STATS 769
Debugging

Paul Murrell

The University of Auckland

October 17, 2019

Overview

- In this section of the course we will discuss some approaches and tools for **debugging R code**.

Debugging

- Use `traceback()` to show the call stack after an error.
- Use `print()` or `cat()` to print values or messages.
- Use `browser()` to interrupt execution and inspect objects.
- Use `setBreakpoint()` to interrupt execution at a line number within a file of code.

Debugging browser

- `where` to show call stack.
- `ls()` to list R objects.
- `str()` or `print()` to view R objects.
- `n` (or `Enter`) to run next expression.
- `s` to step into function.
- `f` to finish current loop or function.
- `c` to continue.
- `Q` to quit the browser.

Debugging

- Use `debug()` to interrupt execution when a function is called.
- Use `trace()` to interrupt execution at an expression number within a function.
- Use `recover()` to browse any currently active function calls.
- Use `options(warn)` to turn warnings into errors.
- Use `options(error)` to call `browser()` or `recover()` after an error.

Debugging R Markdown

- Run code in interactive R session.
  `rmarkdown::render()`

Debugging non-R code

- Beyond our scope.
- Still useful to identify.

Debugging parallel code

The computations running on workers are running in separate R sessions, which makes it difficult to see what is going on (when things go wrong).

- Write results to a file on disk
- `makeCluster(manual=TRUE)`

Reading

- Writing R Extensions
  https://cran.r-project.org/doc/manuals/r-release/R-exts.html#Debugging
- Debugging in R, by Duncan Murdoch
  https://web.archive.org/web/20170706215053/http://www.stats.uwo.ca:80/faculty/murdoch/software/debuggingR/
- Advanced R, by Hadley Wickham
  https://adv-r.hadley.nz/debugging.html
- What They Forgot to Teach You About R, by Jenny Bryan and Jim Hester
  https://whattheyforgot.org/debugging-r-code.html