Introduction

Francisco Villamil

Applied Quantitative Methods II

IC3JM, Spring 2026

Course overview

- This is the second part of the quantitative methods sequence

Course overview

- This is the second part of the quantitative methods sequence
- Focus on **applying** statistical tools in practice

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

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

What will you learn?

- How to choose the right model for your question

What will you learn?

- How to choose the right model for your question
- How to interpret and visualize model results

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

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.)

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

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
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

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
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

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
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

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
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

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

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
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

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
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

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
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

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
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

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
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

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%)

Roadmap

The Big Picture

Version Control and Git

The Big Picture

The research process

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

- Theories make claims about how the world works

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

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

What is a Data Generating Process (DGP)?

- The rules that govern how data comes to exist

What is a Data Generating Process (DGP)?

- The rules that govern how data comes to exist
- Includes:

What is a Data Generating Process (DGP)?

- The rules that govern how data comes to exist
- Includes:
  - → The social or political process we study

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

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

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

Why do we need statistics?

- Our theories deal with processes, not just data

Why do we need statistics?

- Our theories deal with processes, not just data
- Data is a window into the underlying process

Why do we need statistics?

- Our theories deal with processes, not just data
- Data is a window into the underlying process
- Statistics helps us:

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

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

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

Sources of uncertainty

- **Sampling uncertainty**: We observe a sample, not the population

Sources of uncertainty

- **Sampling uncertainty**: We observe a sample, not the population
- **Theoretical uncertainty**: Our theories are simplifications

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

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

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

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

The logic of statistical inference

- **Probability theory**: Given a known process, what data will we see?

The logic of statistical inference

- **Probability theory**: Given a known process, what data will we see?

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?

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?

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

Roadmap

The Big Picture

Version Control and Git

Version Control and Git

The problem: managing files over time

- Have you ever had files like this?

The problem: managing files over time

- Have you ever had files like this?
  - → `thesis_v1.docx`

The problem: managing files over time

- Have you ever had files like this?
  - → `thesis_v1.docx`
  - → `thesis_v2_final.docx`

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`

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`

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

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?

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?

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?

Version control: a better way

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

- One file, complete history

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

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

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

Why version control for research?

- **Reproducibility**: Track exactly what you did and when

Why version control for research?

- **Reproducibility**: Track exactly what you did and when
- **Backup**: Your work is safely stored, even if your laptop dies

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

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

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

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

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

The basic Git workflow

1. **Make changes** to your files (write code, edit text)

The basic Git workflow

1. **Make changes** to your files (write code, edit text)
2. **Stage** the changes you want to save

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”

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

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

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

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

Ways to use Git

- **GitHub web interface**: Create repos, upload files, edit directly

Ways to use Git

- **GitHub web interface**: Create repos, upload files, edit directly

  → Simple but limited

Ways to use Git

- **GitHub web interface**: Create repos, upload files, edit directly
  → Simple but limited
- **Command line**: Most powerful and flexible

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`

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

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

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

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

Assignment 1

- Create a GitHub account (if you don't have one)

Assignment 1

- Create a GitHub account (if you don't have one)
- Create a **public** repository for this course

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

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

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

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

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

What makes a good analysis?

- Clear research question

What makes a good analysis?

- Clear research question
- Appropriate data for the question

What makes a good analysis?

- Clear research question
- Appropriate data for the question
- Right statistical model for the data

What makes a good analysis?

- Clear research question
- Appropriate data for the question
- Right statistical model for the data
- Correct interpretation of results

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

Looking ahead

- Next session: Applied regression

Looking ahead

- Next session: Applied regression
- Regression as conditional expectations

Looking ahead

- Next session: Applied regression
- Regression as conditional expectations
- Multiple regression and control variables

Looking ahead

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

For next week

- Read Urdinez & Cruz (2020), chapters 1–5
- Read Gelman et al., chapters 6–7 and 10
- Read BdM & Fowler, chapters 5 and 10
- Review your notes on OLS from AQMSS-I
- Start Assignment 1

Questions?