Using R for data analysis

Daniel Müllensiefen
Goldsmiths, University of London

August 18, 2009

R is good for

- Flexible Data Analysis (programmable)

R is good for

- Flexible Data Analysis (programmable)
- Using different analysis techniques

R is good for

- Flexible Data Analysis (programmable)
- Using different analysis techniques
- Data Visualisation

R is good for

- Flexible Data Analysis (programmable)
- Using different analysis techniques
- Data Visualisation
- Numeric Accuracy

R is good for

- Flexible Data Analysis (programmable)
- Using different analysis techniques
- Data Visualisation
- Numeric Accuracy
- Rapid prototyping of analysis / process models

R is good for

- Flexible Data Analysis (programmable)
- Using different analysis techniques
- Data Visualisation
- Numeric Accuracy
- Rapid prototyping of analysis / process models
- Pre-processing data from different sources
  - textfiles (.txt) and binary files (e.g. SPSS .sav, Excel)

R is good for

- Flexible Data Analysis (programmable)
- Using different analysis techniques
- Data Visualisation
- Numeric Accuracy
- Rapid prototyping of analysis / process models
- Pre-processing data from different sources
  - textfiles (.txt) and binary files (e.g. SPSS .sav, Excel)
  - Audio files

R is good for

- Flexible Data Analysis (programmable)
- Using different analysis techniques
- Data Visualisation
- Numeric Accuracy
- Rapid prototyping of analysis / process models
- Pre-processing data from different sources
  - textfiles (.txt) and binary files (e.g. SPSS .sav, Excel)
  - Audio files
  - databases

R is good for

- Flexible Data Analysis (programmable)
- Using different analysis techniques
- Data Visualisation
- Numeric Accuracy
- Rapid prototyping of analysis / process models
- Pre-processing data from different sources
  - textfiles (.txt) and binary files (e.g. SPSS .sav, Excel)
  - Audio files
  - databases
  - texts (linguistic data)

R is considered less good for

- Graphical User Interfaces

R is considered less good for

- Graphical User Interfaces
- Internet programming

R is considered less good for

- Graphical User Interfaces
- Internet programming
- Low-level programming

R is considered less good for

- Graphical User Interfaces
- Internet programming
- Low-level programming
- ...

R compares to

- Matlab (open source, community driven, not commercial)

R compares to

- Matlab (open source, community driven, not commercial)
- SPSS, SAS, Stata (programming language, not program)

R compares to

- Matlab (open source, community driven, not commercial)
- SPSS, SAS, Stata (programming language, not program)
- Weka (driven by community, not individuals)

R compares to

- Matlab (open source, community driven, not commercial)
- SPSS, SAS, Stata (programming language, not program)
- Weka (driven by community, not individuals)
- SciPy and other software libraries (entire language specialised for data analysis)

Pros and Cons

- Huge community support

Pros and Cons

- Huge community support
- Cross-plattform and command-line based

Pros and Cons

- Huge community support
- Cross-plattform and command-line based
- Interactive: interpreted not complied

Pros and Cons

- Huge community support
- Cross-plattform and command-line based
- Interactive: interpreted not complied
- Mainly functional

How R came about

- 1976: John Chambers releases 1st version of S: Language for statistics, stochastic simulation and data visualisation

How R came about

- 1976: John Chambers releases 1st version of S: Language for statistics, stochastic simulation and data visualisation
- 1995: Ross Ihaka and Robert Gentleman release R as GPL

How R came about

- 1976: John Chambers releases 1st version of S: Language for statistics, stochastic simulation and data visualisation
- 1995: Ross Ihaka and Robert Gentleman release R as GPL
- 1998: Comprehensive R Archive Network (CRAN) founded

How R came about

- 1976: John Chambers releases 1st version of S: Language for statistics, stochastic simulation and data visualisation
- 1995: Ross Ihaka and Robert Gentleman release R as GPL
- 1998: Comprehensive R Archive Network (CRAN) founded
- 2001: R News published for 1st time

How R came about

- 1976: John Chambers releases 1st version of S: Language for statistics, stochastic simulation and data visualisation
- 1995: Ross Ihaka and Robert Gentleman release R as GPL
- 1998: Comprehensive R Archive Network (CRAN) founded
- 2001: R News published for 1st time
- 2004: 1st *useR!* conference

How R came about

- 1976: John Chambers releases 1st version of S: Language for statistics, stochastic simulation and data visualisation
- 1995: Ross Ihaka and Robert Gentleman release R as GPL
- 1998: Comprehensive R Archive Network (CRAN) founded
- 2001: R News published for 1st time
- 2004: 1st *useR!* conference
- 2009: More than 1000 packages available on CRAN

Basic data in and out

- Start R

Basic data in and out

- Start R
- Save file from http://www.doc.gold.ac.uk/~mas03dm/teaching/r/iris.data.txt to R's working directory (using `getwd()`

Basic data in and out

- Start R
- Save file from http://www.doc.gold.ac.uk/~mas03dm/teaching/r/iris.data.txt to R's working directory (using `getwd()`
- Get data into R using the read.table command (useful operations `help(read.table)` and assignment operator "←")

Basic data in and out

- Start R
- Save file from http://www.doc.gold.ac.uk/~mas03dm/teaching/r/iris.data.txt to R's working directory (using `getwd()`
- Get data into R using the read.table command (useful operations `help(read.table)` and assignment operator "←")
- Change the species label of the 3rd observations to your own first name (using the indexing function `[ , ]`), save this dataset (using `write.table()`)

Basic data in and out

- Start R
- Save file from http://www.doc.gold.ac.uk/~mas03dm/teaching/r/iris.data.txt to R's working directory (using `getwd()`
- Get data into R using the read.table command (useful operations `help(read.table)` and assignment operator "←")
- Change the species label of the 3rd observations to your own first name (using the indexing function `[ , ]`), save this dataset (using `write.table()`)
- Remove the altered dataset (using `rm()`) and get the original dataset in again

Data summary and plots

- Summarise dataset (`summary()`, `str()`)

Data summary and plots

- Summarise dataset (`summary()`, `str()`)
- Plot 1st column vs 2nd column (`plot()`)

Data summary and plots

- Summarise dataset (`summary()`, `str()`)
- Plot 1st column vs 2nd column (`plot()`)
- Attach dataset to search path (`attach()`)

Data summary and plots

- Summarise dataset (`summary()`, `str()`)
- Plot 1st column vs 2nd column (`plot()`)
- Attach dataset to search path (`attach()`)
- Plot *Species* vs *Petal.Width* and give graph a title and axes names

Data summary and plots

- Summarise dataset (`summary()`, `str()`)
- Plot 1st column vs 2nd column (`plot()`)
- Attach dataset to search path (`attach()`)
- Plot *Species* vs *Petal.Width* and give graph a title and axes names
- Plot histogram of *Petal.Length* (`hist()`

Data summary and plots

- Summarise dataset (`summary()`, `str()`)
- Plot 1st column vs 2nd column (`plot()`)
- Attach dataset to search path (`attach()`)
- Plot *Species* vs *Petal.Width* and give graph a title and axes names
- Plot histogram of *Petal.Length* (`hist()`
- Plot scattergram of full dataset (`plot(dataset,col=Species)`)

Data summary and plots

- Summarise dataset (`summary()`, `str()`)
- Plot 1st column vs 2nd column (`plot()`)
- Attach dataset to search path (`attach()`)
- Plot *Species* vs *Petal.Width* and give graph a title and axes names
- Plot histogram of *Petal.Length* (`hist()`
- Plot scattergram of full dataset (`plot(dataset,col=Species)`)
- Add non-parametric smoother (`plot(dataset, col=Species, panel=panel.smooth)`)

More plots and a function

- Do `boxplot(Petal.Length Species,notch=TRUE)`. What are the notches?

More plots and a function

- Do `boxplot(Petal.Length Species,notch=TRUE)`. What are the notches?
- Set the graphical device to be split into a 2x2 panel: `op` $\leftarrow$ `par(mfrow = c(2,2)`