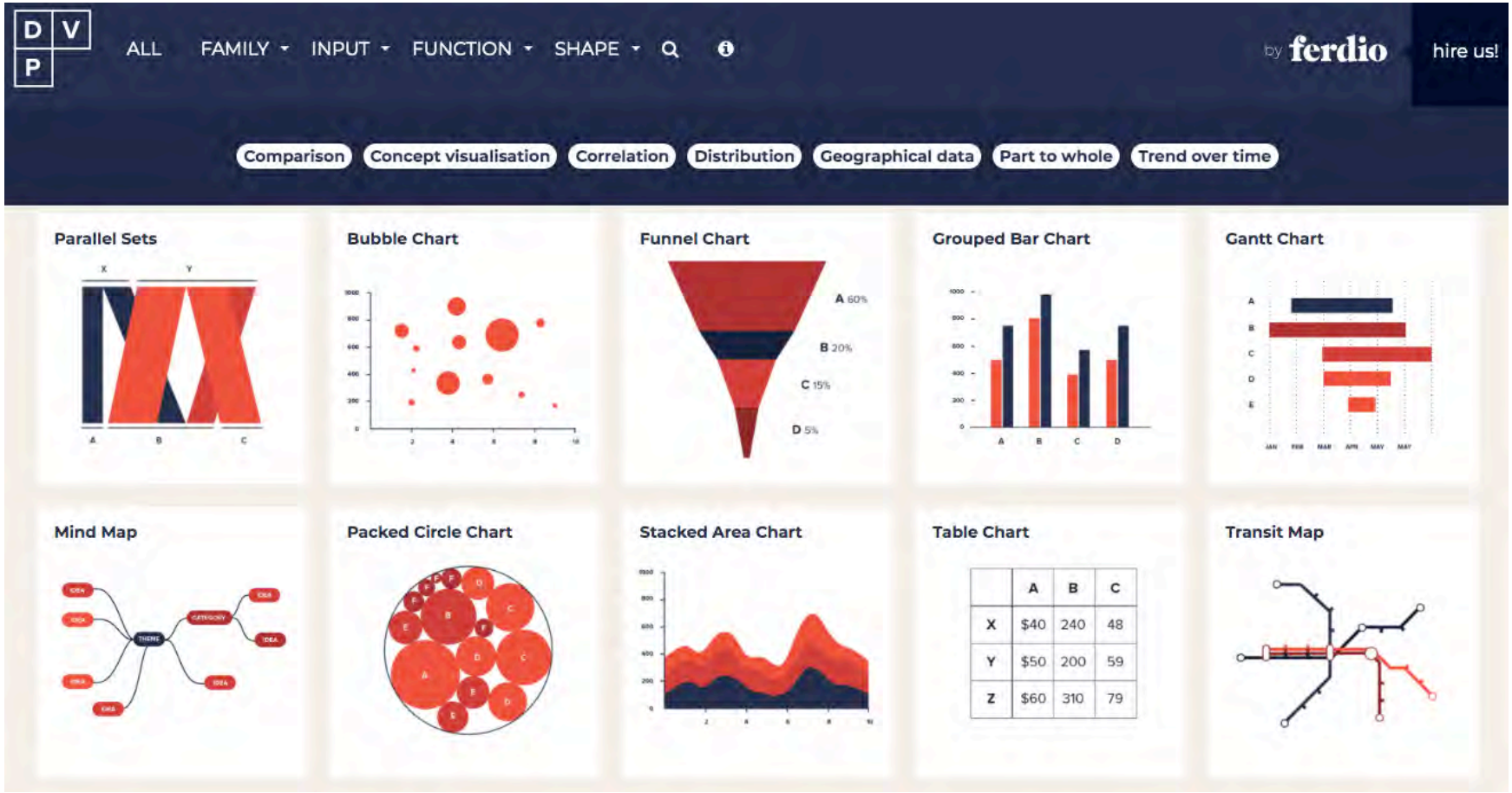








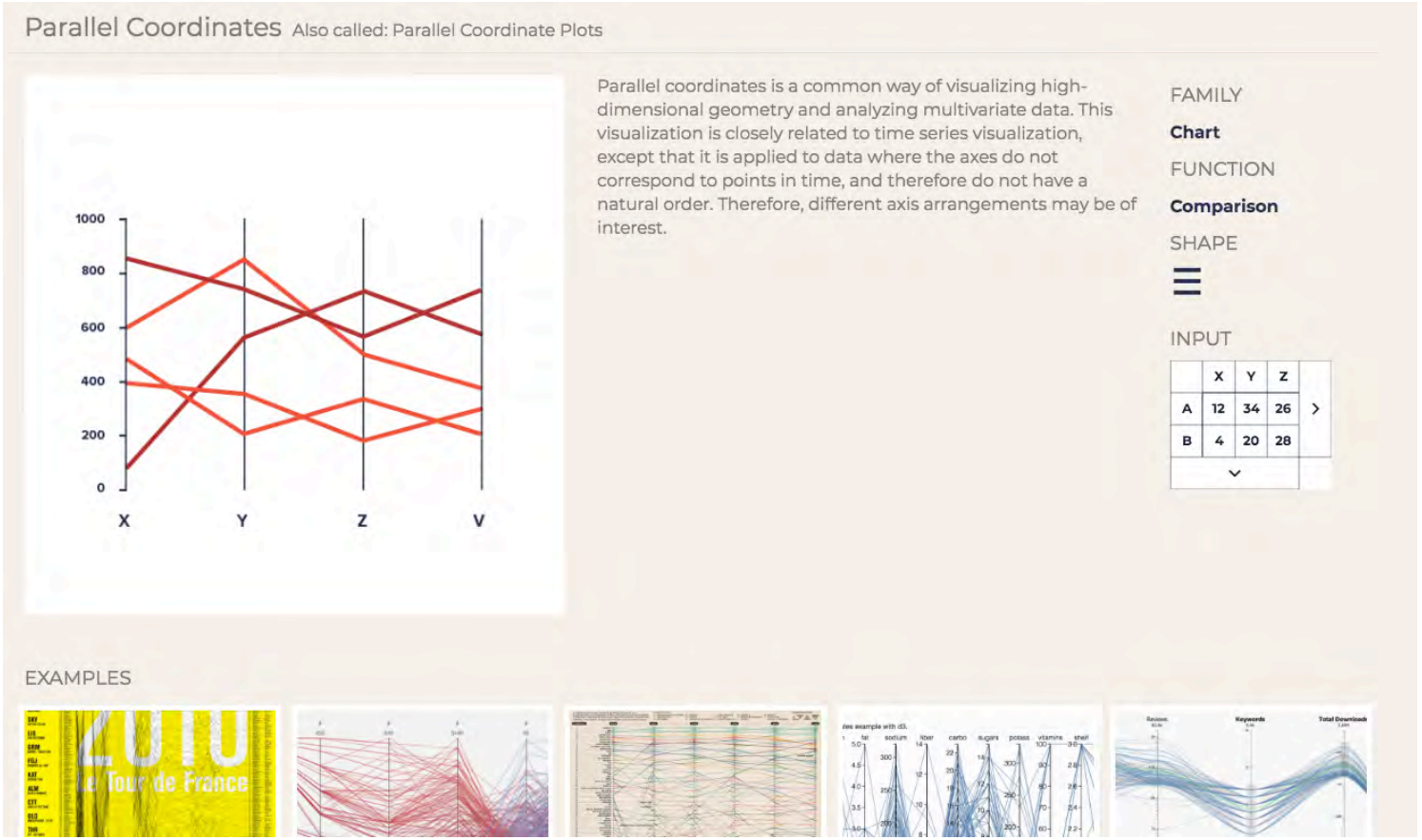
Data Viz Project



	A	B	C
X	$40	240	48
Y	$50	200	59
Z	$60	310	79

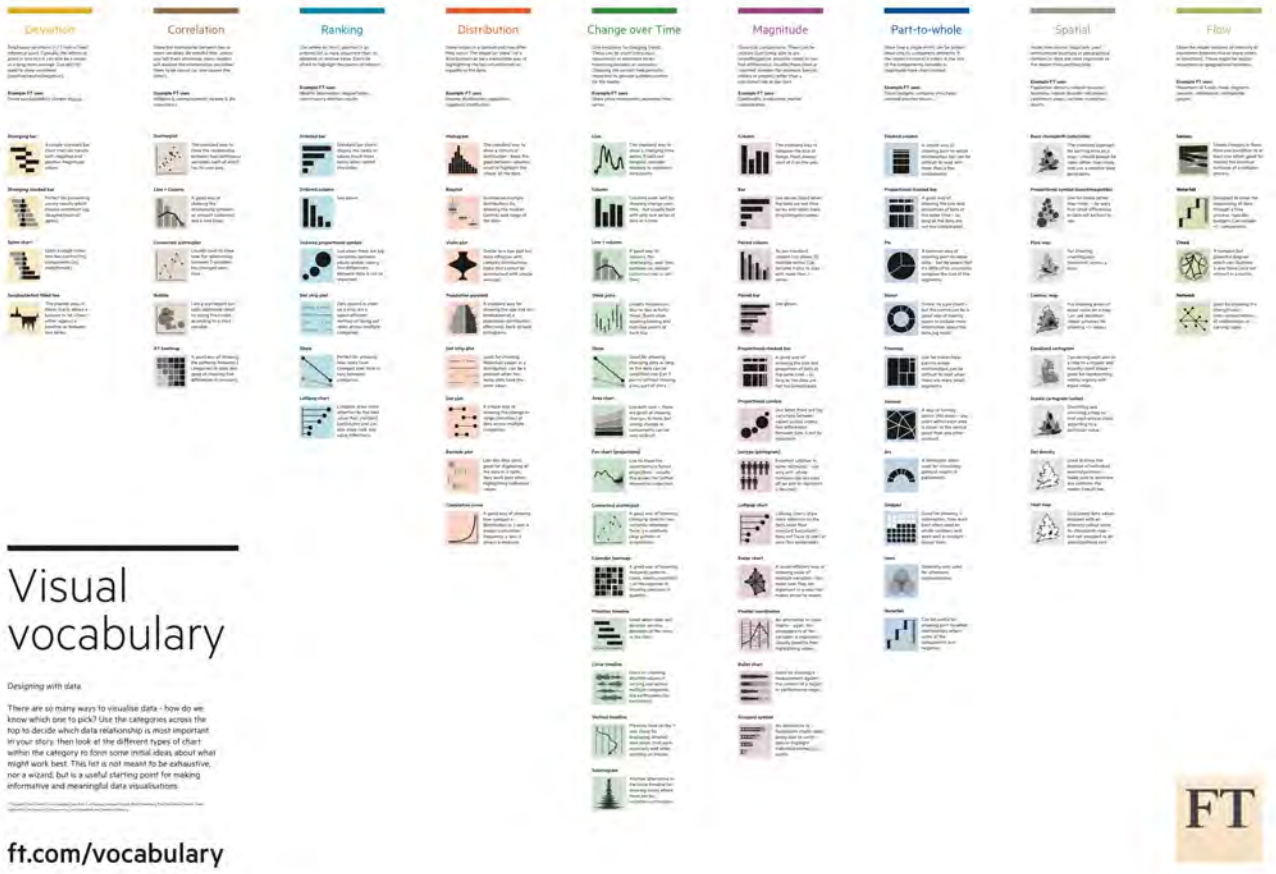
https://datavizproject.com/

Data Viz Project



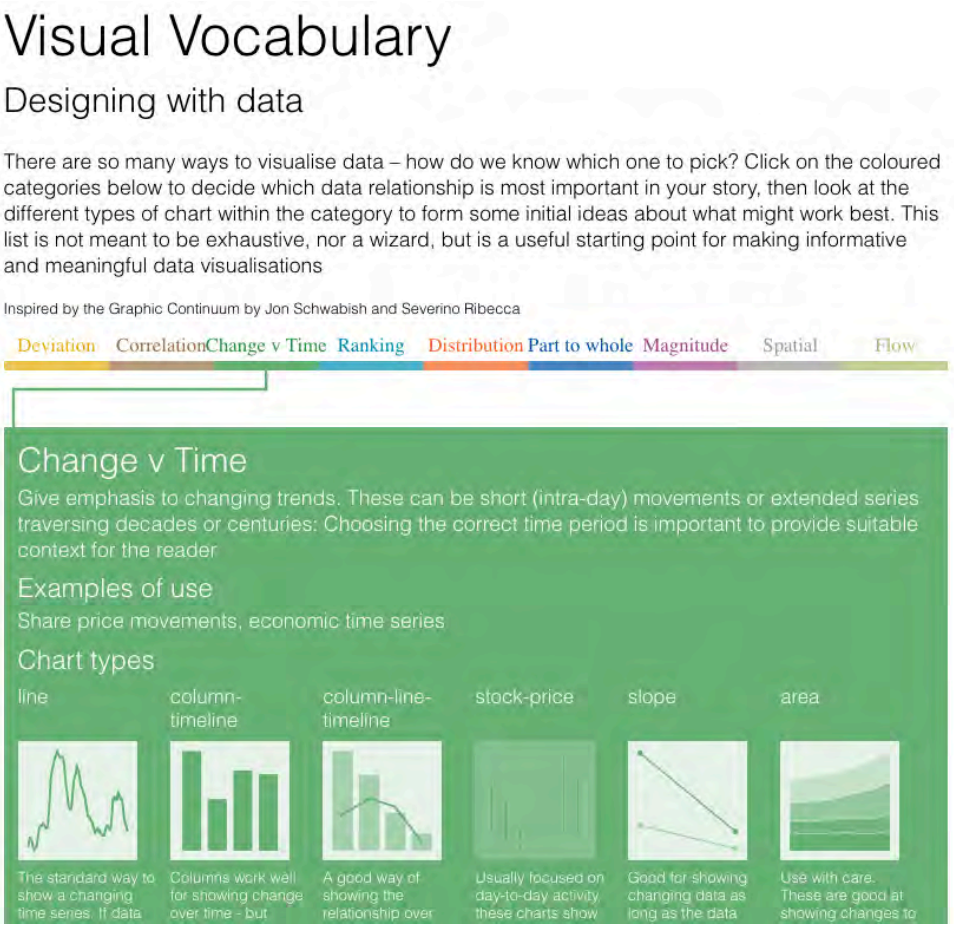
https://datavizproject.com/

FT: visual vocabulary



https://github.com/ft-interactive/chart-doctor/tree/master/visual-vocabulary

Visual vocabulary interactive



https://ft-interactive.github.io/visual-vocabulary/

TED talks on data science

Playlist on data and data science: Making sense of too much data

https://www.ted.com/playlists/56/making_sense_of_too_much_data

In particular: Hans Rosling, David McCandless, Deb Roy, Nate Silver, Mona Chalabi, Jennifer Golbeck – but all worth looking at...

Getting to know a data set

Edgar Anderson's Iris data

50 samples from 3 species:

- Iris setosa, – virginica, – versicolor

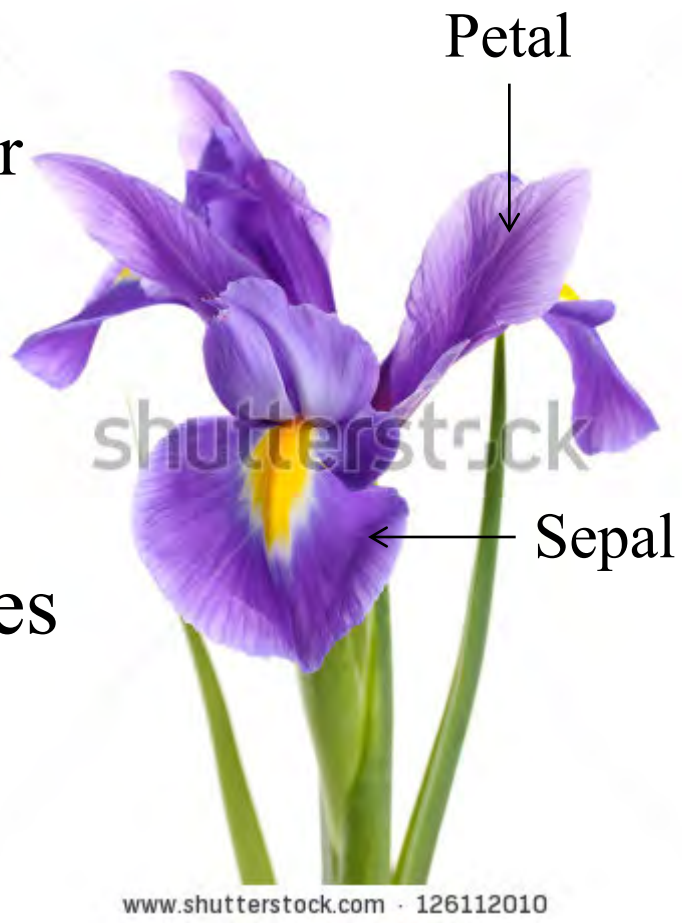
Four features measured:

- Sepal width and length
- Petal width and length

Is it possible to distinguish species using physical measurements?

- Data is packaged with R: "iris"

https://en.wikipedia.org/wiki/Iris_flower_data_set



Print

> iris *# = prints out the data set. Ok for small data sets*

```
  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
...
```

Question 9

How many dimensions in the Iris data?

A. 1

B. 2

C. 3

D. 4

E. More than 4

Dimension, column names, structure

> dim(iris)

```
[1] 150 5
```

> names(iris)

```
[1] "Sepal.Length" "Sepal.Width" "Petal.Length" "Petal.Width"
"Species"
```

> str(iris)

```
'data.frame': 150 obs. of 5 variables:
$ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
$ Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
$ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
$ Petal.Width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
$ Species : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1
1 1 1 1 ...
```

Print head and tail

> head(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
```

> tail(iris)

```
    Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
145          6.7         3.3          5.7         2.5 virginica
146          6.7         3.0          5.2         2.3 virginica
147          6.3         2.5          5.0         1.9 virginica
148          6.5         3.0          5.2         2.0 virginica
149          6.2         3.4          5.4         2.3 virginica
150          5.9         3.0          5.1         1.8 virginica
```

…or a selection of rows

```
> iris[10:15,] # by convention [ ] index rows
   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
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
```

```
> iris[11,] # single row
   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
11          5.4         3.7          1.5         0.2  setosa
```

…or part of a single column

```
> iris[10:20, "Sepal.Length"] # identify column as string
 [1] 4.9 5.4 4.8 4.8 4.3 5.8 5.7 5.4 5.1 5.7 5.1
> # or
> iris$Sepal.Length[10:20] # identify column by name
> [1] 4.9 5.4 4.8 4.8 4.3 5.8 5.7 5.4 5.1 5.7 5.1
> # or
> iris[10:20,1] # identify column by number
> [1] 4.9 5.4 4.8 4.8 4.3 5.8 5.7 5.4 5.1 5.7 5.1
```

Summary

Create a mean + 5 point summary of each numerical column, and list of types for factors.

```
> summary(iris)
```

```
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width          Species
Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100   setosa     :50
1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:50
Median :5.800   Median :3.000   Median :4.350   Median :1.300   virginica :50
Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
```

The real irises

http://dataaspirant.com/2017/01/25/svm-classifier-implemenation-python-scikit-learn/

Question 10

Which species is easiest to differentiate?

A. Versicolor

B. Virginica

C. Setosa

D. Too hard to tell.

Class activity

The data set ‘mpg’ is contained in the ggplot2 package. Let’s get to know it (how many dimensions, types of variables, range etc.) without any graphics.

```
> ?mpg # information about the data
> Head(mpg)
> Str(mpg)
> summary(mpg)
> tail(mpg)
> unique(mpg) #particular columns
```

- See worksheet (MPG Summary) on Moodle

Class activity

```
> str(mpg)
Classes 'tbl_df', 'tbl' and 'data.frame':      234 obs. of  11 variables:
 $ manufacturer: chr  "audi" "audi" "audi" "audi" ...
 $ model       : chr  "a4" "a4" "a4" "a4" ...
 $ displ       : num  1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
 $ year        : int  1999 1999 2008 2008 1999 1999 2008 1999 1999 2008 ...
 $ cyl         : int  4 4 4 4 6 6 6 4 4 4 ...
 $ trans       : chr  "auto(l5)" "manual(m5)" "manual(m6)" "auto(av)" ...
 $ drv         : chr  "f" "f" "f" "f" ...
 $ cty         : int  18 21 20 21 16 18 18 18 16 20 ...
 $ hwy         : int  29 29 31 30 26 26 27 26 25 28 ...
 $ fl          : chr  "p" "p" "p" "p" ...
 $ class       : chr  "compact" "compact" "compact" "compact" ...

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

Class activity

```
> summary(mpg)
 manufacturer          model               displ            year
 Length:234         Length:234         Min.   :1.600   Min.   :1999
 Class :character   Class :character   1st Qu.:2.400   1st Qu.:1999
 Mode  :character   Mode  :character   Median :3.300   Median :2004
                                       Mean   :3.472   Mean   :2004
                                       3rd Qu.:4.600   3rd Qu.:2008
                                       Max.   :7.000   Max.   :2008
      cyl            trans               drv
 Min.   :4.000   Length:234         Length:234
 1st Qu.:4.000   Class :character   Class :character
 Median :6.000   Mode  :character   Mode  :character
 Mean   :5.889
 3rd Qu.:8.000
 Max.   :8.000
      cty             hwy              fl
 Min.   : 9.00   Min.   :12.00   Length:234
 1st Qu.:14.00   1st Qu.:18.00   Class :character
 Median :17.00   Median :24.00   Mode  :character
 Mean   :16.86   Mean   :23.44
 3rd Qu.:19.00   3rd Qu.:27.00
 Max.   :35.00   Max.   :44.00
    class
 Length:234
 Class :character
 Mode  :character
```

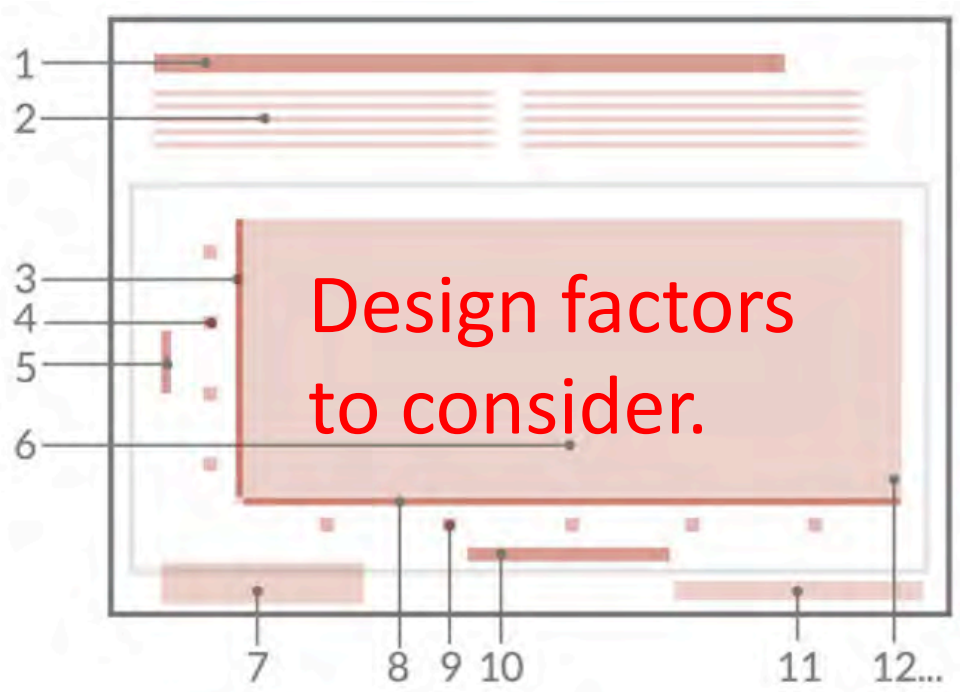
Class activity

```
> tail(mpg)
# A tibble: 6 x 11
  manufacturer  model displ  year   cyl      trans   drv   cty   hwy
         <chr>  <chr> <dbl> <int> <int>      <chr> <chr> <int> <int>
1   volkswagen passat   1.8  1999     4   auto(l5)     f    18    29
2   volkswagen passat   2.0  2008     4   auto(s6)     f    19    28
3   volkswagen passat   2.0  2008     4 manual(m6)     f    21    29
4   volkswagen passat   2.8  1999     6   auto(l5)     f    16    26
5   volkswagen passat   2.8  1999     6 manual(m5)     f    18    26
6   volkswagen passat   3.6  2008     6   auto(s6)     f    17    26
# ... with 2 more variables: fl <chr>, class <chr>

> unique(mpg$manufacturer)
 [1] "audi"       "chevrolet"  "dodge"      "ford"       "honda"
 [6] "hyundai"    "jeep"       "land rover" "lincoln"    "mercury"
[11] "nissan"     "pontiac"    "subaru"     "toyota"     "volkswagen"
```

Graphing the data

Elements of a figure



Typical elements: title (1), subtitle (2), y-axis (3), label (4), name (5), data area (6), legend (7), X-axis (8), label (9), and name (10), sources (11). Further elements: annotations/lines/symbols (12).

Thomas Rahlf: Data Visualisation with R

Base graphics

These are the graphic functions built into the basic R installation.

- High level graphic functions create new graphs with axis, labels and titles.
- Low level graphic functions then annotate plots with points, lines and text.
- See ATHR page 48.

Note: Chapter 3 of
A Tiny Handbook of R
Is worth reading.

Base graphics: high level functions

Some functions we have used/will use include :

> plot *# Scatterplot*

> pairs *# Scatterplot matrix*

> hist *# Histogram*

> stem *# Stem-and-leaf plot*

> boxplot *# Box-and-whisker plot*

> barplot *# Bar plot*

> dotchart *# Dot plot*

- See ATHR page 49

Base graphics: low level functions

Some low level functions include:

> lines *# Draw lines between given coordinates*

> text *# Draw text at given coordinates*

> abline *# Line y = ax + b, horizontal or vertical*

> axis *# Add an axis*

> arrows *# Draw arrows*

> grid *# Add a rectangular grid*

> legend *# Add a legend (a key)*

- See ATHR page 50

Base graphics: graphics parameters

Some low level functions include:

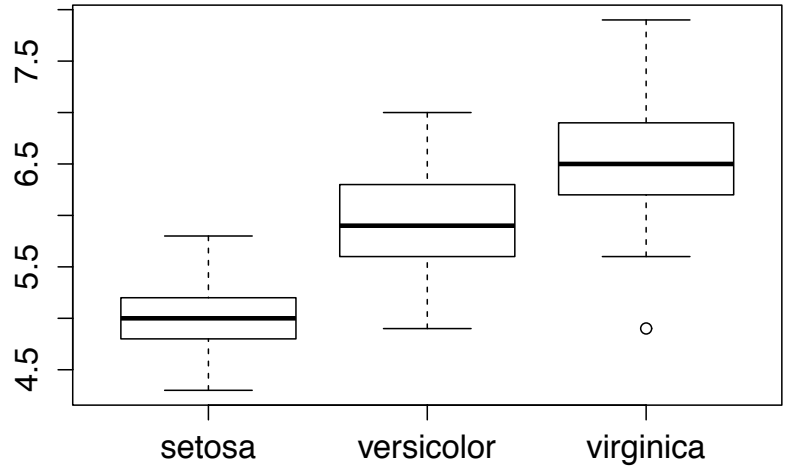
> main *# Title of the plot*
> ylab, xlab *# Labels for the y-axis and x-axis*
> type *# Plot type (points, lines, both, ...),*
> pch *# Plot character (circles, dots, , symbols, ...)*
> lty *# Line type (solid, dots, dashes, ...)*
> lwd *# Line width*
> col *# Colour of plot characters*
> ...and many others, see: help(par)

- See ATHR page 50

Boxplot

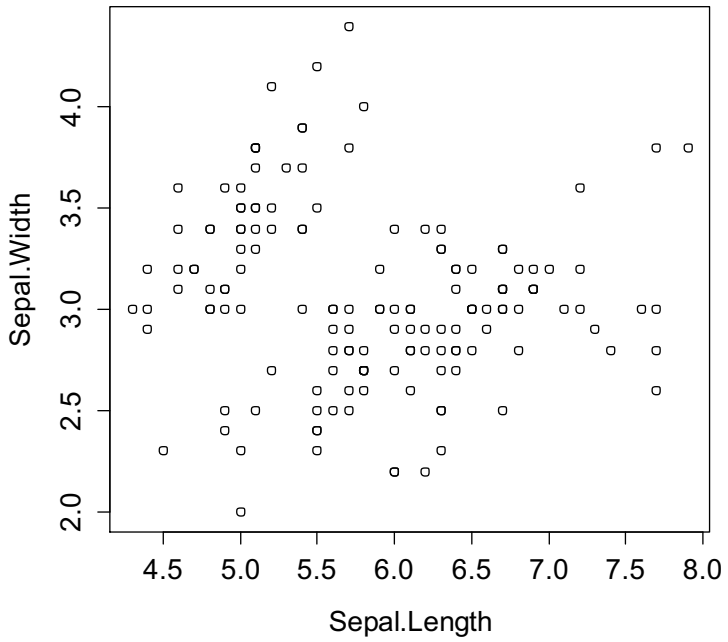
Each variable can be viewed as a boxplot distinguished by level:

```
> boxplot(Sepal.Length ~ Species, data = iris)
```



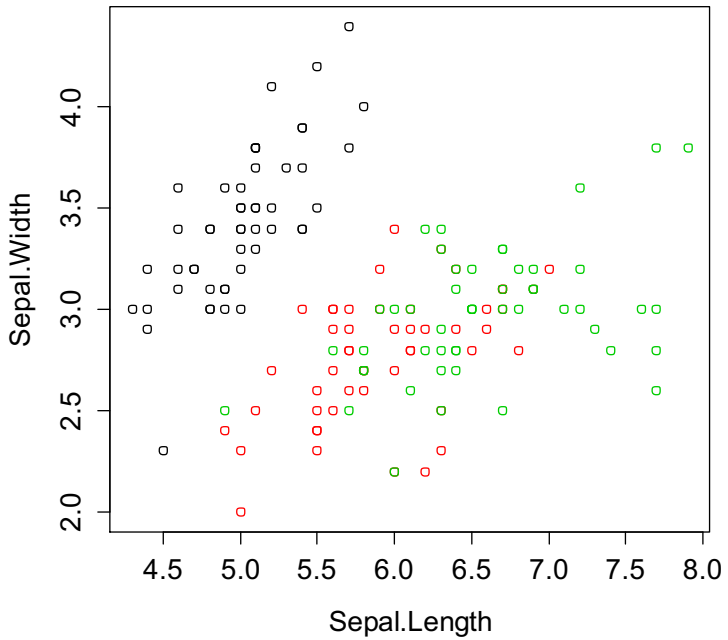
Scatterplot

```
> with(iris, plot(Sepal.Length, Sepal.Width))
> # using 'with' simplifies column names etc.
```



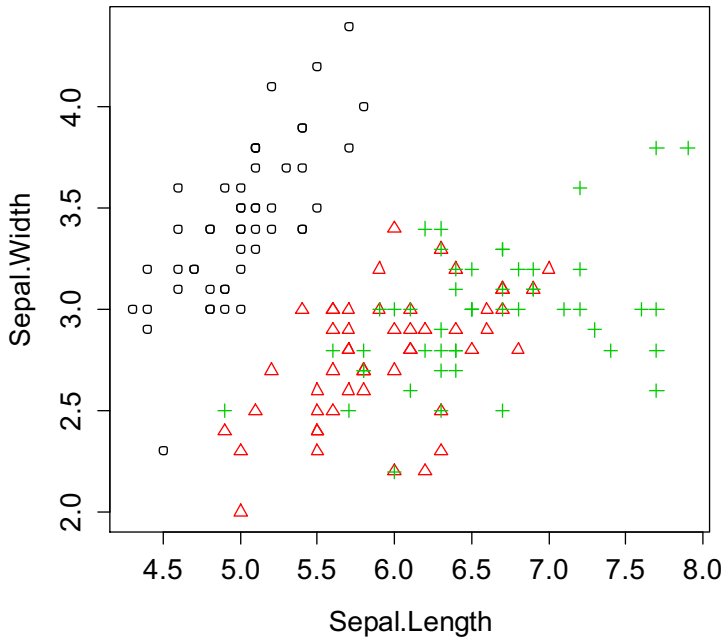
Scatterplot + colour

> with(iris, plot(Sepal.Length, Sepal.Width, col = Species))



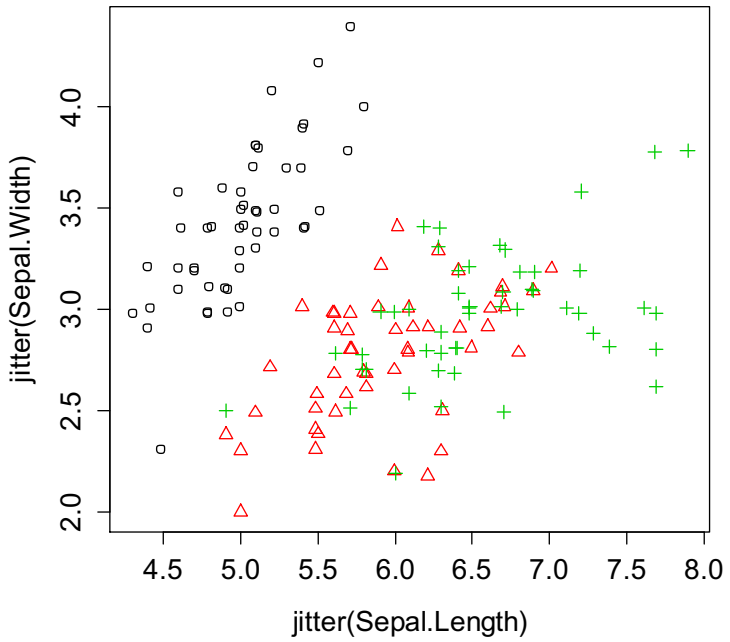
Scatterplot + plot symbol

> with(iris, plot(Sepal.Length, Sepal.Width, col = Species, pch=as.numeric(Species)))



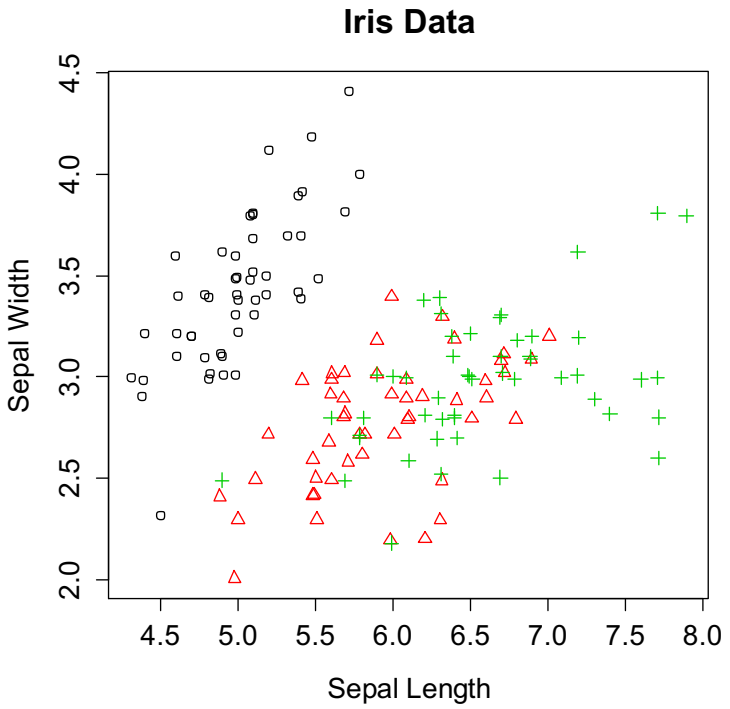
Scatterplot + jitter

```
> with(iris, plot(jitter(Sepal.Length), jitter(Sepal.Width),
  col = Species, pch=as.numeric(Species)))
> # jittering reveals some of the overlapping data points
```



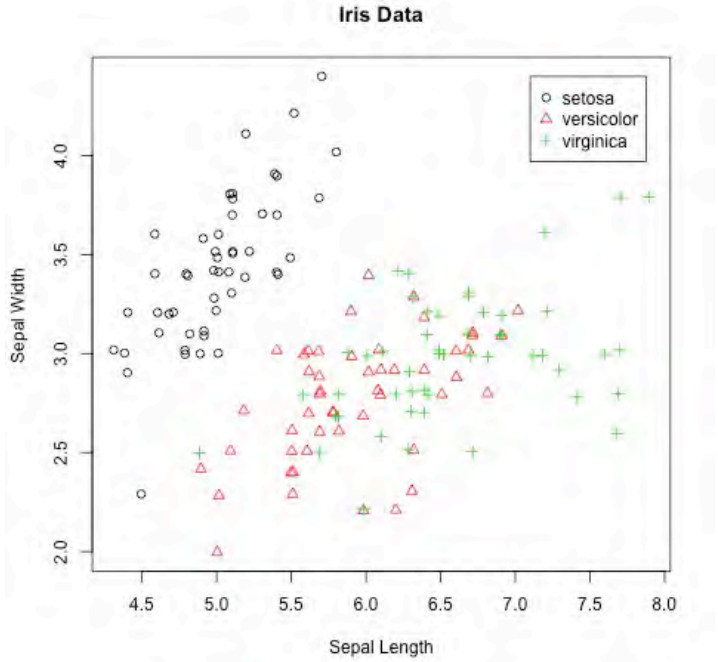
Scatterplot + labels

> with(iris, plot(jitter(Sepal.Length), jitter(Sepal.Width), col = Species, pch=as.numeric(Species), main = ("Iris Data"), xlab = "Sepal Length", ylab = ("Sepal Width")))



Scatterplot + legend

```
> # Follow the plot command with:
> with(iris, legend(7.1, 4.4, as.vector(unique(Species)),
  pch=unique(Species), col = unique(Species)))
```



Complete plot command

```
> with(iris, plot(jitter(Sepal.Length), jitter(Sepal.Width),
  col = Species, pch=as.numeric(Species), main = ("Iris
  Data"), xlab = "Sepal Length", ylab = ("Sepal Width")))
> with(iris, legend(7.1, 4.4, as.vector(unique(Species)),
  pch=unique(Species), col = unique(Species)))
```

Question 11

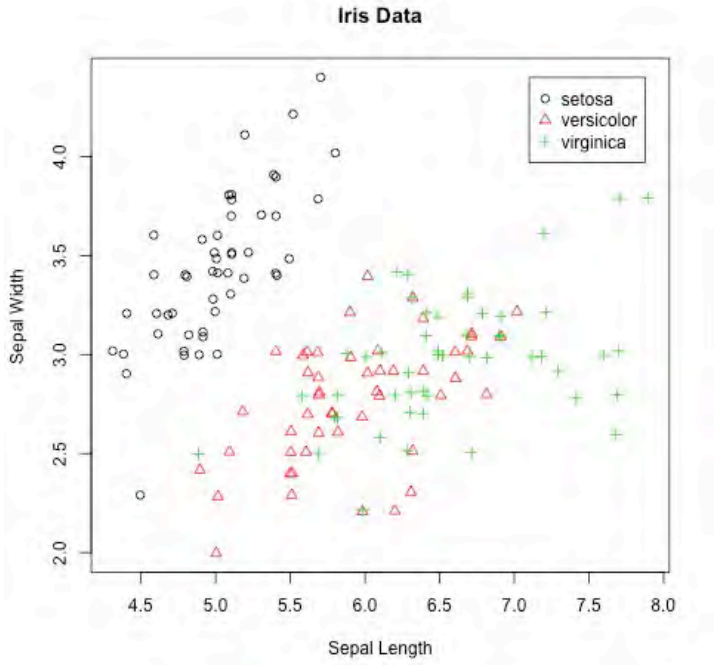
Which species is easiest to differentiate?

A. Versicolor

B. Virginica

C. Setosa

D. Too hard to tell.



Saving graphics

Diverting graphics from RStudio window to a file:

- The code below opens a file, diverts the output from RStudio to a named file (of type jpg in this case) and saves it in the working directory.

```
> jpeg("filename.jpg")
> plot(x,y) # put your plotting commands here
> dev.off()
```

- A simpler method is to use "Export" command under the plot tile in the "help/display" window in Rstudio.

Viewing correlation between variables

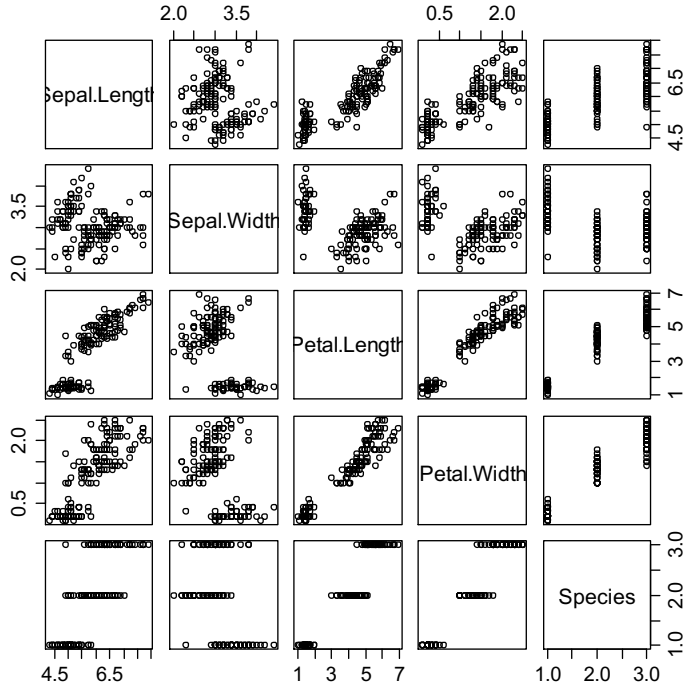
Correlation:

- Gives us an idea of the strength of the (linear) relationship between variables.
- Knowing the strength of this relationship lets us reduce the number of variables we need to analyse: that is, *if two variables are strongly correlated, we may only need to analyse one of them!*

All interactions: scatterplot matrix

The default method for a scatterplot matrix is

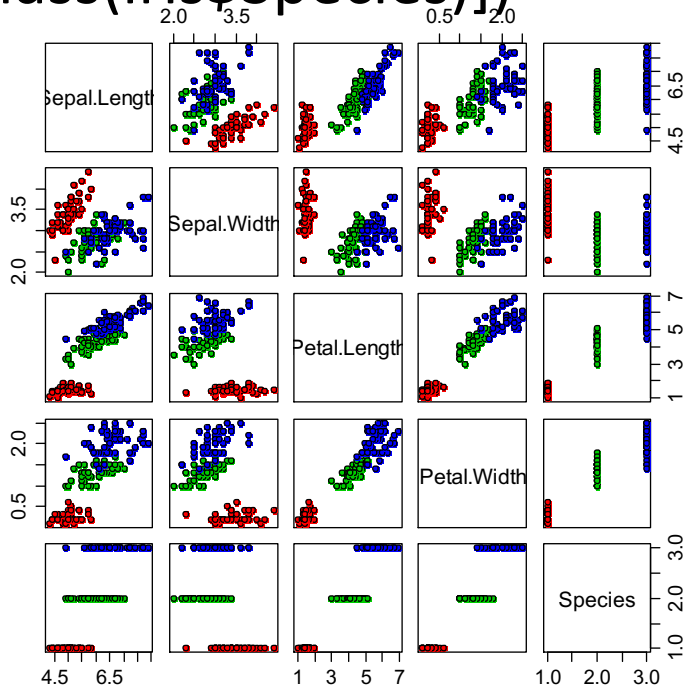
```
> pairs(iris)
```



Scatterplot matrix

Adding colour

> pairs(iris[1:5], pch = 21, bg = c("red", "green3", "blue")[unclass(iris$Species)])



Correlation matrix

The pairwise correlation between each numeric variable

```
> round(cor(iris[1:4]), digits = 3)
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
Sepal.Length	1.000	-0.118	0.872	0.818
Sepal.Width	-0.118	1.000	-0.428	-0.366
Petal.Length	0.872	-0.428	1.000	0.963
Petal.Width	0.818	-0.366	0.963	1.000

Correlation matrix – by factor

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
```

Seeing more variables: lattice

The lattice package has multi-panel graphing functions conditioned on variables, including:

> xyplot *# Multi-panel conditioning scatterplot*
> barchart *# Bar plot*
> dotplot *# Dot plot*
> splom *# Scatterplot matrix*
> bwplot *# Box-and-whisker plot*
> histogram *# Histogram*
> densityplot *# Smoothed histogram*

- See ATHR page 54

lattice

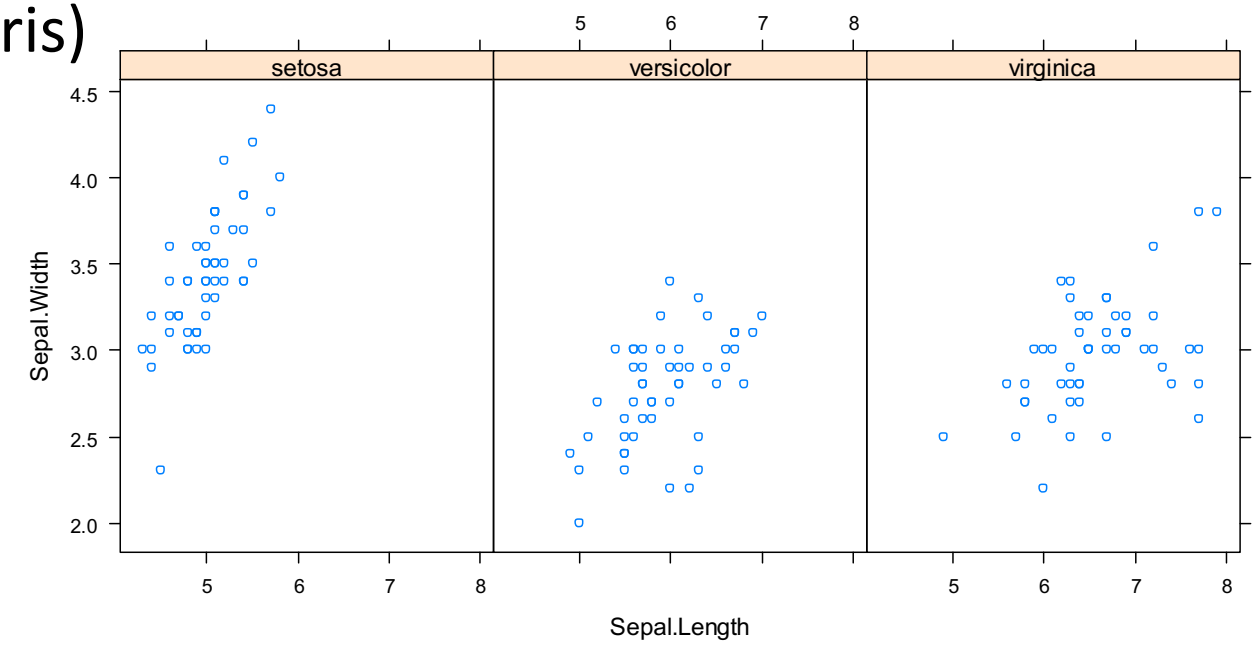
The lattice package comes with the base installation of R so to run:

```
> library(lattice)
```

xyplot

Conditioning on species:

- Syntax: xyplot(y ~ x | g) : *plot y on x grouped by g*
  - > xyplot(Sepal.Width ~ Sepal.Length | Species, data = iris)

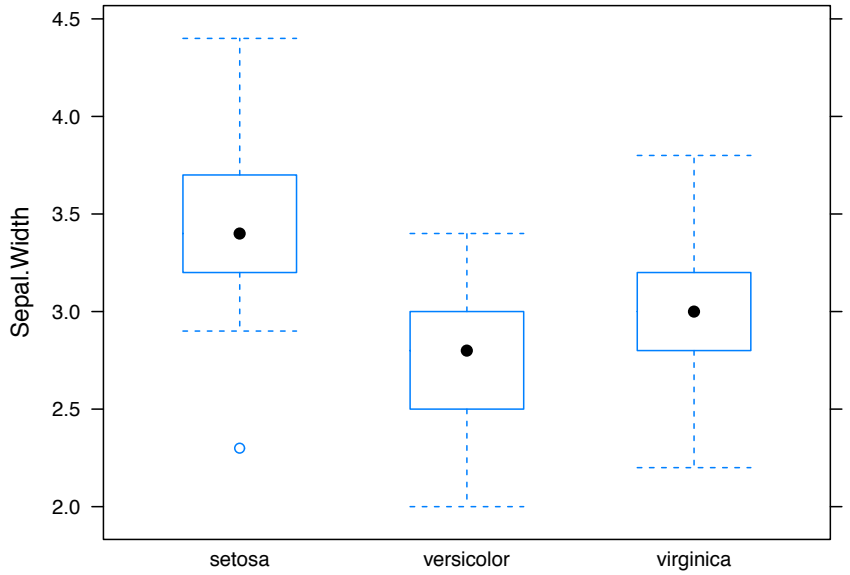


bwplot

Conditioning on species:

- Syntax: bwplot(y ~ g) : *plot y grouped by g*

  > bwplot(Sepal.Width ~ Species, data = iris)

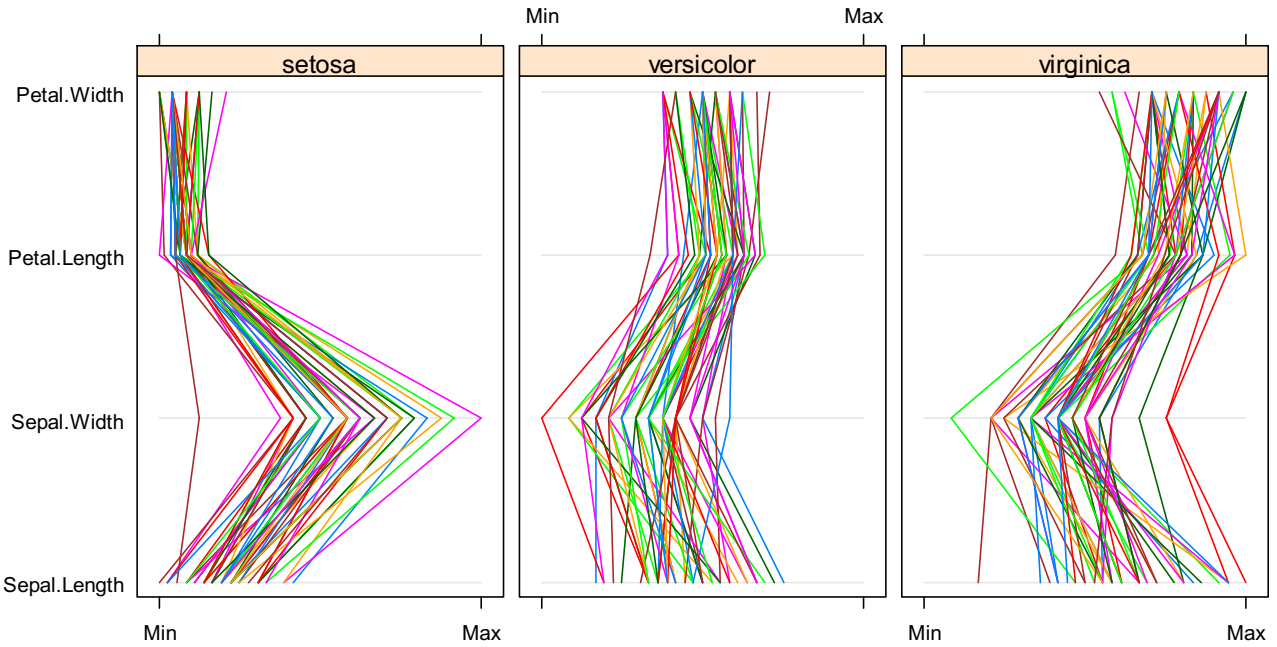


Parallel coordinates

Each data point plotted across 4 numeric variables

- Syntax: parallelplot(~y|g): *plot columns y grouped by g*

  > parallelplot(~iris[1:4] | Species, data = iris)



Presentation quality graphs

ggplot2

- One of the most commonly used packages for display quality graphics
- Written by Hadley Wickham and Winston Chang, it is an implementation of *The Grammar of Graphics* by Leland Wilkinson, and views a graphic as being made up of: data points + scales + annotations + statistical summaries… in a structured way, a grammar. See:

http://vita.had.co.nz/papers/layered-grammar.pdf

Book: ggplot2

Access the book via the Monash library, or

From the package website:

- ggplot2 is a plotting system for R, based on the grammar of graphics, which tries to take the good parts of base and lattice graphics...

https://ggplot2.tidyverse.org/

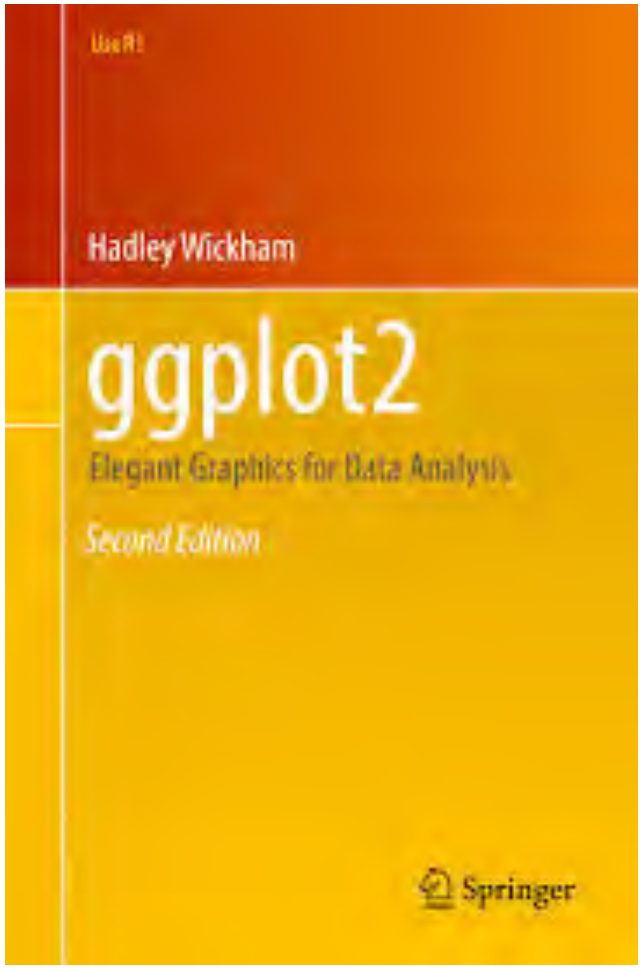
- Online help links from main page and is a useful reference. Many examples with code are given.

https://ggplot2.tidyverse.org/reference/

Book: R for Data Science

- A physical and web-based book by the author of ggplot2, Hadley Wickham, and Garrett Grolemund: https://r4ds.had.co.nz/
- The book takes you through all aspects of the data science workflow (more later)
- It has a good chapter on ggplot2, (Ch. 3) including the syntax for all plot types, for example:

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


Use R!
Hadley Wickham
ggplot2
Elegant Graphics for Data Analysis
Second Edition
Springer


O'REILLY
R for Data Science
Garrett Grolemund &
Hadley Wickham

ggplot2: graphic objects

Some main classes of graphic objects:

- Geoms (geometric objects: think of as type of plot)
- Statistics (summaries, data transformations)
- Scales/coordinate systems
- Faceting (conditional grouping of subsets of data)
- Position adjustments (jitter etc.)
- Annotation
- Aesthetics (colours, line styles etc.)

ggplot2

To run in the labs:

```
> # you may need to authenticate for Internet access
> # if using a browser in the computer laboratories
> # before installing packages...
> install.packages("ggplot2")
> library(ggplot2)
```

? qplot

- Quick plot

  ```
  qplot is the basic plotting function in the
  ggplot2 package ...
  ```

- Usage

  ```
  qplot(x, y = NULL, ..., data, facets = NULL,
  margins = FALSE, geom = "auto",
  stat = list(NULL), position = list(NULL),
  xlim = c(NA, NA), ylim = c(NA, NA),
  log = "", main = NULL,
  xlab = deparse(substitute(x)),
  ylab = deparse(substitute(y)), asp = NA)
  ```

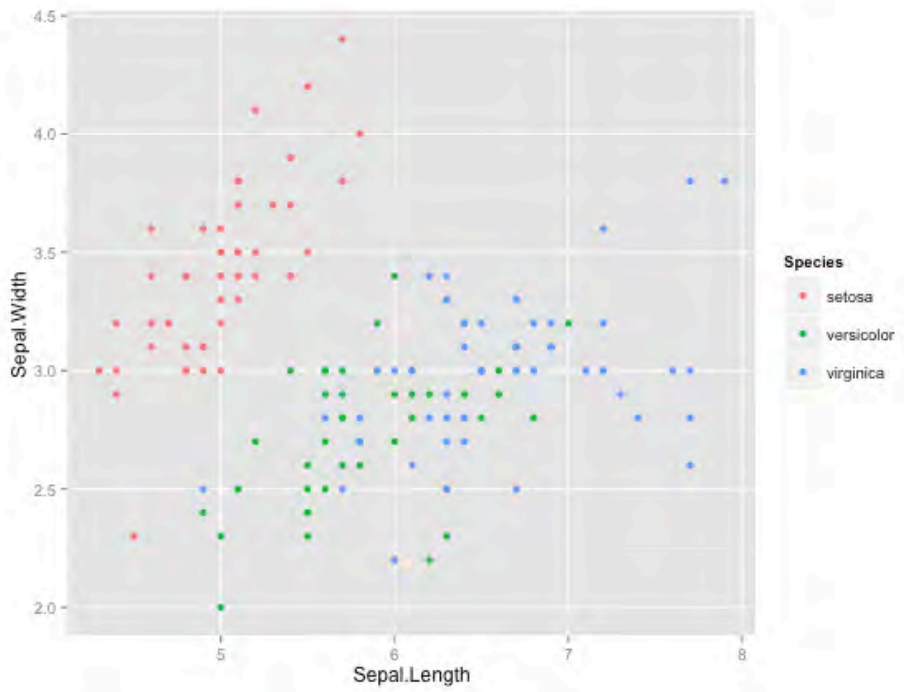
? qplot

- Arguments

```
x, y
...
data
facets
margins
geom
stat
position
xlim, ylim
log
main
xlab, ylab, asp
```

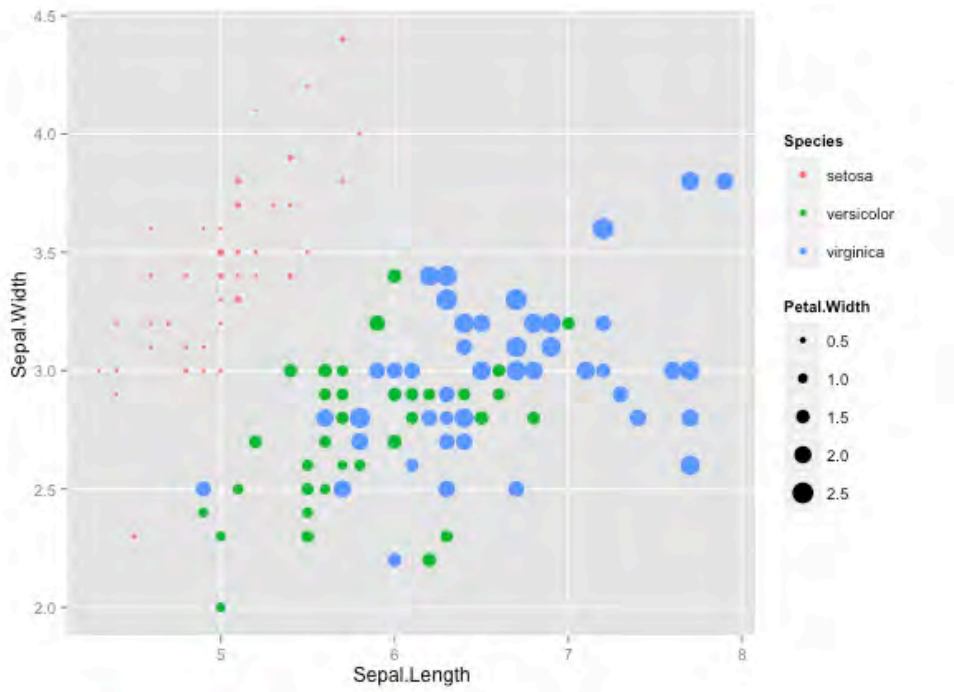
Basic qplot

> qplot(Sepal.Length, Sepal.Width, data = iris, color = Species)



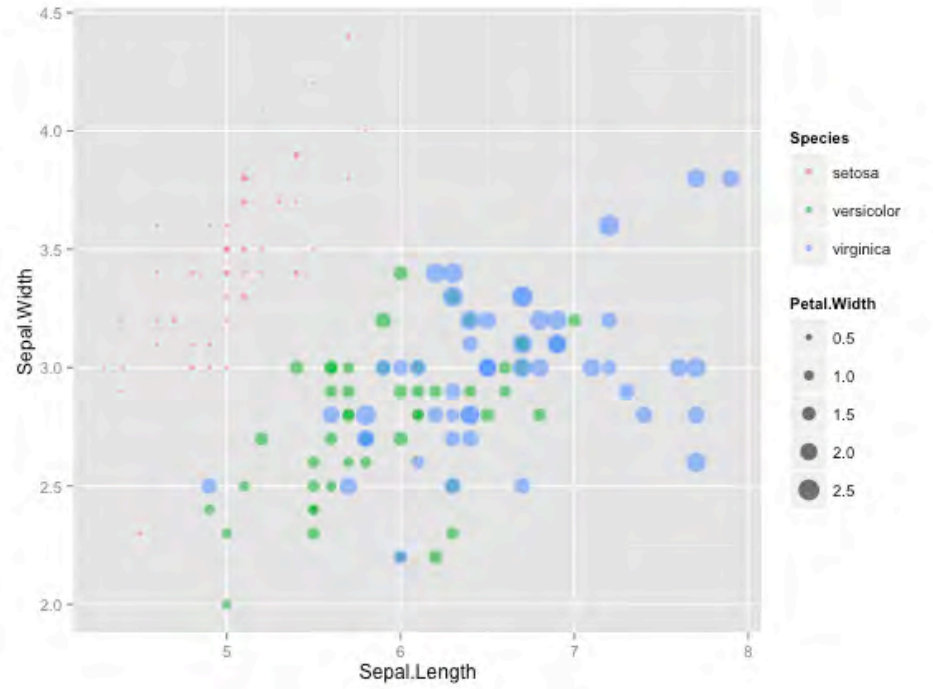
Basic qplot + size

> qplot(Sepal.Length, Sepal.Width, data = iris, color = Species, size = Petal.Width)



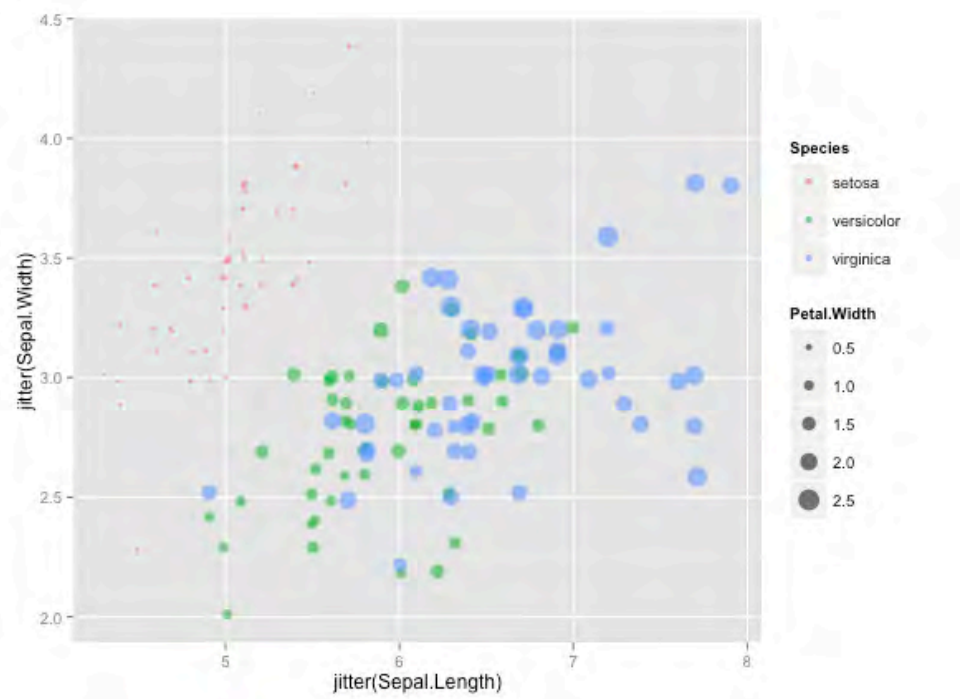
Basic qplot + size + alpha channel

> qplot(Sepal.Length, Sepal.Width, data = iris, color = Species, size = Petal.Width, alpha = I(0.6))



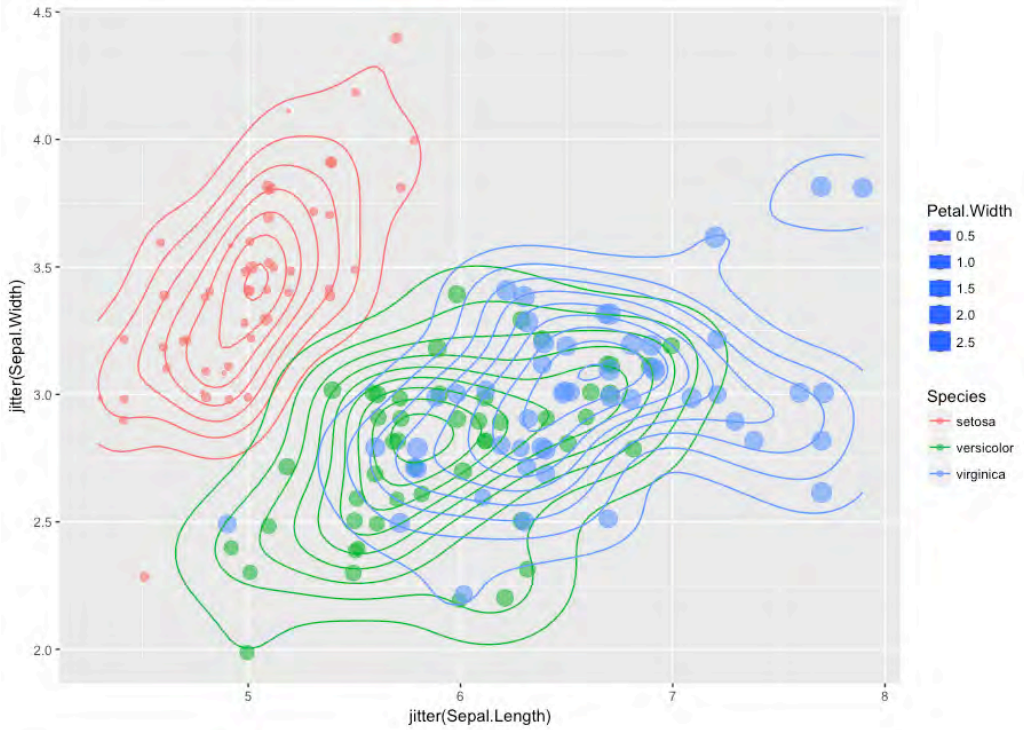
... + jitter

> qplot(jitter(Sepal.Length), jitter(Sepal.Width), data = iris, color = Species, size = Petal.Width, alpha = I(0.6))



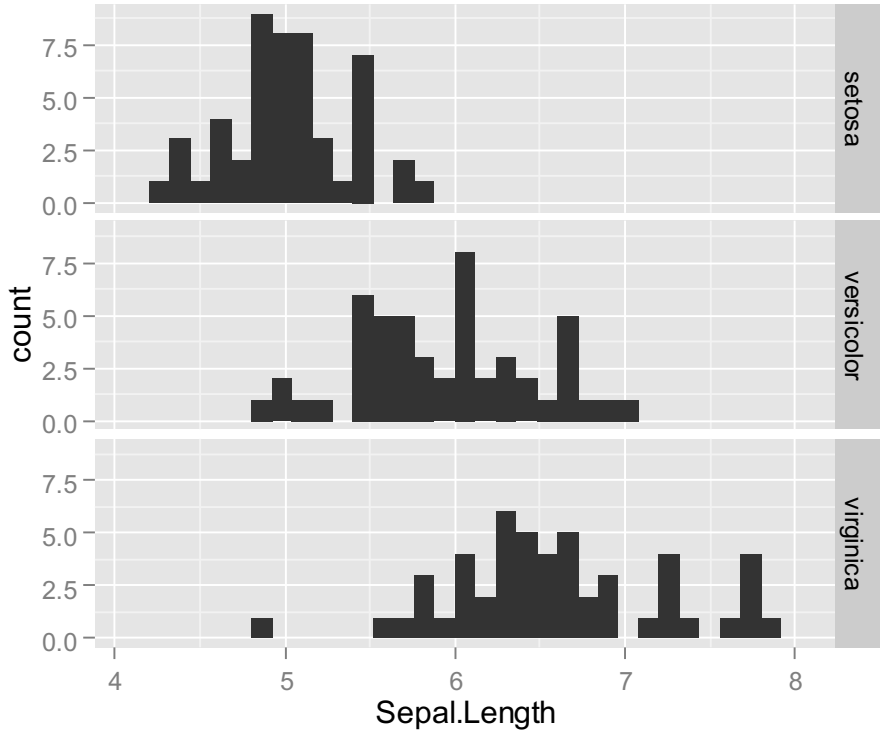
... + 2D contours

> qplot(jitter(Sepal.Length), jitter(Sepal.Width), data = iris, color = Species, size = Petal.Width, alpha = I(0.6)) + geom_density_2d() # one of many overlay stats



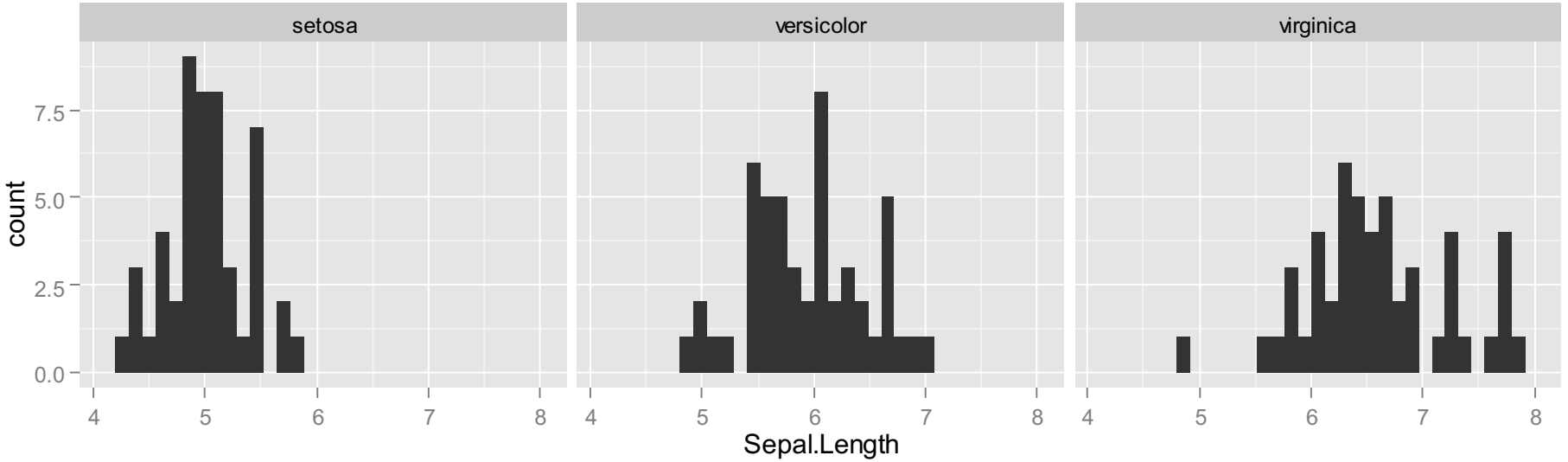
Histogram + facets

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



Histogram + facet_wrap

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



ggplot2: Grammar

Graphs are constructed first with a

- Geom, which specifies the type of plot and the data

Following this, aesthetic elements are added

- Statistics (summaries, data transformations)
- Scales/coordinate systems
- Faceting (conditional grouping of subsets of data)
- Position adjustments (jitter etc.)
- Annotation
- Aesthetics

Creating plots by name

Another way of specifying a plot is to create a plot object and then add components to it.

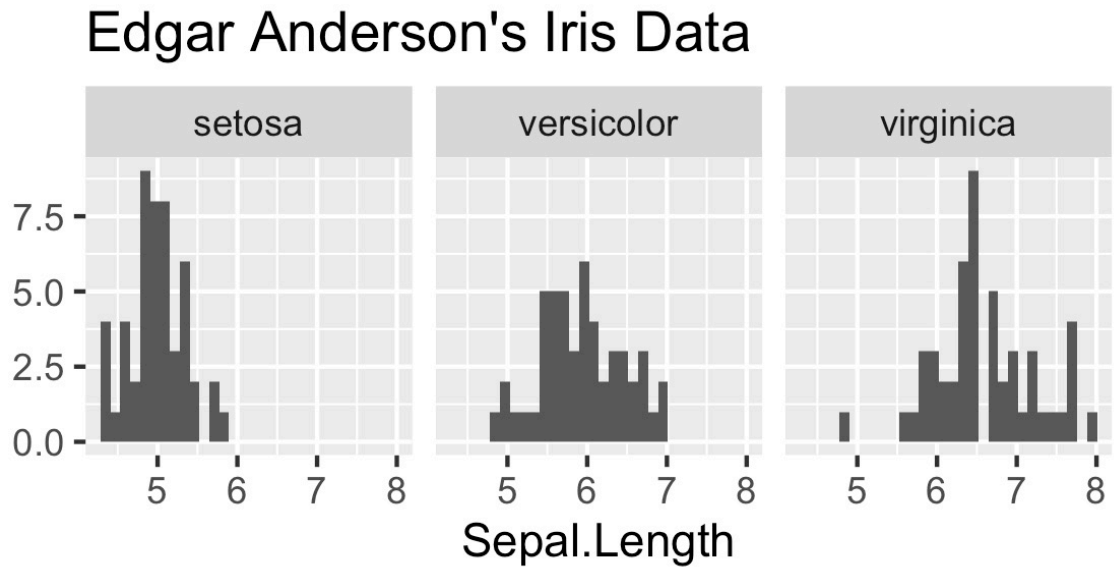
For the previous plot

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

Adding a title and saving

To add a title, and save:

```
> g <- g + ggtitle("Edgar Anderson's Iris Data")
> ggsave("EAI.jpg", g, width = 10, height = 5, units = "cm")
```



Homework activity

Download and install “ggplot2” and the “ggplot2movies” (containing the *movies* data set) packages. Attach the ‘movies’ data.

Are films getting longer over the years?

Graph film *length* against *year* as a scatterplot.

Set sensible time limit for *length*.

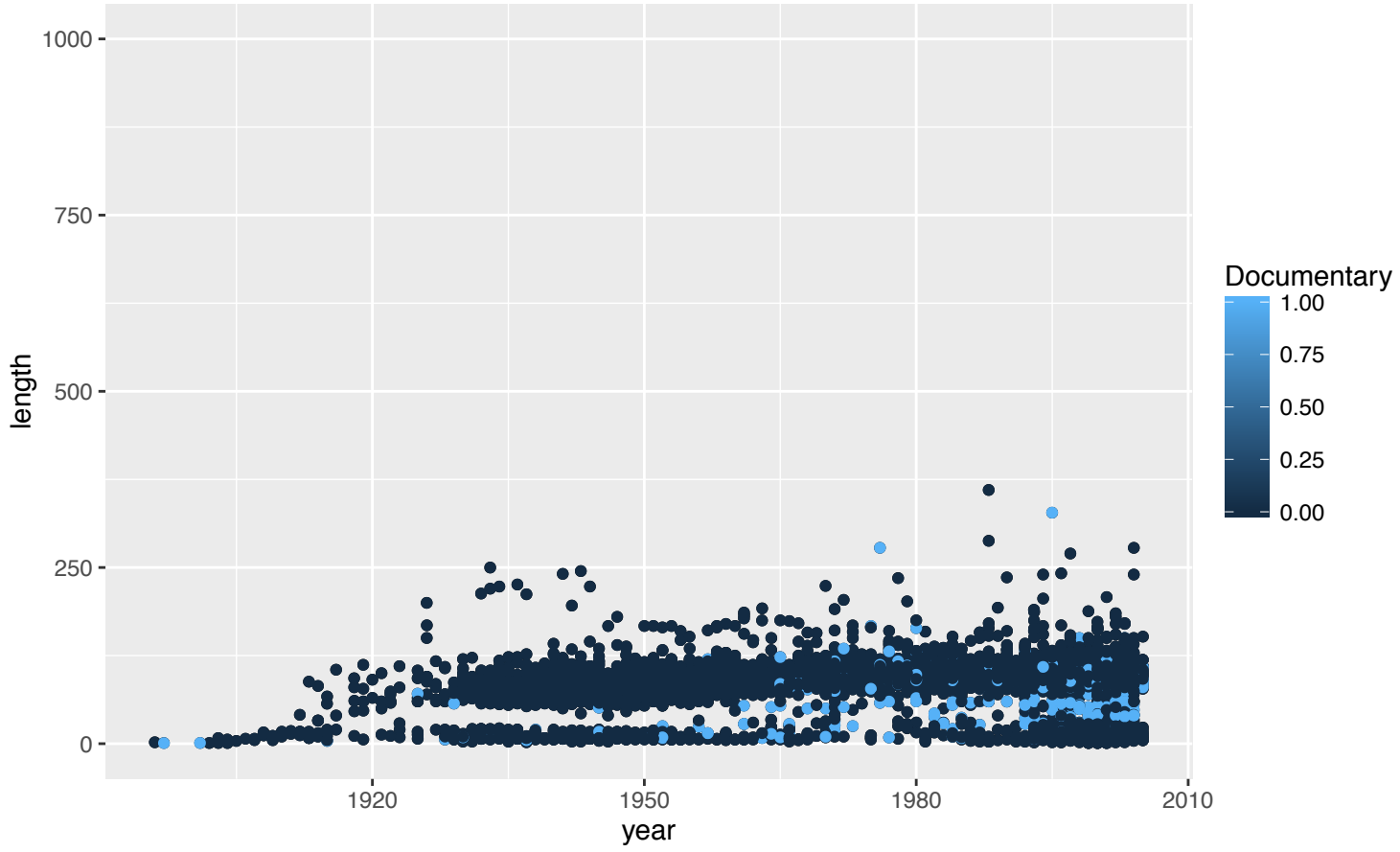
Set the colour of the plotted points to identify documentaries.

Homework activity – solution

```
> install.packages("ggplot2")
> library(ggplot2)
> install.packages("ggplot2movies") # new package
> library(ggplot2movies)
> str(movies) # this shows the structure of data set
> # sample of 5000 rows
> # this just makes plot sparser & plotting faster
> msmall <- movies[sample(nrow(movies), 5000), ]
```

Homework activity + ggsave

```
> msmall <- movies[sample(nrow(movies), 5000), ]
> attach(msmall)
> g = qplot(year,length)
> g
> g = g + ylim(0,1000)
> g
> g = g + geom_point(aes(colour = Documentary))
> ggsave("~/Desktop/msmall.jpg", g, width = 20, height
  = 12, units = "cm")
```


1000
750
500
250
0
length
1920
1950
1980
2010
year
Documentary
1.00
0.75
0.50
0.25
0.00

Next Week: Data manipulation in R

Read this week:

- A tour through the visualization zoo,
- R for Data Science, Chapter 3,
- Lecture 3 pre-reading.

From next week:

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 previous example as a script…

Scripts

ggsave example.R

Source on Save | Run | Source

```
1  # install.packages("ggplot2")
2  library(ggplot2)
3  # install.packages("ggplot2movies")
4  library(ggplot2movies)
5  str(movies) # this shows the structure of data set
6  # sample of 5000 rows
7  # this just makes plot sparser & plotting faster
8  msmall <- movies[sample(nrow(movies), 5000), ]
9  attach(msmall)
10 g = qplot(year,length)
11 g
12 g = g + ylim(0,1000)
13 g
14 g = g + geom_point(aes(colour = Documentary))
15 ggsave("~/Desktop/msmall.jpg", g, width = 20, height = 12, units = "cm")
16
```

Optional, but worth considering: esquisse (package)

https://cran.r-project.org/web/packages/esquisse/index.html

- A ‘shiny’ gadget to create ggplot2 charts interactively with drag-and-drop to map your variables.
- See the following YouTube video, for example:

https://www.youtube.com/watch?v=ih-WfAugyTk

References

Books – online from the Monash Library

- Wickham, H., ggplot2 elegant graphics for data analysis
- Wilkinson, L., and Wills, G., The grammar of graphics
- Rahlf, T., Data visualisation with R, Springer.

Paper by Wickham: Layered grammar of graphics

http://vita.had.co.nz/papers/layered-grammar.pdf

A tour through the visualization zoo

https://dl.acm.org/doi/10.1145/1743546.1743567

ggplot2 Cheat Sheet

https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf