Agentic AI for Economic Research

PIER Research Workshop 2026

Kanis Saengchote
Chulalongkorn Business School
26 June 2026

Today's Agenda

1. Prompting Agents for Research
2. Organizing Your Workspace
3. Using Style Guides
4. Logging Your Sessions
5. Recap & Key Takeaways

Prerequisites: R + Antigravity environment already set up (see setup guide)

What is an Agentic AI Coding Assistant?

- An AI that can **read, write, and execute code** on your machine
- It operates in a loop: **plan** → **act** → **observe** → **refine**
- Unlike chat-only AI, it has **agency** — it can create files, run scripts, install packages
- Think of it as a **research assistant that can code**

Examples: Google Antigravity (Gemini), Claude Code, OpenAI Codex

The Three Platforms

Antigravity (Gemini)

Google DeepMind
Free tier available
Desktop app

Claude Code (Claude)

Anthropic
Terminal-based
Subscription required

Codex (ChatGPT)

OpenAI
Cloud sandbox
ChatGPT Pro/Team required

All three can run R scripts and manage your workspace

Part 1: Prompting Agents for Research

Our Running Example: mtcars

Built-in R dataset with 32 observations on 11 variables for automobile design and performance.

- **Source:** Part of the built-in `datasets` package in R. Load using `data(mtcars)`
- **Key variables:** mpg, hp, wt, am (transmission), cyl, disp

---

Small enough to iterate quickly, rich enough for real econometrics

Prompting for Regression Analysis

> *"Run an OLS regression of mpg on hp, wt, and am using the mtcars dataset. Show results with both classical and robust (HC3) standard errors."*

Key insight: Be specific about what you want — the agent will execute exactly what you ask.

Interactive Prompting: Let the Agent Clarify

Instead of specifying everything upfront, ask the agent to ask *you* for input.

> *"I want to analyze the mtcars data. Please ask me what variables I'd like to use and what model to run."*

This triggers the agent's interactive tool to ask the user for clarification.

Each Platform Has Its Own Tool

Antigravity (Gemini)	`ask_user` tool
Claude Code	`AskUserQuestion` tool
Codex	`ask_user_question` tool *(only available in Plan Mode)*

The concept is the same — the tool name differs per platform

Ask AI to Annotate Your Code

> *"Please fully annotate the R script with detailed comments explaining each step, the econometric logic, and interpretation of results."*

Why this matters: Annotated scripts serve as documentation, teaching materials, and audit trails.

Planning vs Execution

- Antigravity used to have a separate "Plan mode"; **it has been removed**
- The workflow is now: **choose the right model for the right task**
- **Planning phase:** Use a powerful thinking/reasoning model (e.g., Gemini 3.1 Pro, Opus 4.8, GPT-5.5) to design your approach, review architecture, evaluate trade-offs
- **Execution phase:** Switch to a faster, cheaper model (e.g., Gemini 3.5 Flash, Sonnet 4.6, GPT-5.4-mini) for writing code, running scripts, making edits

---

This applies across all platforms, not just Antigravity

Model Selection: Think Big, Execute Fast

🧠	**Planning / Reasoning**	Use a large, powerful model → Gemini 3.1 Pro (high), Opus 4.8 (low-max), GPT-5.5 (low-extra high)
⚡	**Execution / Coding**	Use a fast, efficient model → Gemini 3.5 Flash (low, medium, high), Sonnet 4.6 (low-max), GPT-5.4-mini (low-extra high)

Practical tip: In Antigravity, you can switch models mid-conversation using the model selector dropdown. Start with a thinking model for your research design, then switch to a fast model for implementation.

Thinking models are better at complex reasoning but slower and more expensive. Fast models are great for straightforward tasks.

Example: A Two-Phase Workflow

1. 🧠 *(Thinking model)* “I want to study the effect of transmission type on fuel efficiency in mtcars. What econometric approach would you recommend? Consider endogeneity concerns.”
2. 🧠 *(Thinking model)* Agent produces a detailed plan with model specification, variable selection rationale, robustness checks
3. ⚡ *(Fast model)* “Please implement the analysis plan above. Write and run the R scripts.”
4. ⚡ *(Fast model)* Agent writes code, executes it, generates output

Key insight: Separate thinking from doing — just like you would with a research assistant.

🧪 Exercise: Your First Agent-Driven Analysis

1. Open Antigravity with your workshop folder
2. Prompt: *"Run an OLS regression of mpg on hp, wt, and am using mtcars. Show classical and robust SE. Ask me if I want any additional analysis."*
3. Approve the commands when prompted
4. Ask the agent to fully annotate the resulting script

Time: about 8 minutes

Part 2: Organizing Your Workspace

Why Structure Matters

- Agents work best with **organized projects**
- Clear folder structure = better context for the agent
- **Reproducibility:** others (and future you) can navigate the project
- The agent can help you **design the structure**

Ask AI to Suggest a Structure

> *"Please suggest a workspace structure for an empirical economics research project. Include folders for data, scripts, output, and documentation."*

```
project/
├── data/          # Raw and cleaned datasets
├── scripts/       # R analysis scripts
├── output/        # Figures, tables, exports
├── docs/          # Documentation & guides
├── logs/          # Session logs
└── README.md      # Project overview
```

Agent Instruction Files

Place a file in your project root that the agent reads automatically every session.

GEMINI.md
Instructions for Antigravity (Gemini)

AGENTS.md
Universal convention (works across platforms)

CLAUDE.md
Instructions for Claude Code

These files persist your preferences so you don't have to repeat yourself

What to Put in Your Instruction File

```
# Project Instructions

## R Environment
- R executable: C:\Program Files\R\R-4.5.1\bin\Rscript.exe
- Always use set.seed(42) for reproducibility

## Coding Conventions
- Use tidyverse style
- Save all outputs to the output/ folder
- Annotate scripts with detailed comments

## Style Guide
- See docs/style_guide.md for graph and table formatting
```

🧪 Exercise: Organize Your Project

1. Prompt: *"Please suggest a workspace structure for my research project and create the folders."*
2. Prompt: *"Create a GEMINI.md file with instructions for this project. Include the R path and coding conventions."*
3. Review the generated files

Time: about 5 minutes

Part 3: Using Style Guides

Why Style Guides?

- **Consistency** across all your figures and tables
- Professional, **publication-ready output** from the start
- The agent follows your conventions **every time**
- No more manual reformatting

Graph Style Guide

- **Color palette:** light, clean colors inspired by institutional branding
- **Background:** white or very light gray (#FAFAFA)
- **Fonts:** clean sans-serif, appropriately sized
- **Minimal gridlines,** no unnecessary chart junk
- **Consistent** axis labels and titles

> *"Use light blue (#4A90D9) as the primary color, rose (#B5627E) as accent. White background, minimal gridlines, 300 DPI."*

Table Style Guide

- All coefficients to **2 decimal places**
- Standard errors **in parentheses** below coefficients
- Include: **N, Adjusted R^2, AIC/BIC**
- Significance stars: `+ p<0.1`, `* p<0.05`, `** p<0.01`, `*** p<0.001`
- Export as both **HTML and LaTeX**

Save Your Style Guide as a File

> *"Create a style_guide.md file in docs/ that specifies our graph and table formatting conventions. Reference this file in GEMINI.md."*

Key point: The agent reads this file and applies your conventions automatically in future sessions.

🧪 Exercise: Define Your Style

1. Prompt: *"Create a style guide for my research project. For graphs, use light blue (#4A90D9) as primary and rose (#B5627E) as accent, with white backgrounds. For tables, show all coefficients to 2 decimal places with standard errors in parentheses."*
2. Prompt: *"Now re-run the mtcars regression and generate the table and a coefficient plot using my style guide."*
3. Compare before and after

Time: about 8 minutes

Part 4: Logging Your Sessions

Why Log Your Sessions?

- Track **what you did** and **why**
- Record decisions and their rationale
- Create an **audit trail** for reproducible research
- Easy to resume work after a break
- Introduce yourself to **Markdown**

Quick Markdown Primer

```
# Heading 1
## Heading 2

- Bullet point
- **Bold text**
- *Italic text*
- +"""inline code+"""

> Blockquote for notes

| Column A | Column B |
|----------|----------|
| Data 1   | Data 2   |
```

Markdown is plain text that renders beautifully — perfect for research notes

Session Log Convention

Filename format: `yymmdd-slug.md` (e.g., `260626-mtcars-regression.md`)

```
# 260626  -  mtcars Regression Analysis

## Topic
OLS regression with robust standard errors on mtcars dataset.

## Description
Ran mpg ~ hp + wt + am with classical and HC3 robust SE.
Generated coefficient plot and formatted regression table.

## Decisions Made
- Used HC3 (not HC1) for small-sample robustness
- Chose 2 decimal places for coefficient display
- Applied institutional color palette for graphs

## Outstanding TODOs
- [ ] Add interaction terms (hp x am)
- [ ] Export final table to LaTeX
```

🧪 Exercise: Create Your First Log

1. Prompt: *"Please log today's session. Use the filename format yymmdd-slug.md in the logs/ folder. Include the topic, what we did, decisions made, and any outstanding TODOs."*
2. Review the generated log file
3. Optional: *"Create a todo.md file in the project root tracking our open items."*

Time: about 5 minutes

Recap & Key Takeaways

Key Takeaways

1. **Prompt specifically**, where a more precise prompt yields better output
2. **Use interactive tools**, letting the agent ask for clarification
3. **Organize your workspace**, as folder structure and instruction files yield consistent results
4. **Define style guides**, writing them once to apply them systematically
5. **Log everything** to ensure reproducibility

The Bigger Picture

- Agentic AI does not replace judgment; instead, it amplifies productivity
- The researcher remains the domain expert; the agent handles implementation
- Reproducibility and transparency improve when steps are scripted and logged
- These tools assist with any empirical workflow, including R, Python, Stata, and LaTeX

To automate execution (e.g., running Stata do-files or compiling LaTeX), the executable must be in your system PATH, enabling the agent to invoke the command directly.

Thank You

Kanis Saengchote
Chulalongkorn Business School
PIER Research Workshop 2026