STATS 769
Large Data Problems

Paul Murrell

The University of Auckland

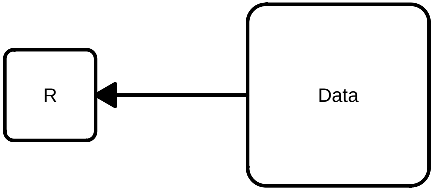
August 4, 2021

Overview

- This lecture explores how to determine the size of problems (in terms of computer memory) in relation to the size of our computational resources.
- “Large” means that a software tool or a computer that we know how to use cannot cope with the data set.

Problem 1

- Our standard tool (R) cannot hold all of the data.



Problem 0

- How do we know that we have a problem?
  - How do we know that the data are too large?
  - How do we know how much RAM we have available?
  - How do we know how much RAM we have used?
  - How do we know how much RAM we need to use?
- Just try-it-and-see is not necessarily a good idea
  (R will expand into shared memory and "thrash" your machine)
- Thrashing your machine is even worse when it is not just your machine (in a multi-user, shared-resource environment)

Calculating memory requirements

On Linux, we can use shell commands to ask the operating system how big the data are and how much memory is available.

- Use `ls -l` to determine the file size.
- Use `du -s` to determine directory size.
- Use `wc` to count the number of lines in a (text) file.
- Use `head` to view the first few lines of a (text) file.
- Use `free` or `top` to show how much RAM is available (and use `top` to monitor memory usage).
- Use `df` to show how much hard drive is available (different mount points can have very different limits).

On Windows we might use “Task Manager”.

Calculating memory requirements

We need to know how R uses memory.

- Data must fit in RAM.
- R automatically allocates new memory as required.
- R reclaims memory through a “garbage collector.”
- R has a limited set of data types (numeric, character, logical, and complex).

Calculating memory requirements

We can ask R how much memory it is using.

- Use `object.size()` to determine the RAM used by an object.
- Use `gc()` to show **maximum** RAM used by R (and to release memory).
- Use `read.table(nrows=)` to see the first few rows (and the number of columns).
- Use `rm()` to remove objects (and release memory).
- Use `profmem::profmem()` to monitor memory allocations (an overestimate of peak memory).

Calculating memory requirements

- We need to know how much memory statistical computations will require.
- Model matrices are $n \times p$.
- Large can mean large n or large p or large $n \times p$.
- Categorical predictors contribute `length(levels) - 1` to p.

Measuring memory usage in the shell

- The `time` command
- `-format="%M"`
- Maximum “resident set size” (a rough measure) in **kilobytes**.

Resources

- "Advanced R" chapter on "Memory"
  `https://adv-r.had.co.nz/memory.html`
- "Writing R Extensions" section on
  "Tidying and profiling R code"
  `http://cran.stat.auckland.ac.nz/doc/manuals/r-devel/R-exts.html#Tidying-and-profiling-R-code`