Use AI as an RA for Economic Analysis

Kohei Kawaguchi, Hong Kong University of Science and Technology

Resources

- git clone git@github.com:kohei-kawaguchi/TestAI.git

Replacing Human RA with AI RA

- 100% replaceable
- Needs the same level of supervision
- Merits
  - Accelerate iterations
  - Parallelize tasks
  - Integrate research and implementation
  - Better implementation
- Demerits
  - Integrity
  - Training opportunity

General Principle

- Follow the best practices of project management
- Supervise AI RA **exactly** like human RA

On-demand data analysis is the best task for AI

- The task can be mathematically defined and described
- The end user is the supervisor themselves
- Little concern for maintainability

What AI is NOT good at

Economics

- Equilibrium, endogenous/exogenous, observable/unobservable, parameters of interest/nuisance parameters are exotic concepts to AI
  → Translate them into CS/statistical concepts
  → Test knowledge with an oral exam

Solving bugs/problems

- AI makes up a fake solution if asked to **solve** a problem
  → Always ask to find the **root cause** of the problem
  → Solution must be determined by the supervisor

Key framework

Pipeline management

- Package-like folder structure, git version control, document → implementation → test → validation workflow, command scripts

Theory-driven approach

- Supervisor writes the ground truth as a document

Multi-level validations

- Consistency check, unit tests, visualized reports, but **no** code inspection

Scope management

- Specify what can be changed and what cannot be changed

Solving and simulating an economic model

Planning

1. Create an issue and create a separate branch
2. Write down the model setting in a document
3. Ask AI to write down the equilibrium condition (oral exam)
4. Ask AI to write down the pseudo code (oral exam)
5. Ask AI to write down the issue and plan in a md file

Solving and simulating an economic model (continued)

Implementation

1. Ask AI to write functions in `src/` following the pseudocode

Verification

1. Ask AI to write unit tests in `scripts/test` and check all pass
2. Ask AI to list any inconsistency between the pseudocode and `src/` implementation

Solving and simulating an economic model (continued)

Validation

1. Ask AI to write an execution file in `scripts/pipeline` and execute it
2. Ask AI to write a reporting file in `scripts/report` and render it
3. Generate relevant comparative statics, spot any strange behavior, speculate the reasons or let AI investigate the root causes (never ask AI to solve the issue)

Rules

- Every time you change the implementation, make sure to repeat the same process
- Every time you streamline the implementation, require no backward compatibility
- At each of the above steps, ask AI to commit changes, so that AI can summarize the changes and status for reference

Running a model-free analysis

- No difference from the previous workflow if we specify tasks using math
- But this can be cumbersome for model-free analysis
- We often use informal instructions for RAs
- We often use economics jargon to describe the task
- This can hinder the use of AI in this context
- Solution: use the **oral exam** method intensively

Running a model-free analysis (continued)

Dynamic reporting

1. Ask AI to make a dynamic report loading the data
2. Ask AI to set up a live server to review the report
3. Ask AI to summarize the variables of the data sets

Oral exam

1. Explain to AI what you want to achieve
2. Ask AI whether it knows how to implement it
3. Ask AI to search relevant sources until it returns a legitimate answer
4. Once a legitimate answer is obtained, ask AI to summarize it in the report

Running a model-free analysis (continued)

Implementation

1. Ask AI to run the analysis in the report

Validation

1. Ask AI to refresh the report and verify the result

This oral exam method works for **data cleaning** tasks as well

Which AI to use

- OpenAI Codex CLI
  - Smart, consistent, but not fun to work with
  - Currently, needs WSL or Linux container on Windows
  - Install the CLI and VS Code/Cursor extension
- Claude Code CLI
  - Less smart than Codex, but lovely
  - Company understands the importance of UI/UX
  - Works with Windows
  - Install the CLI and VS Code/Cursor extension
- Cursor
  - Bad at task management

Demonstration

Tips

- Make a docs/issue/ folder and let AI summarize the problem and plan for each issue
- Make a gitignored scripts/temporary/ folder for AI to write temporary scripts for investigating problems
- Never let AI automatically git add/commit/pull/push, explicitly ask AI to commit
- Periodically ask AI to streamline the document and implementation, make it explicit no backward compatibility is needed
- Let AI write down pseudocode of the implementation
- Let AI add a link to source for any method it suggests
- Let AI read the document and git history to understand the current issue and status