Illustration: Amrei Binzer-Panchal

Basic Biostatistics and Bioinformatics

Git, Github, RStudio

Swedish University of Agricultural Sciences, Alnarp

3 June 2024

Basic Biostatistics and Bioinformatics

A seminar series on the fundamentals

Organised by *SLUBI* and *Statistics at SLU*

Presentation of background and a practical exercise

Topics

- 3 June. Git and Github
- 17 June. Projects in R
- 1 July. Shiny

Topic suggestions are welcome

SLUBI

- SLU bioinformatics center
- Weekly online drop-in (Wednesdays at 13.00)
- slubi@slu.se, https://www.slubi.se
- Alnarp: Lizel Potgieter (Dept. of Plant Breeding)

Statistics at SLU

- SLU statistics center
- Free consultations for all SLU staff
- statistics@slu.se
- Alnarp: Jan-Eric Englund and Adam Flöhr (Dept. of Biosystems and Technology)

Today's content

Introduction to Git and Github

Commands in RStudio and the terminal

Installation and setup

In a nutshell

Git

A distributed version control system

- Version control
  - Log changes during our ongoing work
  - Create snapshots of the full project
- Distributed
  - Split a project into parallel versions (*branches*), each containing the complete project
  - Work on smaller individual problems and merge versions into the main *branch*

Github

A cloud-based platform for version control and collaboration

- Cloud-based
  - Files are stored online and often made publicly available
- Collaboration
  - Projects can be split over several workers

Some historical background

Both Git and Github were developed as technical software development tools

Git was created in the 2000s as part of Linux development

Github was started in 2007 and is owned by Microsoft since 2018

The official help information can often be quite obtuse (*imo*)

But since Git is used in many less technical fields there are lots of resources available

Terminology

- *Staging*. Marking a file for the next commit
- *Commit*. Creating a time-stamp of the current version
- *Branch*. A version of the project
- *Clone*. Creating a local version of a project
- *Fork*. Creating a personal version of a Github project
- *Pull*. Moving the current online (Github) version to the local branch
- *Push*. Moving the current local version to the online (Github) version

Working in Git

We can work in Git using several interfaces

The command line (or Terminal)

Write *Git commands* directly in the terminal

RStudio

A git-connected project will have a specific *Git* tab

Point-and-click-interface to stage, commit, push

There are also R packages to run Git commands

Specific Git interfaces

Git GUI - included in the Git for Windows installation

Github Desktop

VS Code

GitKraken - (gitkraken.com/)

Basic Git work. Local

Initializing

Linking a folder to Git

Can be automatically selected when starting an R project

```
> git init
Initialized empty Git repository in C:/Project folder/.git/
```

Status

See the current status

```
> git status
On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Test-file.txt

nothing added to commit but untracked files present (use "git add" to track)
```

Staging

Setting files where changes should be tracked

Marking in the Git tab in RStudio

```
> git add .
```

Committing

Creating a time-stamped version (a *commit*)

Using the Commit button in RStudio

```
> git commit -m "Add test-file"
[main (root-commit) 474a28e] Add test-file
 1 file changed, 1 insertion(+)
 create mode 100644 Test-file.txt
```

View history

History button in the Git tab in RStudio

```
> git log
commit 474a28e46ef88ac01b1cfa3c1f15ead33267c7da (HEAD -> main)
Author: adamflr <flr.adam@gmail.com>
Date:   Sat Jun 1 15:24:44 2024 +0200

    Add test-file
```

Branches

Create a new branch

New branch button in Git tab in RStudio

```
> git checkout -b new-branch
Switched to a new branch 'new-branch'
```

Branch can be changed in the Git tab (drop menu to the right) or with `checkout`

```
> git checkout main
> git checkout new-branch
```

Changes can be staged and committed to the new branch as usual

Merging branches

Changes in a branch can be merged into another by going to the target branch and using merge

```
> git checkout main
Switched to branch 'main'

> git merge new-branch
Updating 474a28e..463c8ae
Fast-forward
 File in a new branch.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 File in a new branch.txt
```

Basic Git work. Global

If a local Git project is linked to an online Git repository one can pull (get the current online version) and push (change the online version to the current local version)

Pull

Pull button in the Git tab in RStudio

```
> git pull
Already up to date.
```

Push

Push button in the Git tab in RStudio

```
> git push
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Delta compression using up to 4 threads
Compressing objects: 100% (1/1), done.
Writing objects: 100% (1/1), 712 bytes | 142.00 KiB/s, done.
Total 1 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local objects.
To github.com:adamflr/Project-name.git
   0191d8b..59d6e9f  main -> main
```

Cloning

Copies an entire Github repository to a local folder

File > *New Project* > *Version Control* > *Git* in RStudio

```
> git clone git@github.com:adamflr/Presentations.git
Cloning into 'Presentations'...
remote: Enumerating objects: 384, done.
remote: Counting objects: 100% (384/384), done.
remote: Compressing objects: 100% (334/334), done.
remote: Total 384 (delta 65), reused 356 (delta 38), pack-reused 0
Receiving objects: 100% (384/384), 21.00 MiB | 2.89 MiB/s, done.
Resolving deltas: 100% (65/65), done.
Updating files: 100% (543/543), done.
```

Installation and startup

1. Installing R
2. Installing RStudio
3. Installing Git git-scm.com
4. Checking so that Git is reachable from RStudio
5. Registering a Github account
6. Linking the local machine to Github using an SSH key
7. Starting a new Github repository
8. Starting an RStudio project linked to that repository

These instructions follow the steps on a Windows system

3. Installing Git

1. Go to git-scm.com
2. Identify the download page
3. Choose according to operating system and download the installer
4. Run the installer. Pick the default options
5. Open *Git Bash* on the Start menu and run `git --version`

If successful the result should be the version number of the Git installation

4. Checking the Git-RStudio connection

1. Open RStudio and go to *Tools* > *Global Options*…
2. Find the *Git/SVN* settings and check so that there is a path under *Git executable*
3. Go to *File* > *New project*…, select *New directory*, then *New project*, pick a name and a folder on the local drive. Check the *Create a Git repository* button and click *Create project*
4. In the new project, look for the *Git* tab, likely in the same window as *History* and *Environment*

At this point one can do staging, branching and commits

5. Create a new file and save it in the project. It will show up in the Git tab
6. Check the box under *Staged* and click *Commit*
7. Write an comment in the comment box and click *Commit*
8. Check *History* the see the commit

5. Register a Github account

Go to github.com and sign up

6. Link the local machine to Github

Two types of protocols possible: SSH and HTTPS

This suggestion is for SSH, since I find it a bit easier in RStudio

1. Go back to RStudio and go to *Tools* > *Global Options*…
2. Under Git/SVN click *Create SSH key* and then *Create* and *OK*
3. Click *View public key* and copy the text
4. Go to Github. Click the icon on the top right and go to *Settings*
5. Find *SSH and GPG keys* in the list to the left
6. Click *New SSH key*. Give a suitable title (like *work-2024*) and paste the SSH key from RStudio in the area for *Key*

7 & 8. Start a repository and an RStudio project

Finally a Github repository needs to be linked to an RStudio project

The easiest way is to create an empty repository and start a new project

1. Go to the Github start page
2. Start a new repository (button on top left) and give it some name (like *test-project*). Click *Create repository*
3. Copy the SSH link from the next page
4. Go to RStudio and *File* > *New Project*…
5. Select *Version control* and *Git*. Paste the SSH link under *Repository URL*. Place the new project somewhere on the local drive (usually C:)
6. A project window will open. Find the Git tab. Stage the files and commit. The click *Push*. Write *yes* if there is a question prompt.
7. Update the Github page of the repository and check if the pushed files appear

Best Practices

Tips for Effective Use

- **Commit Often**: Make frequent commits with meaningful messages.
- **Branching Strategy**: Use branches for features, fixes, and experiments.
- **Code Reviews**: Regularly review code via pull requests.
- **Documentation**: Maintain good documentation for your projects.
- **Backup**: Regularly push changes to remote repositories.

Resources

- **Official Git Documentation**: git-scm.com/doc
- **Happy Git and GitHub for the useR** by Jennifer Bryan: happygitwithr.com/
- **Git in Practice** by Mike McQuaid