

Week 2-4: Unix, Git, Containers & Scientific Python

Chapter 2: The Unix Operating System

What is Unix?

- Fundamental operating system used to control how computers execute programs
- Originated in the 60s
- Important for data analysis, programming and system management
- Designed to produce output that can be used as input to other programs
- Enables the creation of pipelines using small tools

The Shell

- A command line interface (CLI)
- Allows users to:
 - Run programs
 - Control files and folders
 - Receive output in text form
- Operated as a RELP
 - Read Evaluate Print Loop
- Common in modern system (macOS, Linux, Windows via Git Bash)

Basic Commands

Command	Description
<code>ls</code>	List the contents of the current directory.
<code>cd</code>	Change the current directory.
<code>pwd</code>	Print the path of the current directory.
<code>mkdir</code>	Create a new directory.
<code>touch</code>	Create a new file.
<code>cp</code>	Copy a file or directory.
<code>mv</code>	Move or rename a file or directory.
<code>rm</code>	Remove a file or directory.

<code>cat</code>	Print the contents of a file to the terminal.
<code>less</code>	View the contents of a file one page at a time.
<code>grep</code>	Search for a pattern in a file or files.
<code>sort</code>	Sort the lines of a file.
<code>find</code>	Search for files based on their name, size, or other attributes.
<code>wc</code>	Print the number of lines, words, and bytes in a file.
<code>chmod</code>	Change the permissions of a file or directory.
<code>chown</code>	Change the ownership of a file or directory.
<code>head</code>	Print the first few lines of a file.
<code>tail</code>	Print the last few lines of a file.
<code>diff</code>	Compare two files and show the differences between them.

Exploring the Filesystem

- When shell starts, it begins in home directory
- Adding flags changes command behavior
 - Example: `ls -F` adds `/` to folders
- Absolute path:
 - Always starts from root `/`
 - Example: `cd /Users/arokem`
- Relative path:
 - Depends on current directory
 - Example: `cd Documents`

Path Shortcuts

- `..` – parent directory
 - Example: `cd ..` goes up one level
- `~` - home directory
 - Example: `cd ~`
 - Example: `cd ~/Documents`

Pipe Operator (|)

- Connects commands so output of one becomes input of another

Why Unix Matters

- Provides powerful control over:
 - Files and folders
 - Program execution
 - Automation through pipelines
- Becoming comfortable with Unix makes data work faster and more efficient

Chapter 3: Version Control (Git)

What is Git?

- A widely-used version control tool
- Works via a command line

Initialize a Repository

- Creating a Project
 - Mkdir my_project
 - Cd my_project
 - Get init
- Add a file
 - Touch my_file.txt.
 - Git add my_file.txt
- Check status
 - Git status
- Commit Changes
 - Git commit -m "Statement here"
- Check commit history
 - Git log
- Important concepts
 - Commit: saves a snapshot of your project
 - SHA: unique identifier for each commit
 - HEAD: current state of the repository

Tracking Changes

- Stages
 - Unstaged
 - Stages
 - Committed changes
- View changes
 - Git diff
- Workflow
 - Make changes
 - Git add
 - Git commit
- Undoing changes
 - To revert a file to a previous commit
 - Git checkout <SHA> myfile.txt

Branching and Merging

- Enables experimenting without affecting main code
- Keep main branch stable

- Merge only when ready
- Create and switch branches
 - Git branch feature_x
 - Git checkout feature_x
- Merge branch into main
 - Git checkout main
 - Git merge feature_x
- Delete branch
 - Git branch -d feature_x

Collaborating with Github

- Remote repository
 - A copy of your repo stored online
 - GitHub is the most common remote
- Add remote and push
 - Git remote add origin <URL>
 - Git push -u origin main
- Authentication
 - GitHub requires
 - Personal Access Token (PAT) OR
 - SSH Keys

Collaborating Workflow

- Cloning
 -