AI Agents Review

21 Days with Claude Code

Zhao Jiaxi (NUS) & Claude Code (Anthropic)

September 1, 2025

Outline

What is Agent and Why Agent?

How to Build Agent?

Agent Literature Review

Case Study: Claude Code

What is Agent and Why Agent?

What is an Agent?

Formal Definition (Russell & Norvig)

”An agent is anything that perceives its **environment** through sensors and acts upon that environment through **actuators**.”

Key Characteristics:

- Autonomy: Acts independently
- Perception: Senses environment
- Action: Changes environment
- Goal-directed: Works toward objectives

In AI Context:

- Software that performs tasks with minimal supervision
- Makes decisions to maximize goals
- Can learn and adapt over time
- Ranges from thermostats to LLMs

Etymology: From Latin *agere* (to do) - ”action on behalf of”

Tool Use: Signal of Early Intelligence

- Anthropological perspective: Tool use marks cognitive revolution
  - Stone tools → Agriculture → Writing systems
  - Cognitive leap: from reactive to proactive behavior
- AI parallel: From language generation to action
  - GPT-3 (2020): Pure text generation
  - WebGPT (2021): Web browsing capability
  - ChatGPT Plugins (2023): Third-party tool ecosystem
  - Claude Code (2025): Full IDE integration

Eliminating Hallucination Through Grounding

Problem: LLMs generate plausible but incorrect information

Solution: Ground responses in external tools and data

Without Tools

```
Q: What is sqrt(144)?
A: sqrt(144) = 13
```

× Hallucination

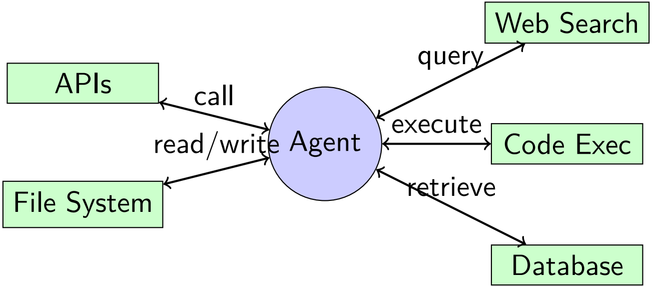
With Calculator Tool

```
Thought: Use calculator
Action: calc(sqrt(144))
Result: 12
A: sqrt(144) = 12
```

✓ Grounded

Key Insight: Grounding in external tools eliminates hallucination

Environmental Interaction Paradigm



Perception: Read from environment (search, query, read)

Action: Modify environment (write, execute, call)

Learning: Adapt based on feedback

Agent Learning Through Environment

Key Concept: Agents can evolve and improve through reinforcement learning

- State: Current context + tool outputs
- Action: Tool calls + text generation
- Reward: Task success + human feedback

Learning Goals:

- Learn optimal tool selection
- Improve reasoning strategies
- Adapt to user preferences

Evolution: From Prompting to Agents

Capability	**Prompting**	**Tool Use**	**Agents**	**Advanced**
User Control	High	Medium	Low	Guided
Context Length	Limited	Limited	Extended	Extended*
Error Recovery	Manual	Manual	Semi-auto	Semi-auto
Task Complexity	Simple	Medium	Complex	Complex
Example	ChatGPT	Plugins	ReAct	Claude Code

Prompting Era (2020-2022): Chain-of-thought, few-shot learning

Tool Use Era (2022-2024): Function calling, structured outputs

Early Agent Era (2024-2025): Planning, memory, basic workflows

Future (2025+): Enhanced reliability, domain specialization

How to Build Agent?

Tool Use Mechanism: System prompts v.s. Function calling

Pattern Recognition for Tool Calls

LLMs output text → System recognizes patterns → Triggers tool execution

OpenAI Function Calling:

```
{
  "role": "assistant",
  "content": null,
  "function_call": {
    "name": "get_weather",
    "arguments": "{\"location\": \"SF\"}"
  }
}
```

- JSON Schema: Structured output for reliable parsing (Claude's tool use format)
- XML Tags: Better prompt readability and flexibility

MCP: Model Context Protocol

Standardized protocol for LLM-tool communication that separates tools from AI agents

Key Components:

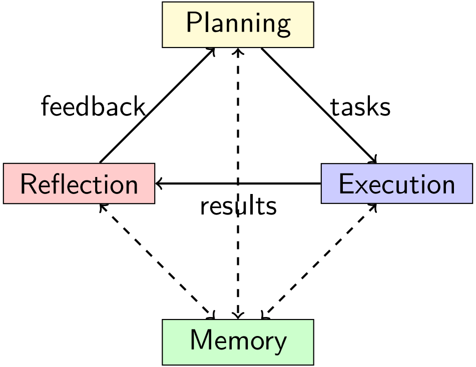
- MCP server (calls tools), agent (MCP client)
- Tool registration and discovery
- Execution sandbox
- Result formatting

Benefits:

- Tool interoperability
- Security isolation
- Consistent interfaces

More info: Bilibili Video

Agentic Workflows



Planning: Task decomposition

Execution: Tool calls, actions

Reflection: Self-critique

Memory: Context persistence

Training Agents: Beyond Pre-training

Stage	**Pre-training**	**Fine-tuning**	**Agent Training**
Objective	Next token prediction	Task-specific	Tool use + Planning
Data	Web text	Labeled examples	Trajectories
Scale	Trillions of tokens	Thousands	Millions of steps
Method	Self-supervised	Supervised	IL/RL/Self-play

- Imitation Learning: Learn from human demonstrations
  - WebGPT: 6K demonstrations of web browsing
- Reinforcement Learning: Optimize for task rewards
  - RLHF for preference alignment
- Self-improvement: Generate and learn from own data
  - Toolformer: Self-supervised API call insertion

Agent Frameworks: LangChain & LangGraph

LangChain:

- Chain-based (DAG) architecture
- Sequential, linear workflows
- High-level abstractions

```
from langchain.chains import LLMChain
from langchain.llms import OpenAI

chain = LLMChain(
    llm=OpenAI(),
    prompt=prompt_template,
    memory=ConversationBufferMemory()
)
result = chain.run(query)
```

LangGraph:

- Graph-based with cycles
- Stateful, complex workflows
- Low-level control

```
from langgraph.graph import StateGraph

workflow = StateGraph(state_schema)
workflow.add_node("plan", planning_node)
workflow.add_node("act", action_node)
workflow.add_edge("plan", "act")
workflow.add_conditional_edges("act",
    should_continue)
app = workflow.compile()
```

Agent Literature Review

WebGPT: First Large-Scale Web-Browsing LLM

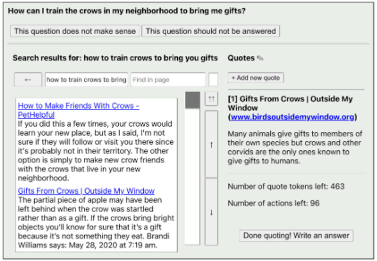
Text-based browser environment

Imitation learning from humans: 6K human demonstrations

RLHF for alignment: 20K preference comparisons

Performing rejection sampling against a reward model: 56% preferred over humans, 69% vs Reddit answers

Provides factual answers with citations


This question does not make sense
This question should not be answered
←
Find in page
↑↑
↑
↓
+ Add new quote
Done quoting! Write an answer

How can I train the crows in my neighborhood to bring me gifts?

Search results for: how to train crows to bring you gifts

how to train crows to bring

How to Make Friends With Crows - PetHelpful
If you did this a few times, your crows would learn your new place, but as I said, I'm not sure if they will follow or visit you there since it's probably not in their territory. The other option is simply to make new crow friends with the crows that live in your new neighborhood.

Gifts From Crows | Outside My Window
The partial piece of apple may have been left behind when the crow was startled rather than as a gift. If the crows bring bright objects you'll know for sure that it's a gift because it's not something they eat. Brandi Williams says: May 28, 2020 at 7:19 am.

Quotes

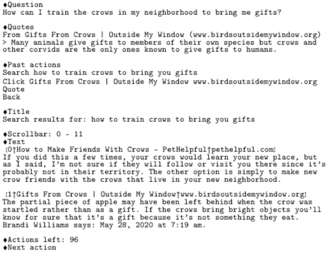
[1] Gifts From Crows | Outside My Window (www.birdsoutsidemywindow.org)

Many animals give gifts to members of their own species but crows and other corvids are the only ones known to give gifts to humans.

Number of quote tokens left: 463

Number of actions left: 96

(a) Screenshot from the demonstration interface.



```
♦Question
How can I train the crows in my neighborhood to bring me gifts?

♦Quotes
From Gifts From Crows | Outside My Window (www.birdsoutsidemywindow.org)
> Many animals give gifts to members of their own species but crows and
other corvids are the only ones known to give gifts to humans.

♦Past actions
Search how to train crows to bring you gifts
Click Gifts From Crows | Outside My Window www.birdsoutsidemywindow.org
Quote
Back

♦Title
Search results for: how to train crows to bring you gifts

♦Scrollbar: 0 - 11
♦Text
【0†How to Make Friends With Crows - PetHelpful†pethelpful.com】
If you did this a few times, your crows would learn your new place, but
as I said, I'm not sure if they will follow or visit you there since it's
probably not in their territory. The other option is simply to make new
crow friends with the crows that live in your new neighborhood.

【1†Gifts From Crows | Outside My Window†www.birdsoutsidemywindow.org】
The partial piece of apple may have been left behind when the crow was
startled rather than as a gift. If the crows bring bright objects you'll
know for sure that it's a gift because it's not something they eat.
Brandi Williams says: May 28, 2020 at 7:19 am.

♦Actions left: 96
♦Next action
```

(b) Corresponding text given to the model.

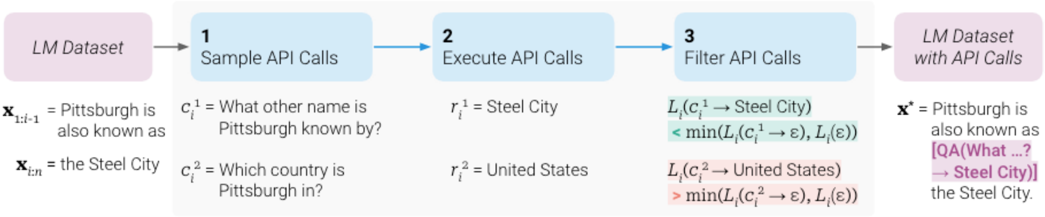
---

[1] Nakano, Reiichiro, et al. "WebGPT: Browser-assisted question-answering with human feedback." arXiv preprint arXiv:2112.09332, OpenAI, December 2021.

Toolformer: Self-Supervised Tool Learning

Self-Supervised Training:

- Step 1: Sample potential API call positions
- Step 2: Execute calls and compute L_i^+, L_i^-
- Step 3: Filter useful calls: $L_i^+ < L_i^- - \tau$
- Step 4: Fine-tune on filtered dataset



$\mathbf{x}_{1:i-1}$ = Pittsburgh is also known as

$\mathbf{x}_{i:n}$ = the Steel City

c_i^1 = What other name is Pittsburgh known by?

c_i^2 = Which country is Pittsburgh in?

r_i^1 = Steel City

r_i^2 = United States

$L_i(c_i^1 \rightarrow \text{Steel City}) < \min(L_i(c_i^1 \rightarrow \varepsilon), L_i(\varepsilon))$

$L_i(c_i^2 \rightarrow \text{United States}) > \min(L_i(c_i^2 \rightarrow \varepsilon), L_i(\varepsilon))$

$\mathbf{x}^*$ = Pittsburgh is also known as **[QA(What ...? → Steel City)]** the Steel City.

Toolformer: Self-Supervised Tool Learning

Weighted Cross-Entropy Loss:

$$L_i(z) = -\sum_{j=i}^{n} w_{j-i} \log P_M(x_j | z, x_{1:j-1})$$

Filtering Criterion:

$$L_i^+ = L_i(e(c_i, r_i)) \tag{1}$$

$$L_i^- = \min(L_i(\epsilon), L_i(e(c_i, \epsilon))) \tag{2}$$

Where: c_i = API call, r_i = response, $e(\cdot)$ = embedding

Key Insight: API calls are kept only if they reduce prediction loss for future tokens

Tools: Calculator, QA, Search, Translation, Calendar

Limitations:

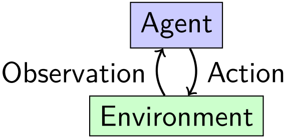
- Cannot learn tool usage in chains
- Limited to single-step tool interactions

---

[2] Schick, Timo, et al. "Toolformer: Language Models Can Teach Themselves to Use Tools." arXiv preprint arXiv:2302.04761, Meta AI, February 2023.

ReAct Framework: Synergizing Reasoning and Acting

Traditional Action-Only Loop:



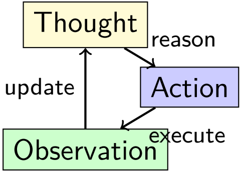
Agent

Observation

Action

Environment

ReAct Framework:



Thought

reason

update

Action

execute

Observation

Key Innovation: Interleaved reasoning traces and actions via few-shot in-context examples

Uses a frozen language model (PaLM-540B) with few-shot in-context examples

Baselines: Standard (w/o reasoning + observation), CoT (w/o observation + action), ACT (w/o reasoning)

---

[3] Yao, Shunyu, et al. "ReAct: Synergizing Reasoning and Acting in Language Models." International Conference on Learning Representations (ICLR), 2023.

ReAct: Example Trace

ReAct Trace Example

```
Question: What is the elevation of Mt. Everest?
Thought: I need to search for Mt. Everest
Action: search[Mt. Everest]
Observation: Mt. Everest is Earth's highest mountain
Thought: I need the specific elevation
Action: lookup[elevation]
Observation: 8,849 meters (29,032 ft)
Thought: I have the answer
Answer: 8,849 meters
```

Results:

HotpotQA: 27% improvement, ALFWorld: 34% over RL baselines, WebShop: 10% improvement

Reduces error propagation in multi-hop QA

Case Study: Claude Code

Claude Code: Introduction

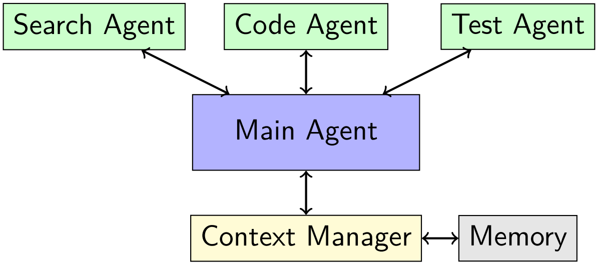
What is Claude Code?

- Anthropic's official CLI for Claude, an AI-powered coding assistant
- Released in February 2025
- Terminal-based, similar to OpenAI's Codex & Google's Gemini CLI, different from Cursor and Windsurf
- Source code available (partially obfuscated)

Models Used:

- Haiku 3.5 (simple tasks, high throughput)
- Sonnet 4 (main agent)
- Opus 4.1 (complex tasks)

Architecture: Multi-Agent System



Main Agent: Orchestrates overall task

Sub-agents: Handle specific subtasks in isolation

Context Manager: Optimizes token usage

Reverse Engineering: System Prompts

Key System Prompt Elements:

Core Instructions

"You are Claude Code, Anthropic's official CLI for Claude"

"Be concise, direct, and to the point"

"Minimize output tokens while maintaining quality"

"Use tools to complete tasks, not for communication"

Behavioral Guidelines:

Proactive: Use TODO lists

Defensive: Follow conventions

Efficient: Batch operations

Safe: Never commit without asking

Clear: Explain complex commands

Adaptive: Learn from context

Source: github.com/Yuyz0112/claude-code-reverse

Agentic Workflow

1. **Quota Check** (Haiku 3.5)
    - Lightweight API verification
    - Context initialization
2. **Task Analysis** (Main Agent)
    - Parse user request
    - Create TODO list
    - Plan execution strategy
3. **Execution Loop**
    - Execute tools in parallel when possible
    - Update TODO status
    - Handle errors and retry
4. **Context Compaction**
    - Isolate ”dirty context” in sub-agents
    - Return only essential results
    - Maintain conversation history

TODO List: How It Works

Core Mechanism:

- **Proactive Creation**: Agent creates TODO list for complex tasks (2+ steps)
- **Real-time Updates**: Status changes as work progresses
- **Atomic Operations**: Each task marked complete immediately upon finish

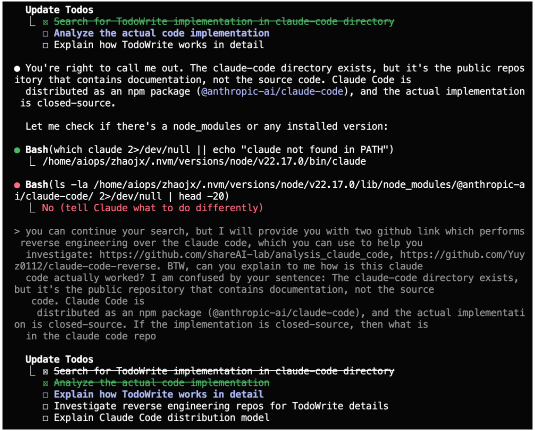
JSON Structure

```
{
  "todos": [
    {
      "content": "Fix authentication bug",
      "activeForm": "Fixing authentication
     bug",
      "status": "in_progress"
    },
    {
      "content": "Run tests",
      "activeForm": "Running tests",
      "status": "pending"
    }
```

Workflow:

1. User requests complex task
2. Agent creates TODO list
3. Exactly ONE task ”in_progress”
4. Complete → mark done → start next
5. Continue until all complete

TODO List: Dynamic Task Management



```
   Update Todos
     ⎿ ☒ Search for TodoWrite implementation in claude-code directory
       ☐ Analyze the actual code implementation
       ☐ Explain how TodoWrite works in detail

● You're right to call me out. The claude-code directory exists, but it's the public repos
itory that contains documentation, not the source code. Claude Code is
  distributed as an npm package (@anthropic-ai/claude-code), and the actual implementation
 is closed-source.

  Let me check if there's a node_modules or any installed version:

● Bash(which claude 2>/dev/null || echo "claude not found in PATH")
  ⎿  /home/aiops/zhaojx/.nvm/versions/node/v22.17.0/bin/claude

● Bash(ls -la /home/aiops/zhaojx/.nvm/versions/node/v22.17.0/lib/node_modules/@anthropic-a
i/claude-code/ 2>/dev/null | head -20)
  ⎿  No (tell Claude what to do differently)

> you can continue your search, but I will provide you with two github link which performs
 reverse engineering over the claude code, which you can use to help you
  investigate: https://github.com/shareAI-lab/analysis_claude_code, https://github.com/Yuy
z0112/claude-code-reverse. BTW, can you explain to me how is this claude
  code actually worked? I am confused by your sentence: The claude-code directory exists,
but it's the public repository that contains documentation, not the source
   code. Claude Code is
    distributed as an npm package (@anthropic-ai/claude-code), and the actual implementati
on is closed-source. If the implementation is closed-source, then what is
  in the claude code repo

   Update Todos
     ⎿ ☒ Search for TodoWrite implementation in claude-code directory
       ☒ Analyze the actual code implementation
       ☐ Explain how TodoWrite works in detail
       ☐ Investigate reverse engineering repos for TodoWrite details
       ☐ Explain Claude Code distribution model
```

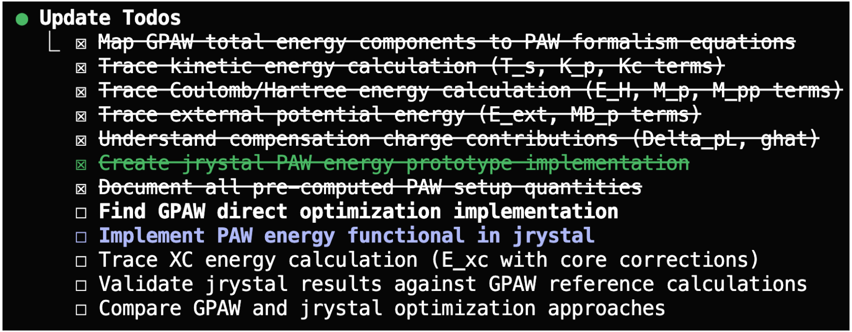
Implementation:

- Stored in `~/.claude/todos/`
- JSON format
- Three states:
  - `pending`
  - `in_progress`
  - `completed`

Benefits:

- User visibility
- Progress tracking
- Task decomposition
- Error recovery

TODO List: Real Example



● **Update Todos**
⎿ ☒ ~~Map GPAW total energy components to PAW formalism equations~~
☒ ~~Trace kinetic energy calculation (T_s, K_p, Kc terms)~~
☒ ~~Trace Coulomb/Hartree energy calculation (E_H, M_p, M_pp terms)~~
☒ ~~Trace external potential energy (E_ext, MB_p terms)~~
☒ ~~Understand compensation charge contributions (Delta_pL, ghat)~~
☒ ~~Create jrystal PAW energy prototype implementation~~
☒ ~~Document all pre-computed PAW setup quantities~~
☐ **Find GPAW direct optimization implementation**
☐ **Implement PAW energy functional in jrystal**
☐ Trace XC energy calculation (E_xc with core corrections)
☐ Validate jrystal results against GPAW reference calculations
☐ Compare GPAW and jrystal optimization approaches

Automatic creation when task complexity detected

Real-time updates as work progresses

Hierarchical task breakdown

Test Case: Read Scientific Code (99%)

Task: Understand complex numerical package GPAW (electronic structure calculation, 300K+ lines of code)

- ✓ Trace functions and classes without Go to Definition
- ✓ Connect mathematical derivations and code implementation
- ✓ Analyze the implementation efficiency

Advanced task:

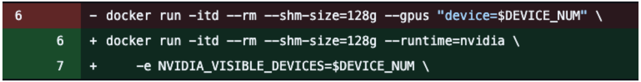
Ask CC to implement a Poisson solver for radial functions based on multipole expansion

Status: Difficult to verify correctness and design accuracy tests

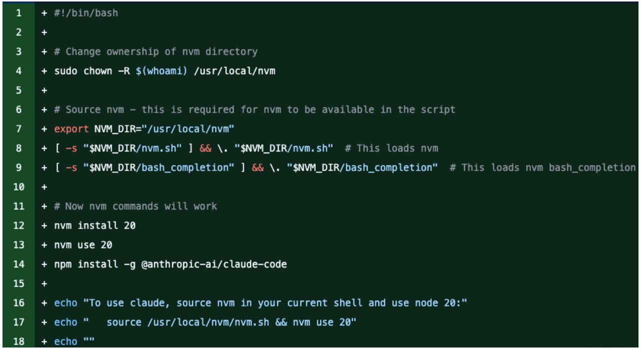
Test Case: Resolve Environment & Docker error (99%)

Task: Modify Docker configuration

- ✓ Debug docker error
- ✓ Run CC in a docker container
- ✓ Perfect solution with a few iterations



```
- docker run -itd --rm --shm-size=128g --gpus "device=$DEVICE_NUM" \
+ docker run -itd --rm --shm-size=128g --runtime=nvidia \
+       -e NVIDIA_VISIBLE_DEVICES=$DEVICE_NUM \
```



```
+ #!/bin/bash
+
+ # Change ownership of nvm directory
+ sudo chown -R $(whoami) /usr/local/nvm
+
+ # Source nvm - this is required for nvm to be available in the script
+ export NVM_DIR="/usr/local/nvm"
+ [ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh"  # This loads nvm
+ [ -s "$NVM_DIR/bash_completion" ] && \. "$NVM_DIR/bash_completion"  # This loads nvm bash_completion
+
+ # Now nvm commands will work
+ nvm install 20
+ nvm use 20
+ npm install -g @anthropic-ai/claude-code
+
+ echo "To use claude, source nvm in your current shell and use node 20:"
+ echo "   source /usr/local/nvm/nvm.sh && nvm use 20"
+ echo ""
```

Test Case: Integration Testing (90%)

Task: Write tests for the invertibility of normalizing flow models

Normalizing flow implemented via *distrax* with clear interface

Extremely clear instructions

✓ Implemented the desired tests but with some code redundancy

```python
# Claude generated test
def test_rqs_identity_composition(self):
    """Test that forward * inverse = identity for RQS transformation."""
    # initialization of the NF...

def test_rqs_inverse_forward_composition(self):
    """Test that inverse * forward = identity for RQS transformation."""
    # initialization of the NF...

def test_rqs_jacobian_consistency(self):
    """Test that the Jacobian of the forward mapping is consistent."""
    # initialization of the NF...
```

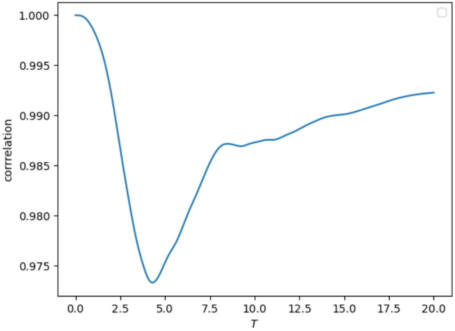
✓ Generated several meaningful new test cases

See more in CC test, refactor test

Test Case: Debug NS Solver (20%)

Task: Debug hand-written spectral solver for 2D Navier-Stokes

- ✓ Figure out the incorrect patterns of the temporal correlation functions
- × Failed to identify incorrect real-valued FFT (rfft) treatment ⟹ Required self-correction based on human feedback
- × Failed to identify incorrect dealiasing treatment ⟹ Required self-correction based on human feedback



Test Case: Hydra + Ray Training Configuration (0%)

Task: Set up Hydra + Ray for multi-run experiments

- Personal codebase with a relative training interface file train_jax.py
- Clear config system that support hydra multi-run experiments
- Integrated with Ray for distributed training (1 run per GPU)
- × Generated overly complex configuration files
- × Cannot configure multiple runs with separate GPUs

Performance Summary

Task Type	**Score**	**Key Factors**
Code Reading	99%	Pattern recognition, documentation
Environment Setup	99%	Standard practices, clear goals
Integration Testing	90%	Clear interfaces, specifications
Complex Debug	20%	Needs domain expertise
Framework Config	0%	Limited training data
Novel Algorithms	N/A	Beyond current capabilities

Key Takeaway:

Claude Code is a powerful amplifier for human developers,
not a replacement for domain expertise

Claude Code as a Tool for Humans

Key Insight: Dramatically lowers barriers for many tasks

Before Claude Code:	**With Claude Code:**
Hours reading documentation	Instant codebase understanding
Manual environment setup	Automated setup scripts
Trial-and-error debugging	Guided exploration
Context switching overhead	Maintained context

Best Use Cases

- Learning new repositories: Navigate unfamiliar codebases
- Environment setup: Docker, dependencies, configuration
- Boilerplate generation: Tests, documentation, CI/CD
- Refactoring: Systematic code improvements

Personal Thoughts

Optimization over the space of orthogonal matrices

$$\mathbf{X} \xrightarrow{\text{qr}} \mathbf{QR} \rightarrow L(\mathbf{Q}) \qquad (\textit{reparametrizationtrick})$$

$$\mathcal{L}_\lambda(\mathbf{X}) = \mathcal{L}(\mathbf{X}) + \lambda \left\| \mathbf{X}\mathbf{X}^T - \mathbf{I} \right\|_2^2 \qquad \text{(penalty method)}$$

Neural network training with equivariant constraint: special architecture design enforces the equivariance, e.g. e3nn; Penalty method or data augmentation

Desired properties of LLM agents: Predefined workflow for agent system; Prompt engineering

Thank You!

Questions?