Claude Code for Academic Researchers

A Methods Workshop in AI-Assisted Research

Instructor: Brady Allardice (PhD Candidate, UPF / IBEI) **Contact:** `brady.allardice@upf.edu`

Format: Five 2-hour sessions, hands-on

Course description

AI coding agents — Claude Code, Cursor, and similar — sit inside your project folder, read your files, and edit them in place. They are a step beyond chatbots and Copilot-style autocomplete, and using them well takes a different mental model than either. This workshop teaches that model for academic research: agentic AI as a working collaborator across the empirical pipeline, from data cleaning and merging through model estimation, replication, writing, and version control. The course is hands-on; each session is built around exercises on real data, and the capstone is a small replication of a published paper.

Audience and prerequisites

For grad students, postdocs, and faculty across the social sciences. Comfort with one statistical environment (Python, R, or Stata) is assumed; no prior agent or LLM-tooling experience required. Before Session 1, participants install Claude Code CLI, Python 3.9+ with `pandas` / `numpy` / `statsmodels` / `matplotlib`, git, VS Code with the Claude Code and LaTeX Workshop extensions, and a TeX distribution. A prework setup guide is distributed one week in advance.

Learning outcomes

By the end of the workshop, participants will be able to:

- Distinguish agentic AI from chatbots and autocomplete tools, and pick the right approach for the task at hand.
- Manage context windows, scoping, and verification across long agentic sessions.
- Use git, GitHub, and LaTeX in VS Code as one integrated environment alongside Claude Code.
- Build custom skills, subagents, hooks, and MCP connections to extend the agent for recurring research tasks.
- Replicate a published empirical paper from raw data to final figure with agentic AI as the primary tool, while retaining methodological control.

Schedule

Session 1 — Introduction to Claude Code. What Claude Code is and where it sits in the spectrum from manual coding through autocomplete, web chat, and full agentic systems. The RA analogy: delegating concrete tasks, verifying outputs, retaining control. Terminal, VS Code, scripts vs. notebooks, virtual environments, context windows, and permission modes. *Exercise:* set up the project, run a small data task end to end, write a first CLAUDE.md.

Session 2 — Research Pipelines. Working within the context window: five practical strategies for keeping long sessions coherent. The core agentic workflow — *describe* → *review* → *generate and validate* → *save*. Six validation strategies for catching agent errors before they propagate. *Exercise:* the Swiss Franc Shock data pack — merge, explore, specify, estimate, run robustness, and produce a publication-ready LaTeX regression table.

Session 3 — Git and LaTeX in VS Code. Why version control matters *more*, not less, when an agent edits your files directly. Minimum-viable git: `init`, `add`, `commit`, `diff`, `log`, `checkout`. `.gitignore` and `.claudeignore`. Moving from Overleaf to LaTeX in VS Code so paper, code, and data all live in one project. *Exercise:* build a short paper from scratch using git, LaTeX, and Claude Code together.

Session 4 — Skills, MCPs, and GitHub Collaboration. GitHub for team work: branches, pull requests, the three-step ritual, conflict resolution. Skills as packaged knowledge for recurring tasks (abstract audits, regression tables, spec validation) and the routing rule for where they belong. MCPs as connections to external tools, with the Zotero MCP for literature work and the GitHub MCP for collaboration. *Exercise:* build a custom skill; run a mini literature review through the Zotero MCP; submit a PR to the course repo.

Session 5 — Subagents, Hooks, and Capstone. Subagents as *roles* (a referee, a copyeditor, a methodologist) versus skills as *knowledge*. Multi-agent pipelines and composition patterns. Hooks for automation: research audit logs, auto-commit, `claude-ignore`. Choosing between hook, skill, and subagent. Beyond the terminal: Claude in the cloud, scheduled tasks, and the Claude API. *Capstone:* replicate Figure 1 of a published empirical paper from raw data, run a small extension, and write up the result.

Assessment

No grades. Two tracks for the capstone: (a) a guided replication of a provided paper, or (b) integration of Claude Code into one stage of your own ongoing project, using at minimum a CLAUDE.md, plan mode, git, and one power feature (hook, skill, MCP, or subagent). Deliverable is a version-controlled project folder containing data, code, output, and a compiled .tex write-up.

Selected resources

- *Ten Ways to Use Agentic AI in Academic Research* — course notes (distributed)
- Anthropic Claude Code documentation
- Course repository and exercise materials distributed at enrollment