Introduction

Francisco Villamil

Applied Quantitative Methods II

MA in Social Sciences, Spring 2026

Welcome everyone. Introduce yourself briefly. Mention this is the second part of the quantitative methods sequence – they already have the foundations from AQMSS-I, now we build on that.

Course overview

- This is the second part of the quantitative methods sequence
- Focus on **applying** statistical tools in practice
- Less theory, more hands-on work with data
- Goal: go from research question to answer

Emphasize the contrast with AQMSS-I: less abstract theory, more applied work. The goal is that by the end they can take a research question, find or collect data, choose an appropriate method, and present results. Mention that we will also spend time on computing skills and workflow.

What will you learn?

- How to choose the right model for your question
- How to interpret and visualize model results
- How to evaluate whether a model is appropriate
- How to work with different types of data (panel, spatial, etc.)
- Best practices in computing and reproducibility

Go through each point briefly. Stress that "choosing the right model" means matching the model to the question and the data – not just running the fanciest thing available. Panel and spatial data are new types they probably haven't seen. Computing and reproducibility will be a recurring theme.

Course structure

Feb 5	Introduction
Feb 12	Applied regression
Feb 19	Applied regression II (binary)
Feb 26	Interpretation and diagnostics
Mar 5	Best practices in computing *(move just before break?)*
Mar 12	Panel data I
Mar 19	Panel data II
Mar 26	Spatial data
Easter break	
Apr 9	Spatial data
Apr 16	Other outcomes
Apr 23	Project presentations
Apr 30	Exam + Review

Walk through the schedule session by session. Mention that the order might change slightly (e.g. the computing session could move). Highlight that the course alternates between learning new models and applying them in problem sets. Note the Easter break and that project presentations happen right after.

Evaluation

- Problem sets (20%)
  - → Started in class, finished at home
  - → Short deadlines
- Proposal presentation and peer review (10% + 10%)
- Final essay (30%)
  - → Small research note (max 3,000 words)
  - → Original data analysis using R
- Exam (30%)

Problem sets are mostly graded on completion – the point is to practice, not to get everything perfect. The final essay is the main individual assessment: they pick a question, find data, and do an original analysis. Emphasize that the proposal presentation is also a chance to get feedback from peers before writing. The exam will cover both concepts and applied interpretation.

Roadmap

The Big Picture

Version Control and Git

The research process

Theory $\longleftrightarrow$ Data Generating Process $\longleftrightarrow$ Data

- Theories make claims about how the world works
- These claims imply certain patterns in data
- We observe data and try to learn about the underlying process
- Our research strategy connects theory to data

This is the key conceptual diagram for the course. Spend some time here. Theory tells us what the world should look like; the DGP is the actual mechanism generating the data we observe; data is what we get to see. The double arrows mean we go back and forth: theory informs what patterns to expect, and data helps us refine our theories. The last point (research strategy connects theory to data) sets up the next slide.

Theory first, methods second

- The research question and theory should drive everything:
  - → What unit of analysis to use
  - → What variation to look at
  - → What empirical strategy to follow
- Methods are tools to implement that strategy
- Common mistake: picking a method and then looking for a question
- In this course: we learn tools, but always ask *why this tool for this question?*

This is a crucial point, especially for students who might be tempted to learn a fancy method and then look for a problem to apply it to. Give a concrete example: if your theory is about individual-level behavior, you need individual-level data and a strategy that captures individual variation – running a country-level regression with a cool estimator doesn't help. Connect to the research design course: the sequence is always question, theory, strategy, then method. This course teaches the tools, but the question "why this tool?" should always come first.

What is a Data Generating Process (DGP)?

- The rules that govern how data comes to exist
- Includes:
  - → The social or political process we study
  - → How observations end up in our dataset
- We never observe the DGP directly
- We use statistical models to make inferences about it

Make it concrete. Example: civil war onset. The DGP includes the actual political, economic, and social factors that lead to conflict, but also how conflicts get recorded (coding decisions, media coverage, threshold definitions). We never see the "true" process – we see a dataset with rows and columns that reflect both the real process and a lot of measurement and selection choices. That's why understanding the DGP matters for choosing the right model.

Why do we need statistics?

- Our theories deal with processes, not just data
- Data is a window into the underlying process
- Statistics helps us:
  - → Separate signal from noise
  - → Quantify uncertainty
  - → Make valid inferences

Bridge from the DGP to methodology. We can't just look at raw data and draw conclusions – data is noisy, incomplete, and shaped by the DGP. Statistics gives us a principled way to learn from data despite this. The three functions (signal vs noise, uncertainty, inference) will come up throughout the course: every model we fit is trying to do these things.

Sources of uncertainty

- **Sampling uncertainty**: We observe a sample, not the population
- **Theoretical uncertainty**: Our theories are simplifications
- **Fundamental uncertainty**: Some processes are inherently random

- All of these create "noise" in our data
- Statistical models help us deal with this noise

Give examples for each type. Sampling: we survey 1,000 people but want to say something about millions. Theoretical: our model of voting assumes a few variables matter but the real process is much more complex. Fundamental: even if we knew everything, some outcomes have inherent randomness (e.g. election margins in very close races). The key takeaway: noise is unavoidable, so we need tools that account for it rather than pretending our data is perfect.

The logic of statistical inference

- **Probability theory**: Given a known process, what data will we see?
- **Statistical inference**: Given observed data, what can we learn about the process?
- We're doing the reverse: from data back to process

This is a good place to pause and check understanding. Probability theory is the "forward" problem: if I know the coin is fair, I can predict roughly 50% heads. Statistical inference is the "inverse" problem: I see 53 heads out of 100 flips, is the coin fair? This inversion is what makes statistics hard – there's always uncertainty about the answer. Everything we do in this course (confidence intervals, hypothesis tests, model selection) is about managing that uncertainty.

Roadmap

The Big Picture

Version Control and Git

Learning to use computers as tools

- World of quantitative methods is changing fast
  - → e.g. Claude Code
- I think it'll be more important to be really literate with computers
- Part of this course will also involve learning how to properly use computers
  - → Not using only RStudio, R Markdown, etc, but being ready to do big data-based projects
- We'll have a session on computing, project management, etc – but today, some notes on version control

Set the scene for why we're spending time on computing. AI tools like Claude Code are changing the workflow fast – students who are comfortable with the command line, file systems, and version control will adapt much more easily. This isn't about becoming programmers, it's about being literate enough to use these tools effectively for research. Mention that we'll dedicate a full session later to project management and reproducibility, but version control is so fundamental that we start with it today.

The problem: managing files over time

- Have you ever had files like this?
  - → `thesis_v1.docx`
  - → `thesis_v2_final.docx`
  - → `thesis_v2_final_REAL.docx`
  - → `thesis_v2_final_REAL_submitted.docx`
- What changed between versions?
- Which version has the correct analysis?
- How do you collaborate without overwriting each other's work?

This usually gets a laugh – everyone recognizes themselves. Let them share their own horror stories if they want. The point is that this "system" of naming files is fragile, unscalable, and doesn't work for collaboration at all. Ask: how would you collaborate on a data analysis project with a co-author using this approach?

Version control: a better way

Version control is a system that records changes to files over time

- One file, complete history
- Every change is recorded with a description
- Can go back to any previous state
- Multiple people can work simultaneously

Introduce the concept without jargon. The key idea is that instead of saving multiple copies, you save "snapshots" of your project over time, each with a description of what changed. You can always go back. This is like track changes in Word but much more powerful and for any kind of file.

Why version control for research?

- **Reproducibility**: Track exactly what you did and when
- **Backup**: Your work is safely stored, even if your laptop dies
- **Collaboration**: Work with others without email chains of files
- **Transparency**: Share your code with the research community

- Many journals now require or encourage sharing code via GitHub

Connect to their future as researchers. Reproducibility is increasingly expected – some journals won't publish without replication materials. Mention specific examples: AJPS data policy, QJE replication packages. Collaboration is also practical for their MA thesis or co-authored papers. The transparency point matters for their credibility as researchers.

Git and GitHub

Git

- A version control system
- Runs locally on your computer
- Tracks changes to files

GitHub

- A web platform that hosts Git repositories
- Stores your code online
- Enables sharing and collaboration

Important distinction: Git is the engine, GitHub is the garage. Git runs on your machine and tracks changes locally. GitHub is where you store and share your work online. You can use Git without GitHub, but GitHub without Git doesn't make much sense. Analogy: Git is like saving versions on your computer; GitHub is like backing them up to the cloud and making them available to others.

The basic Git workflow

1. **Make changes** to your files (write code, edit text)
2. **Stage** the changes you want to save
   → "These are the files I want to include in my next snapshot"
3. **Commit** the staged changes with a message
   → A snapshot of your project at this moment
4. **Push** your commits to GitHub
   → Upload your local changes to the cloud

Walk through each step slowly. Staging is the step that confuses people most – explain that it's like putting items on a table before packing a box. You decide what goes in the next snapshot. A commit is the snapshot itself, with a message explaining what you did and why. Pushing uploads everything to GitHub. Emphasize that if they forget to push, their work is only on their laptop.

Ways to use Git

- **GitHub web interface**: Create repos, upload files, edit directly
  → Simple but limited
- **Command line**: Most powerful and flexible
  → `git add`, `git commit`, `git push`
- **RStudio**: Built-in Git integration
  → Point-and-click interface

- All do the same thing—choose what works for you

Reassure them: they don't need to master the command line right now. The web interface is perfectly fine for getting started. The assignment document gives step-by-step instructions for all three approaches. Encourage them to try the command line eventually because it's the most flexible, but there's no penalty for using the web interface or RStudio.

Assignment 1

- Create a GitHub account (if you don't have one)
- Create a **public** repository for this course
- Set up your README and folder structure
- Create a simple `.R` file

- This repository is where you'll submit all your assignments
- Detailed instructions in the assignment document

Go through the assignment briefly. The main goal is just to get everyone set up with Git and GitHub – the actual R work is minimal. Emphasize that the repository must be public so you can check their submissions. Mention they should start early in case they run into setup issues. Point them to the detailed assignment document and the appendices on Windows setup, Positron, and Sublime Text.

What makes a good analysis?

- Clear research question
- Appropriate data for the question
- Right statistical model for the data
- Correct interpretation of results
- Honest about limitations and uncertainty

Tie back to the Big Picture section. This is the checklist for the entire course: every analysis we do should satisfy these criteria. A clear question means knowing what you're trying to learn. Appropriate data means the data actually lets you answer the question (connects to the "theory first" point). Right model means matching the method to the data structure. Correct interpretation means not overclaiming. Honesty about limitations is what distinguishes good research from bad.

Looking ahead

- Next session: Applied regression
- Regression as conditional expectations
- Multiple regression and control variables
- Interaction effects and presenting results

Brief preview of what's coming. Next session will be a proper deep dive into regression – they've seen it before in AQMSS-I but now we'll focus on applying it in R and interpreting results carefully. Mention that interactions and presenting results are things they'll use constantly in their own work.

For next week

- Check readings if needed
- Review your notes on OLS from AQMSS-I
- **Finish Assignment 1**

Remind them to start the assignment early. The readings are optional review if they feel rusty on OLS. The most important thing is to have their GitHub set up and working before next session.

Questions?

Open the floor. If no questions, can use this time to start working on Assignment 1 together – walk them through creating a GitHub account and repository on the projector.