AI for Everyday Coding

Practical Tools for Research

Jenna Landy | Wednesday December 10, 2025

Dana Farber Cancer Institute Data Science Training Session

Preface

- I do not claim to be an AI expert, but I code a lot and have benefited from these tools
- My background:
  - PhD in Biostatistics @ Harvard, BS in Statistics and Data Science @ Cal Poly SLO
  - I've made R packages, python packages, jupyter lab extensions, shiny apps, bookdowns, quarto and rmd websites, parameterized rmd reports — AI tools make all of these things easier
- You may benefit from this tutorial if…
  - You write code regularly and work with data
  - You're interested in lightweight tools that fit naturally into your existing workflow
  - You want to get better output from LLMs but don't need to understand their internal architecture
  - You want AI to *help you* do research
- You may not like this if…
  - You're hoping to learn how LLMs work or how to train AI models
  - You're looking for cutting-edge agentic workflows, autonomous systems, or the heaviest models.
  - You're a software engineer or otherwise needing system-level integration or production ML engineering
  - You want AI to do research *for you*

Coding Poll

- R vs Python
- DFCI Cluster vs Local

AI Use Poll

- Online chat (e.g., with ChatGPT)
- AI Autocomplete (e.g., with GitHub Copilot or Cursor)
- Within-IDE chat (e.g., with GitHub Copilot or Cursor)

Outline

- Introduction: key uses, limitations, and tools
- Understanding environmental impact
- Efficient prompting
- Case study with tutorials
    - Question: how do the dynamics of vaccine rates throughout 2021 differ across California counties?
    - Part 1: prompting for research with Chat GPT
    - Parts 2-3: in-IDE chat and autocomplete with GitHub Copilot in RStudio and VS Code
    - Parts 4-5: refactoring code and agentic AI with Cursor
- Office hour for tool setup

Introduction

How can AI help with coding?

- Coding speed
- Refactoring
- Documentation
- Debugging
- Generating boilerplate

- Explaining unfamiliar code
- Cross-language translation
- Command line and bash

…

What are it's limitations?

- AI may hallucinate package names or functions (always test!)
- Not always correct in statistical reasoning without oversight
- General advice:
  - Never implement suggestions you don't understand
  - Always test suggestions
  - Use your brain for research and reasoning, only use AI to automate simple or repetitive tasks

To get this out of the way…

- NEVER input private data into LLMs (patient data, financial information, etc.)
- NEVER input something you wouldn’t want public to an LLM

Tools Overview

- ChatGPT
  - Free option, Harvard faculty and students have access to enterprise (better models, more private)
  - Ideas, debugging, explanations, multi-step workflows
  - *Best for big-picture discussions*
- GitHub Copilot (extensions available in RStudio, VSCode, and Jupyter Lab)
  - Free option, Pro version free for students, teachers, open source maintainers, or $10/month
  - Autocomplete and inline suggestions informed by context of current/open files
  - Copilot chat for quick Q&A on current code (not available in RStudio)
  - *Best for speeding up everyday coding, especially for small scale projects*
- Cursor
  - Free option, Pro is free for students for one year or $20/month
  - Autocomplete and inline suggestions informed by full project context
  - Chat for Q&A with awareness of the entire codebase
  - Agentic tools for multi-step coding tasks with several specialized models
  - *Best for global changes and large scale projects*

Tools Overview

- All are generative pre-trained large language models (LLMs)
- Utilizes a massive and diverse set of publicly available text and code
  - Public code repositories
  - Documentation and issues
  - StackOverflow and similar forums
  - Academic articles and books
  - General web text
- Trained with a predict-next-token objective
- Additional fine-tuning improves usefulness and safety
  - Supervised examples
  - Reinforcement learning from human feedback

Environmental Impact

A note on energy consumption & environmental impact

- Training LLMS has significant environmental and hardware cost
- Querying LLMs use energy and emit CO_2

A note on energy consumption & environmental impact

- Training LLMS has significant environmental and hardware cost
- Querying LLMs use energy and emit CO_2

- Query type length matters
  - Long prompts / responses cost more (you can ask for concise responses!)
  - Searching is cheaper than deep reasoning or agentic workflows

- Utilization vs single-query costs
  - Tab complete is cheaper per-query, but hundreds of queries per hour in the background add up

Relative energy consumption of query types

• Order of increasing impact	Energy
1. Textbook search / asking a friend	0
2. Google search	~0.3 Wh
3. GitHub copilot autocomplete (lightweight model, small context window)	
4. Cursor autocomplete (larger context)	
5. GitHub copilot chat (midsize “chat” model, limited context)	
6. Cursor chat (stronger models, larger context)	
7. ChatGPT (especially reasoning models)	~3 Wh
8. Cursor agent / multi-step tasks (many LLM calls + iterative refinement)	
9. 10W LED bulb for 1 hour	10 Wh
10. Driving 5 miles in an EV	1500 Wh

Average Electric Car kWh Per Mile (2024)
Search Engines vs AI: energy consumption compared (2025)

But alternatives can be costly too

- Efficient code can save energy long-term
  - If I repeatedly run buggy code, that uses resources too. Could Cursor have helped avoid this?

- Better tools can require fewer queries
  - If it takes me 10 google searches to find a useful Stack Overflow, asking ChatGPT would be comparable.

We already make these types of decisions daily

- Driving vs biking to work
- Buying new vs fixing or building from scratch
- Ordering out vs cooking at home
- Central heat vs bundling up

An intuition about relative costs is key

Human time and effort

Environmental cost

A Few Resources

- Energy and Policy Considerations for Deep Learning in NLP (Strubell et al., ACL, 2019)
- Carbon emissions and large neural network training (Patterson et al., 2021)
- The growing energy footprint of artificial intelligence (de Vries, Joule, 2023)
- Explained: Generative AI's environmental impact (Zewe, MIT News, 2025)

Effective Prompting

Prompting: overview

- A good prompt reduces the total number of interactions required
- Be specific!
- Include context (documents are allowed!)
  - I like to think: *If I were to ask a person for help*
    - *What kind of person would I ask?* I should provide context for **field / specialty and preferences for types of tools**.
    - *What information would I need to provide?* I can assume they understand the field, so I should focus on the **specifics of my data / model / limitations / requirements / considerations / edge cases / ideas / worries / results / issues so far /** …
    - *What do they need to know about me?* I can **explain my background, experience level, and goals** so suggestions are tailored to me
  - Required context for code prompts
    - *Language (e.g., R, Python)*
    - *Packages you want to use (e.g., tidyverse, Bioconductor, scikit-learn)*
    - *Data structure (e.g., data frame with given column names)*
    - *Constraints (e.g., use base R, max scale of data, vectorized for speed)*

Prompting: overview

- ChatGPT's default behavior is wordy with a lot of explanation. You can change this with a preface to the conversation.
  - My default: *"Always be concise. Use small tables when applicable (should fit on printer paper). Use the package::function notation whenever calling functions."*
  - Strict option: *"Always output one R code block with efficient code and minimal comments"*
  - Personality requests*: "Pretend you are a reviewer for the journal Biostatistics and be very critical"*
  - Interest requests: *"I'm interested in learning how R is different from other languages (like lazy evaluation). If you provide code, include a short 'R tips' section at the end"*

Prompting: structure

- **First query only:**
  - Preface for preferred output (with “always” or “throughout this chat”)
  - This is my big picture scientific question and goals (publication, exploratory, etc.).
  - This is my experience level and specialty.
  - These are my limitations. For example:
    - *Can only use open access data*
    - *Don’t have cluster access*
    - *Only want to use standard statistical approaches, don’t want to develop a new method*
    - *Need an algorithm that works to analyze 1M+ cells at a time*
  - In this chat, I’ll be asking for help with this specific component.
- **All queries:**
  - Specific question:
    - *What do you need help with (e.g., implementing this algorithm, refactoring my code, narrowing in on a research question)*
    - *Why do you want to do that (may lead to alternative suggestions)*
    - *How you want it done (e.g., search the web, look at my code base, use R code, use tidyverse)*
  - Preface for preferred output (with “for this query”)

Prompting: key research questions

- **For a new dataset/question:** What method is standard in the field right now?
  - What are its key limitations or assumptions?
  - Why might it work well/poorly for my data?
  - List some key connections between this analysis and <secondary field>
  - Provide key sources to understand the landscape of options (review papers, textbooks, blog posts)
  - *Helps create an outline for literature review and initial papers to dive into*

- **For a new model/use case:** Has this type of analysis been done in this application area before?
  - What about in other domains?
  - What are the names of key methods or frameworks related to what I'm proposing, regardless of the domain?
  - *Helps understand novelty and connections to other areas (you can't google a term you don't know!)*

Prompting: key research questions

- **For a new field/skill:** What’s the best way to learn this?
    - What are the most important building blocks I should learn to be able to do research in this field?
    - Provide a timeline of key methods in this field and rank how important they are to understand from “niche” to “foundational”
    - Is there a publicly available, clean, simple dataset that I can explore and test methods on?
    - *Helps guide your study plan*

Prompting: key research questions

- **For a new coding project:** How to get started?
  - This is the data, model, and my preferred language and experience level
  - This is the kind of code I want to write
    - *I want to create an interactive plot that I can host on the web*
    - *I want to create a tutorial, textbook, or website*
    - *I want to implement a user-friendly R software package*
    - *I want to implement the model in easy-to-use R functions*
    - *I want to fit the model on a single dataset in an Rmd/qmd notebook*
  - Are there tools I can use to make implementing this model easier? AI may introduce you to:
    - *Shiny for interactive dashboards*
    - *quarto for websites or textbooks*
    - *usethis for R package creation*
    - *optim() for MLEs*
    - *Stan, NIMBLE, or JAGS for MCMC sampling*

Case Study:
2021 COVID Vaccine Rate Dynamics across CA Counties

Case study

- How do the dynamics of vaccine rates throughout 2021 differ across CA counties?.
- Part 1: with Chat GPT
  - Prompting for research
- Parts 2-3: with GitHub Copilot
  - Part 2: Data filtering and algorithm development in VS Code
  - Part 3: Data visualization in RStudio
- Parts 4-5: with Cursor
  - Part 4: Multi-file refactoring
  - Part 5: Example of agenitc AI