NOT ALL STATISTICAL COMPUTING INVOLVES R

Computational Skills for Biostatistics I: Lecture 8

Amy Willis, Biostatistics, UW

THERE IS TIDY, AND THERE IS...

- Sometimes you will be sent *not-even-data*

NOT EVEN DATA

- Sometimes you will be sent text files with data somewhere between line 400 and line 800
- Files in random formats
  - every field has its own programs which have their own files
- Data at the bottom of an e-mail
  - e.g., digests
- Sometimes you will need to generate it yourself
  - e.g., web/e-mail scraping

REPETITIVE TASKS

- Dealing with these files often involves applying the same rule repeatedly
- Knowing how to apply the same rule repeatedly and automatically is time-saving and scalable
  - Do not do this by hand!
    - Slow, risk of errors, repetitive strain injury!

REPETITIVE TASKS

- In the same way as with scripting in R, this repetition needs to be repeatable
- For collaborators
- For reviewers
- For future you

REPETITIVE TASKS

- Sometimes this involves data analysis
  - In which case you can use R
- Sometimes it does not
  - In which case you need to work outside R

COMMAND LINE

- The command line allows you to control your computer
  - Unix: "Terminal"
  - Windows: "Command Prompt" or "cmd.exe"
- It does not have a graphical user interface

COMMAND LINE

- Everything you can do on your computer, you can do with command line*

- * just because you can do it doesn't mean you and I can figure out how

COMMAND LINE

- When you open command line, you are in a location on your computer

  - pwd

COMMANDLINE

- You can see what is available with your current location
  - ls
  - ls -a

COMMANDLINE

- Let's dive into the syntax of ls and ls -a

CONTROLLING YOUR COMPUTER

- Calling functions in the command line can take multiple forms
- So far we have seen two
  - [function]
  - [function] [argument-flag]

NAVIGATING YOUR COMPUTER

- From a location, we can move locations using cd
  - cd my_subdirectory
  - cd my_subdirectory/my_subsubdirectory
  - cd ..

NAVIGATING YOUR COMPUTER

- Useful shortcuts
  - here: .
    - (full stop)
  - one-level-up: ..
    - (two full stops)

NAVIGATING YOUR COMPUTER

- ls -a my_subdirectory

CONTROLLING YOUR COMPUTER

- Functions in the command line:
  - [function] [argument]
  - [function] [argument-flag]
  - [function] [argument-flag1] [argument1]
  - [function] [argument-flag1] [argument1] [argument-flag2] [argument2] ...

CONTROLLING YOUR COMPUTER

- Everything you can do manually can be done using the command line

  - copy: cp [myfile] [mynewfile]

  - rename: mv [oldname] [newname]

CREATING FILES

- Creating a new file containing the string "I love the command line!" in a new file called declaration.txt:
  - echo "I love the command line" > declaration.txt
  - echo 'I love the command line' > declaration.txt
  - echo 'I love the command line!' > declaration.txt

LOOKING AT FILES

- Print the output of a file
  - cat [filename]
- Repeat the output that was given to it
  - echo [statement]
  - Useful when scripting!

DELETING FILES

- Removing files
  - rm [unwantedfile]
- Be careful with deleting files!
  - Files deleted with rm are generally not recoverable!
  - No Recycle Bin

DELETING FILES

- You do not get warnings at the command line!
  - echo 'I love learning about the command line' > declaration.txt
    - overwrites old declaration.txt without warning

DIRECTORIES (FOLDERS)

- make a directory: mkdir [folder-name]
- delete an empty directory: rmdir [folder-name]
- delete a directory: rm -r [folder-name]

CONTROLLING YOUR COMPUTER VIA TERMINAL

- You can open programs using the command line
  - open
  - open file.txt
  - R
  - open -a [app] [file]
  - open -na RStudio
  - atom

VIM

- Fantastic approach to editing non-standard files: vim
- Open vim (or vi, which is a slightly lighter version)
  - vi
- Close vim
  - :q
  - :q!

VIM

- Exercise: 5 minutes in pairs
- Learning about vim
  - openvim.com
- Please put your hand up if you are stuck/have questions/don't understand the point/think you'll never need this*
- * Having worked in industry (finance and tech), government, and academia, I have needed these skills in *every one of these jobs*

REGEX

- Regular expressions with vim
- Case study: removing all underscores from contigs.fa

VIM + REGEX

- What's a .fa file?
    - It doesn't matter! Your collaborator sent it to you you need to deal with it

VIM

- vi contigs.fa

SUMMARY STATISTICS

- Get some information about contigs.fa
  - How many lines: wc -l contigs.fa
  - Print first line: head -n1 contigs.fa
  - Print last 5 lines: tail -n5 contigs.fa

EDITING FILES

- Change text in a file
- Lots of ways to do this!
- Important: make a copy; edit the copy! (space permitting)

TEXT EDITING

- I like stream editor (sed)
- sed -i -e 's/original/new/' mycopy.fa
  - -i = in place
  - -e = this regular expression
  - s = substitute

TEXT EDITING

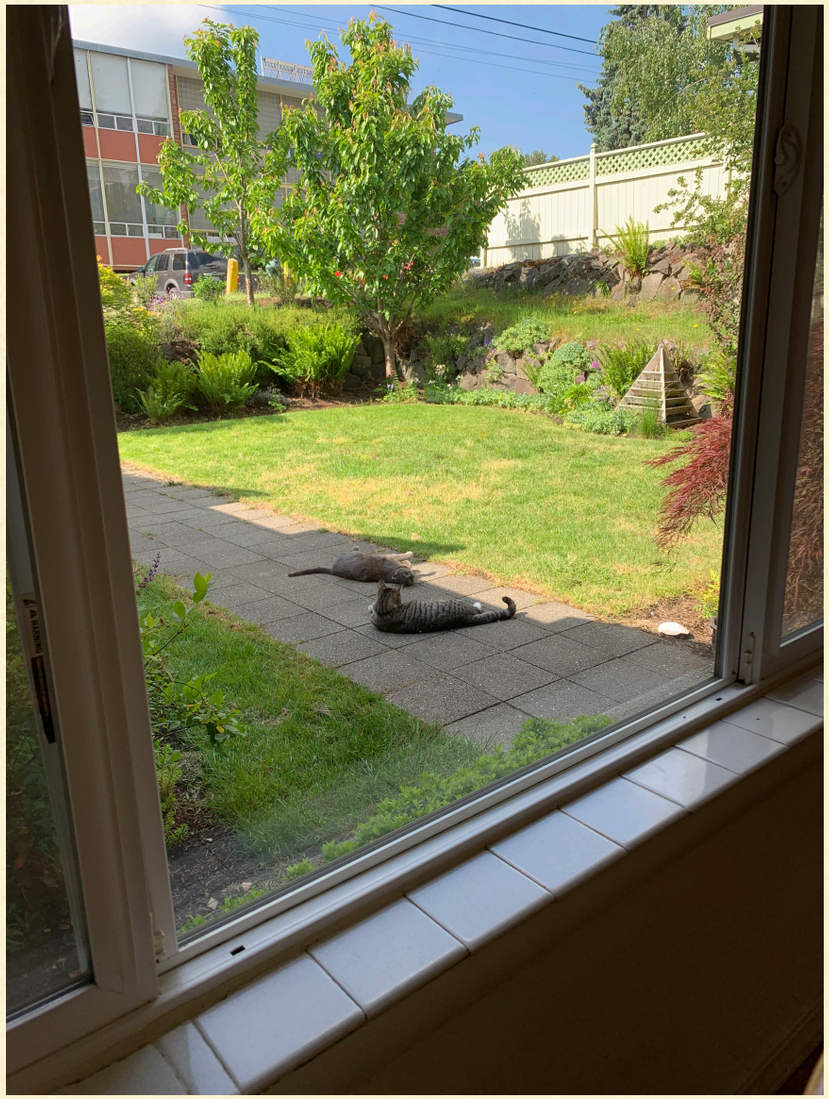
- stream editor
  - Need backslashes for special characters: $ * . [\ ^
  - Quite flexible!
  - I mostly use it for find and replace

TEXT EDITING

- Exercise: 3 minutes
  - Delete all underscores from contigs.fa using the command line

NESTING COMMANDS

- The command line has a pipe!
  - |
  - Instead of
    - tail -n5 contigs.fa
    - cat contigs.fa | tail -n5
- cat: print to standard output



PIPING

- Piping commands
  - Don't need to save every step of a sequential command!

PIPING

- Exercise: 1 minute
- What does the following command do?
  - cat contigs.fa | tail -n5 | head -n1

PIPING

- Fun example

- echo "The slow brown unicorn jumped over the hyper sleeping dog" > tmp

- cat tmp | sed s/unicorn/fox/ | sed s/hyper/lazy/ | cat

SCALING UP

- How do we...
- Create lots of files/directories?
- Delete lots of files/directories?

SCALING UP

- We have .R files that contain R "scripts"
- We have .sh files that contain "bash scripts"
  - Sequences of commands that are run at the command line

A SIMPLE SHELL SCRIPT

```
#!/bin/sh
for file in *.tmp
do
  sed -i -e 's/c_/c/' $file
done
```

EXECUTING PROCESSES

- Bash/shell files
  - Save as .sh files
  - Execute using ./myscript.sh
    - Need permissions!
    - chmod u+x myscript.sh
      - grants yo**u** (file owner) permission to e**x**ecute the file

EXAMPLES OF MY SCRIPTS

- Updating CV
  - ./sph-to-website.sh
- Running the same process on all files in a directory
- Cleaning data/metadata
- Downloading files with address listed in spreadsheet

OTHER EXAMPLES

- Any repetitive task can be scripted!
- From a review I received today:
  - *It is also important to note that the dedication to code documentation, availability, and reproducibility is impressive; the authors should be very proud.*

WHY LEARN SCRIPTING?

- Let's be honest -- who do we really care about?
  - FUTURE YOU!

NOT ALL STATISTICAL COMPUTING INVOLVES R

Computational Skills for Biostatistics I: Lecture 8

Amy Willis, Biostatistics, UW