Exploring data with R

What it is, what you can do with it

The R Programming Language

* Good at:

    * Rapid statistical analysis (4000+ R libraries)

    * Rapidly-created static graphics

* Bad at:

    * Web applications (although you can use Shiny)

    * Pretty interactive graphics (use D3)

    * Iteration (use almost anything else)

Download R from http://www.r-project.org/

Starting R

- 4 ways to run R:

    - Terminal window: type "r" (and "q()" to quit)

    - Rstudio: click on Rstudio tool

    - R script from the command line: "R <filename.r —no-save"

    - R script from inside R: source('myscript.r')

    - From Python: use the rpy2 library

Getting help

- Inside R or Rstudio:
    - help(functionname)
    - example(functionname)
    - e.g. help(sum), example(sum)
- R community:
    - R bloggers
    - R local user groups
    - R mailing lists
    - Stack overflow

Variables

- data types:
  - 2.456
  - 'This is a string'
  - TRUE, FALSE
- Variables:
  - x <- 15
  - x/4
  - y <- 'This is also a string!'
  - print(x)

Vectors

- Vector = same as Python's list
    - a <- c(1,5,2)
    - note the 'c' function ('combine')
- NB if you try creating a mixed-variable vector, R will create a vector of strings. Try this:
    - x <- c(1,'five',2,TRUE)
- Indices:
    - x[3]
    - x[2] <- 'four'
    - x[2:4]
    - NB: R indices start at 1 (unlike Python, which start at 0)

Sequences

- 1:5

- seq(1,5)

- seq(1,5,0.3)

- x[2:4]

Functions
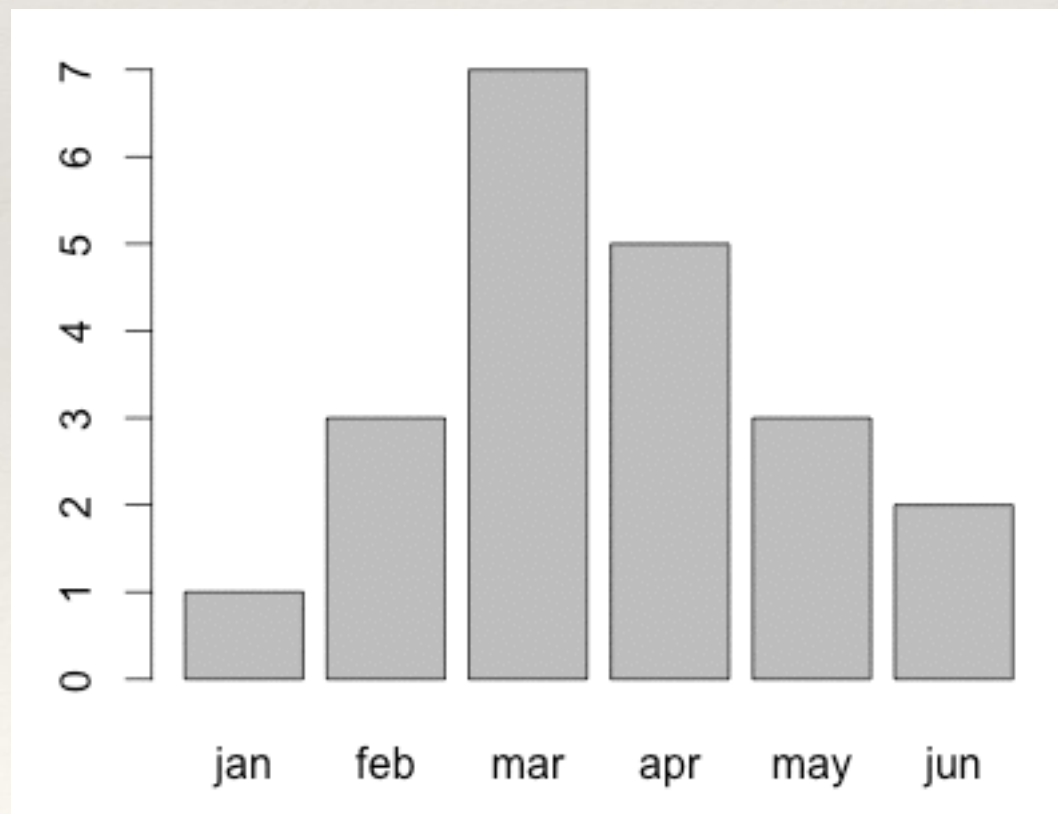
- sum(1,2,3,7,3,4)

- a <- c(1,2,3,7,3,4)

- sum(a)

- max(a)

- sqrt(a)

Vector names

❖ You can name each entry in your vector:

   ❖ x <- c(1,'five',2,TRUE)

   ❖ names(x) = c('1st', '2nd', 'third', 'rainbow')

   ❖ x['2nd']

   ❖ x['rainbow'] <- FALSE

Visualisations

- b <- c(1,3,7,5,3,2)

- names(b) <- c('jan','feb','mar','apr','may','jun')

- barplot(b)

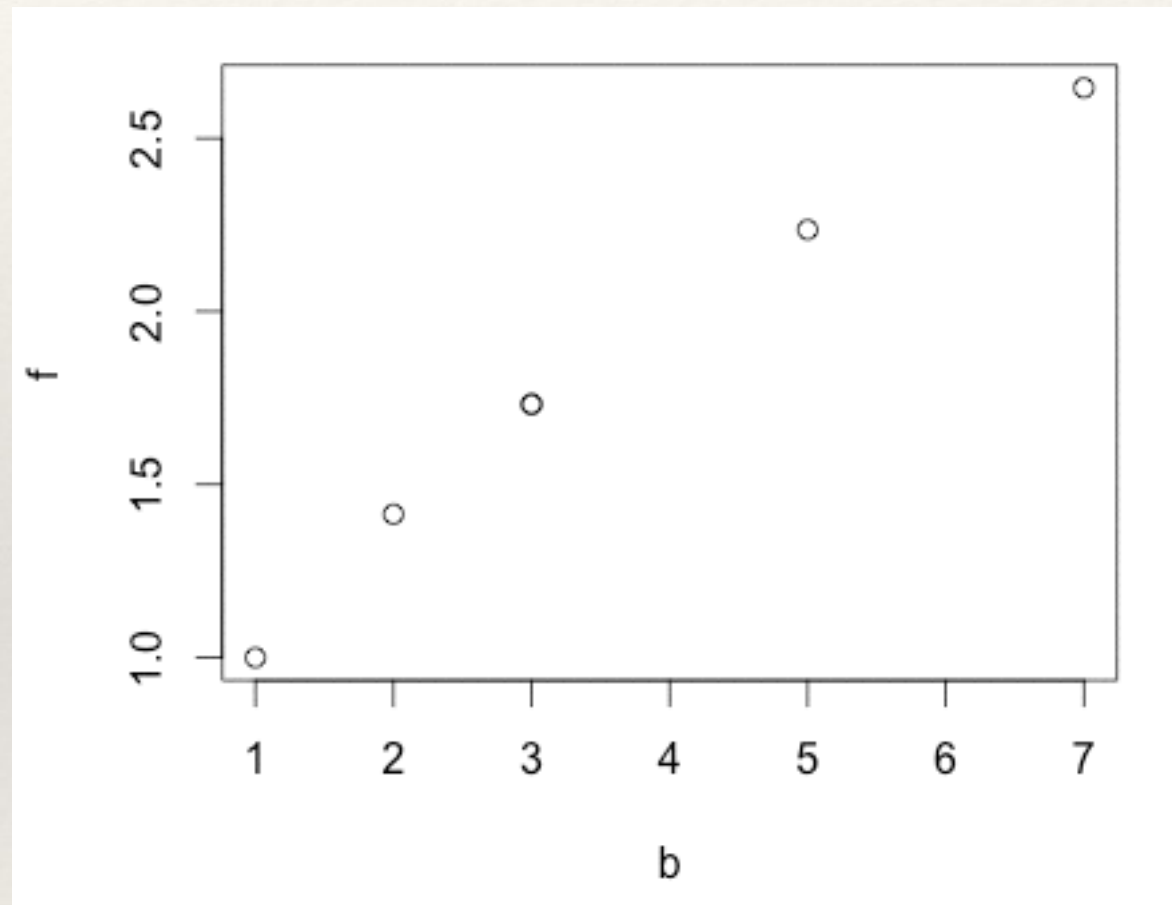# Vector Maths

- b <- c(1,3,7,5,3,2)

- b+1

- b/4

- sqrt(b)

- d <- c(3,6,5,4,3,7)

- b*d

- b == d

More visualizations

❖ b <- c(1,3,7,5,3,2)

❖ f <- sqrt(b)

❖ plot(b,f)

Handling missing values
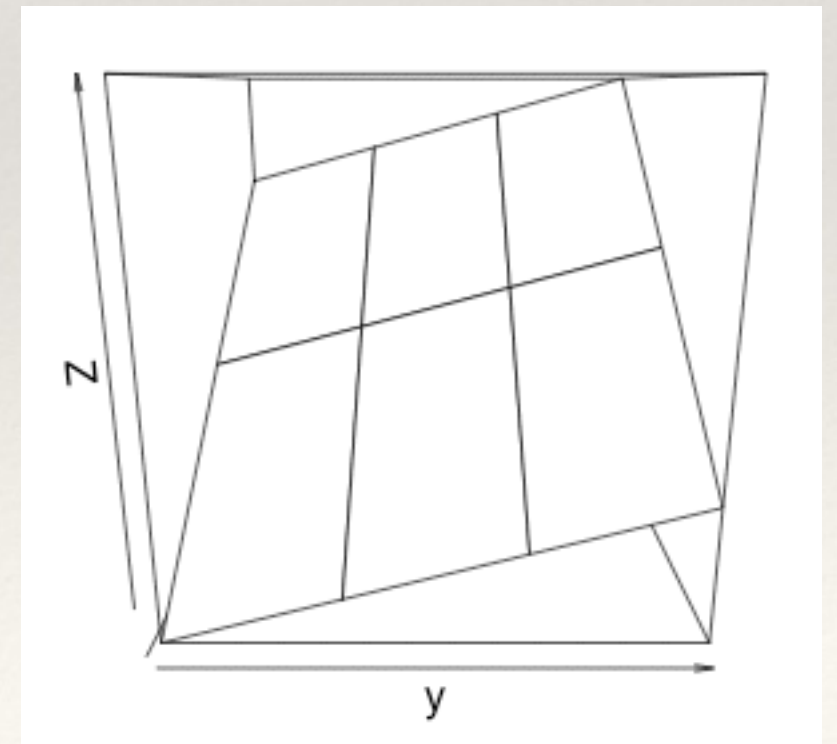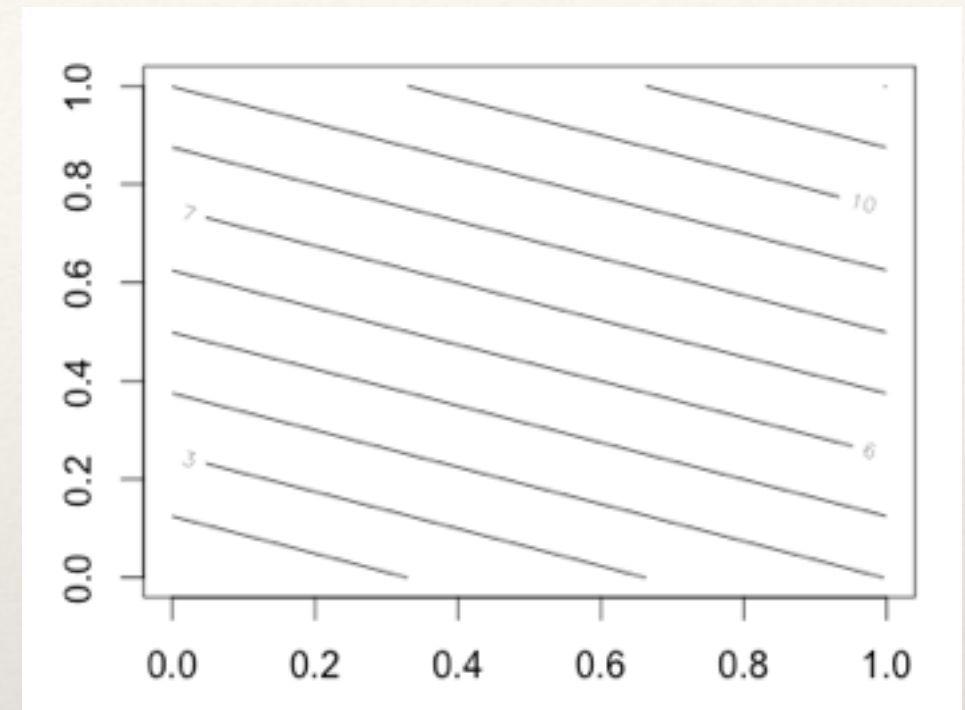
❖ b <- c(1,3,7,NA,3,2)

❖ sum(b)

❖ sum(b, na.rm=TRUE)

Matrices (two-dimensional arrays)

- ❖ x <- matrix(0, 4, 3)

- ❖ y <- matrix(1:12, 4, 3)

- ❖ y[2,3] <- 15

- ❖ y[,3]

- ❖ y[2,2:3]

- ❖ print(y[1,3])

- ❖ z <- 1:15

- ❖ dim(z) <- c(3,5)

- ❖ print(z)

More visualisation

- y <- matrix(1:12, 4, 3)

  - contour(y)

  - persp(y)

- R comes with example datasets:
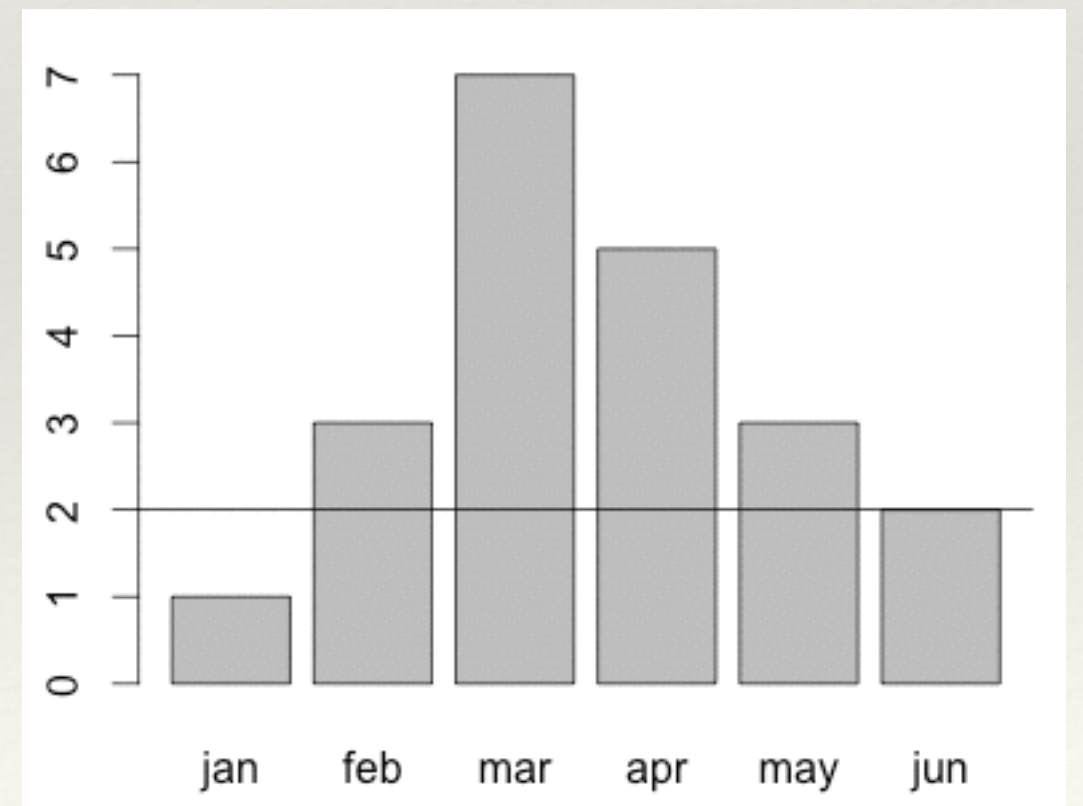
  - contour(volcano)

  - persp(volcano)

Getting Statistics

- ❖ b <- c(1,3,7,5,3,2)

- ❖ mean(b)

- ❖ median(b)

- ❖ sd(b)

Adding to visualizations

- ❖ b <- c(1,3,7,5,3,2)

- ❖ names(b) <- c('jan','feb','mar','apr','may','jun')

- ❖ barplot(b)

- ❖ abline(h=2)

Factors

- a <- c('developed', 'transitioning', 'developed', 'developing', 'developed', 'developing', 'developed')

- classes <- factor(a)

- print(classes)

- print(levels(classes))

DataFrames (tables)

- refugees = c(23,175,15543,11,338796,1244,63,53)

- asylumseekers = c(140,77,2872,6,255,91,6,7)

- origins = c('Burundi','Dem. Rep. of the Congo', 'Eritrea', 'Rwanda', 'Somalia','South Sudan','Sudan','Uganda')

- countries <- factor(origins)

- popstats <- data.frame(refugees, asylumseekers, countries)

- print(popstats)

- print(popstats$asylumseekers)

Reading CSV files into Dataframes

❖ List all files:

  ❖ list.files()

❖ Read a CSV into a data frame:

  ❖ refugees <- read.csv('popstats_clean.csv')

❖ Read a tab-separated file into a data frame:

  ❖ population <- read.table('sp_pop.tsv', sep='\t', header=TRUE)

Exploring a Dataframe

❖ refugees <- read.csv('popstats_clean.csv')

❖ dim(refugees)

❖ **str(refugees)**

❖ head(refugees)

❖ tail(refugees)

Merging dataframes

❖ merge(x = refugees, y = population, by.x='Country.of.origin', by.y='Country.Name')

❖ Spaces in headings got converted to dots

❖ Use by.x, by.y *unless* you have at least one column with the same name in both files

Science: correlation

❖ plot(column1, column2)

❖ cor.test(column1, column2)

❖ Is the p-value smaller than 0.05? Yes = strong evidence for correlation.

Science: linear models

- plot(column1, column2)

- line <- lm(column2 ~ column1)

- abline(line)

R libraries

❖ R libraries are called "packages". They live here: http://cran.r-project.org/

❖ To install package x, start R and type:

    ❖ install.packages('x')

❖ To get information about a package, type:

    ❖ help(package = 'x')

❖ You'll probably want these packages:

    ❖ ggplot2

    ❖ plyr

    ❖ xlsx

Using ggplot2

❖ refugees = c(23,175,15543,11,338796,1244,63,53)

❖ asylumseekers = c(140,77,2872,6,255,91,6,7)

❖ origins = c('Burundi','Dem. Rep. of the Congo', 'Eritrea', 'Rwanda', 'Somalia','South Sudan','Sudan','Uganda')

❖ countries <- factor(origins)

❖ library(ggplot2)

❖ qplot(refugees, asylumseekers, color = countries)
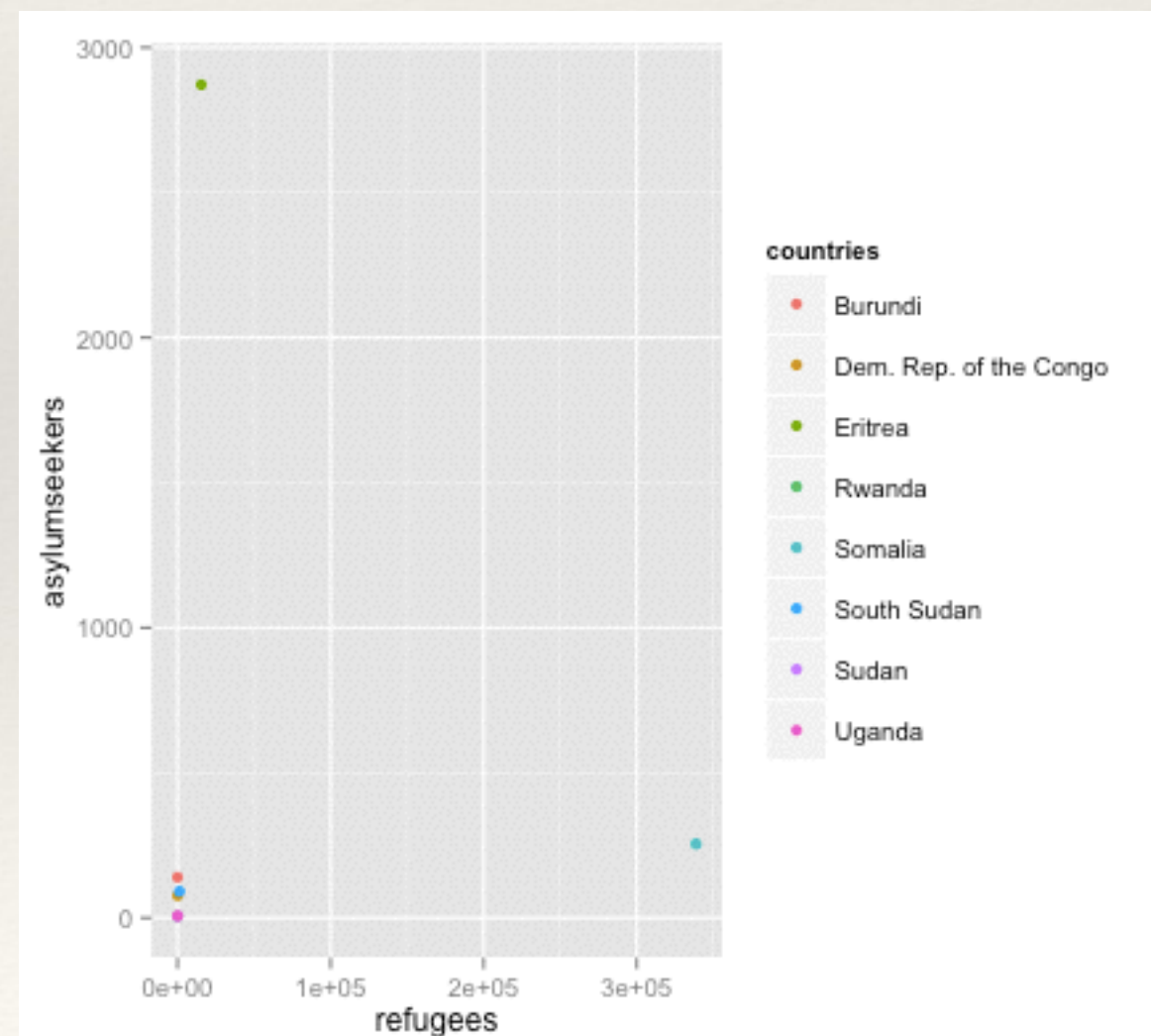
Continuing your R Journey

❖ Websites:

  ❖ http://www.cookbook-r.com/

  ❖ http://www.statmethods.net/

  ❖ The community sites from slide 3

❖ Books:

  ❖ Paul Teetor, "The R cookbook"

  ❖ Examples in O'Neil's "Doing data science"