Lecture 27

How to use AI to turbo-charge your research productivity

Tyler Ransom
ECON 5253, University of Oklahoma

Today's plan

1. What are LLM Agents?
2. What tools are available?
3. Why should academics care?
4. How can we better use these tools?
5. Application

What are LLM Agents?

Agents

- LLMs by themselves aren't solving a dynamic optimization problem
  - This severely limits what they can do
- However, recent advances have made "reasoning" possible
  - The LLM iterates on itself to make sure its final output makes sense
  - This reduces the prevalence of "hallucinations"
- "Agents" also arise from giving LLMs a "goal" to achieve
  - Similar to reinforcement learning
  - Hope is that agents can autonomously solve problems without human intervention

What tools are available?

Chat-based tools

There are 4 major players for chat-based LLMs (January 2026):

- OpenAI's `GPT-5.2` - 400K context window, can create spreadsheets & presentations
- Google's `Gemini 3 Pro` - state-of-the-art reasoning, multimodal
- Anthropic's `Claude Opus 4.5` - best-in-class for coding and agents
- `DeepSeek R1` - open-source, comparable to frontier models at fraction of cost

Each product may have differing derivative features

- Gemini can search YouTube; Microsoft Copilot uses GPT models
- Freemium model; $20/month unlocks most features (Premium: $200/month)

AI coding agents and IDEs

A major development: AI can now write code autonomously, not just autocomplete

- `Cursor` - AI-native IDE; $500M ARR; expects 20% of coding via agents by end of 2026
- `Claude Code` - Anthropic's CLI tool for agentic coding
  - Explores codebases, writes code, runs tests autonomously
  - Works in your terminal alongside your existing tools
- 85% of developers now regularly use AI tools for coding

GitHub Copilot coding agent

GitHub Copilot now includes a full **coding agent** that can submit pull requests:

- Assign an issue or describe a task; Copilot creates a PR autonomously
- Works in its own ephemeral dev environment (powered by GitHub Actions)
- Can explore code, make changes, run tests and linters
- When done, requests your review; you can leave comments for iteration
- All PRs require human review; Copilot can't approve/merge its own work
- **Agentic memory** (Jan 2026): learns from your repos over time
- Free for academics!

NotebookLM for research communication

Google's `NotebookLM` transforms how you communicate research:

- **Audio Overviews ("podcasts"):** Two AI hosts discuss your paper in conversation
  - Great for making dense papers accessible
- **Slide Decks:** Generate presentation slides from your sources in seconds
  - "Detailed" slides for handouts; "Presenter" slides for talks
- **Infographics:** Create visual summaries without design skills
- **Deep Research mode:** Synthesizes dozens of sources into long-form reports
- Upload PDFs of papers, and it handles citation and synthesis

Other AI-based tools

- `Elicit.org` and `Consensus.app` for literature reviews
- GitHub Copilot (free for academics!) for code completion
  - LaTeX/RMarkdown files can be thought of as code
  - Code completion then becomes "writing completion" in these instances
- `lex.page` for writing and editing
- `Microsoft Copilot` for Word, PowerPoint, etc.
- `Perplexity` for AI-augmented internet search; interfaces with DeepSeek-R1

Which product(s) should I use?

- Each product on the previous slides has strengths and weaknesses
  - You must experiment to find the best fit for your needs
- **Claude Code** is an extremely capable general research assistant
- **For coding:** Claude Opus 4.5 or Claude Code; Cursor for IDE experience
- **For research/writing:** Claude or GPT-5.2 for long-context work
- **For literature reviews:** Elicit, Consensus, or NotebookLM
- **For presentations:** NotebookLM can generate slides from your papers

Why should academics care?

How AI Could Transform Academic Research

- `Joshua Gans` had o1 write a paper that got `published` at *Economics Letters*
- Shows that AI can dramatically compress research timelines
- Gans argues in a `blog post`:
  - The current "presearch" model may become obsolete
  - Research could become cheaper than search
  - Why look up papers when AI can generate research on demand?
  - The academic system will need to adapt
  - Direction of scientific progress likely to shift markedly

My own experience

- In October 2024, I released a health economics paper (see **here**)
- This was far afield from my normal research expertise
- The idea came from reading 10-12 popular science books on the topic
- Producing this paper by myself would not have been possible pre-2024
  - I used Elicit for literature review
  - GPT-4 and Claude 3.5 Sonnet for coding
  - Claude 3.5 Sonnet for technical explanations outside my expertise
  - Claude 3.5 Sonnet for outlining paper structure and writing refinement

How can we better use these tools?

Claude Code demonstration

Live demo: Reviving a 5-year-old research project with Claude Code

- Old projects often have:
  - Undocumented code
  - Disorganized file structures
  - Missing dependencies or outdated packages
  - No README or setup instructions
- Claude Code can help bring them back to life

What we'll do in the demo

1. Point Claude Code at a dusty old project
2. Ask it to explore and understand the codebase
3. Have it create documentation (README, code comments)
4. Organize files and clean up the structure
5. Update dependencies and fix any issues
6. Create visualizations or outputs

Pro tips for using AI agents

Use AI to document and organize your files

- "Create a README for this project explaining what each file does"
- "Organize these scripts into logical folders"
- "Add a CLAUDE.md file so future AI sessions understand this codebase"

Pro tips for using AI agents

Have AI be your graphic design expert

- "Create a flowchart showing the data pipeline"
- "Make this figure publication-ready with better colors and fonts"
- LaTeX/TikZ diagrams from hand-drawn sketches

Pro tips for using AI agents

Have AI document all of your code

- "Add docstrings to every function in this file"
- "Create a data dictionary for this dataset"
- "Write comments explaining the non-obvious parts of this code"

So far, I've used LLMs to help me ...

- Fill out bureaucratic forms
- Write code that automates grading
- Write code that systematizes data analysis (for ease of replication)
- Write code to create data visualizations
- Write unit tests of code
- Prepare discussion slides for a conference
- Prepare peer review reports
- Prepare this slide deck
- Reduce word count of an abstract
- Improve sentence clarity in a paper
- Write survey questions that a survey methodologist would approve of
- Explain poorly written abstracts / papers in simpler terms
- Invert mathematical functions
- ... not to mention a bunch of stuff in my personal life

Staying on top of new developments

The following sources are helpful for keeping on top of new developments:

- `One Useful Thing` Substack by Ethan Mollick
- `Marginal Revolution` blog by Tyler Cowen & Alex Tabarrok