The Analysis Workflow

Cradle to Grave Care for Your Data

Leo Schalkwyk

SGDP, IoP

SGDP Summer School

Outline

Use informative sample names

- you need to be involved in the experiment from the start
- make sure samples are properly randomised or blocked
- also take the opportunity to agree naming conventions
  - LCS_ddmmyyyy_control_C57BL
  - no spaces saves trouble
  - best to start with a letter
  - `[A-Za-z0-9_]`

Derive analysis factors automagically

- try to plan for a workflow that does not involve collating information by hand
- ideally the main analysis factors fall out of sample identifiers
  - `t(data.frame(strsplit(colnames(data), '_')))`
- there will be errors, it's important to keep them to a minimum
- use double entry or two-person proofing for anything hand-entered

Get data in a single storage place

- excel is the most common data curation tool for students
- it has subtle and not-so subtle limitations
  - there is no record of edits or manipulations
  - it is very easy to sort a single column
  - typically communicated by replication (email)
- consider a database if data is at all complex
  - MS access is actually not bad
  - MySQL isn't that hard either
  - it's easier to do database operations (merging data) with a database
  - it's important not to have multiple versions of data

Load Data from Central Store

- if possible it's a good practice to read data directly from the horse's mouth
  - read xls or csv locally or from URL
  - connect to MySQL database
- set `options(stringsAsFactors=FALSE)`
  - the default behaviour is to convert character data to factors
  - this is slow for large data
  - can be confusing when the data is supposed to be numeric
- keep informative row and column names

Check Data Type and Missingness

- `str(data)` gives a list of data columns and their types
- tells you if you need to modify your data reading command
  - are there usable column names?
  - are there usable row names?
  - are numeric columns labelled `int` or `num` ?
  - are the number of rows and columns what you expect?
- `summary(data)`
  - 5-number summary for numeric columns
  - count of each level for factors
  - length for characters
- `table(data$categories)`

Check Distribution and Missingnesss

- surfeit of zeroes, 99s, NAs or other oddities flag problems
- `boxplot(volcano)` - graphical equivalent of summary
- `boxplot(weight ~feed, chickwts)`
- `plot(density(volcano[,1]))`
  - kernel density plot often nicer than a histogram `hist()`
  - you can add further columns:
    `lines(density(volcano[,2]),col=2)`
- is the data normally distributed? `qqnorm(volcano[,50])`

Correlation

- scatterplots can be quite informative if the frame is not huge
- `plot(DNase)`
- pairwise relationships are summarized by correlation
- correlation matrix: `cor(longley)`
  - above about 5 variables correlation matrices are hard to understand
  - `image(cor(USArrests))`
  - `hclust(dist(t(USArrests)))`
- PCA is another way to dissect correlations
- sometimes you can identify mislabelled samples
  - correct these in your central data store

History and Workspace Files

- you can save the data from a session
  `save.image('myproject.RData')`
- you can also save the command history
  `savehistory('myproject.Rhistory')`
- you can load these back up and continue where you left off
- this does not constitute proper record keeping
- it can be time-consuming to reconstruct your analysis

Notes

- it's important to keep notes
  - any way that works for you
  - I usually use a MS word document with graphs pasted in
- this keeps an explicit record of the rationale and methods

Scripts

- R history files contain all the mistakes and false starts
- this usually makes them unsuitable to be run directly
- they can be edited into usable scripts
  - delete rubbish
  - debug
  - document with many comments
- the analysis script plus data allow the results to be repeated
  - by yourself if you need to change something
  - by others , who can see every detail

Version Control

- if your work is at all complicated and/or collaborative
- if your memory is not perfect
- if your hardware is not infallible

you should consider using a version control system

- subversion
- git