LLM coding agents in HEP

Claude Code as the example

Krvavec
September 29, 2026

Andraž Tomšič
Jožef Stefan Institute

Why an agent: it sees the whole repository

Chat in the browser

- No access to your files: copy-paste.
- Sees a keyhole, not the room.
- Cannot run, test or check anything.
- The context is your clipboard.

Agent in the repository

- Reads and writes your files.
- Runs the build, tests, jobs.
- Searches the whole codebase itself.
- Keeps state in files across sessions.

- **Code indexing is the point**: it finds the code it needs, not the snippet you pasted.

- Claude Code as the example, since we use it.
- Codex, Gemini CLI, Copilot: same principles. `CLAUDE.md` is `AGENTS.md` or `GEMINI.md` there.

I adapted this slide from Lenart's presentation (thank you, Lenart!)

Install: terminal or VS Code

terminal, on your machine or over ssh

```
curl -fsSL https://claude.ai/install.sh | bash   # once
cd my-analysis && claude     # log in with your plan
> /init                      # writes a starter CLAUDE.md
```

Terminal (CLI)

- Runs where the data and jobs are: ssh, containers.
- In `tmux`: long sessions keep going with the laptop closed.
- Cannot paste screenshots.

VS Code extension

- Just as powerful: same engine.
- Paste screenshots, inline diffs.
- Session ends when the laptop closes.
- Brings its own copy: `claude` not on PATH.

- I use both: same `CLAUDE.md`, same settings.

Getting a seat

- **F9 is looking into Claude for all employees**.
- Hope: 1 year free through the Team plan for scientists.

seat	per month	usage
basic	$\sim$20 EUR	$1\times$
premium	$\sim$100 EUR	$5\times$ basic

Otherwise, Team plan list prices.

My suggestion

- Almost all seats basic.
- I use basic heavily and rarely hit the limit.
- The tricks: next slides.

Team plan for scientists
`support.claude.com/en/articles/16634237`

All plans
`claude.com/pricing`

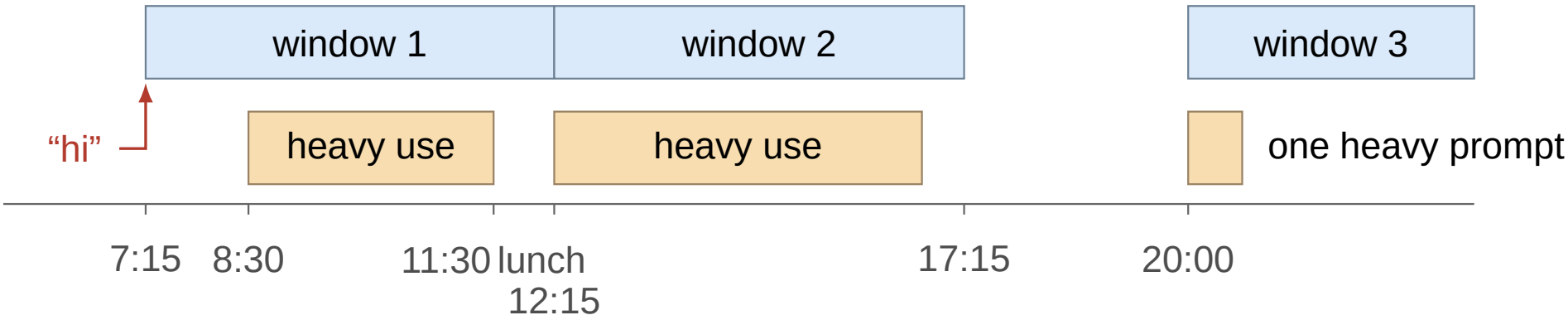
Limits: plan around the 5 h window

How they work

- A 5 h limit and a weekly limit.
- The 5 h window starts with your first prompt.
- Weekly: almost never hit. 5 h: that is the one.

Using it heavily

- Start the window early: a “hi” is enough.
- Spend it in one focused block.
- Evening: a long-thinking prompt, e.g. a proposed code change to read tomorrow.



Use case 1: coding, and git

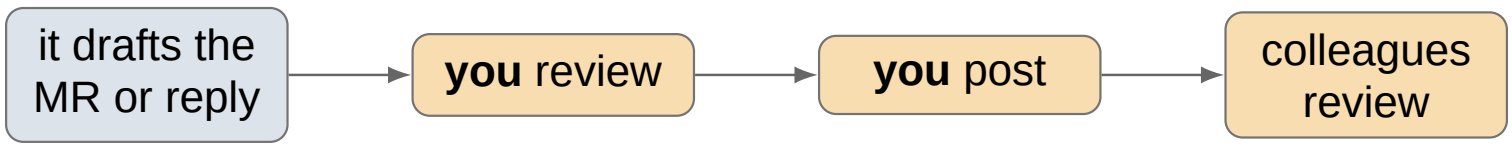
Coding

- Plan first: it reads the code and proposes the change.
- You approve; it edits, builds, runs, reads the errors.
- Strong at reading code nobody here wrote.

Git

- It can pull, commit, push, open MRs, comment.
- MRs, comments: needs a GitLab/GitHub access token.
- **MRs and comments: by hand**.
- Sometimes comments even when forbidden.

- **I always bypass permissions**: `claude --dangerously-skip-permissions`, in VS Code a toggle.
- Otherwise you confirm every command: useless. A year of full access, no medium or large accident.

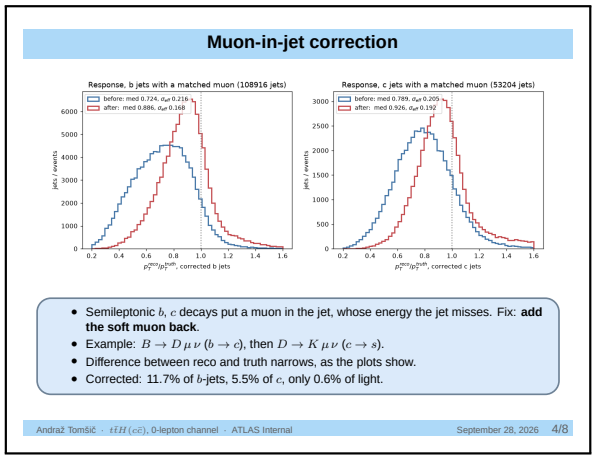


- Do not spend colleagues' time on MRs (PRs) you have not reviewed yourself.

Use case 2: slides

`slides_krvavec_0L/gen_deck.py`

```
MAIN.append(frame(
    "Muon-in-jet correction",
    "  \\vspace{1mm}\n  \\centering\n"
    + fig("muinjet_response.png", r"0.97\deckw")
    + "\n  \\vspace{2mm}\n"
...
```



- Start: give it an old deck (pptx, keynote, PDF), it takes the style.
- It writes `gen_deck.py`: python that emits beamer.
- Describe each slide: plots, numbers, message.
- Point it at earlier decks to copy from: v1, v2, v3 updates.
- Repetitive slides in one loop, no dragging and aligning.
- Weak spot: it cannot see the page.

Tip 1: `CLAUDE.md`

- A glorified notes file / README, read at the start of **every** session.
- Code structure: it does not rescan the repo each time.
- Your preferences on code changes.
- Git workflow: what it must not do.
- Progress on what you are doing.

- `/init` writes the first one.
- Keep it short: every line costs tokens, every session.

`CLAUDE.md, the four parts`

```
# Code structure
- src/fill/: event loop
- src/plot/: plotting
- build: see README

# Code changes
- Minimal diffs, local style
- Comments: why, 1-2 lines

# Git
- Never push unless asked
- Never post MR comments
- No slash in branch names

# Current state
- 0L fit ok, fix not committed
```

Tip 2: daily logs, and clearing often

Daily logs

- One sentence in `CLAUDE.md`: it keeps a log per day.
- An index of your work: pick up old tasks, old talks.
- Maybe the thesis one day.

Context

- Heavy chat: longer answers, more tokens, slowly worse.
- Update the daily log often, `/clear` often.
- Fresh session: “read today’s log”, carry on.

`CLAUDE.md, the rule (condensed)`

```
- One log per day:
  daily_logs/claudeYYMMDD.md
- Keep its ## Report current
```

```
$ ls daily_logs/
claude260914.md  claude260916.md
claude260917.md  claude260921.md
claude260923.md  claude260924.md
claude260925.md  claude260929.md
```

`ttHccAnalysis/daily_logs/claude260921.md`

```
## Report

Cleaned up the repo: the 0L v5
work now sits on
`mlregions-0l-v5`, a fresh branch
that already contains latest
`origin/main`, and it is pushed.
...
```

Tip 3: model and prompts

Model

- Almost always the newest, heaviest one: Opus 5.5 now.
- Tune speed and tokens with the effort: `/effort`, I use high or extra high most often.

Prompts

- Essays are fine: 1000 to 2000 words, even more.
- Slides: every bullet, every plot, in words.
- The repetitive part is then its job, not yours.
- **Give paths**: no repo scan, fewer tokens, faster.

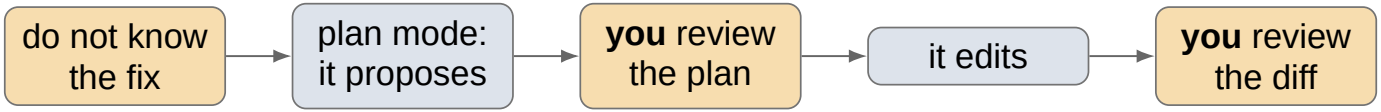
a real slide prompt, excerpt

```
Use my styling, be brief in words.
First say Run 2 0L, state the preselection.
Then show TNA plots of the discs and discuss the significance we
will use (show rebinned vs not) ...
Show prefit plots of all CRs on one slide, comment that it looks ok,
show the relevant discriminant ... log versions in backup ...
[...]
fit output /data0/.../wsmaker_tthqq/output/ttHcc_0L-01_mc16ade...
TNA plots /data0/.../ttHcc/results_v3_inference_CMOKv5_0L_run2/...
```

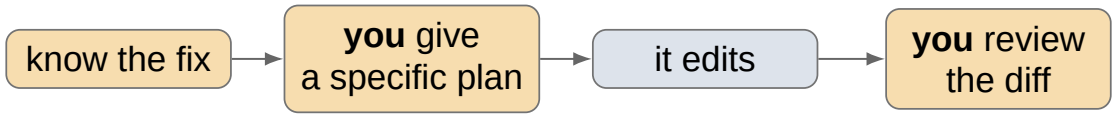
Review: you mentor it

- Understand every change, and what you ask it to fix. Otherwise the result is bad.

a)



b)



Our ATLAS group: half a year, plans from IJS

- Bad MRs, bad code practices: we learned from them.
- A real learning curve, not the easiest tool.
- Gains already: more time on physics, less on bugs.

Everything else

- Guides, tips, settings: caveman tools (Google) or ask the AI.
- **Starting out: ask me, or any of us in ATLAS**.

- **Everything gets reviewed.** It is fast, you are responsible.

Claude Code is to me what Štef is to prof. Cizelj

Backup

All plans, and data handling

$/month	monthly	annual	Claude Code
Free	$0	$0	no
Pro	$20	$17	yes
Max	from $100	n/a	yes, $5\times$ or $20\times$ Pro
Team, standard seat	$25	$20	yes
Team, premium seat	$125	$100	yes, $5\times$ standard
Enterprise	custom	custom	yes, seats plus API-rate usage

Team plan for scientists

- Standard seats free, premium $15/user/month, 12 months.
- PI at a university or nonprofit institute: JSI qualifies.
- 1 to 25 seats per group, verified in about 7 working days.

- Team and Enterprise data: not used for training by default.
- Prices as of 2026-09-24, check before signing up.

Read these

Quickstart

`/quickstart`

Best practices

`/best-practices`

Common workflows

`/common-workflows`

Memory and `CLAUDE.md`

`/memory`

VS Code

`/vs-code`

- All under `code.claude.com/docs/en`.

What else is in the box

feature	what it does
Plan mode	read-only exploration, a plan to approve before any edit
Subagents	parallel workers with their own context, only the conclusion comes back
Skills	packaged instructions it loads when a task matches
Hooks	shell commands run on events, e.g. a formatter after every edit
MCP	connectors to external tools and data
Headless	`claude -p`, in scripts and CI
Agent SDK	the same loop as a library, for your own agents

- None of it needed to start; all of it in the docs.