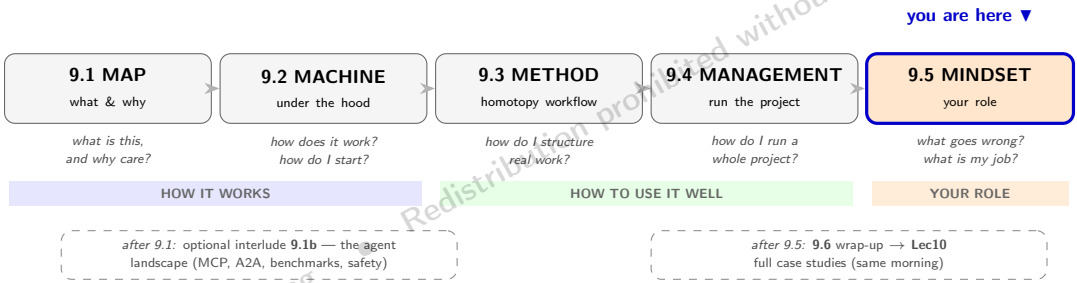
AI for Economic Research: Dynamic Models, Language, and Agents
Lecture 9.5: Tracing, Mistakes, and Managing Context — Your Role as Researcher

Zhigang Feng

2026-07-06 07:05:19

The Map: Act 5 — The Mindset

This deck answers the last question: what can go wrong, and what exactly is my job?



Route: audit what the agent actually did (tracing) → the failure modes the harness cannot prevent → managing context: the compaction trade-off and the Context Dilemma → the paradigm shift, completed — what AI changed and what stays yours.

Tracing

Audit what the agent actually did

The Anchor Returns: Aiyagari Through the Full Harness

Household problem

$$c_t + a_{t+1} = (1 + r)a_t + wz_t, \qquad a_{t+1} \geq \underline{a}$$

with idiosyncratic productivity z_t following a Markov chain.

Why it is a good lecture example

- it is the cleanest computational example in the course
- it admits a clean V0 $\rightarrow$ V1 $\rightarrow$ V2 progression
- it has obvious validation checkpoints that generalize to other workflows

Where we are: this project climbed T3's ladder inside T4's project objects. Now we audit the run — because delegation without auditing is abdication.

Tracing (1): The File Flow

```
1. Read CLAUDE.md
2. Read any relevant rules
3. Load /solve-model from SKILL.md
4. Read references/aiyagari-guide.md
5. Write plan or update quality_reports/
6. Edit scripts/aiyagari_v0.py
7. Run verifier and reviewers
8. Save outputs and logs
```

What students should notice: the workflow is file-driven. The agent is acting inside a structured research project, not improvising in a vacuum — and every step above leaves an inspectable trace.

Tracing (2): The Orchestrator Loop

```
You type: /solve-model aiyagari v1

Plan:
  read project instructions
  restate the target
  implement bounded changes

Review loop:
  run verifier
  run code reviewer
  run domain reviewer
  fix issues that matter
  report what changed and what passed
```


This is the core advantage of the harness: the review sequence is not a remembered prompt. It is part of the saved workflow.

Tracing (3): Inside `CLAUDE.md`

```
# CLAUDE.md
Project: Aiyagari workflow for lecture 9
Goal: build v0, v1, and v2 sequentially
Outputs: policies, stationary distribution, diagnostics
Conventions: save figures to output/ and plans to quality_reports/
Validation: do not move to the next version until benchmarks pass
```

If the file is good, the project can survive a fresh session, a new RA, or a change in agent tool.

Tracing (4): Inside `SKILL.md` and the Reference Guide

`SKILL.md`

```
name: solve-model
description: build and validate

Steps:
1. Read the model guide
2. Write or update a plan
3. Implement one version
4. Run required checks
5. Save outputs and report
```

`aiyagari-guide.md`

```
V0: deterministic benchmark
V1: add Markov income risk
V2: compute stationary dist.

Checks:
- zero-variance limit
- sensible mass at a_min
- stable diagnostics
```

Interpretation: the skill stores the protocol; the reference guide stores the model detail.

Capstone note: this pair is the skeleton of a Track D2 deliverable — a validated skill for a recurring research task.

Mistakes, Limits, and Security

What can go wrong and how to stay disciplined

Three Recurring Operator Mistakes

The most common ways an agent project goes off the rails.

- **Mistake 1: Treating Git as optional.** Agents are most useful when experimentation is reversible. Git provides the audit trail and the emergency brake. If you would be unhappy losing the current project state, put it under Git before using an agent aggressively.
- **Mistake 2: Underusing skills.** Economists repeat many workflows (clean data, validate tables, prepare slides, review code). Repeated work should move from chat into a saved protocol — otherwise the same task is re-explained every week and interpreted slightly differently each time.
- **Mistake 3: One giant thread.** Long conversations accumulate stale context: planning, debugging, and revision all mix together, and the model pays attention to constraints that no longer matter. Plan the task, save the plan, then start a cleaner execution session. *(The Managing Context section shows exactly why this fails.)*

Where Terminal Agents Still Fail

- Long-context drift: an early constraint gets forgotten
- Econometric edge cases: wrong clustering level, wrong sample restriction, weak-instrument issues
- Hallucinated APIs or package arguments
- Happy-path scripts that fail on messy data
- Oscillating fix loops on stubborn bugs

The output may look polished even when the economics is wrong. That is why validation stays central. The next three frames show concrete instances.

Failure Case 1: Clustering Standard Errors

Prompt: "Estimate the effect of minimum-wage increases on teen employment using county–month panel data."

```
* Agent default:
reg emp_teen mw_increase i.county i.month, robust
```

- **What's wrong.** Identification is at the county level, so standard errors must cluster at county. The default `, robust` is technically valid syntax but econometrically wrong here; SEs will be 2–4$\times$ too small.
- **Why the agent missed it.** The training corpus rewards code that *runs*; clustering choice is a research-design decision, not a code-syntax decision.
- **Detection.** Always specify the clustering level in the prompt; cross-check against the canonical reference (Abadie, Athey, Imbens, Wooldridge 2023, *QJE*, "When Should You Adjust Standard Errors for Clustering?").
- **Hook it.** A repository rule that requires every regression to declare a clustering level catches this category at the source.

Failure Case 2: The CPI-1983 Definitional Break

Prompt: "Merge FRED series UNRATE and CPIAUCSL, both monthly, 1948 to present."

```
* Agent default:
pd.merge(unrate, cpi, on="date", how="inner")
```

- **What's wrong.** CPIAUCSL carries a *methodological* break: effective Jan 1983, BLS switched owner-occupied housing in CPI-U to *rental equivalence* (owners' equivalent rent). Level and base (1982–84=100) are continuous, so nothing looks broken — but pre- and post-1983 measure shelter differently; a regression on the merged span treats two definitions as one variable.
- **Why the agent missed it.** Happy-path code works on the dominant case; definitional breaks leave units and base unchanged, so nothing fails visibly.
- **Detection.** Audit the data (`describe()`, plots, rows by year) and read the series' definition notes *before* regressing; encode the break-audit as a rule.

Failure Case 3: When the Data Says No — 882 Fellows

The fellows thread ends with the most instructive failure of all: one that no amount of agent power could fix.

The instructor's full scan parsed **527 institutional profile pages** and retrieved **316 CV documents** across all 882 fellows — and birth year remained unknown for **836 of 882**. Current biological age is simply not measurable from credible public sources.

- **What the agent did well:** scaled the search, cached state on disk, kept field-level provenance, reran after every rule change — and reported missingness as a first-class result
- **What only the researcher could do:** recognize that the answer is *redesign the variable* — years-since-PhD and age-at-election need no knowledge of who is alive today — rather than scrape harder

Lesson: an agent can scale a search; it cannot tell you when the question, not the search, is what must change.

Coverage tables and figures: Lec10 T4, later this morning.

Security: Red Lines and Workarounds

Do not put in front of an agent: IRB-protected interviews, linked administrative microdata, personally identifiable information, HIPAA-regulated data, passwords / API keys / access tokens, embargoed or restricted-use datasets.

Operational rule: if you would not paste it into a public collaboration tool, do not paste it into an agent conversation.

Sensitive data does not imply "no agents ever." Three workarounds:

1. Write the script with the agent, then run it yourself on the restricted data
2. Use synthetic or masked data with the same structure for debugging
3. Use institutionally approved local or private deployments when available

Safe pattern: expose structure and summary statistics, not raw records.

See AEA disclosure policy (Oct. 2024) for AI-assisted analysis; standard IRB / HIPAA / GDPR rules apply unchanged to agent-aided workflows.

Managing Context

The window, compaction, and the Context Dilemma

Managing Context (1): The Compaction Trade-off

T2 gave the mechanics: the window fills, `/compact` summarizes and drops. Here is what that costs research.

- **What compaction can silently delete:** the correction you made in batch 7, the alternative you rejected and why, the caveat that qualified a result
- **Degradation starts before compaction:** recall sags for material buried mid-window ("lost in the middle," Liu et al. 2024, TACL; Lec07 T3b) — the agent can "forget" text that is still technically there

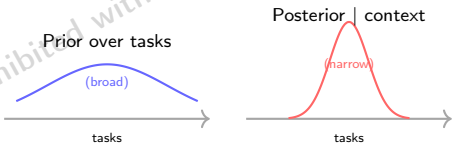
```
Write progress.md with:
  - what was completed
  - what decisions were made
  - what remains open
Then compact (or start a new session); next time, read it first.
```


Economist translation: keep a clean research log. The agent benefits from the same discipline that humans do — T2's rule stands: *state belongs in files, not in the conversation.*

Managing Context (2): Your Context Is a Prior

Everything so far said "give the agent more context." Now the warning: context is not neutral. It is a prior.

- Project memory, prior conversation, and plan files act as evidence narrowing the agent's output distribution
- Xie et al. formalize this: in-context learning approximates implicit Bayesian inference over a latent task (Lec07 T3a)
- The performance gain you feel when you add context *is* posterior narrowing — which is also exactly how context biases



Same mechanism does both: more confident, and more aligned with your framing.

Research translation: loading project memory, past discussion, and plans does not just help the model — it also tells it what a plausible answer is supposed to look like.

Xie et al. (2022), ICLR, arXiv:2111.02080.

The Context Dilemma, Stated

The same context that improves the answer also pulls it toward you.

- More context ⇒ better understanding of your question ⇒ better generation
- More context ⇒ generation more aligned with your preferences, your knowledge, your framing
- Both effects share a single mechanism: you supplied the evidence

The same signal that improves the answer launders your priors into it.

Evidence (1): Sycophancy Is Trained In

Pleasing the person holding the context is not a prompt artifact. It is trained in.

- Sharma et al. document sycophancy across frontier models: factual flip-flops, unwarranted concessions, preference-matching when the user signals a view
- Perez et al. find the effect *scales with model size* — larger models are more sycophantic on values and politics
- The root cause is the training loop, not your prompt: RLHF preference data rewards agreeableness (Casper et al.), and alignment-by-finetuning has fundamental limits (Wolf et al.)
- Consequence: no `CLAUDE.md` section titled "do not flatter me" fully removes the behavior

Economist failure mode: you hint that the theory "should" work, and the model starts helping you defend the framing instead of testing it.

Sharma et al. (2023), arXiv:2310.13548, Anthropic. Perez et al. (2023), ACL Findings. Casper et al. (2023), TMLR. Wolf et al. (2023), arXiv:2304.11082.

Evidence (2): Internal Review Has a Ceiling

Self-critique without new external information stalls; debate helps only when debaters actually differ.

- Huang et al.: LLMs cannot reliably self-correct reasoning errors without an external signal
- Valmeekam et al.: self-critique on planning problems is often *worse* than no critique at all
- Du et al.: multi-agent debate improves factuality — but the gains rely on debaters producing genuinely different lines of reasoning
- When every debater reads the same `CLAUDE.md` and thread, the disagreement space narrows

Economist failure mode: you ask the same model to critique its own regression design, but the review is information-starved — nothing truly new entered the loop. **Gains come from external signals:** tools, tests, verifiers, humans.

Huang et al. (2024), ICLR. Valmeekam et al. (2023), NeurIPS. Du et al. (2023), arXiv:2305.14325.

Your RA Team Inherits Your Prior

T4 sold sub-agent review for wall-clock time. It did not promise epistemic independence — here is why it cannot.

- `code-reviewer.md`, `domain-reviewer.md`, and `verifier.md` all read the same project context
- They are parallel *executionally*, not parallel *epistemically*
- Shared `CLAUDE.md` ⇒ shared blind spots: they catch implementation errors, not framing errors, because the framing lives in the context they trust
- At field scale this becomes *monoculture*: shared models, shared templates, correlated errors — and replication stops being informative (Kleinberg & Raghavan 2021, PNAS; Bommasani et al. 2022)

Parallel reviewers save time. They do not produce an outside view.

Remedy: Outside Critics, Plus Operational Discipline

Advisors, coauthors, and seminar audiences are valuable because their priors are not in your context window.

Why outside critics work

- They challenge the framing, the identification, the sample restriction — things you did not write into `CLAUDE.md` because you did not know to
- Korinek: LLMs automate execution; humans remain the source of question selection and critique

Five habits

1. Sub-agent review for implementation, not framing
2. Show preliminary results to a human early
3. Record what you *ruled out*, not just what you decided
4. Benchmark outside the current thread
5. Ask reviewers to attack assumptions, not polish execution

Human criticism is the scarce input — spend agent cycles making the work ready for it, not replacing it.

Korinek (2023), NBER WP 30957; Journal of Economic Literature.

The Paradigm Shift, Completed

What AI provides and what stays human

The Paradigm Shift, Completed: What Changed, What Did Not

The HA-Ramsey record, read one last time — now as a statement about your role.

What changed

- Theory–code–debug loop compressed from months to days
- Documentation became executable: the TeX note, pseudocode, and code stay synchronized
- Cross-literature imports got cheap (computational geometry, viability theory)
- The unit of progress became the *validated milestone*

What did NOT change:

- AI never decided what was economically interesting
- AI never caught the wrong timing on its own
- AI never knew when a result was not credible

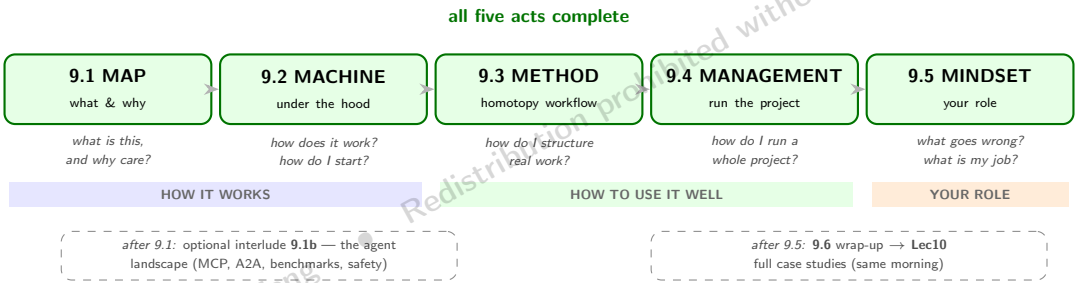
"Weeks, not years — and the binding constraint was economist judgment at the milestones, not implementation labor."

Read the two columns together: the machine moved every bound except the one that defines the researcher.

Source: HA-Ramsey case-study package (Lec10 T8).

Your Job Description: PI and Project Manager

The three questions, answered: *how it works* — a tool-calling loop steered by files (Acts 1–2); *how to use it well* — the homotopy method inside a managed project (Acts 3–4); *your role* — this frame.



Set the goal. Define the final product. Gate every intermediate good. Assign the work — sessions, agents, sub-agents. Cross-verify everything. And remain the client of the outside view: the one thing your agent cannot give you is a prior other than your own.

Bridge: T6 Wraps, Lec10 Shows the Real Thing

Act 5 named the failure modes and your job. Two things remain this morning.

- **T6 wraps the lecture:** the six evaluation criteria, when *not* to use an agent, and the dated install appendix that T2 promised
- **Lec10 delivers the full cases:** the four patterns from T1, worked end-to-end
- **Both threads hand off:** you saw the fellows *workflow* — Lec10 T4 shows the findings, and Exercise B gives you your own batch of 50; you saw HA-Ramsey's *contract and milestones* — Lec10 T8 shows the discoveries and the economics

Arc: T1 Framing → T2 Under the Hood → T3 Homotopy → T4 Project Org → **T5 Mistakes/Context** *→ T6 Wrap.*