Computational Fluency Short Course

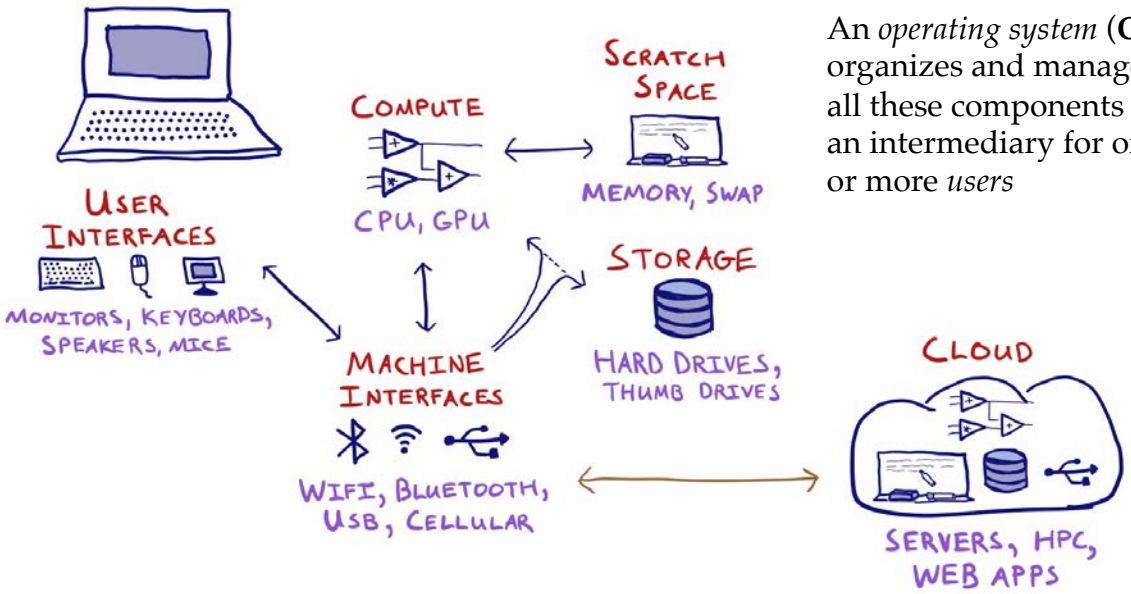
Architectures and interfaces

Jason Ritt
jason_ritt@brown.edu
Scientific Director of Quantitative Neuroscience

ROBERT J. & NANCY D. CARNEY
INSTITUTE FOR BRAIN SCIENCE
BROWN UNIVERSITY

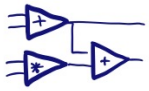
https://github.com/brownritt/cfsc25

Back to basics: What is a computer?

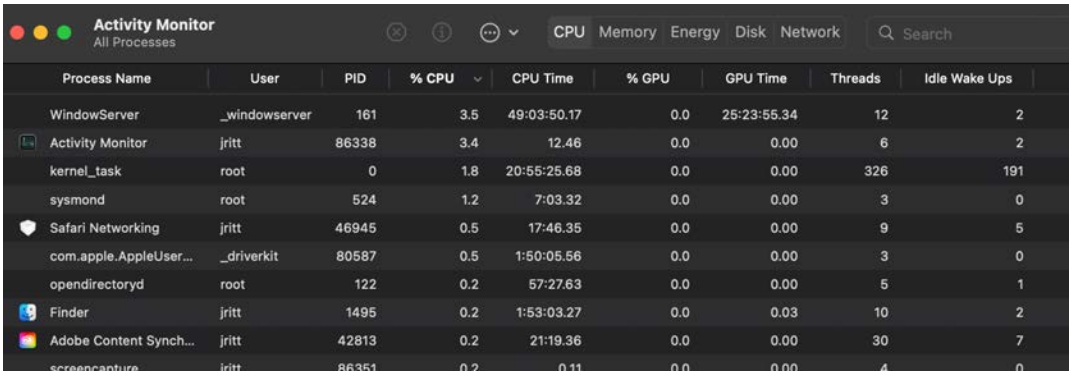


An *operating system* (**OS**) organizes and manages all these components as an intermediary for one or more *users*

How is Compute organized?



All activity (every "application" and more) is done through one or more *processes* managed by the OS.



Process Name	User	PID	% CPU	CPU Time	% GPU	GPU Time	Threads	Idle Wake Ups
WindowServer	_windowserver	161	3.5	49:03:50.17	0.0	25:23:55.34	12	2
Activity Monitor	jritt	86338	3.4	12.46	0.0	0.00	6	2
kernel_task	root	0	1.8	20:55:25.68	0.0	0.00	326	191
sysmond	root	524	1.2	7:03.32	0.0	0.00	3	0
Safari Networking	jritt	46945	0.5	17:46.35	0.0	0.00	9	5
com.apple.AppleUser...	_driverkit	80587	0.5	1:50:05.56	0.0	0.00	3	0
opendirectoryd	root	122	0.2	57:27.63	0.0	0.00	5	1
Finder	jritt	1495	0.2	1:53:03.27	0.0	0.03	10	2
Adobe Content Synch...	jritt	42813	0.2	21:19.36	0.0	0.00	30	7
screencapture	jritt	86351	0.2	0.11	0.0	0.00	4	0

Every process has some key properties:

- Who am I? ***Accounts***
- What am I allowed to do? ***Permissions, Priority***
- Where am I? ***Working directory (path)***

Editor
- Code as the center
- Use to make standalone apps

Notebooks
- Code mixed with its outputs (e.g. figures)
- Favors interactive and narrative use

Compiler or Interpreter ("kernel")
- Makes code executable

Linter
Syntax highlighter
Debugger
Project management
- Version control
- Virtual environments

Terminal

Shell

Low level interfaces: CLI via Terminal and Shell

A *command line interface* executes commands given by text input. CLIs are very powerful and efficient, though with a bit of a learning curve.

Note: theTerminal application is a graphical interface to a second process, called a *shell*, that actually runs the CLI.

```
em-event-detection-demo — -bash — 69×17
jritt: ~ $ cd Code/EM_event_detection/GitLab/
jritt: GitLab $ ls
.DS_Store                       em-event-detection-demo/
jritt: GitLab $ cd em-event-detection-demo/
jritt: em-event-detection-demo $ ls
.DS_Store                       EM_algorithm_demo.pdf
.git/                           LICENSE
.gitignore                      README.md
.ipynb_checkpoints/             README.md~
EM_algorithm_demo.ipynb         environment.yml
jritt: em-event-detection-demo $ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
jritt: em-event-detection-demo $
```

CLIs are a common example of a Read-Eval-Print Loop (REPL) interface.

Low level interfaces: Text editor

A *text editor* manipulates arbitrary text-based files.

```
em-event-detection-demo — nano README.md — 83×17
  UW PICO 5.09                       File: README.md

  EM Event Detection Demo #

A demonstration of using expectation-maximization to find "spiking" events in calc$

**You will need a numpy data file** to run the notebook yourself (except for secti$

```python
A = np.load('F.npy')
df_data = A[0,:]
```

Only `data_df` is used from there on. Probably any such file will work, or you can$

^G Get Help ^O WriteOut ^R Read File ^Y Prev Pg   ^K Cut Text ^C Cur Pos
^X Exit   ^J Justify   ^W Where is ^V Next Pg   ^U UnCut Text^T To Spell
```

Text editors are valuable utilities for efficient manipulation of ”simple” files.

Editor
- Code as the center
- Use to make standalone apps

Notebooks
- Code mixed with its outputs (e.g. figures)
- Favors interactive and narrative use

Compiler or Interpreter ("kernel")
- Makes code executable

Linter
Syntax highlighter
Debugger
Project management
- Version control
- Virtual environments

Terminal

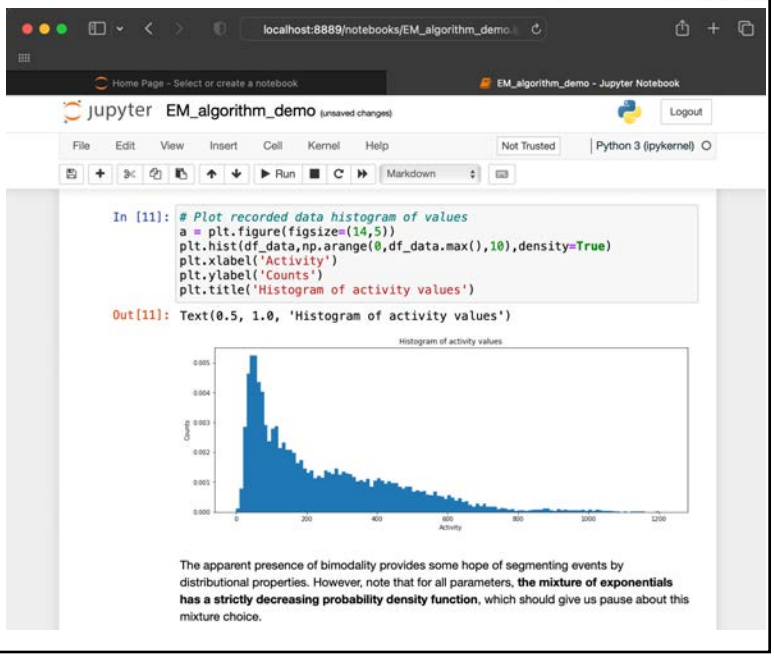
Shell

High level interfaces: Notebooks

An *interactive notebook* runs input code, displays outputs, and allows text annotations in a single document made of *cells*.

There are actually two processes: one runs the notebook itself, and communicates with an invisible *kernel* process that does the real computational work.

Beware: is a REPL that keeps its history, but can get "out of order"!



Editor
- Code as the center
- Use to make standalone apps

Notebooks
- Code mixed with its outputs (e.g. figures)
- Favors interactive and narrative use

IDE

Compiler or Interpreter ("kernel")
- Makes code executable

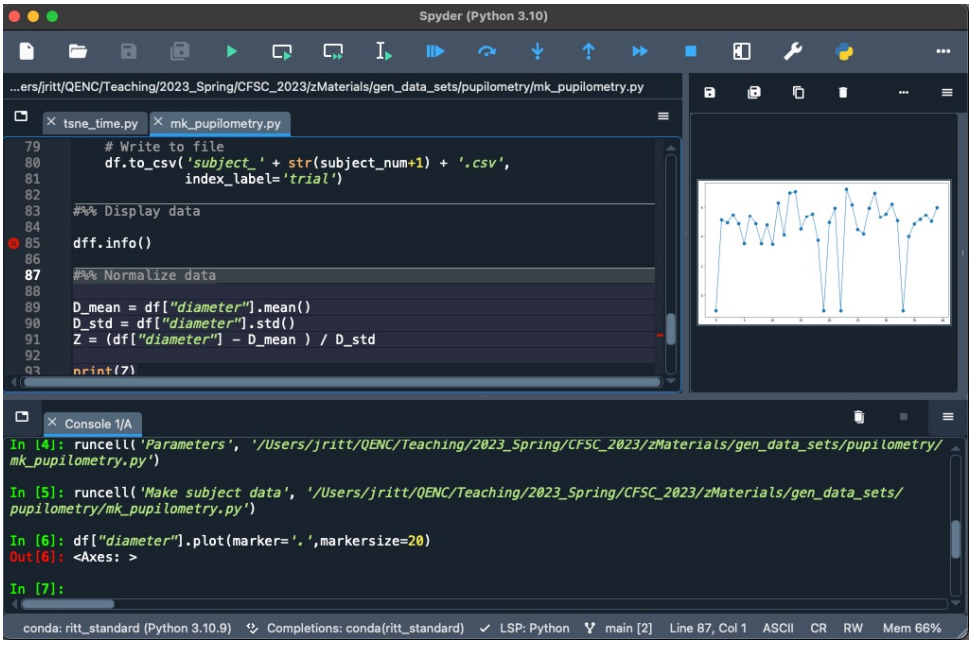
Linter
Syntax highlighter
Debugger
Project management
- Version control
- Virtual environments

Terminal

Shell

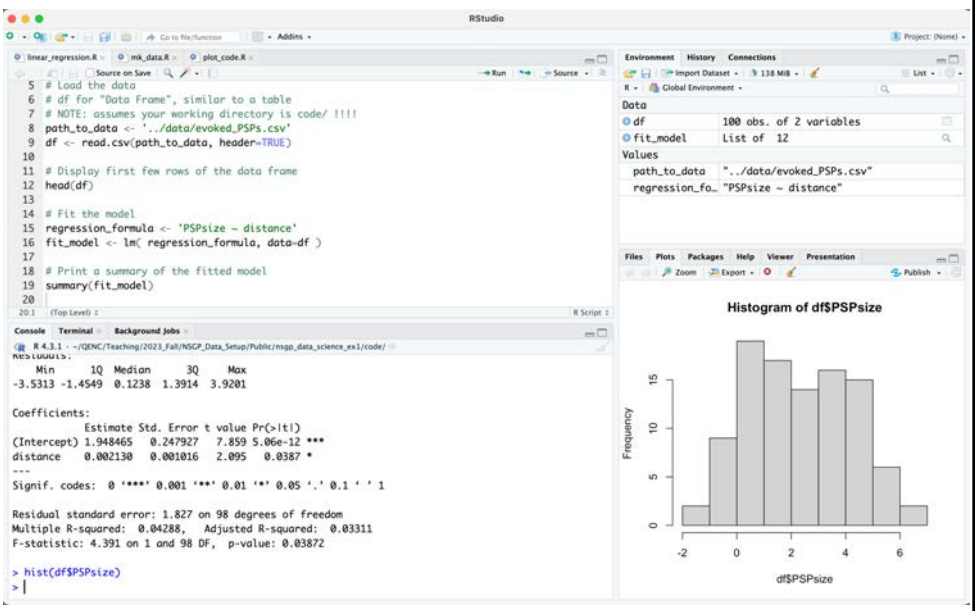
All-in-one interfaces: IDEs

An *integrated development environment* (**IDE**) combines a "smart editor" (syntax highlights, error checks, code hints, etc), a (REPL) *console* for running interactive commands, and other coding and file handling utilities.



All-in-one interfaces: IDEs

An *integrated development environment* (**IDE**) combines a "smart editor" (syntax highlights, error checks, code hints, etc), a (REPL) *console* for running interactive commands, and other coding and file handling utilities.



Editor
- Code as the center
- Use to make standalone apps

Notebooks
- Code mixed with its outputs (e.g. figures)
- Favors interactive and narrative use

IDE

Compiler or Interpreter ("kernel")
- Makes code executable

Linter
Syntax highlighter
Debugger
Project management
- Version control
- Virtual environments

Terminal

Shell

Which is the right interface for the job?

There are **many** other tools for computational projects, and everyone has their own preferred tool chain.

Common use cases:

- CLI - Direct interaction with the OS, processes, and filesystem
- Text editor - “Simple” files like scripts, READMEs, and configuration files
- Notebook - Exploratory data analysis, and “narrative” coding
- IDE - Exploratory data analysis, and “standalone” or complex coding

Do not use Excel:

nature

NEWS | 13 August 2021 | Correction 25 August 2021

Autocorrect errors in Excel still creating genomics headache

Despite geneticists being warned about spreadsheet problems, 30% of published papers contain mangled gene names in supplementary data.

https://www.nature.com/articles/d41586-021-02211-4