

From chatbot to coworker

The Claude landscape — and one physics problem, solved
four ways

Prof. Murillo · NRT Agentic AI



OUTLINE

Today, in six parts

- | | | |
|---|--------------------------------------|---|
| 1 | The landscape | What agents are, and how fast the tools change |
| 2 | Four interfaces | Chat, Cowork, desktop Code and terminal: what each can reach and remember |
| 3 | The example, as an experiment | One physics problem, and how to compare tools on it fairly |
| 4 | Hands-on | You try each interface on a piece of the problem |
| 5 | Results | Four setups, scored against the answer key |
| 6 | Your turn | Your group project |

The order is deliberate: ideas first, then the example and how to study it, then your hands, then the evidence.

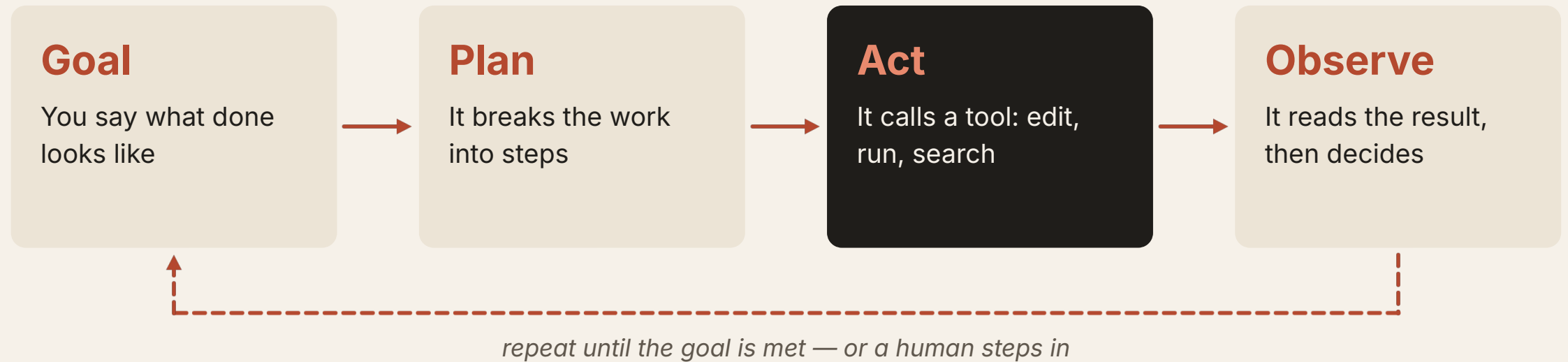
SETTING THE STAGE

The agentic ecosystem, October 2026

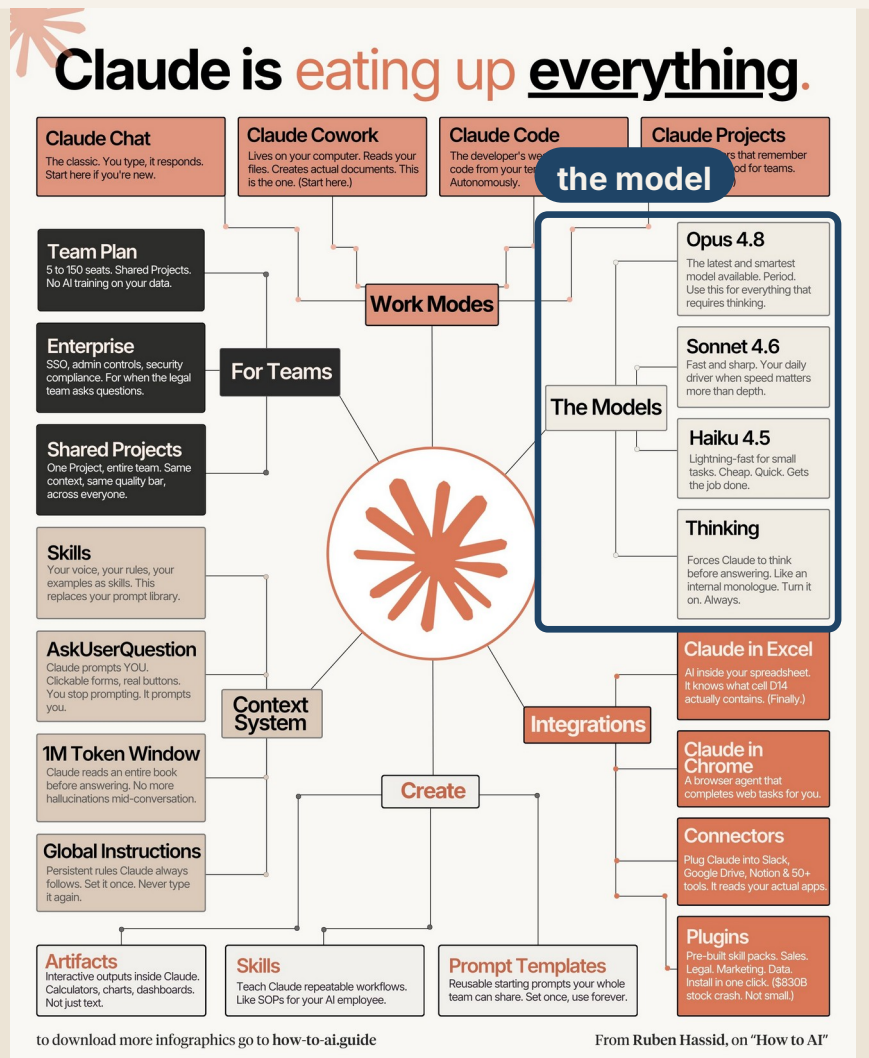
ANTHROPIC	OPENAI	GOOGLE	MICROSOFT & OTHERS	Apps
Claude: chat + Cowork (merging), Code tab	ChatGPT desktop app: Chat, Work, Codex	Gemini app · Antigravity 2.0 desktop	Microsoft Copilot · Cursor · GitHub Copilot	Terminal agents
Claude Code	Codex CLI	Antigravity CLI (agy) — replaced Gemini CLI in June	Open source: OpenHands, Aider, ...	Build-your-own SDKs
Claude Agent SDK	OpenAI Agents SDK	Google ADK · Antigravity SDK	Microsoft Agent Framework · AWS Strands	
Open frameworks — any model	LangGraph · CrewAI · LlamaIndex · Pydantic AI · smolagents · ...			
Protocols — how pieces connect	MCP (agents ↔ tools) · A2A (agent ↔ agent)			
Dark cells = today	One corner of one column. The frameworks and protocols come later in the course.			

LAST WEEK WITH PROF. HOU, IN ONE PICTURE

The agent loop



Last week you saw this loop run in three places: **the browser, the CLI, and Cowork**. Today: what changes from one place to the next — and why it matters.



A MAP SOMEONE ELSE DREW — AND A DEFINITION

agent = model + harness

On this map, only one branch is the **model** (boxed). Every other branch — work modes, context, integrations, create — is **harness**.

HARNESS

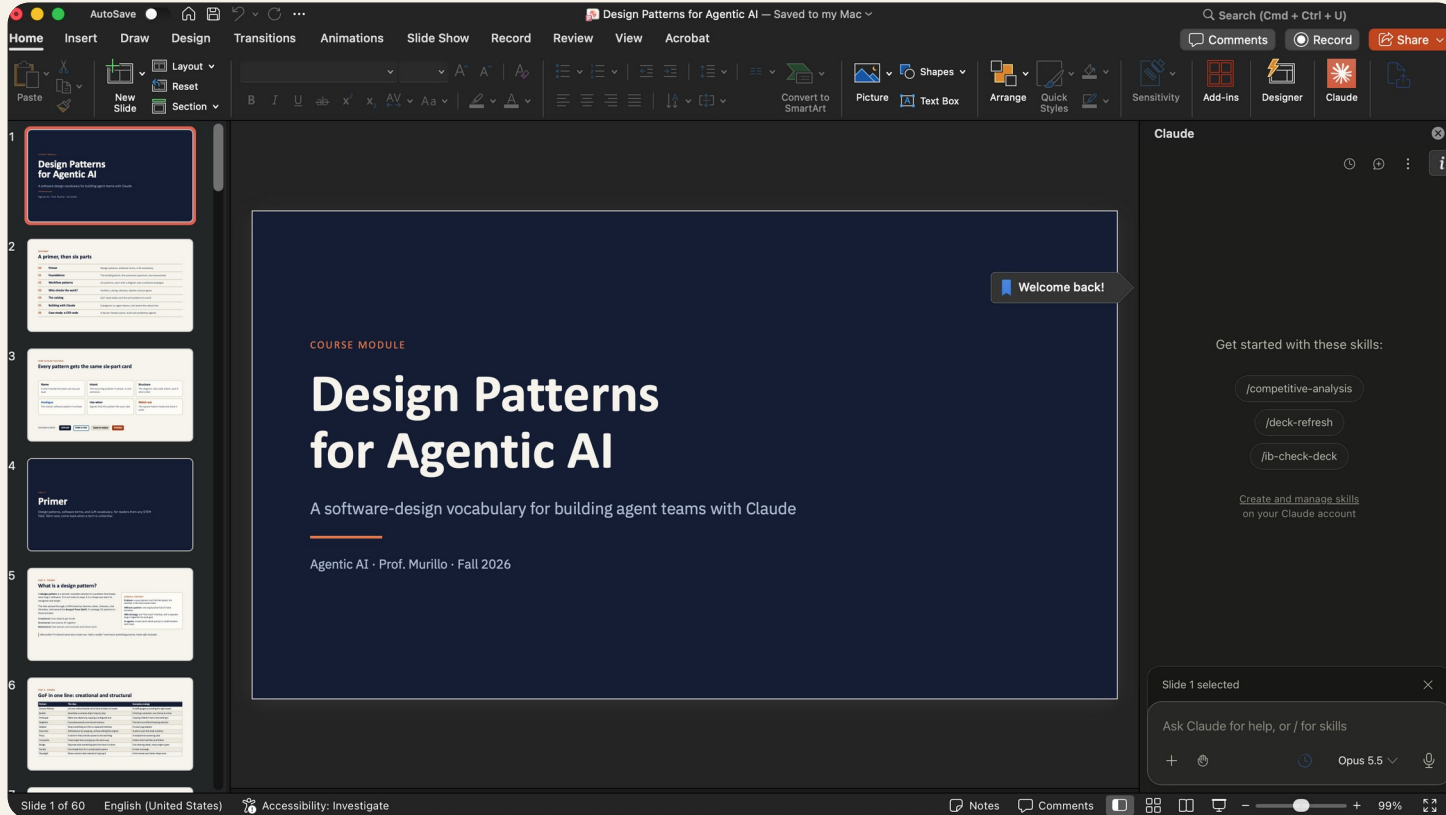
Everything around the model that makes it an agent: the loop, the tools it can call, what it can reach, what it remembers, and how you approve its actions.

NOT TO BE CONFUSED WITH

An *evaluation* harness: the code that runs a benchmark against a model. Same word, different job.

THE CLAUDE FAMILY

Claude where you already work



PowerPoint



A side panel in your deck — with skills like `/deck-refresh`

Chrome



A browsing agent that clicks, reads and fills in forms

Excel



Works inside your spreadsheet, cell by cell

Design



A canvas for mockups, posters and visuals

Slack



Tag `@Claude` in a thread to hand off a task

CHECKPOINT 1 • 3 MINUTES

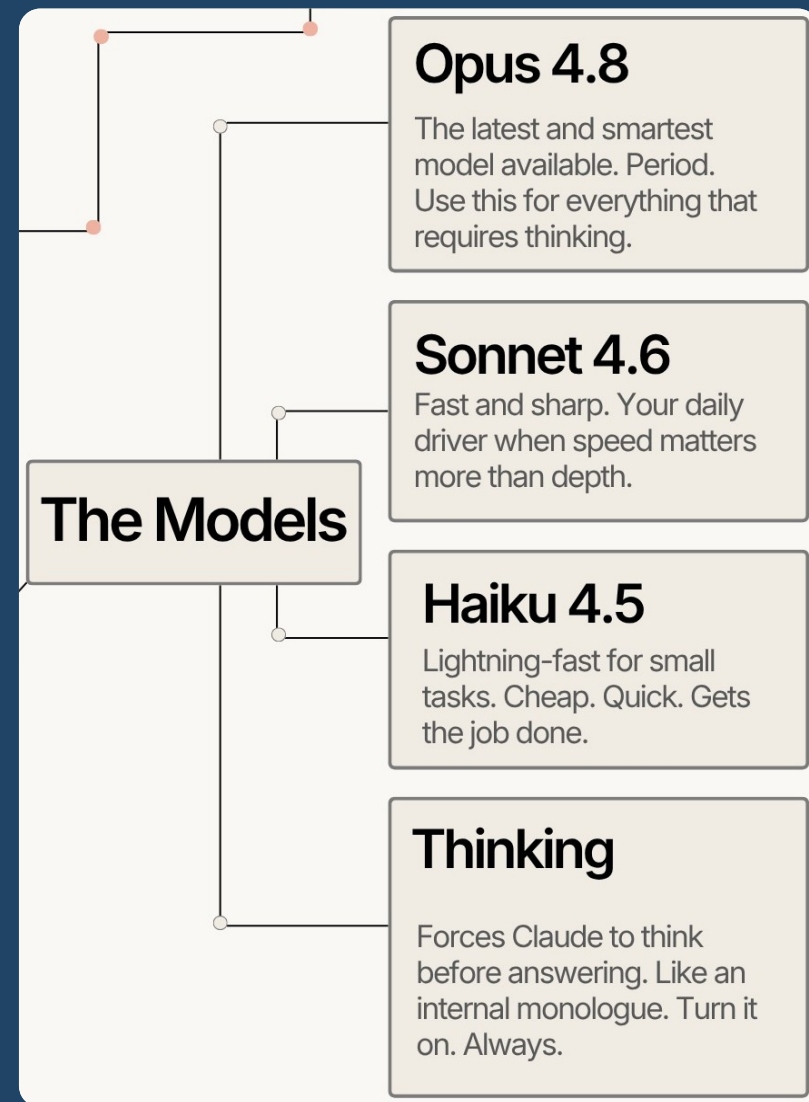
Fact-check the models panel

Find three claims that are out of date — or were never quite right.
Then: how would you get an agent to check them for you?

1

2

3



CHECKPOINT 1 · ANSWERS

What the models panel gets wrong

"Opus 4.8", "Sonnet 4.6"

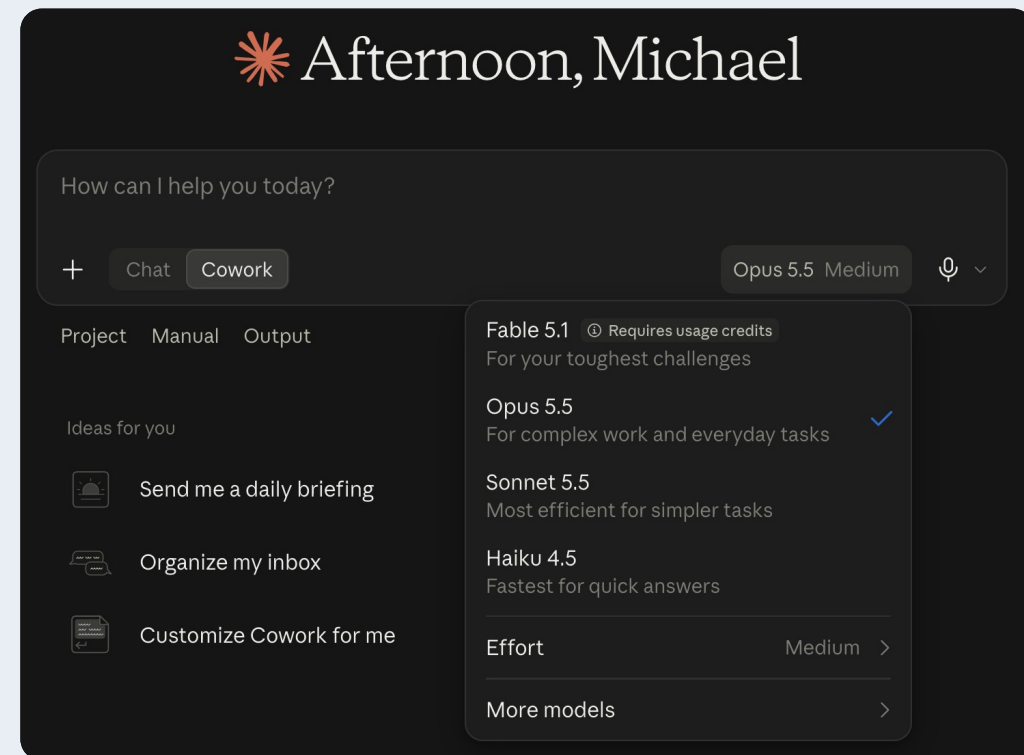
Version numbers go stale fast. Today's picker lists Opus 5.5, Sonnet 5.5 and Haiku 4.5.

"The latest and smartest model. Period."

There is now a tier above Opus: Fable 5.1, "for your toughest challenges" — and it needs usage credits.

"Thinking: turn it on. Always."

It is a dial now — the Effort setting. Worth turning up for a derivation; wasted on a lookup.



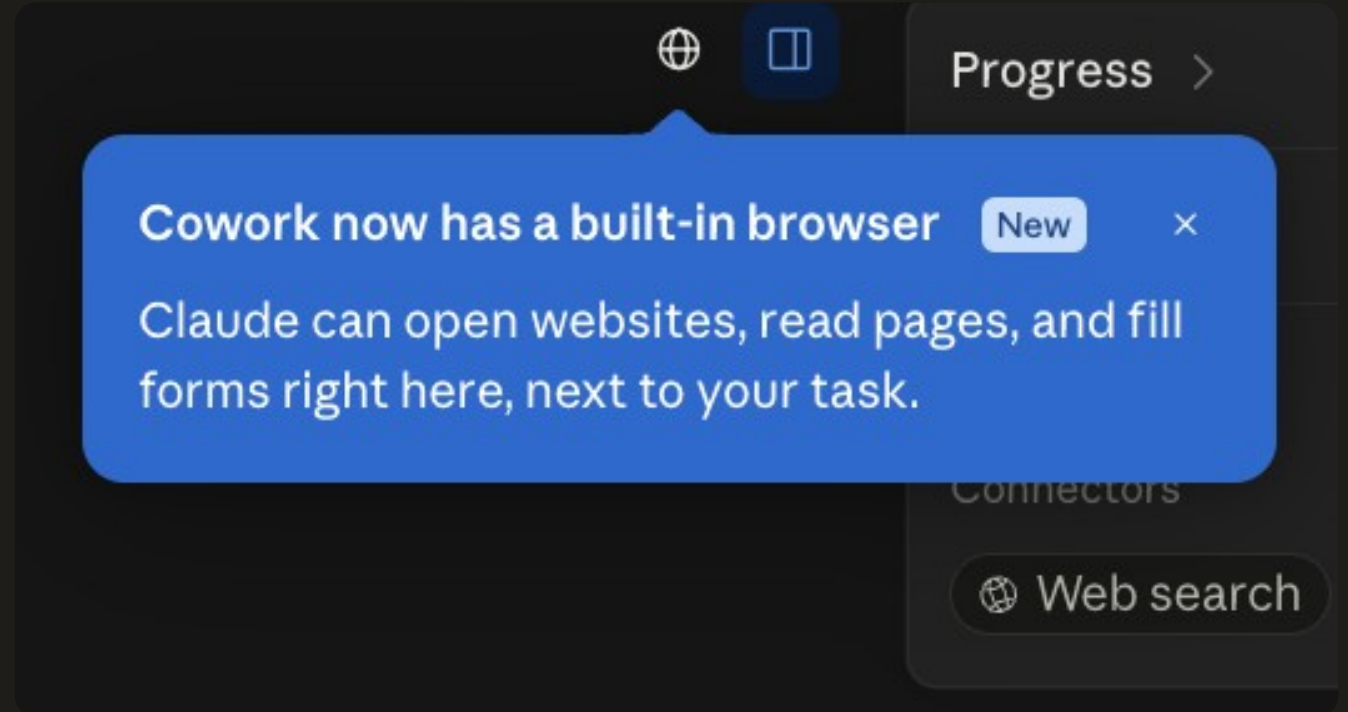
Version numbers are claims to verify. Ask the agent to check the official docs and cite what it finds.

MEANWHILE, WHILE I MADE THESE
SLIDES...

This just changed.

Cowork gained a built-in browser. One slide in this very deck was already out of date — a few hours after I wrote it.

Sun 4 Oct 2026 · 6:13 pm



AND THIS IS CHANGING RIGHT NOW

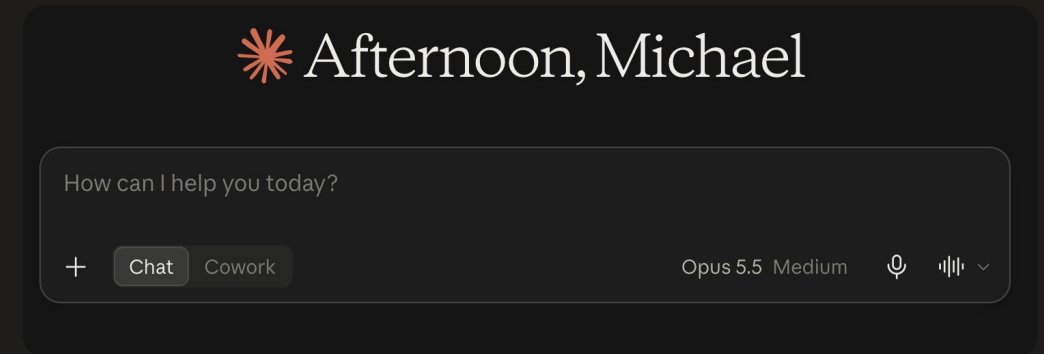
Chat and Cowork are becoming one box.

Announced 16 September 2026. Rolling out to Pro and Max first — on web, desktop and mobile — with other plans to follow. Once an account switches, it can't switch back.

FOR TODAY

No toggle in your app? You have the new version. Just ask — Claude decides whether to answer or start a task.

WHAT I STILL SEE (4 OCT)



WHAT MANY OF YOU MAY SEE

One message box. No Chat / Cowork buttons.

Work on local folders, the built-in browser and computer use still go through the desktop app.

CHECKPOINT 2 · 3 MINUTES

True, half-true, or false?

Pick one card on the right and defend your verdict in a sentence.
Three to argue about:

- 1 Does a bigger context window stop hallucinations?

- 2 What is a skill, mechanically?

- 3 Where do "global instructions" live in Claude Code?

Skills

Your voice, your rules, your examples as skills. This replaces your prompt library.

AskUserQuestion

Claude prompts YOU.
Clickable forms, real buttons.
You stop prompting. It prompts you.

1M Token Window

Claude reads an entire book before answering. No more hallucinations mid-conversation.

Global Instructions

Persistent rules Claude always follows. Set it once. Never type it again.

Context System

```
graph LR; Skills --- CS[Context System]; AskUserQuestion --- CS; 1MTokenWindow[1M Token Window] --- CS; GlobalInstructions --- CS;
```

Context is engineered, not given

Does a bigger context window stop hallucinations?

False

A window is how much it can see, not whether it is right. Reliability comes from checks: tests, symbolic verification, confirmed citations.

What is a skill, mechanically?

A folder of instructions

A SKILL.md plus optional scripts and examples. Only a short description sits in context; the rest loads when a task matches.

Where do “global instructions” live in Claude Code?

In CLAUDE.md files

~/.claude/CLAUDE.md applies everywhere; a CLAUDE.md in the repo applies to that project. Both are read at the start of every session.



Written-down context — files, rules, skills — is what an agent can re-read. Our physics example tests exactly how much that matters.

PART 2 OF 6

Four interfaces

Chat, Cowork, desktop Code and terminal: what each can reach and remember

1 Landscape

2 Interfaces

3 The example

4 Hands-on

5 Results

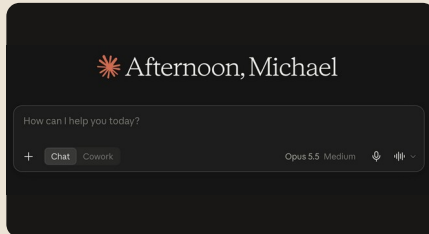
6 Your turn

INTERFACE TOUR

From chatbot to coworker, step by step

1 Chat

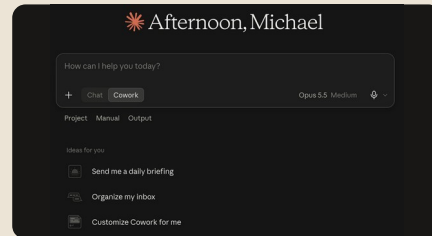
🏠 Browser · desktop app
· phone



You ask, it answers. Code runs only in its own sandbox.

2 Cowork

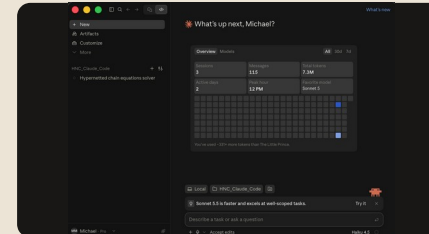
🏠 Browser · desktop app
· phone
(your folders: desktop app)



Same app, new mode: it works through your files, step by step.

3 Desktop Code

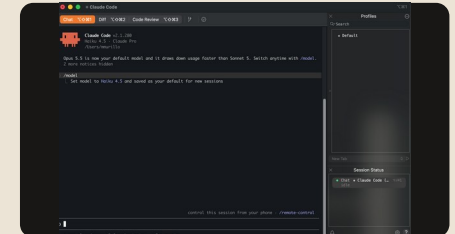
🏠 Claude desktop app →
Code tab



Claude Code in a window: your repo, your shell, edits you accept.

4 Terminal

🏠 iTerm2, Terminal, or
any shell



The same agent in your terminal: scriptable, steerable from your phone.

Answers questions

Does the work

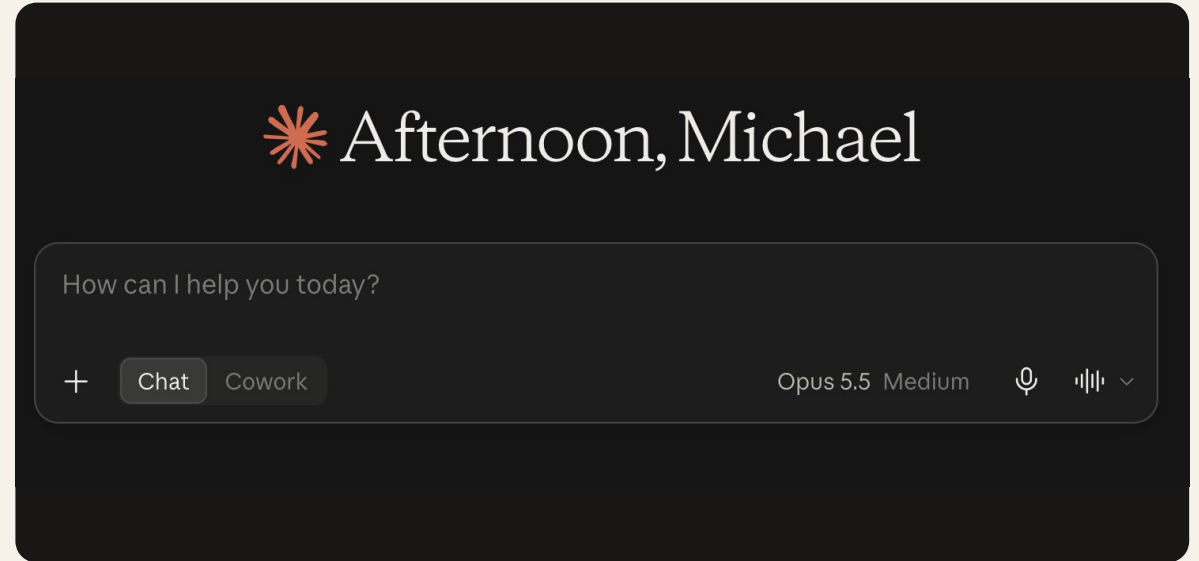
INTERFACE TOUR · 1 OF 4

Claude chat

claude.ai in a browser, the mobile apps, or the Chat toggle in the desktop app.

REACH FOR IT WHEN

You're thinking out loud, drafting, or doing a one-off calculation — on any device



IT CAN

- ✓ Explain, draft, and reason through a derivation step by step
- ✓ Run Python — SymPy, NumPy, plots — in a sandbox, once code execution is on
- ✓ Read files you upload, and use connectors you switch on (Drive, Gmail...)

IT CAN'T

- ⚠ See files on your computer unless you upload them
- ⚠ Keep its sandbox: files vanish unless it hands you a download
- ⚠ Run your code, your packages, or your tests

Cowork

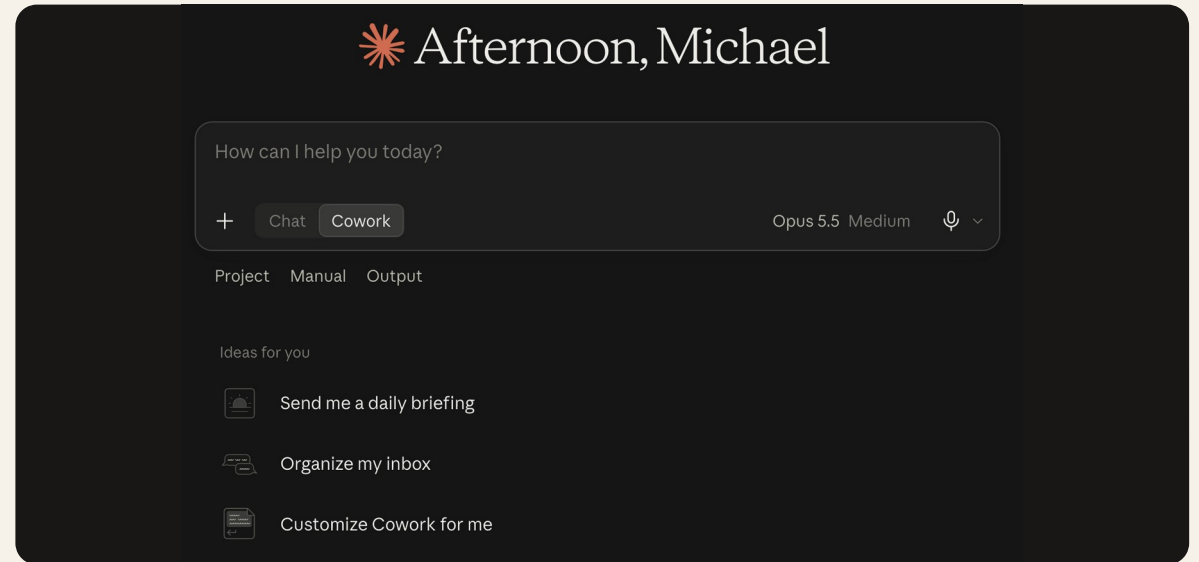
Browser, desktop app or phone — but working on your folders needs the desktop app. (Merged app: no toggle, just ask.)

REACH FOR IT WHEN

The job is many files and many steps — documents, not a codebase — and you'd rather not use a terminal

IT CAN

- ✓ Read, create and reorganize files in folders you give it
- ✓ Work through long, multi-step tasks — or wait for your OK at each step (Manual)
- ✓ Open websites, read pages and fill in forms in its built-in browser — new this week



IT CAN'T

- ⚠ See folders you haven't given it
- ⚠ Stand in for a developer setup: it isn't wired into your repo, shell and tests
- ⚠ Skip your approval on actions you haven't allowed

INTERFACE TOUR · 3 OF 4

Claude Code in the desktop app

The Code tab of the desktop app, pointed at a folder on your machine ("Local").

REACH FOR IT WHEN

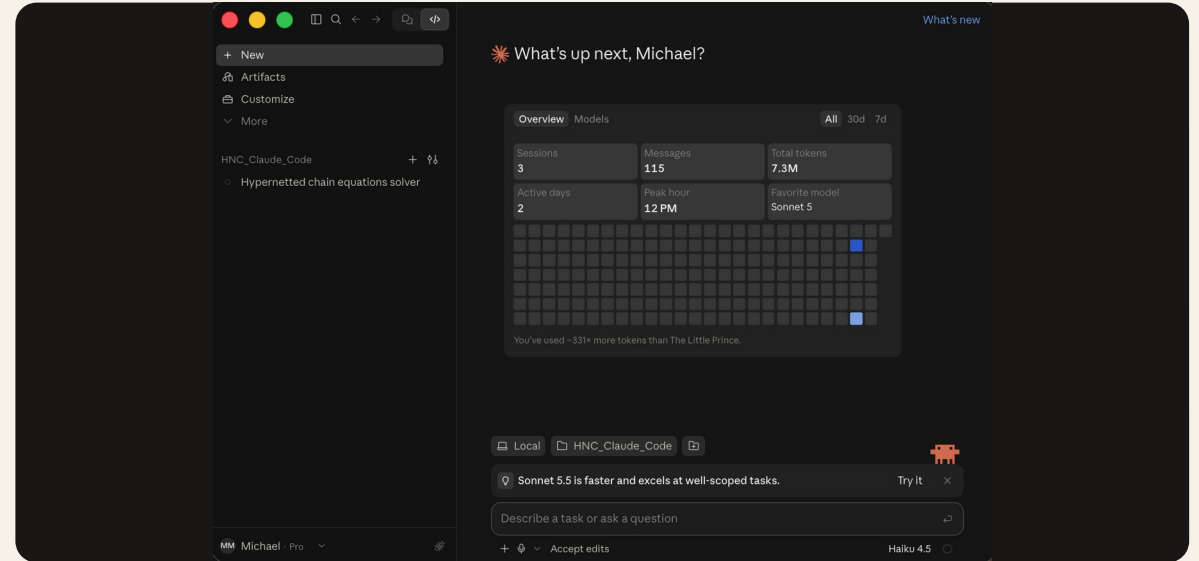
You want Claude Code's full power in a familiar app window

IT CAN

- ✓ Edit files across your project, with edits you accept or reject
- ✓ Run commands with your Python and packages: scripts, tests, git
- ✓ Read CLAUDE.md, and use skills and subagents

IT CAN'T

- ⚠ Work outside the folder you chose without asking
- ⚠ Run a command you haven't approved (in the default modes)
- ⚠ Help much with no project: an empty folder means it starts from zero



INTERFACE TOUR · 4 OF 4

Claude Code in the terminal

Your terminal: open any folder and type `claude`. Same agent as the Code tab.

REACH FOR IT WHEN

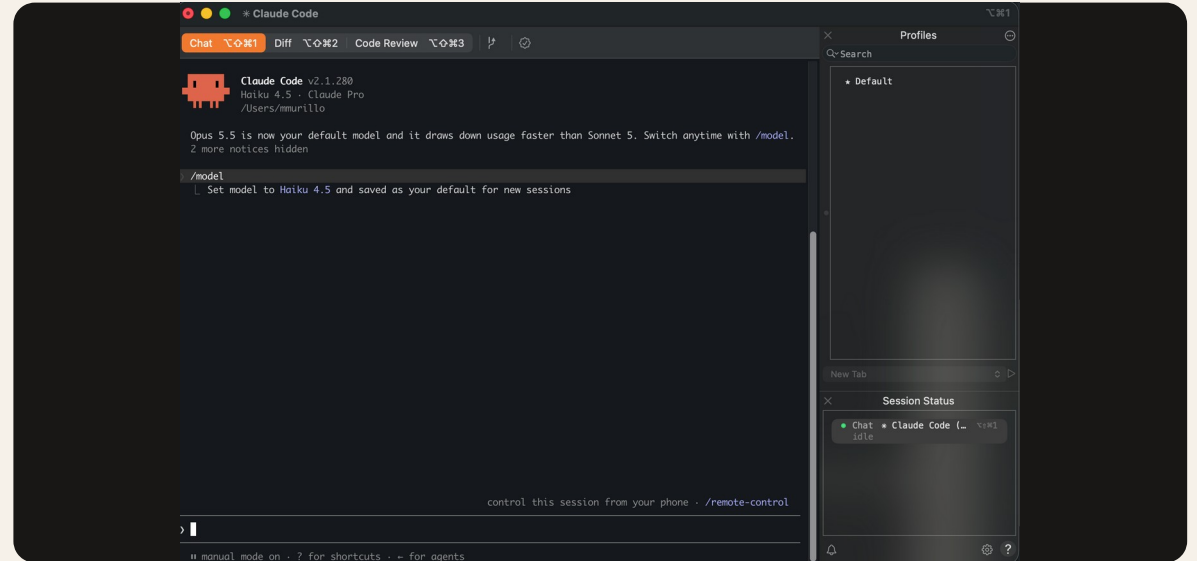
Long autonomous runs, automation, or you live in the terminal anyway

IT CAN

- ✓ Do everything the Code tab does — it is the same agent
- ✓ Run from scripts with `claude -p "..."`, e.g. a nightly test summary
- ✓ Switch models with `/model`; steer a running session from your phone with `/remote-control`

IT CAN'T

- ⚠ Hold your hand: it's text-first, so you need to be at home in a shell
- ⚠ Show buttons and visual diffs (some terminals and IDEs add them)
- ⚠ Work outside the folder you started in without asking



Four interfaces, compared

	1 · Chat	2 · Cowork	3 · Desktop Code	4 · Terminal
Where it runs	Web, phone, app	Desktop app	Desktop app	Terminal
What it can touch	Uploads, a sandbox	Folders you grant	Repo + shell	Repo + shell
Where code runs	Sandbox, if enabled	Its own sandbox	Your Python	Your Python
How you approve	Little to approve	Modes, e.g. Manual	e.g. Accept edits	e.g. manual mode
What persists	Projects, memory, skills	Projects, skills	CLAUDE.md, skills, subagents	CLAUDE.md, skills, subagents
Best for	Thinking, drafting	Multi-step document work	Coding in a window	Long runs, scripting
In my experiment (Part 3)	Setup 1	—	Setups 2–4	Setups 2–4

Columns 3 and 4 are **the same agent** — pick the window you prefer.

CHECKPOINT 3 · 2 MINUTES

Which room would you use?

More than one answer can be defensible. Be ready to argue yours.

Rename and sort 300 PDFs in your Downloads folder

→ _____

Refactor a plasma solver until its tests pass

→ _____

Sanity-check a derivation on your phone, on the bus

→ _____

Fix the formulas in a grant budget spreadsheet

→ _____

Pick the room by what the agent must touch

Rename and sort 300 PDFs in Downloads

→ **Cowork**

Local files, many steps, no code required. Claude Code works too if you live in a terminal.

Refactor a plasma solver until its tests pass

→ **Claude Code**

It needs the repo, a shell, the test suite and git — so it can run the tests and iterate until they pass.

Sanity-check a derivation on the bus

→ **Chat, on your phone**

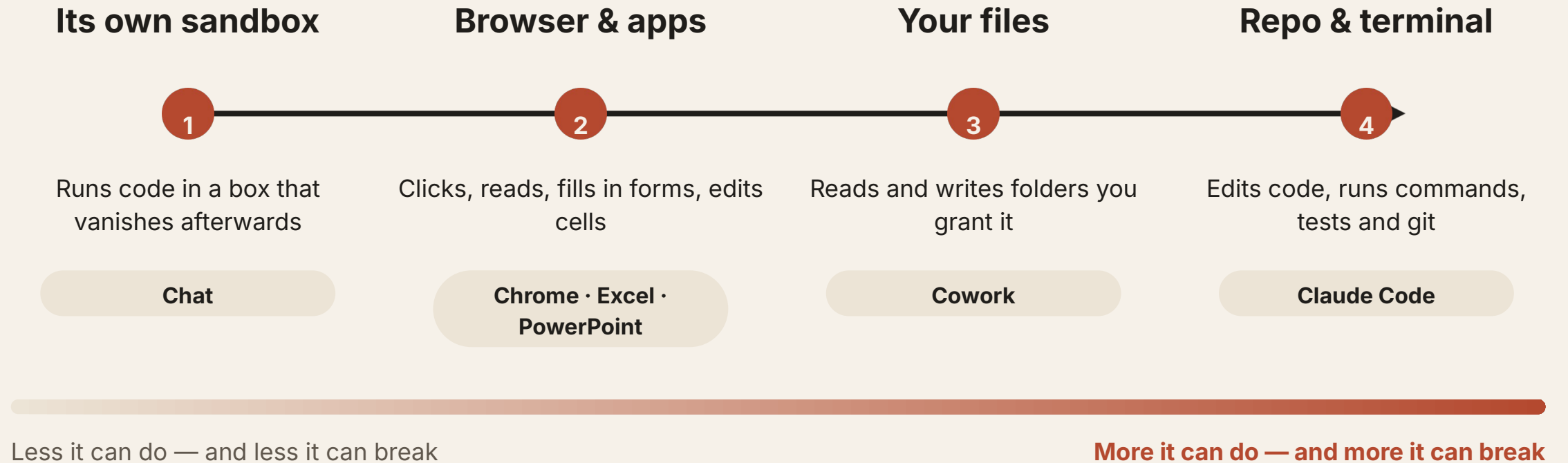
Pure reasoning: nothing to touch. (Claude Code sessions can also be steered remotely from the phone.)

Fix the formulas in a grant budget

→ **Claude in Excel**

It works on the actual cells and formulas in place, not on a pasted copy that you must then paste back.

Reach: what can the agent touch?



Memory: what persists between sessions?

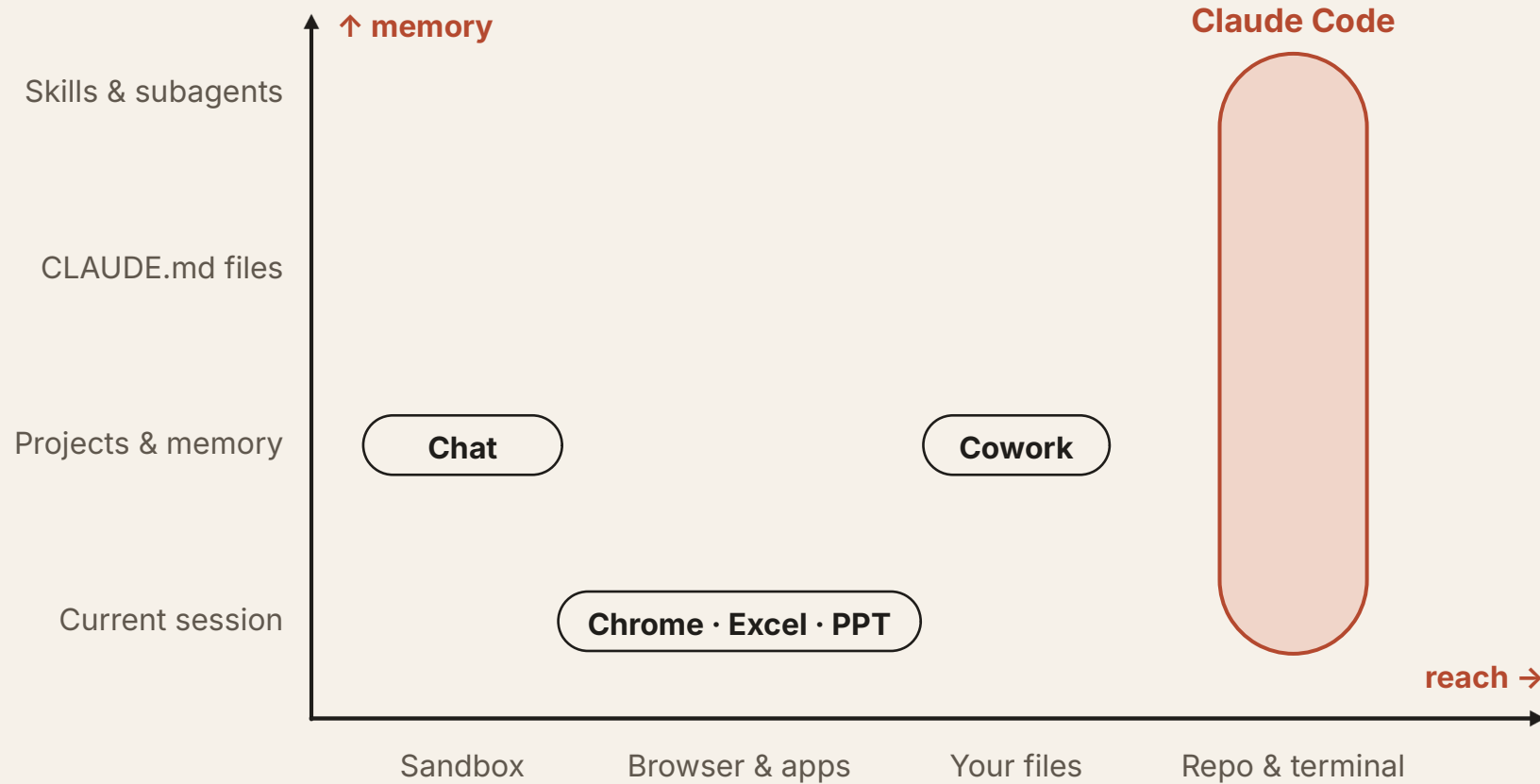
↑ lasts longer, reused more

Skills & subagents	Reusable workflows loaded on demand; helpers that work in their own context	Code · Cowork · Chat
CLAUDE.md files	Rules and notes the agent re-reads at the start of every session	Claude Code
Projects & memory	Uploaded files, project instructions, remembered facts about you	Chat · Cowork
Current session only	What you've said in this chat or session — gone when you start a new one	Everywhere

Memory here is **text the agent re-reads**. If it isn't written down somewhere, the next session doesn't know it.

TWO DIALS · TOGETHER

Every interface sits somewhere on this map



DISCUSS

Where do you spend most of your time?

For a research problem, which dial matters more?

PART 3 OF 6 · FROM OVERVIEW TO EXAMPLE

The example, as an experiment

Now we use these tools on a real problem. First the problem — then how to compare tools on it fairly.

1 Landscape

2 Interfaces

3 The example

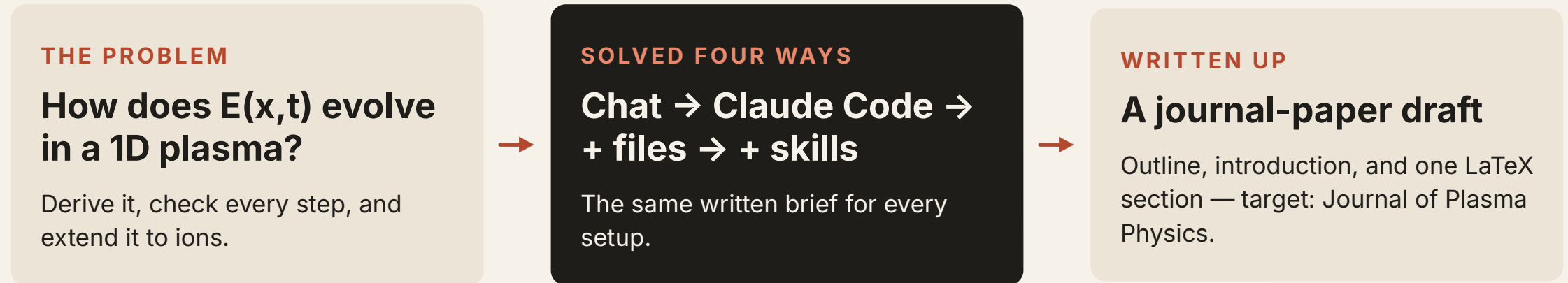
4 Hands-on

5 Results

6 Your turn

THE EXAMPLE FOR THE REST OF TODAY

My goal: from a physics problem to a paper



To compare the four fairly, we run it like an experiment. That design comes right after the physics.

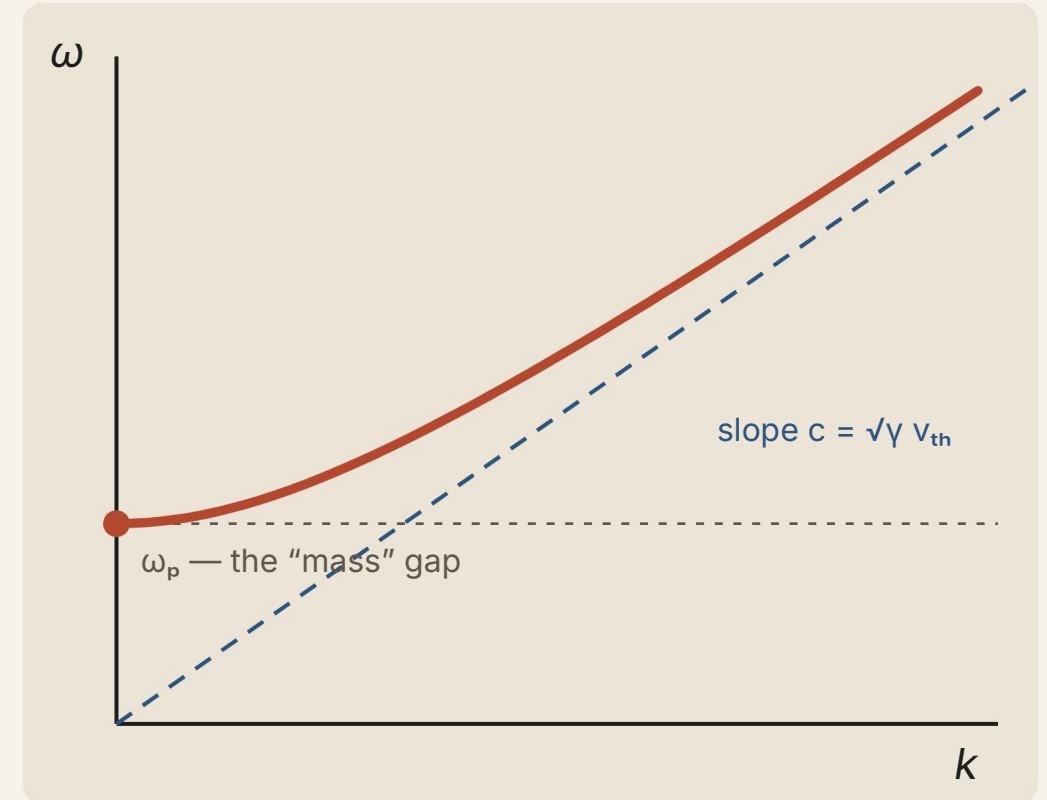
THE EXAMPLE

How does $E(x,t)$ evolve in a 1D plasma?

Electrons: 1D Vlasov–BGK, reduced to a fluid. Ions: a fixed background. Then: mobile ions with screening electrons.

$$\partial_t^2 E + \nu \partial_t E - \gamma v_{th}^2 \partial_x^2 E + \omega_p^2 E = 0$$

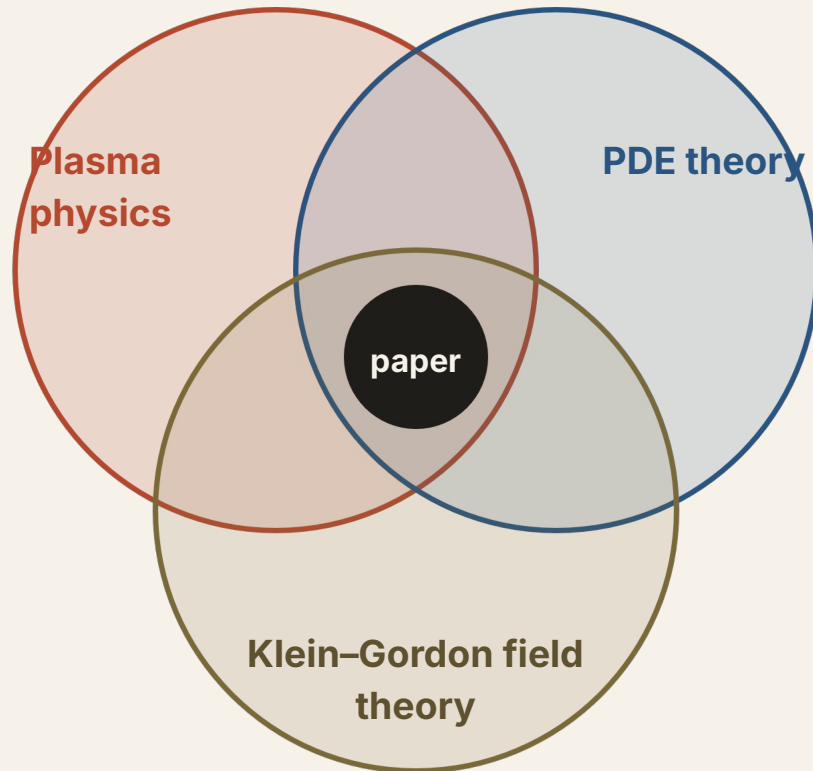
Our working guess: a damped Klein–Gordon equation. Not yet shown.



Bohm–Gross: $\omega^2 = \omega_p^2 + \gamma k^2 v_{th}^2$ — a massive relativistic particle in disguise

THE EXAMPLE

Three communities that rarely cite each other



BELIEVED

Each piece exists somewhere in one of these literatures — but no single work pulls them together.

BELIEVED

The damped, electron-screened ion case has not been treated.

SHOWN

Nothing yet. That's Question 7: search, with evidence, before claiming novelty.

HOW WE'LL STUDY IT

Run it like an experiment, not a demo



Question

Does a richer harness produce better research — with the model held fixed?



We change

The harness: chat → Claude Code bare → + project files → + skills and subagents



We hold fixed

The brief (seed.md v2), the model and effort, web access, memory, and every setting we can see



We measure

A rubric: results actually checked, §3 errors caught, real citations, novelty evidence, rerunnable figures, draft quality



We keep blind

My original answers sit in answer_key.md. No setup sees it; we score against it afterwards.



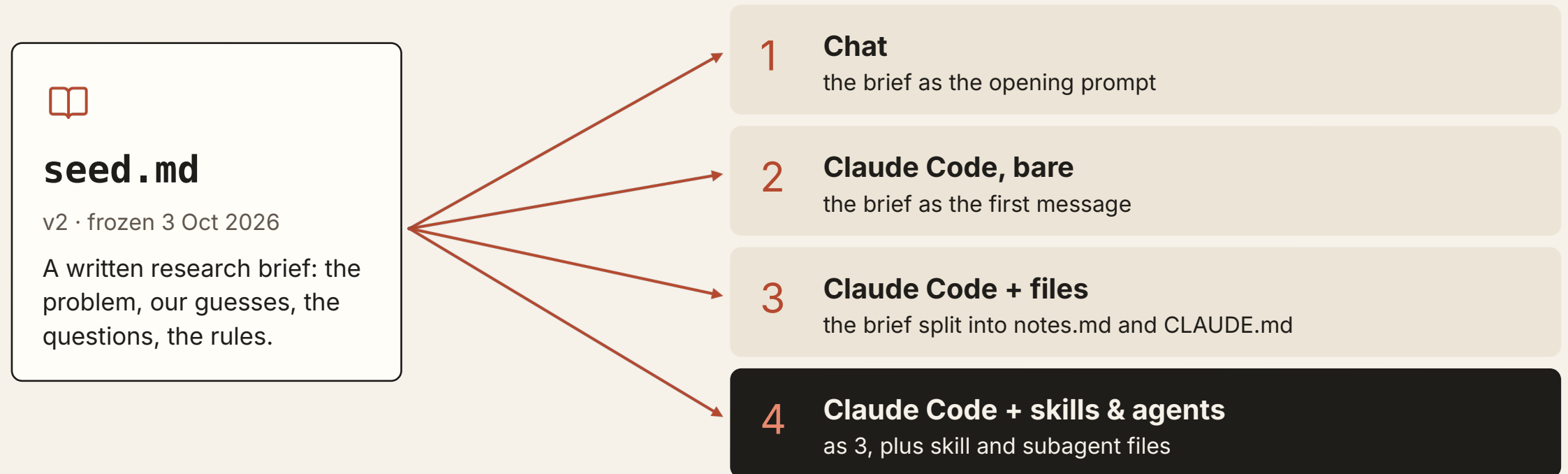
We record

A run log per setup: versions, settings, timings, and every follow-up I type

THE EXAMPLE

A fixed test for the progression

To compare setups fairly, each one gets **the identical brief**. That brief is seed.md — so any difference in results comes from the setup, not from how we asked.



THE EXPERIMENT

One seed, four setups

Same Sections 1–7. Same model. Only the packaging changes.

1

Chat

Seed pasted as the opening prompt

Context: one conversation

2

Claude Code, bare

Seed as the first message

+ a shell, files, Python

3

+ Project files

§1–4, 7 → notes.md

§5–6 → CLAUDE.md

+ rules that persist across sessions

4

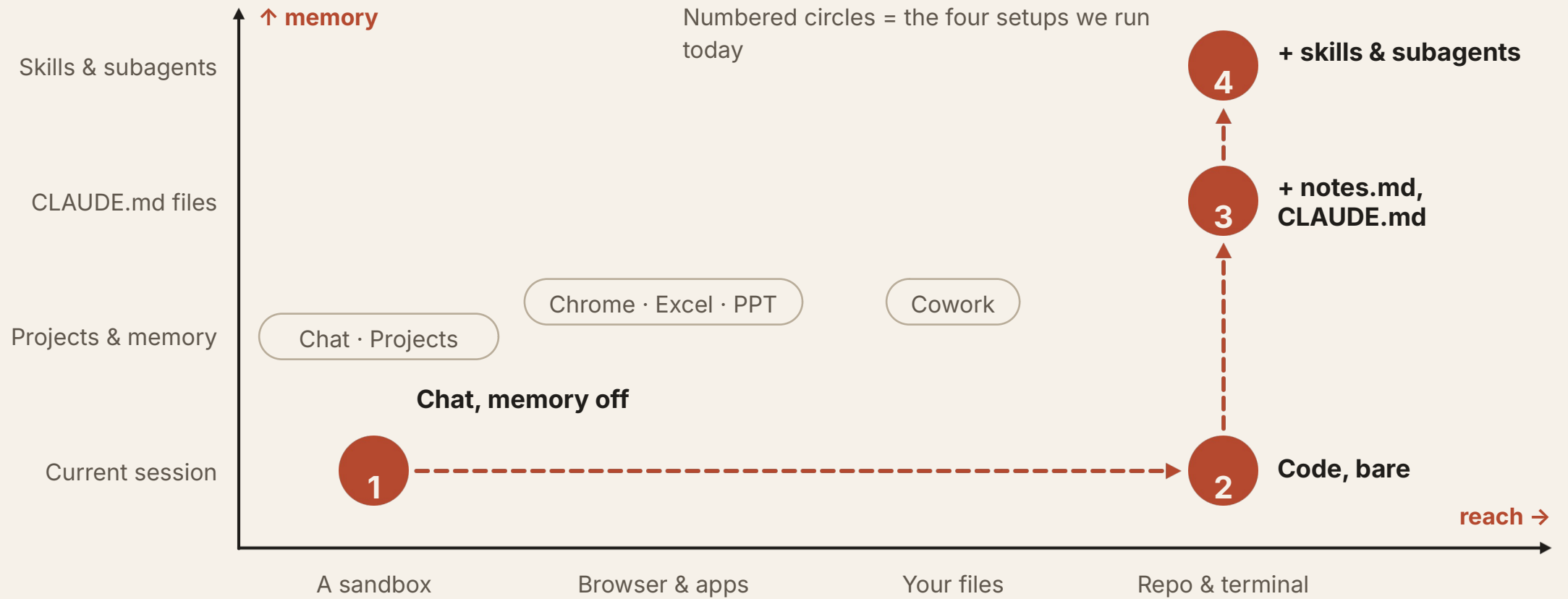
+ Skills & subagents

As 3, plus skill and agent files

+ reusable workflows, delegation

THE EXPERIMENT

The four setups, on the map



THE BRIEF

The top of seed.md, read closely

```
# seed.md – Electric-field dynamics in 1D plasmas

<!-- EXPERIMENTER HEADER (not given to any approach)
- Version: v2, 2026-10-03. Freeze before the first... ①
- Detailed results ... are WITHHELD → answer_key.md ②
- Every approach receives Sections 1–7, unchanged. ③
  Approach 1 (chat): pasted as the opening prompt
  Approach 2 (Code bare): pasted as first message
  Approach 3 (+ files): §1–4,7 → notes.md;
                        §5–6 → CLAUDE.md
  Approach 4 (+ skills/agents): as 3, plus files
- Before each approach: memory paused, chat ④
  search off, ~/.claude/CLAUDE.md empty, same model.
-->

## 1. Goal ⑤
Write a journal paper (target: JPP) on how  $E(x,t)$ ...
```

① Frozen

No edits mid-study. Changes become v3, with a note.

② Blind

My earlier answers are withheld in answer_key.md.

③ Same content

Only the packaging changes between setups.

④ Controlled

Settings checked before every run.

⑤ The task

Sections 1–7: identical for every setup.

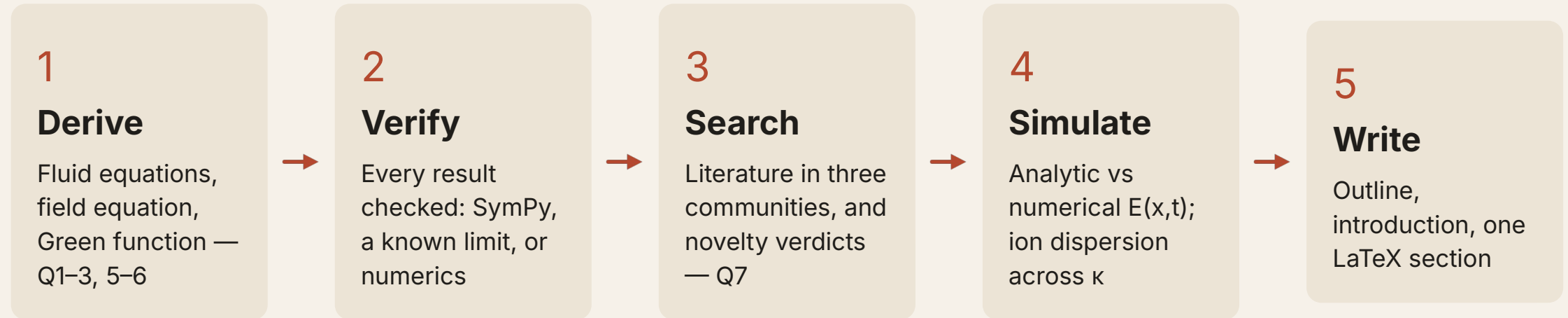
THE EXAMPLE

Inside seed.md

—	Experimenter header	Version, packaging per setup, controls. Never shown to any setup.
§1	Goal	A paper on $E(x,t)$ in a 1D plasma that unifies three literatures
§2	Model	Vlasov–BGK electrons, two collision models, closures, screened ions
§3	What we think might be true	Working guesses — Klein–Gordon form, Green function, ion dispersion. Any may be wrong.
§4	Questions	Seven questions, from derivation to novelty
§5	Working rules	Check every result; never invent a citation; keep shown apart from believed
§6	Deliverables	Derivations, literature map, novelty verdict, numerics, paper draft
§7	Out of scope	Experiments, strong coupling, drifting plasmas

THE TASK

A research program, not a single prompt



The working rules from seed.md §5 — these are what we grade:

Never invent a citation

Keep shown apart from believed

Every figure from rerunnable code

THE EXPERIMENT

Controls: hold everything else fixed



Seed frozen

v2, dated. Later edits go in v3.



Memory paused

No carry-over from past chats



Past-chat search off

"Search and reference chats" — it can't find my earlier work



Empty CLAUDE.md

The global one in ~/.claude — no hidden rules



Same model

For all four runs



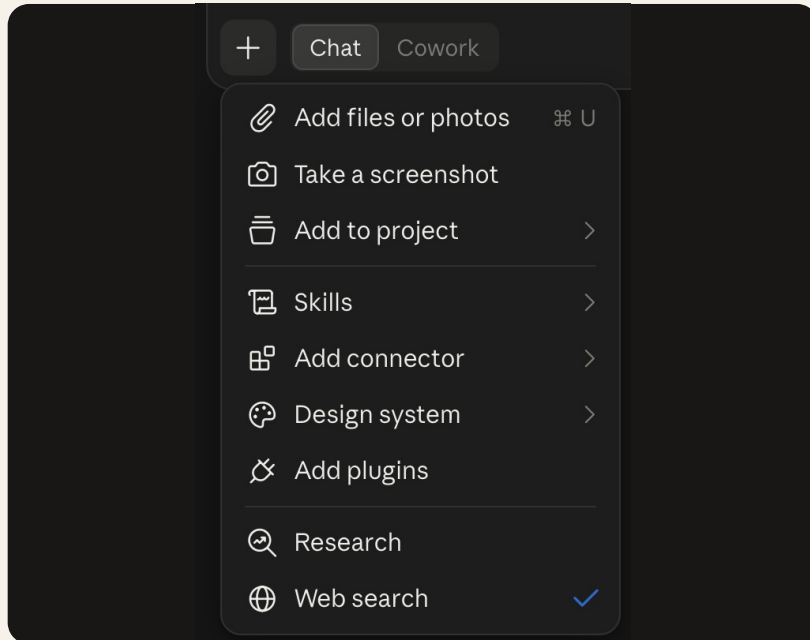
answer_key.md holds the original results, references and numerics. No setup sees it. We score every run against it afterwards.

CONTROLS

Same knobs, set the same way, in every setup

Knob	Why it matters	Chat · setup 1	Claude Code · setups 2–4
Model & effort	Results depend on it	Picker: same model & effort	/model → the same
Memory	Could leak my earlier work	Settings → Memory: both off	Empty ~/.claude/CLAUDE.md
Past chats	My original chat is findable	Past-chat search off	Fresh folder; no --resume
Web search	Literature questions need it	On	Allowed (WebSearch, WebFetch)
Code execution	\$5 checks need it	Code on · network off	Runs in your own Python
File access	What it reads is context	No uploads	Only that setup's folder
Instructions	Hidden rules change behavior	None in Settings or Projects	Only what the setup adds
Your follow-ups	Each one is an intervention	Log every one	Log every one

Setup 1, configured: what's on, what's off



Web search ON

The literature questions need it. Research, projects, skills and connectors stay off.

Memory

Search and reference chats

Allow Claude to search for relevant details in past chats. [Learn more](#)



Generate memory from chats

Allow Claude to generate memory from your chats. [Learn more](#)



Memory: both OFF

No carry-over from the chat where I first worked this out

Code execution and file creation

Cloud code execution and file creation

Claude can execute code on a server and create and edit docs, spreadsheets, presentations, PDFs, and data reports. Required for skills.



Allow network egress

Give Claude network access to install packages and libraries in order to perform advanced data analysis, custom visualizations, and specialized file processing. Monitor chats closely as this comes with [security risks](#).



Code ON · network OFF

It can run SymPy, but can't install anything new

What could still bias the comparison



Leakage

Anything the agent can see is context: memory, past chats, stray files in the folder — even my phrasing. Close every door.



Drift

The tools change under us: the Chat/Cowork merge, model updates. Record the version and date of every run.



Intervention

Every follow-up I type steers the result. Same kickoff message for all; log each nudge, word for word.



Chance

One run per setup is $n = 1$. The same setup can do better or worse next time. Rerun when a difference looks decisive.



When scoring the drafts: hide which setup wrote which, so the grader can't favor the one they expect to win.

WRITE IT DOWN

A run log for every setup

```
# Run log – setup 1 (chat)
date:      2026-10-__
interface:  claude.ai chat (Chat/Cowork toggle)
model:     Opus 5.5 · effort Medium
memory:    off · past-chat search off
web/code:  search on · code on · network off
brief:     seed.md v2, §1–7, pasted once
time:      __:__ → __:__
follow-ups: 1. “...” 2. “...”
outputs:   paper-setup1-chat.pdf, transcript
```

IN THE ZIP, UNDER EXAMPLE/

controls.md

Every knob: how to set it, in chat and in Claude Code

run_log_template.md

Fill in one per setup

rubric.md

How each run is scored — before we see results

setup3-files/

notes.md + CLAUDE.md: seed.md, split

setup4-skills/

... plus two skills and one subagent

CHECKPOINT 4 · SHOW OF HANDS

Predict before we run it

Which setup will...	1	2	3	4
run a symbolic check without being reminded?	☐	☐	☐	☐
catch what changes when electrons collide with each other, not with ions?	☐	☐	☐	☐
cite a paper it can't back up?	☐	☐	☐	☐
say plainly that a guess in Section 3 is wrong?	☐	☐	☐	☐

Which setup will run a symbolic check without being reminded?

THE ANSWER

Setup 4 most likely; chat least. Every setup is told to check (§5), but only setup 4 has a skill that makes checking a step it runs after every derivation.

WHY IT MATTERS

"Verified" with no visible code is a claim, not a check. Research needs checks you can rerun.

PREDICTED BEFORE RUNNING · RESULTS IN PART 5

- | | | |
|---|---|-------------|
| 1 | Chat
Can run SymPy, but often just asserts | Maybe |
| 2 | Claude Code, bare
Python at hand; the rule is one line in a long prompt | Maybe |
| 3 | + project files
The check rule stays in context | Likely |
| 4 | + skills & subagents
A skill turns checking into a procedure | Most likely |

Which setup will catch what changes when electrons collide with each other, not with ions?

THE ANSWER

The physics: self-collisions conserve momentum, so there is no $-vu$ drag term — with a polytropic closure, no damping at this order. Expect no clear winner: no harness supplies this insight.

WHY IT MATTERS

It separates reasoning from recall, and shows what a harness can't do. Procedure can be scaffolded; insight comes from the model.

PREDICTED BEFORE RUNNING · RESULTS IN PART 5

- | | | |
|---|---|-------|
| 1 | Chat
Same model, same physics | Maybe |
| 2 | Claude Code, bare
More tools, no more insight | Maybe |
| 3 | + project files
Rules don't supply physics | Maybe |
| 4 | + skills & subagents
A check might flag it — if it tests momentum | Maybe |

Which setup will cite a paper it can't back up?

THE ANSWER

Chat is most at risk; setup 4 least, with a citation skill and a literature scout that must open every DOI.

WHY IT MATTERS

Here "likely" is the bad outcome. One invented reference can sink a paper — and it's the failure readers are most likely to miss.

PREDICTED BEFORE RUNNING · RESULTS IN PART 5

- | | | |
|---|---|----------|
| 1 | Chat
Easy to cite from memory | Likely |
| 2 | Claude Code, bare
Can search; nothing forces it to | Maybe |
| 3 | + project files
The rule says never invent — but no procedure | Maybe |
| 4 | + skills & subagents
Skill and scout must open each DOI | Unlikely |

Which setup will say plainly that a guess in Section 3 is wrong?

THE ANSWER

Setups 3 and 4 most likely. The invitation to challenge the brief sits in CLAUDE.md, always in context, and setup 4 tests each result.

WHY IT MATTERS

The brief invites disagreement. A setup that quietly adopts every guess is agreeing, not checking.

PREDICTED BEFORE RUNNING · RESULTS IN PART 5

- | | | |
|---|--|-------------|
| 1 | Chat
The invitation is one line in a long prompt | Maybe |
| 2 | Claude Code, bare
Same prompt, more tools | Maybe |
| 3 | + project files
The rule stays in context | Likely |
| 4 | + skills & subagents
Each result gets tested | Most likely |

PART 4 OF 6

Hands-on

You try each interface on a piece of the problem

1 Landscape

2 Interfaces

3 The example

4 Hands-on

5 Results

6 Your turn

FOLLOW ALONG · KEEP THIS SLIDE IN MIND

What's in agentic-ai.zip

One folder, so every interface works on the **same files**.

In ~/agentic-ai/	What it is	When you use it
README.md	What's here, plus the before-class checklist	Before class
prompts.md	Prompts 1–6, copy-ready — no typing Greek letters	Every follow-along step
papers/	2–3 PDFs	Cowork step · Prompt 3
plasma/	Empty on purpose — Claude Code fills it	Desktop Code and terminal · Prompts 1, 4–6
example/seed.md	The written brief for my full experiment	The experiment — and your project template
example/controls.md · rubric.md · run_log_template.md	How the study is controlled, scored and recorded	Your group project
example/setup1-chat/ ... setup4-skills/	What each setup receives — and setup 1's paper draft	When we compare results

FOLLOW ALONG

Get ready before class



Download **agentic-ai.zip** from the course Drive and unzip it in your home folder.

~ is your home folder: /Users/you on a Mac, C:\Users\you on Windows.

Give each interface only the folder its step names: **papers/** for Cowork, **plasma/** for Claude Code.

CHECKLIST

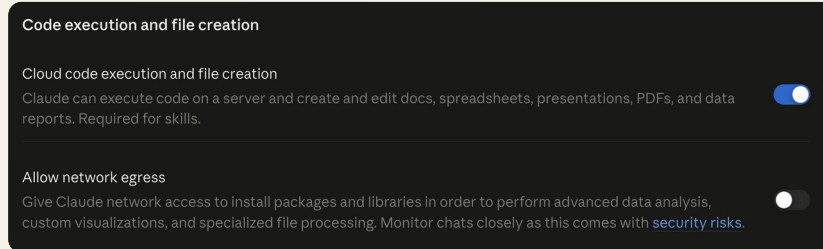
- ✓ Signed in to the course's Claude Team account
- ✓ Claude desktop app installed
- ✓ Cloud code execution on in Settings (for chat)
- ✓ Python 3 with SymPy
`python3 -m pip install sympy`
- ✓ Claude Code installed (for the terminal)

TRY IT NOW · 1 OF 4

Follow along: chat

1 Open claude.ai or the desktop app. Start a new chat.

2 In Settings, turn on **Cloud code execution and file creation**:



3 From `prompts.md`, paste Prompt 1. Then Prompt 2.

PROMPT 1 · THE SAME IN EVERY ROOM

Linearize the 1D electron fluid equations (continuity, momentum with pressure $\gamma T \cdot n$ and drag ν , Poisson) and derive one equation for $E(x,t)$. Verify it with SymPy.

PROMPT 2 · CHAT

Plot $\omega(k)$ for the $\nu = 0$ case and give me the figure as a PNG.

WATCH FOR

Did it really run SymPy? Look for a code block and its output. A sentence saying “verified” is a claim, not a check.

TRY IT NOW · 2 OF 4

Follow along: Cowork

- 1 Check that papers/ holds the PDFs from the course zip.
- 2 Desktop app → flip the toggle to Cowork (merged app: no toggle) → give it this folder:

```
~/agentic-ai/papers
```
- 3 Choose **Manual**, so it waits for your OK at each step.
- 4 Paste Prompt 3. Then Prompt 1.

PROMPT 1 · THE SAME IN EVERY ROOM

Linearize the 1D electron fluid equations (continuity, momentum with pressure $\gamma T \cdot n$ and drag v , Poisson) and derive one equation for $E(x,t)$. Verify it with SymPy.

PROMPT 3 · COWORK

Read every PDF in this folder. Write papers.md: a table of authors, year, DOI, and whether each is plasma physics, PDE theory, or field theory.

WATCH FOR

papers.md appears in your folder. Then ask it to open each DOI in its built-in browser — and check one yourself.

TRY IT NOW · 3 OF 4

Follow along: desktop Code

- 1 Desktop app → Code tab → Local → choose this folder:

```
~/agentic-ai/plasma
```

- 2 Paste Prompt 1.
- 3 If SymPy is missing, it asks to install it. Read the command before you approve.
- 4 Then Prompt 4. Accept or reject each edit.

PROMPT 1 · THE SAME IN EVERY ROOM

Linearize the 1D electron fluid equations (continuity, momentum with pressure $\gamma T \cdot n$ and drag ν , Poisson) and derive one equation for $E(x,t)$. Verify it with SymPy.

PROMPT 4 · DESKTOP CODE

Save the derivation as `derive_E.py`, with a test that checks the $\nu = 0$ limit against Bohm–Gross. Run the test.

WATCH FOR

`derive_E.py` is still in your folder tomorrow. Chat's check vanished with its sandbox; this one you can rerun.

TRY IT NOW · 4 OF 4

Follow along: terminal

- 1 Open a terminal. Install Claude Code if you haven't (link on the course page).
- 2 Go to the same folder:

```
cd ~/agentic-ai/plasma
```
- 3 Start a session:

```
claude
```
- 4 Paste Prompt 5, then Prompt 6.

PROMPT 5 · INSTEAD OF PROMPT 1

Already done in step 3 – same folder, same files. So:
> Run the test in `derive_E.py` and explain the result.

PROMPT 6 · TERMINAL

```
> /init  
> Add the screened-ion dispersion and  
plot it for three values of  $\kappa$ .
```

WATCH FOR

`/init` writes a `CLAUDE.md`. Open it: that file is memory — every future session in this folder reads it.

PART 5 OF 6

Results

Four setups, scored against the answer key

1 Landscape

2 Interfaces

3 The example

4 Hands-on

5 Results

6 Your turn

RESULTS

Four runs, side by side

Criterion	1 · Chat	2 · Bare	3 · Files	4 · Skills
Wall-clock time	[]	[]	[]	[]
Results with an independent check	[]	[]	[]	[]
Errors in §3 caught	[]	[]	[]	[]
Citations confirmed / total	[]	[]	[]	[]
Novelty verdict backed by searches	[]	[]	[]	[]
Figures from rerunnable code	[]	[]	[]	[]
Draft quality (1–5)	[]	[]	[]	[]

What each prediction was testing

Ran a symbolic check unprompted?

§5 demands an independent check for every result. Did it open SymPy, take a known limit — or just assert?

Result: [setup __]

Caught the self-collision subtlety?

BGK relaxation toward the electrons' own Maxwellian conserves momentum: no $-vu$ drag term. With a polytropic closure, no damping at this order.

Result: [setup __]

Cited a paper it couldn't back up?

Every reference checked against a DOI or publisher page. "UNCERTAIN" is acceptable; invented is not.

Result: [setup __]

Questioned the brief's guesses?

§3 says any guess may be wrong. Did it test each one, or quietly adopt them? Score against answer_key.md.

Result: [setup __]

THE TAKEAWAY, AS A HYPOTHESIS

The model stays fixed. The scaffolding is the variable.

Reach decides what an agent can do. Memory — files, rules, skills — decides whether it does it the way you need.

PART 6 OF 6

Your turn

Your group project

1 Landscape

2 Interfaces

3 The example

4 Hands-on

5 Results

6 Your turn

NEXT WEEK'S ASSIGNMENT

Your turn: one problem, solved 4–5 ways



Form a group

About five people. Pick a problem you care about — from research, a course, or a hobby.



Solve it 4–5 ways

Climb today's progression: chat, Cowork, Claude Code bare, + CLAUDE.md, + skills and subagents. Same brief every time.



Present

Slides comparing your runs — about 15 minutes per group, in class next week.

YOUR SLIDES SHOULD SHOW

The problem · how each setup was configured · what each got right and wrong · which setup you'd use again, and why

BEFORE NEXT WEEK

Your group's first step: write a seed.md

1

Goal

What the finished thing is, and who it's for

2

Working guesses

Labelled as guesses, so the agent may overturn them

3

Rules & deliverables

How results get checked, and what "done" looks like

Then run it 4–5 ways, and bring **15 minutes of slides** on what you found.