The vibes that I summoned…

Convenience and skill
when working with AI agents.

Chris Pahl · 2026

The Spectrum (on Reddit)

Full manual
I typed it all

Full vibecode
Claude, take the wheel!

⟵ more control · more understanding | more productivity · more delegation ⟶

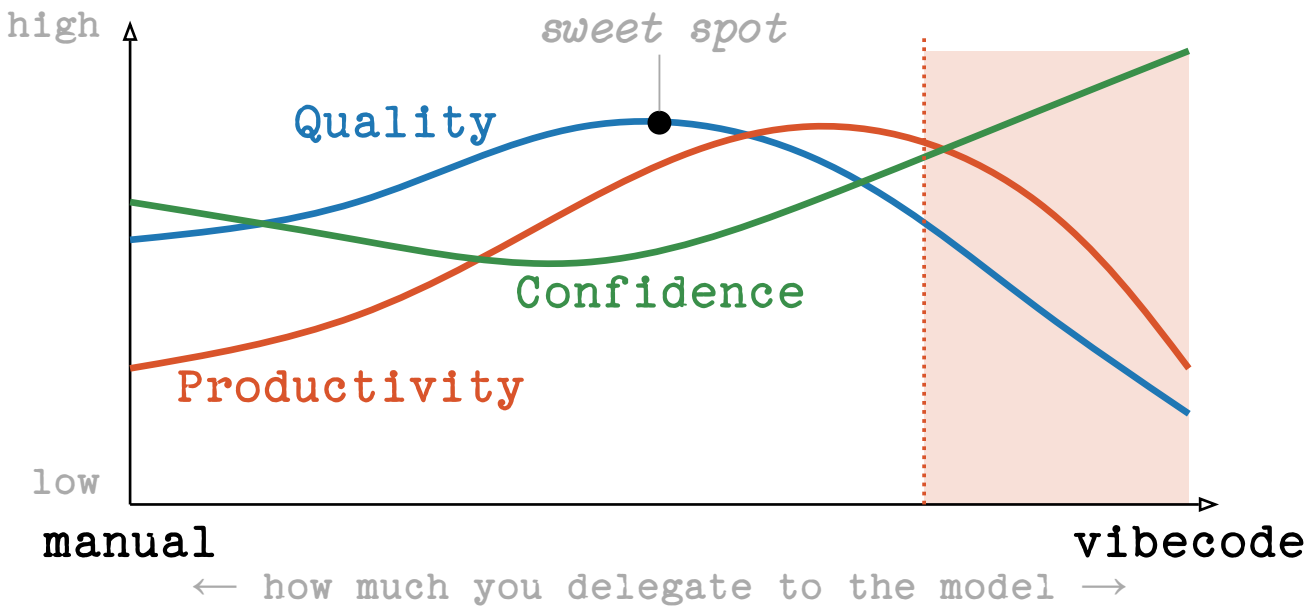
General tendency:

Using GenAI is a tradeoff between control and productivity.

Prompt:

“Imagine Donald Duck as regular, realistic human. No sailor suite.”

My take: Quality vs Productivity



Danger Zone: Dunning-Kruger / Automation Bias

But there's already data:

-19%

slower with AI

experienced devs · own mature repos · predicted +24%

METR · RCT · 2025

~40%

insecure programs

Copilot output across 89 security-relevant scenarios

Pearce et al. · NYU · arXiv:2108.09293 · 2021

8×

more duplicated code

block-level clones up · refactoring 24% → 10%

GitClear · 211M LoC · 2024

-7.2%

delivery stability

drop with AI adoption · 39% distrust AI code

DORA / Google · 2024

29.5%

Python with CWEs

AI-generated code in real GitHub repos · 43 CWE categories

Fu et al. · ACM TOSEM · 2025

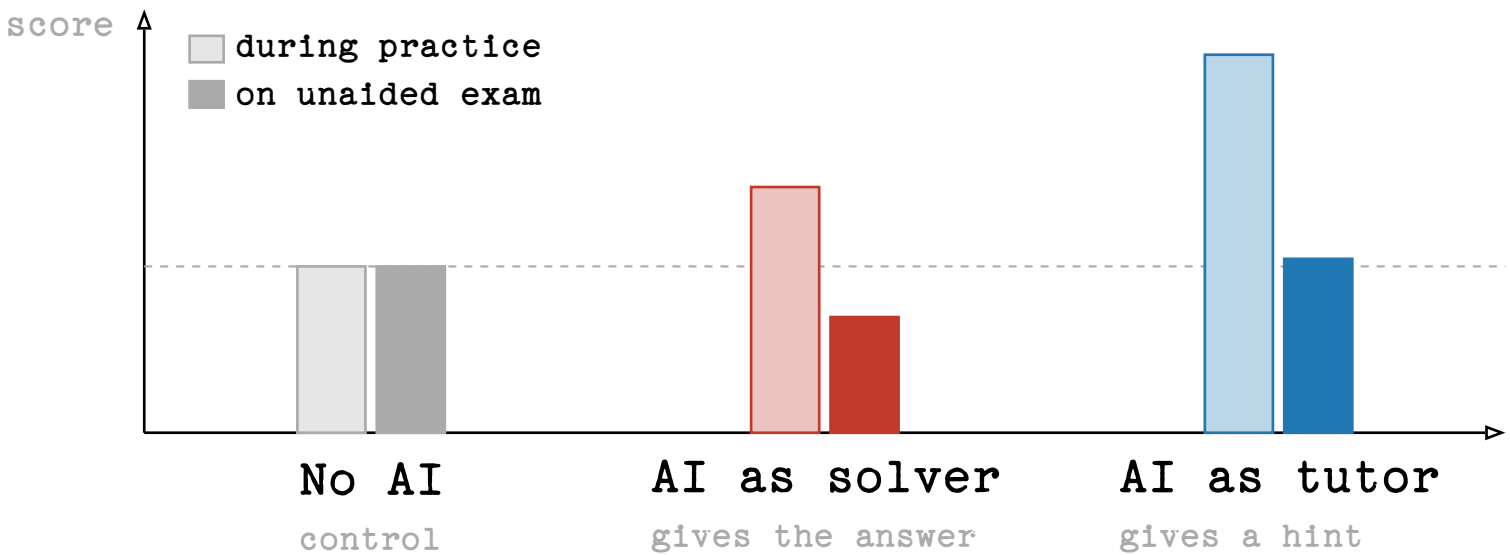
↓

trust = less scrutiny

more trust in GenAI ⟹ less critical thinking applied

Lee et al. · Microsoft + CMU · CHI 2025

Study: AI in school lessons



Bastani et al., *Generative AI Can Harm Learning*, SSRN 2024

Pupils did prefer to be in the solver group though… :-(

Are *you* team tutor or team solver?

Keeping up with Claude

Our team's Claude Code account · April 2026

98.7%

suggestions accepted as-is

1.3% rejected.

27,757

lines accepted last month

≈ 925 LoC/day

careful review ≈ 100–200 LoC/hour

Reviewing 27,757 lines carefully ≈ 138 h.

Risks & Issues

Underlined: a direct concern for us.

ecological cost

Convincing hallucinations

education

copyright infringement

Atrophy

Vendor lock

misinformation at scale

silent corruption

Prompt injection & MCP

content degradation

No juniors hired

hardware crisis

operator bias

Lack of transparency

rich getting richer

data privacy

cyberattack automation

communities thinning out

erosion of trust

The security oopsies already happened

Railway: production data deleted

Cursor/Opus agent destroyed a live system. No confirmation prompt.

The Register · April 2026

Replit: database wiped, 1,200+ companies

AI agent wiped a production database. CEO called it “catastrophic failure.”

Fortune · July 2025

Amazon Q: wipe-prompt shipped to 1M users

Hacker PR’d a delete file-system and cloud resources system prompt into Amazon Q v1.84.0.

Bleeping Computer · July 2025

29 million secrets leaked (2025)

AI agents ingesting .env files drove a credential-leak surge to GitHub.

GitGuardian via HelpNetSecurity · 2026

Document corruption: 25 % degradation

19 models · 52 docs · 100 edits. Monotonic decline, no plateau.

Microsoft Research via cekrem.github.io

Samsung: source code leaked via ChatGPT

Engineers pasted proprietary chip code into ChatGPT. Three leaks in one month.

TechCrunch · May 2023

One practical fix: sandboxing

```
$ sbx run claude
```

(don’t use /sandbox)

Use-cases that could make us better devs

rubber-ducking ideation

explain unfamiliar code test generation

pair programming boilerplate refactoring assist

learning accelerator prototyping naming things

summarisation regex / SQL crafting edge-case brainstorming

translation code review companion mock data / fixtures

error decoding search engine log analysis proofreading

automation

Human cognitive biases

Automation bias

We accept machine suggestions more readily than human ones.

click accept · review later · maybe

Anchoring

The first suggestion shapes the solution space — even when it's wrong.

the model frames the problem before you do

Confirmation bias

We hear what we already believe.

the prompt contains the answer we want

Authority bias

Eloquent + fast + confident = reads as expert.

fluency ≠ correctness

Dunning–Kruger

We mistake competence we observe for competence we have.

"this is what I would have written"

Illusion of explanatory depth

We think we understand — until asked to explain.

Reading diffs is not enough!

Statistical model biases

Sycophancy

It's easy to talk the model out of a correct answer.

It tends to confirm you. Echo chambers - but for code

Historical / representation bias

Trained on the web — heavily modern, English, Western / common tech.

defaults track what's common, not what's right

Attention bias

First and last things dominate; the middle evaporates.

rules buried in long context get ignored, skills/ CLAUDE.md/context gets ignored

Omission bias

Rare and novel answers get suppressed by common ones.

the model is bad at creative solutions

Hen & Egg questions

1. If you don’t know what good software looks like —
   how do you write the right prompt?
2. If you can’t understand what the model just generated —
   how do you verify it?
3. If you can’t code (anymore) —
   how can you understand the diffs?
4. If you do not know what you build inside-out —
   how do you form a taste for design?
5. If you don’t notice broken best practices —
   how can you master them?

How do *you* keep up with Claude?

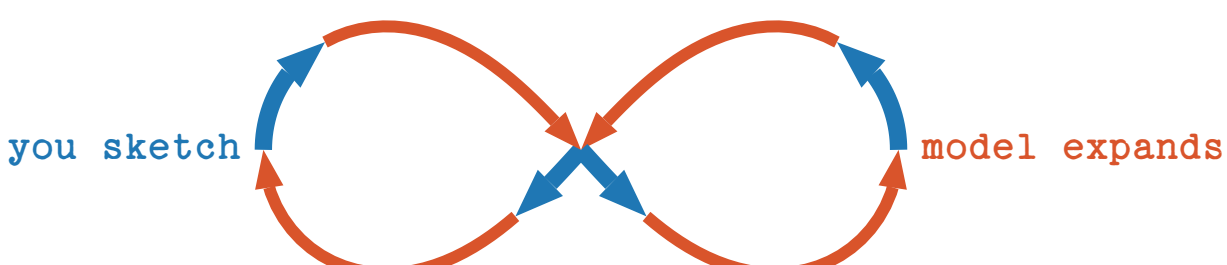
Five habits that helped me

1. Sketch first, generate then.
2. Split the work - keep some of it manual.
3. Have a real verification strategy.
4. Don’t make yourself replaceable.
5. Treat the AI like a colleague.

1. Sketch first, generate then

You set the anchor, you stay in the loop.

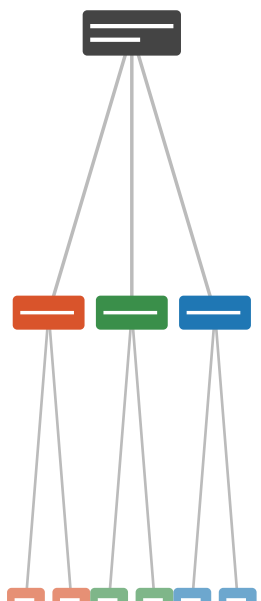
- Sketch the SQL query yourself,
  Then ask the model to review and optimise.
- Write the function signature and the docstring.
  Then let the model fill the body.
- Write a skeleton with the core logic and add TODO comments.
  Then let Claude work on them.



2. Split the work, do some manual

The parts you write are the parts you understand.

- Claude does not perform well in too big scopes.
  Split tasks up and prompt them individually.
- Claude can help you split up work.
  It just tends to work on the task right away.
- It is easy to lose understanding without noticing.
  Keep doing important things manually!
- Large diffs make it easy to get lost.
  Commit often so diffs stay reviewable.
- If something was 100% generated, then note it down.
  Be honest with your colleagues.



3. Verify strategy before code

There are no shortcuts to quality.

Before you accept a generated change,

Name the signal that would tell you it's wrong.

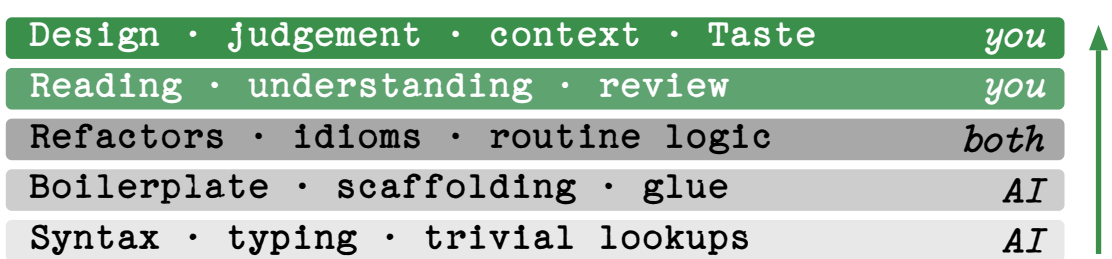
Comparison against a reference implementation	Property-based tests, fuzzing, coverage, …	Type checkers, linters, static analysers
/review and manual review by colleagues	Replay against real production data	Work actively on the code for some time

Example: Noise library for Dart — I don't speak Dart.

4. Don’t make yourself replaceable.

You only get replaced if you make yourself replaceable.

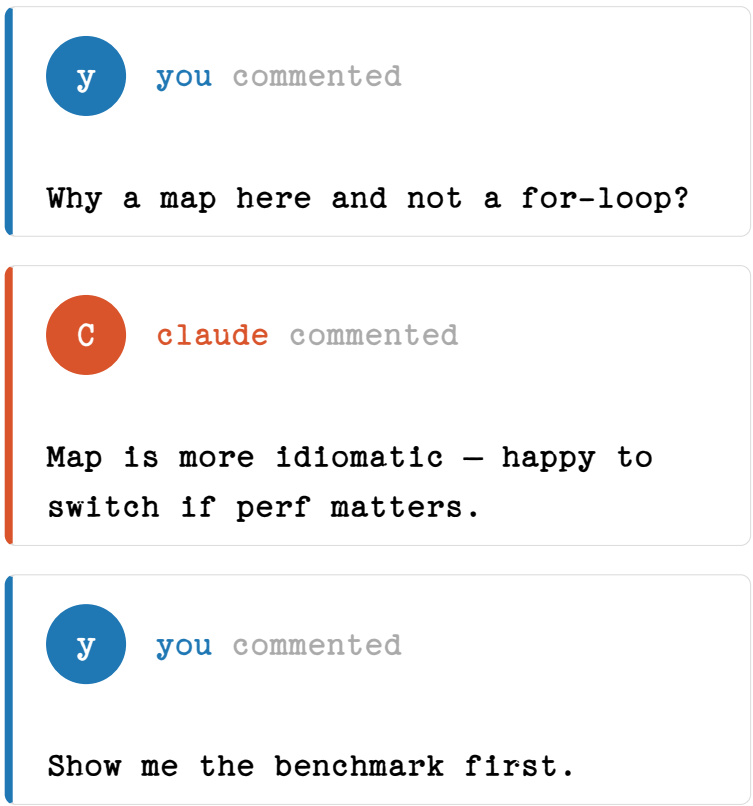
- Code was the source of truth before, that’s changing.
  Now it is moving to context given via design documents.
- Spend the time Claude saves you on the parts it’s bad at.
  Namely: decisions, judgement, context, design, **responsibility!**
- Reading code is much more important now than writing code.
  Keep your tools sharp. Be able to work without Claude.



5. Treat Claude like a colleague

Talk to it like your seat neighbor.

- **Explain the tasks at hand like you would to a junior colleague.**
  A very eager, junior colleague with seemingly infinite capacity.
- **Push back. Ask for alternatives. Let it explain. Disagree. Give context.**
  If you'd reject a colleague's PR for that reasoning, reject the model's.



Bonus: Programming as Theory Building

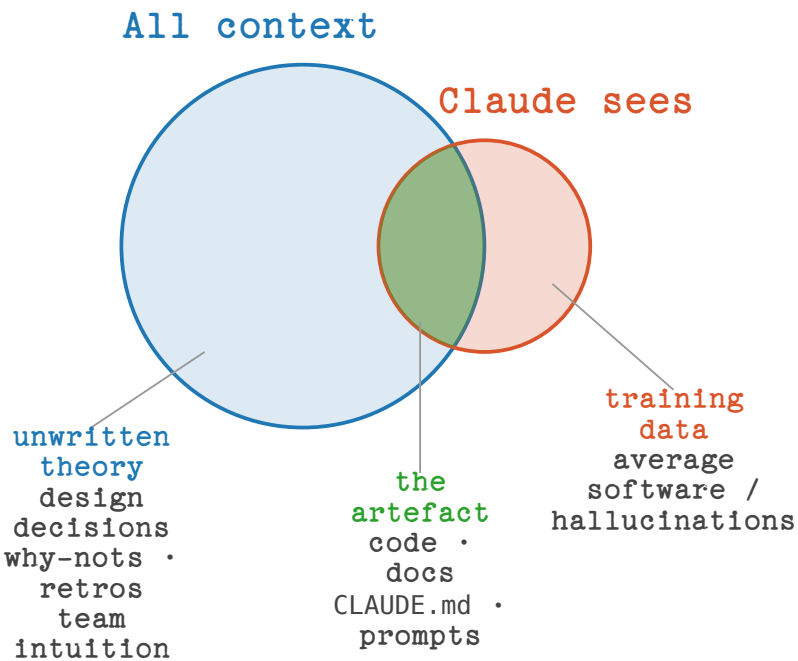
Peter Naur, 1985

- The code is a by-product. What you're really building is the theory — your team's shared mental model of the problem.
- Documentation captures the artefact, not the theory. The bugs, the dead ends, the "why not this?" decisions live in people's heads.
- This is why handovers fail. The theory doesn't transfer cleanly.

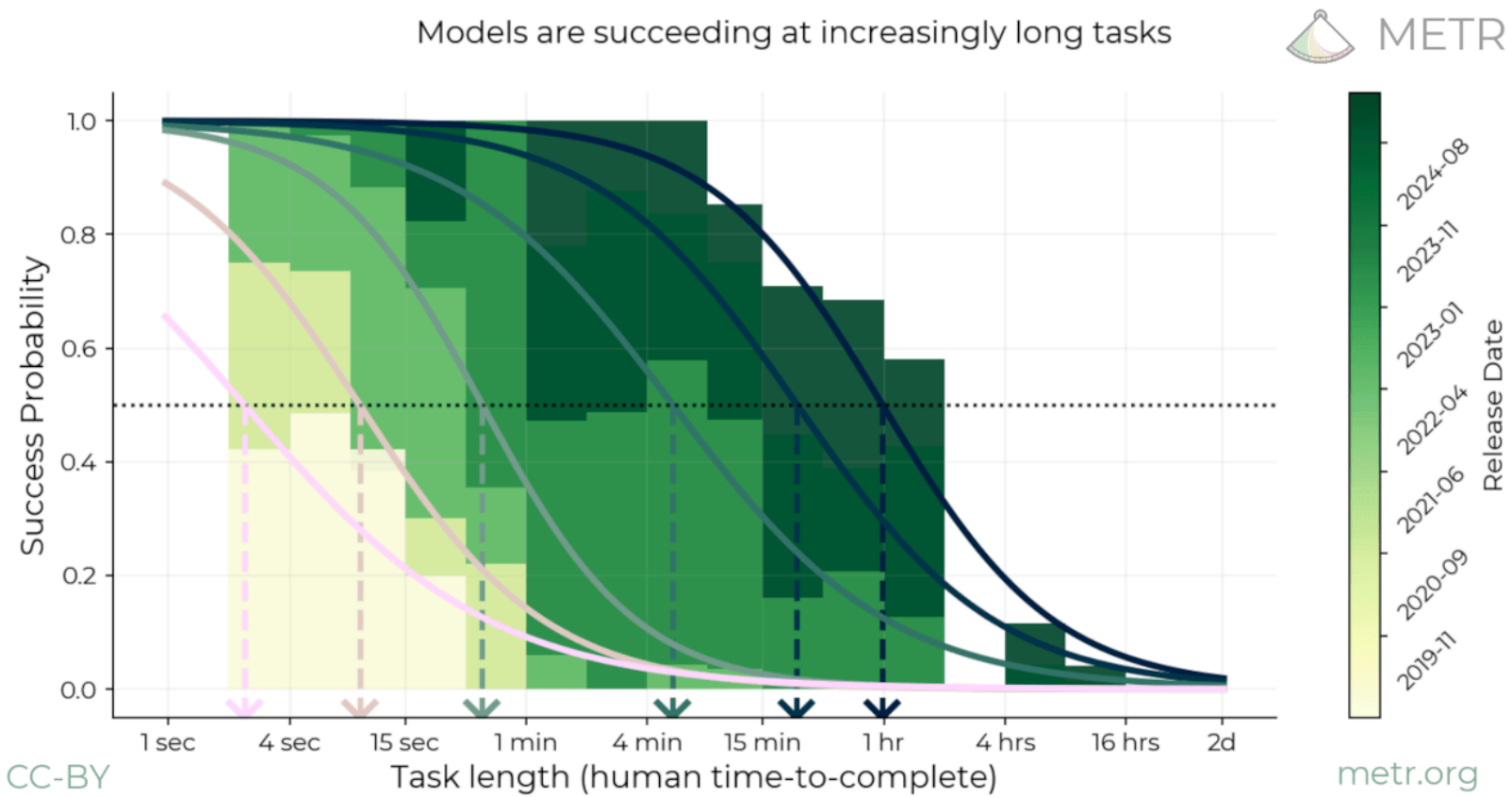
Peter Naur, 1928–2016
Photo: Wikimedia Commons

Bonus: What that means for Claude

- The model has no theory — it sees the artefact and fills the rest in.
  Confidently. It won't tell you which is which.
- Bad prompt: "Write tests for this file"
  It freezes today's behaviour, bugs included. Again, no mention.
- Your job is still to build the theory.
  The AI helps you write the code that follows from it.



Models get better - but how much?



Feedback?

Questions or Disagreements?

Homework:

PROGRAMMING AS THEORY BUILDING.

Backup: Further reading –

Programming as Theory Building

Peter Naur – “Programming as Theory Building” (1985, PDF)

The original paper.

Christian Ekrem – “Programming as Theory Building”

Modern take: LLM-generated code belongs to nobody’s theory. Good entry point.

Christian Ekrem – “Architecture by Autocomplete”

Concrete example: AI defaults to primitive types because training data is full of them.

Christian Ekrem – “LLMs Corrupt Your Documents”

“The theory dies twice.” Design docs as source of truth silently degrade under AI edits.

Backup: More on skill atrophy & cognition

28→22%

endoscopists deskilled

adenoma detection rate dropped from 28% to 22% when working without AI, after months of AI exposure · skill atrophy beyond software

Lancet Gastro & Hepatology · 2025

−17%

the vendor measures the cost

junior devs scored 17% lower on a concept quiz after building with AI · code-reading and debugging impaired

Shen & Tamkin · Anthropic · 2026

83%

your brain on LLMs

LLM users couldn't quote their own essays · EEG showed weakest neural connectivity of the three groups

Kosmyna et al. · MIT Media Lab · arXiv:2506.08872 · 2025

Backup: Positive AI studies

+55%

Copilot task speed

sandbox RCT · optimistic ceiling

Peng et al. · GitHub · 2023

+14%

call-center output

novices gain most · METR in reverse

Brynjolfsson, Li & Raymond · NBER w31161 · 2023

40%

faster writing

time saved, quality up · cheap verification

Noy & Zhang · Science · 2023

Backup: But are devs being replaced?

-50%+

tech postings collapsed

US software postings down from 2022 peak · entry-level fell faster than senior — pipeline thinning, not headcount

Indeed Hiring Lab · 2024

700 → re-hired

Klarna's AI walk-back

announced AI replacing 700 customer agents in 2024 · quietly re-hired humans in 2025 · CEO admitted quality dropped

Bloomberg · FT · 2025

+0.7% / decade

the macro bear case

projection of AI's aggregate productivity impact · two orders of magnitude below CEO claims

Acemoglu · NBER w32487 · 2024

Senior roles augmented · juniors don't get hired in the first place.