Tools for Reproducible Research in Psychology and Neuroscience: Emphasizing GitHub/Git for Version Control

Yibei Chen
Department of Communication
Researcher at Yu Emotion Science (YES) Lab
ReproNim/INCF Fellow 2022-2023

ReproducibiliTea - May 10th, 2023

UC SANTA BARBARA

What is Reproducible Research?

- Research that can be independently verified
- Transparent methodology, data, and code
- Key component of open science

How to do Reproducible Research

- Share data and materials
- Provide clear documentation
- Use version control and data management tools
- Collaborate and communicate effectively

Tools for Reproducible Research

- GitHub/Git: Version control and collaboration
- Datalad: Data management and tracking
- Open Science Framework (OSF): Project management and sharing
- Cloud storage (Amazon S3, Google Cloud Storage): Storing and accessing data

Tools for Reproducible Research

- **GitHub/Git: Version control and collaboration**
- Datalad: Data management and tracking
- Open Science Framework (OSF): Project management and sharing
- Cloud storage (Amazon S3, Google Cloud Storage): Storing and accessing data

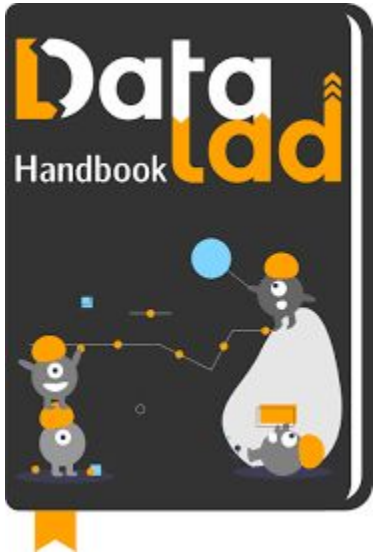
Overview of Reproducible Research Tools

GitHub/Git for Version Control

- **Version control:** A system to manage and track changes in files (source code, documents, etc.) over time
  - Keep a record of every modification made to the files
  - Allow multiple users to collaborate on the same project without conflicts
  - Enable easy retrieval of previous versions or revert changes if needed
- **Git:** A distributed version control system used to track changes and collaborate with others
- **GitHub:** A web-based platform that hosts Git repositories and provides additional collaboration features
- **Key features:** Branching, merging, tagging, and project history management
- Integration with various platforms and tools for a comprehensive workflow

Datalad for Data Management

- **Datalad:** An extension of Git designed specifically for managing large datasets and their versions (data files, metadata, etc.)
- **Differences from Git:**
  - Handles large files (e.g., brain data) and binary data more efficiently
  - Simplifies access and retrieval of data using various storage backends (e.g., local, cloud, or distributed)
  - Built-in data privacy features for sensitive data
- Seamlessly integrate with GitHub/Git and other tools for a unified workflow



When to Use What?

Datalad

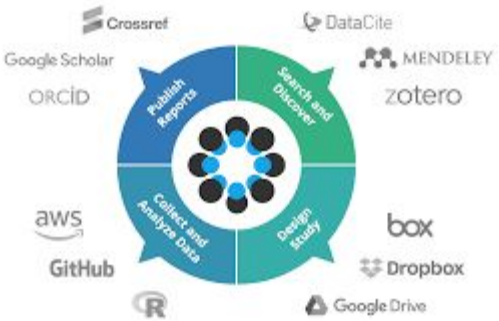
1. Managing large datasets, especially in the context of research data
2. Needing a more efficient way to handle binary files
3. Requiring data privacy and controlled access features

Git/GitHub

1. Managing mainly source code and text-based documents
2. Not dealing with large datasets or binary files

Open Science Framework (OSF)

- **What is OSF?**
  - A free, open-source web application developed by the Center for Open Science (COS)
- **Key Features:**
  - Create, manage, and organize research projects
  - Collaborate with team members and assign roles and permissions
  - Share research materials, data, and preprints with others
  - Integrate with various research tools, including GitHub/Git, Datalad, and cloud storage services
  - Support for versioning, archiving, and DOI assignment for persistent access



Cloud Storage Options

- **What is Cloud Storage?**
  - Remote storage infrastructure that allows users to store, manage, and access data over the internet
  - Scalable, secure, and reliable storage solutions
  - Accessible from anywhere with an internet connection
- **Popular Cloud Storage Services:**
  - Amazon Simple Storage Service (Amazon S3)
  - Google Cloud Storage

GitHub/Git for Version Control: Basics

What is Git?

- A distributed version control system
- Tracks changes in files **over time**
- Facilitates collaboration among **multiple users**
- Works well with text-based files like source code (e.g., Python/R/bash/text files, jupyter notebooks, etc.) and documents (e.g., .pdf, .jpg, .png, etc.)

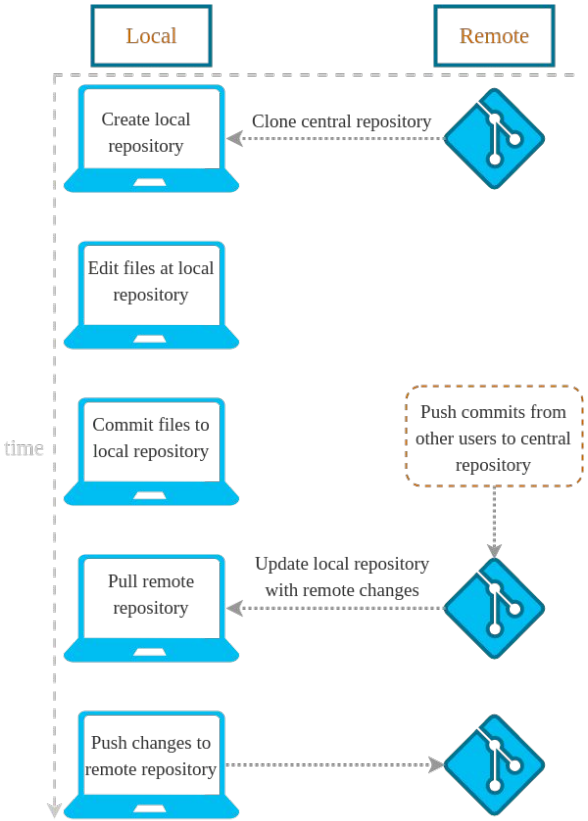
Introduction to GitHub

- Web-based platform for Git repositories
- Repository hosting with a **user-friendly interface**
- Additional **collaboration features** like issue tracking, project boards, and pull requests
- Integration with other tools and platforms

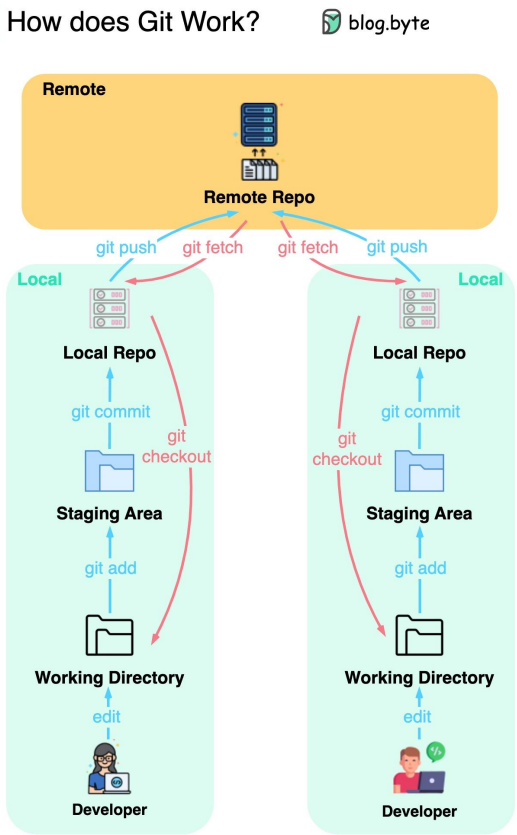
Git Workflow: Basic Concepts

- Repository (Repo): A collection of files and their history
- Commit: A snapshot of changes made to files in a repository
- Branch: An independent line of development within a repository
- Merge: Combine changes from multiple branches into a single branch

Git Workflow: Basic Concepts



Source: https://www.tutorialkart.com/git/git-project-life-cycle/



Source: https://blog.bytebytego.com/p/ep-40-git-workflow

Git Workflow: Commands Overview

- git clone: Create a local copy of a repository
- git add: Stage changes to files for a commit
- git commit: Save the staged changes with a descriptive message
- git push: Upload local commits to the remote repository
- git pull: Download and integrate remote changes into the local repository
- git branch: Create, list, or switch between branches
- git merge: Combine changes from one branch into another

Git Workflow: Collaborating with Others

- Fork: Create a personal copy of another user's repository
- Pull request: Request to merge changes from a forked repository into the original repository
- Resolve conflicts: Address differences between branches before merging

Git Best Practices

- Write clear and descriptive commit messages
- Keep related changes in separate commits
- Use branches for new features or bug fixes
- Regularly sync local and remote repositories to stay up-to-date
- Collaborate effectively with pull requests and code reviews

Practice: ReproducibiliTea_Workshop Repo