Problem Set 1: Setting Up Git and GitHub

Applied Quantitative Methods for the Social Sciences II

Carlos III–Juan March Institute, Spring 2026

Instructions:

- Deadline: **February 12, before class**
- This problem set walks you through setting up Git and GitHub
- You will use this setup to submit all assignments throughout the course
- Complete all the tasks below and send me the link to your GitHub repository

1 Task 1: Create a GitHub Account

If you don't already have a GitHub account, create one at `https://github.com`.

1. Go to `https://github.com` and click "Sign up"
2. Choose a **professional username**—you may use this for years in your academic career
3. Use your university email or a professional email address
4. Complete the verification process
5. Optionally: Add a profile picture and brief bio

2 Task 2: Create a Repository for This Course

Create a new **public** repository. Name it something like `aqmss2` or `quant-methods-2026`.

2.1 Option A: Using the GitHub Web Interface

1. Click the "+" icon in the top-right corner of GitHub
2. Select "New repository"
3. Fill in the form:
   - **Repository name**: `aqmss2` (or similar)
   - **Description**: "Problem sets for AQMSS II, Spring 2026"

    - **Visibility**: Select **Public**
    - Check the box "Add a README file"

4. Click "Create repository"

2.2 Option B: Using the Command Line

First, make sure Git is installed on your computer. Then run:

```
# Create a new directory and initialize Git
mkdir aqmss2
cd aqmss2
git init

# Create a README file
echo "# AQMSS II - Problem Sets" > README.md

# Stage and commit the file
git add README.md
git commit -m "Initial commit"

# Set up the remote repository (create it on GitHub first, without README)
git branch -M main
git remote add origin https://github.com/YOUR-USERNAME/aqmss2.git
git push -u origin main
```

Note: Replace `YOUR-USERNAME` with your actual GitHub username.

2.3 Option C: Using RStudio

1. First, create an empty repository on GitHub (without README)
2. In RStudio: File → New Project → Version Control → Git
3. Paste your repository URL: `https://github.com/YOUR-USERNAME/aqmss2.git`
4. Choose a location on your computer
5. Click "Create Project"

3 Task 3: Edit the README File

Your README is the "front page" of your repository. Edit it to include:

- Your name

- A brief description (e.g., "Problem sets for AQMSS II, Spring 2026")
- Optionally, a list of what will be in the repository

3.1 Option A: On the Web

1. Click on `README.md` in your repository
2. Click the pencil icon (edit) in the top-right of the file view
3. Make your changes using Markdown syntax
4. Scroll down and click "Commit changes"
5. Add a commit message like "Update README with my info"

3.2 Option B: Command Line

```
# Edit README.md with any text editor, then:
git add README.md
git commit -m "Update README with my info"
git push
```

3.3 Option C: RStudio

1. Edit the `README.md` file in the Files pane
2. Go to the Git pane (usually top-right)
3. Check the box next to `README.md` to stage it
4. Click "Commit"
5. Write a commit message and click "Commit"
6. Click "Push" to upload to GitHub

4 Task 4: Create a Folder and R File

Create a folder called `problem_sets` and add your first R file.

4.1 Option A: On the Web

1. Click "Add file" → "Create new file"
2. In the filename box, type: `problem_sets/ps1.R`
    - This creates the folder and file at once
3. Add the following content:

    ```
    # Problem Set 1
    # AQMSS II, Spring 2026
    # [Your Name]

    # This file will contain my solutions for PS1
    print("Hello, Git!")
    ```

4. Scroll down and commit with message "Add ps1.R"

4.2 Option B: Command Line

```
# Create the folder
mkdir problem_sets

# Create the R file (use any text editor)
# Then stage, commit, and push:
git add problem_sets/ps1.R
git commit -m "Add ps1.R"
git push
```

4.3 Option C: RStudio

1. Create a new folder `problem_sets` in the Files pane
2. Create a new R script: File → New File → R Script
3. Add the header content and save as `problem_sets/ps1.R`
4. In the Git pane, stage the new file, commit, and push

5 Task 5: View Your Commit History

Check that your commits were recorded properly.

5.1 Option A: On the Web

1. Go to your repository page on GitHub
2. Click on "Commits" (or the clock icon with a number)
3. You should see your commits listed with messages and timestamps

5.2 Option B: Command Line

```
git log --oneline
```

This shows a compact list of your commits.

5.3 Option C: RStudio

1. In the Git pane, click "History" (clock icon)
2. Browse through your commits

Take a screenshot of your commit history showing at least 2–3 commits.

6 Submission

Send me an email with:

1. The URL of your GitHub repository
   (e.g., `https://github.com/username/aqmss2`)
2. A screenshot of your commit history

I will check that:

- Your repository is **public**
- It contains a README with your name
- It has a `problem_sets` folder with at least one `.R` file
- There are multiple commits in the history

7 Optional: Installing Git Locally

If you want to use Git from the command line, you need to install it first.

7.1 Installation

- **Mac**: Git comes pre-installed. Or install via Homebrew: `brew install git`
- **Windows**: Download from https://git-scm.com/download/win
- **Linux**: Use your package manager, e.g., `sudo apt install git`

7.2 First-Time Configuration

After installing, configure your identity (run once):

```
git config --global user.name "Your Name"
git config --global user.email "your@email.com"
```

7.3 Cloning an Existing Repository

If you created the repository on GitHub first, download it to your computer:

```
git clone https://github.com/YOUR-USERNAME/aqmss2.git
cd aqmss2
```

7.4 Configuring RStudio

To use Git in RStudio:

1. Go to Tools → Global Options → Git/SVN
2. Make sure "Git executable" points to your Git installation
3. Restart RStudio

8 Quick Reference: Common Git Commands

Command	What it does
`git status`	Show which files have changed
`git add <file>`	Stage a file for commit
`git add .`	Stage all changed files
`git commit -m "msg"`	Save staged changes with a message
`git push`	Upload commits to GitHub
`git pull`	Download changes from GitHub
`git log --oneline`	Show commit history (compact)
`git diff`	Show changes not yet staged

9 Resources

- GitHub's official guides: https://docs.github.com/en/get-started
- Happy Git with R: https://happygitwithr.com
- Git cheat sheet: https://education.github.com/git-cheat-sheet-education.pdf
- Interactive Git exercises: https://gitexercises.fracz.com
- Pro Git book (free): https://git-scm.com/book/en/v2