Applied Quantitative Methods for the Social Sciences II
MA in Social Sciences, UC3M, Spring 2026

Assignment 5: Best Practices in Computing

Instructions:

- Deadline: **March 12, before class**
- Submit via your GitHub repository in a separate folder (`assignment5`)
- **Always use comments** in your R code – and use them to answer questions
- You are encouraged to work together, but each person must submit their own code
- Plan is to start Part 1 in class and complete Part 2 at home
- I'll upload a solution file to the website after next class

Contents

1 Part 1: In-Class (Project Organization)

In this lab, you will create a properly organized mini-project that follows the `workflow_example` pattern from the lecture. Recall the key principle: separate tasks into folders, each with its own script and output subfolder. You will use the `corruption.dta` dataset from Assignment 4 to build a complete, reproducible pipeline.

1.1 Folder structure

Create a new folder `assignment5` in your course repository with the following structure:

```
assignment5/
  Makefile
  README.md
  data/
    corruption.dta
  analysis/
    models.R
    output/
  plots/
    figures.R
    output/
```

a) Create all the folders and empty files listed above. You can do this from the terminal using `mkdir` and `touch`, or manually.

b) Download the `corruption.dta` dataset and place it in the `data/` subfolder. The dataset is available at:

   - github.com/franvillamil/AQM2/tree/master/datasets/other

c) In `README.md`, write a brief description of the project and what each folder contains (3–5 sentences). For example: what question does the analysis address? What does the `data/` folder contain? What do the `analysis/` and `plots/` folders produce?

1.2 Analysis script

Create the file `analysis/models.R` with the following structure. The script should be self-contained: anyone should be able to run it from the `assignment5` folder and reproduce the results.

a) Start the script by loading necessary packages and reading the data:

```
# setwd("~/path/to/assignment5")
options(stringsAsFactors = FALSE)

library(readstata13)
library(modelsummary)
```

```
df = read.dta13("data/corruption.dta")
```

b) Define constants at the top of the script for the key variables:

```
dep_var = "ti_cpi"
indep_var = "undp_gdp"
```

c) Drop observations with missing values on these two variables. Then print the number of observations after cleaning:

```
df = df[!is.na(df[[dep_var]]) & !is.na(df[[indep_var]]), ]
cat("Observations:", nrow(df), "\n")
```

d) Add an assertion after cleaning to check that the dataset is not unexpectedly small:

```
if(nrow(df) < 10) stop("Too few observations")
```

e) Estimate two models — one in levels and one using the log of GDP:

```
m1 = lm(ti_cpi ~ undp_gdp, data = df)
m2 = lm(ti_cpi ~ log(undp_gdp), data = df)
```

f) Use `modelsummary()` to save a LaTeX table to `analysis/output/table_models.tex`:

```
modelsummary(
  list("Level" = m1, "Log" = m2),
  output = "analysis/output/table_models.tex",
  stars = TRUE,
  gof_map = c("r.squared", "nobs"))
```

g) Print a message when the script completes:
`cat("Analysis complete.  Table saved.\n")`

1.3 Plots script

Create the file `plots/figures.R`. This script reads the same data and produces a scatter plot.

a) Load packages (`readstata13`, `ggplot2`), read the data, and drop missing values.
b) Create a scatter plot of corruption (y-axis) vs. log GDP (x-axis) using `geom_point()`. Add a linear fit with `geom_smooth()`.
c) Use `theme_minimal()`, add informative axis labels and a title.
d) Save to `plots/output/scatter_corruption.pdf` using `ggsave()`, specifying width and height explicitly.
e) Print a message when done: `cat("Scatter plot saved.\n")`

1.4 Makefile

Create a `Makefile` in the `assignment5` folder that automates the pipeline. Recall from the lecture that a Makefile specifies targets, dependencies, and recipes.

a) Write an `all:` rule that lists both outputs as targets:

```
all: analysis/output/table_models.tex \
     plots/output/scatter_corruption.pdf
```

b) Write a rule for the analysis table that depends on the R script and the dataset:

```
analysis/output/table_models.tex: analysis/models.R \
                                  data/corruption.dta
        Rscript --no-save analysis/models.R
```

Important: the recipe line (the `Rscript` command) must be indented with a **tab character**, not spaces. This is a Makefile requirement.

c) Write a similar rule for the scatter plot output, which depends on `plots/figures.R` and the dataset.

d) Test it by running `make` in the terminal from the `assignment5` folder. In a comment in your `README.md`, note whether it worked and what output was produced.

2 Part 2: Take-Home (Code Quality and Git)

2.1 Code improvement

The following R script works but has multiple style and organization problems. Your task is to rewrite it following the best practices from the lecture (DRY, constants, meaningful names, comments, assertions).

```
x=read.csv("mydata.csv")
x=x[x$year>=2000,]
x=x[x$year<=2020,]
m=lm(x$outcome~x$gdp+x$pop)
summary(m)
m2=lm(x$outcome~x$gdp+x$pop+x$education)
summary(m2)
m3=lm(x$outcome~x$gdp+x$pop+x$education+x$health)
summary(m3)
pdf("fig.pdf")
plot(x$gdp,x$outcome)
dev.off()
pdf("fig2.pdf")
plot(x$education,x$outcome)
dev.off()
```

Rewrite this script and save it as `assignment5/improved_script.R`. Your improved version should include:

a) Meaningful variable names instead of `x`, `m`, `m2`, `m3`.

b) Constants defined at the top of the script for the year range (e.g., `start_year = 2000`, `end_year = 2020`).

c) Comments explaining what each section of the code does, with section dividers (`# ============`).

d) A function to avoid repeating the `pdf()/plot()/dev.off()` pattern. For example:

```
save_scatter = function(data, xvar, yvar, filename) {
  pdf(filename, width = 7, height = 5)
  plot(data[[xvar]], data[[yvar]],
       xlab = xvar, ylab = yvar)
  dev.off()
}
```

e) An assertion after loading the data (e.g., check that the number of rows is greater than zero, or check that the expected columns exist).

f) Proper formatting: spaces around operators, consistent indentation, use of the `data =` argument in `lm()` instead of the `$` notation.

g) A diagnostic print statement after filtering, e.g.:

```
cat("Rows after filtering:", nrow(df), "\n")
```

2.2 Git practices

This exercise asks you to practice the Git habits discussed in the lecture: frequent commits with meaningful messages, and a proper `.gitignore`.

a) Make sure your `assignment5` folder is tracked by Git (it should be part of your course repository).
b) Create at least **3 separate commits** for this assignment. Each commit should correspond to a logical unit of work (e.g., one commit for the folder structure, one for the analysis script, one for the Makefile). Do **not** make one big commit with everything at the end.
c) Each commit must have a **meaningful message** — not "update" or "changes". Recall from the lecture: the message should complete the sentence "This commit will...". Examples:
   - `Add folder structure and README for assignment 5`
   - `Create analysis script with regression models`
   - `Add Makefile to automate pipeline`
d) Create a `.gitignore` file inside `assignment5/` that ignores generated and system files:

```
*.pdf
*.aux
*.log
.DS_Store
analysis/output/
plots/output/
```

e) Push everything to GitHub.
f) In a comment at the end of your `README.md`, list the commit messages you used for this assignment.

2.3 Reflection

At the end of your `README.md`, answer the following questions in a few sentences each:

a) What was the most useful practice you learned from the computing lecture?
b) How would you apply these practices to your final project for this course?
c) What is one thing from the "bad script" example (in the code improvement exercise) that you recognized from your own coding habits?

3 Data Sources

The corruption dataset is available at the course GitHub repository:

- github.com/franvillamil/AQM2/tree/master/datasets/other (`corruption.dta`)

4 Submission

Commit your work to your GitHub repository before the deadline. Everything should be inside the `assignment5` folder. Make sure your repository is public so I can access it.

Your `assignment5` folder should contain:

- `README.md` — project description, Makefile test results, commit messages, and reflection
- `Makefile` — automation rules for the analysis and plots
- `.gitignore` — ignoring generated output and system files
- `data/corruption.dta` — the dataset
- `analysis/models.R` — regression analysis script
- `analysis/output/` — (generated table, may be git-ignored)
- `plots/figures.R` — scatter plot script
- `plots/output/` — (generated figures, may be git-ignored)
- `improved_script.R` — rewritten version of the bad script

All R scripts should run without errors from the `assignment5` directory.