

Using Large Language Models for Bioinformatics

Harvard FAS Bioinformatics Virtual Training
March 12, 2026

- (1) A quick overview of the AI landscape
- (2) Example use cases for bioinformatics
- (3) Limits, caveats, and challenges

The AI Landscape in March 2026

 **OpenAI****ANTHROPIC** **Meta**

Claude Opus

Claude Sonnet

GPT 5.4

GPT 5.4 Thinking

GPT 5.4 Codex

Gemini 3.1 **ChatGPT**

Claude Code

NotebookLLM

Codex

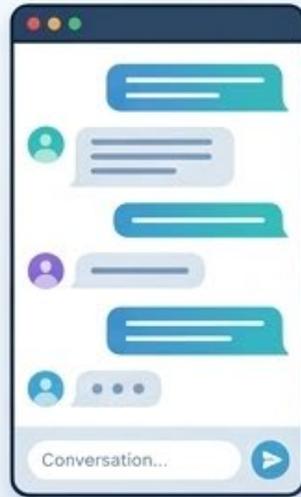
Antigravity

THREE WAYS TO INTERACT WITH AI

1. CHATBOT



NATURAL DIALOGUE



2. TOOL USER / SANDBOX



Analyze this data and create a plot

DATA INPUT



CODE EXECUTION



3. AGENTIC AGENT



Build a web application



generate & test



RECURSIVE EXECUTION

AUTONOMOUS SYSTEM



Harvard provides access to generative AI tools

| Tool | Models | Use | Access |
|------|--------|-----|--------|
|------|--------|-----|--------|

Source: <https://www.huit.harvard.edu/ai/tools>

Harvard provides access to generative AI tools

| Tool | Models | Use | Access |
|------------|--|---------|---|
| AI Sandbox | Multiple models from different providers | Chatbot | https://www.huit.harvard.edu/ai-sandbox |

Source: <https://www.huit.harvard.edu/ai/tools>

Harvard provides access to generative AI tools

| Tool | Models | Use | Access |
|------------|---|--|---|
| AI Sandbox | Multiple models from different providers | Chatbot | https://www.huit.harvard.edu/ai-sandbox |
| Gemini | Gemini 3, NotebookLLM, Antigravity (coding agent) | Chatbot, data analysis, agentic coding | https://gemini.google.com/ (g.Harvard account) |

Source: <https://www.huit.harvard.edu/ai/tools>

Harvard provides access to generative AI tools

| Tool | Models | Use | Access |
|--------------------|---|--|--|
| AI Sandbox | Multiple models from different providers | Chatbot | https://www.huit.harvard.edu/ai-sandbox |
| Gemini | Gemini 3, NotebookLLM, Antigravity (coding agent) | Chatbot, data analysis, agentic coding | https://gemini.google.com/ (g.Harvard account) |
| OpenAI ChatGPT Edu | GPT family, Codex (coding agent) | Chatbot, data analysis, agentic coding | https://www.huit.harvard.edu/openai-chatgpt-edu (request access) |

Source: <https://www.huit.harvard.edu/ai/tools>

Benefits of Harvard AI Tools: Privacy and Security

| | | |
|--|---|--|
| <p>Example: Public Information</p> | <p>Example: Unpublished Research</p> | <p>Example: Student Records</p> |
| <p>Level 1: No impact on university operations, research, reputation, or finances.</p> | <p>Level 2: Minimal impact on university operations, research, reputation, or finances.</p> | <p>Level 3: Moderate impact on university operations, research, reputation, or finances.</p> |

OK

AGAINST GUIDELINES

Personal accounts (e.g, free ChatGPT or a personal Google account)

OK

Harvard-provided generative AI resources

Benefits of Harvard AI Tools: Cost

Free ChatGPT
Limited models
No reasoning models
No agentic tools



Harvard Provided Access
Extending thinking models
Deep Research models
Coding agents



| | | | |
|--|---|--|---|
| <p>Free Intelligence for everyday tasks</p> <p>\$0 / month</p> <p>Get Free ↗</p> | <p>Go Keep chatting with expanded access</p> <p>\$8 / month</p> <p>Get Go ↗</p> | <p>Plus Do more with advanced intelligence</p> <p>\$20 / month</p> <p>Get Plus ↗</p> | <p>Pro Full access to the best of ChatGPT</p> <p>\$200 / month</p> <p>Get Pro ↗</p> |
|--|---|--|---|

(OpenAI example; Google pricing and capabilities are similar)

Example Use Cases for Bioinformatics

(1) “Debugging bioinformatics” with chatbots

(2) Interactive data analysis

(3) Agentic coding for tool and workflow development

(1) “Debugging bioinformatics” with chatbots

(2) Interactive data analysis

(3) Agentic coding for tool and workflow development

Scenario 1: How do I do X?

TS

Tim Sackton

I have a fasta file of a genome that I downloaded from NCBI. I downloaded the "masked" version. I need a hard-masked version for my analysis, but I'm not sure the difference between "masked" and "hard masked". Can you explain?



Scenario 1: How do I do X?



GPT-5.2

In NCBI genome downloads, “masked” usually means the assembly has had **repeat regions identified** (e.g., interspersed repeats, low complexity), and those regions have been **marked in the FASTA** in one of two ways:

Soft-masked (often what NCBI labels simply as “masked”)

- The **actual bases are still present**, but they’re written in **lowercase**.
- Example:
`ACGTTGCAaaaattttGGCC`
Here, `aaaatttt` is the masked (repeat/low-complexity) part.
- Why it’s useful:
 - You *retain* the underlying sequence information.
 - Many aligners/annotators can be told to **treat lowercase as repeats** (ignore them for seeding, down-weight them, etc.), while still allowing alignment through them if needed.

Hard-masked

- The masked regions are **replaced**, typically with `N` (sometimes `X`, but `N` is most common for genomes).
- Example:
`ACGTTGCANNNNNNNGGCC`
- Why it’s useful:
 - The masked sequence is effectively “removed” from consideration because it’s now unknown.
 - Some tools/pipelines *require* this because they don’t interpret lowercase masking and will otherwise treat repeats as normal sequence.

How to tell which you currently have

Open your FASTA and look at the sequence characters:

- If you see **lowercase a/c/g/t**, that’s **soft-masked**.
- If masked parts are mostly **N runs** where you expect repeats, that’s **hard-masked**.

Gives some text, explaining things.

Tells me what to look for.

Scenario 1: How do I do X?

TS

Tim Sackton

how do i tell if there are lowercase bases in the fasta file? it is too big to look at



GPT-5.2

You can check for lowercase **without opening the whole file** by searching for any `a/c/g/t` in **non-header** lines.

Fast "does it contain lowercase?" check (Linux/macOS)

bash

Copy code

```
grep -n -m 1 -E '^[^>]*[acgt]' genome.fa
```

- If it prints a line, you **have lowercase bases** (soft-masked).
- If it prints nothing (exit code 1), there may be **no lowercase** (or it's hard-masked / unmasked).

If the FASTA is gzipped (.gz)

bash

Copy code

```
zgrep -n -m 1 -E '^[^>]*[acgt]' genome.fa.gz
```

Outputs code to run.

If you don't understand code, ask AI to explain before you cut and paste!

Scenario 1: How do I do X?

TS Tim Sackton

can you explain what this does? `grep -n -m 1 -E '^[^>]*[acgt]' genome.fa`

< 2 / 2 >

GPT-5.2

That command is a quick way to ask: "Does this FASTA contain any **lowercase** bases in the **sequence lines** (not the header lines)? If yes, show me the first place."

The command

bash

Copy code

```
grep -n -m 1 -E '^[^>]*[acgt]' genome.fa
```

What `grep` options mean

- `-E`: use *extended* regular expressions (so the pattern syntax is a bit richer; here it mainly just makes the regex style standard).
- `-n`: print the **line number** of the match (useful for locating it).
- `-m 1`: stop after **1 match** (so it's fast on big files—`grep` quits as soon as it finds the first lowercase base).

What the regex means: `^[^>]*[acgt]`

Breakdown:

- `^` = start of the line
- `[^>]*` = "zero or more characters that are **not** `>`"
 - This effectively ensures the line does **not start with** `>` (FASTA headers start with `>`).
 - Also, in normal FASTA sequence lines, there usually is no `>` at all, so this just means "scan along the sequence line".
- `[acgt]` = at least one **lowercase** base (a, c, g, or t)

You can ask more if you don't understand the explanation.

AI can make mistakes but it often corrects itself if you ask it to explain.

Scenario 1: How do I do X?

TS Tim Sackton

okay i ran it and it gave output like you showed. so my sequence is soft masked. i need to convert to hard masked now

 **GPT-5.2**

To hard-mask a **soft-masked** FASTA, you typically **replace any lowercase base with N**, while leaving headers (`>...`) unchanged.

Convert soft-masked → hard-masked (recommended)

bash

 Copy code

```
perl -pe 'if(!/^>/){ s/[a-z]/N/g }' genome.fa > genome.hardmasked.fa
```

What it does:

- `if(!/^>/)` = only modify **non-header** lines (sequence lines)
- `s/[a-z]/N/g` = replace **any lowercase letter** (a-z; includes lowercase IUPAC ambiguity codes like `n`, `r`, `y`, etc.) with `N`

Now it is giving some explanation with its code.

You can still ask it to explain more.

Scenario 1: How do I do X?

On a problem like this, most frontier AI models are pretty consistent.

But if you are not sure about the command it provides, you should still ask it to verify.

I asked this question about 10 times to different models. Nine out of ten times I got correct code. One time I got this.

```
awk '/^>/ {print; next} {print toupper($0) | "sed s/[ACGT]/N/g"}' input.fa
```

Scenario 1: How do I do X?

TS Tim Sackton

can you explain what this does? `awk '/^>/ {print; next} {print toupper($0) | "sed s/[ACGT]/N/g"}' input.fa`

Net effect on `input.fa`

- Header lines (starting with `>`):
 - Printed exactly as in the input.
- Sequence lines:
 - First forced to uppercase.
 - Then all occurrences of A, C, G, T are replaced with `N`.
 - So a line like:
 - `acgtacgtRYN` → `ACGTACGTRYN` in `toupper` → `NNNNNNNNRYN` after `sed`.

So overall, it **passes FASTA headers through unchanged** and **mutates each sequence line so that any A/C/G/T (in any case) becomes N** in the output.

Model response

Scenario 1: How do I do X?

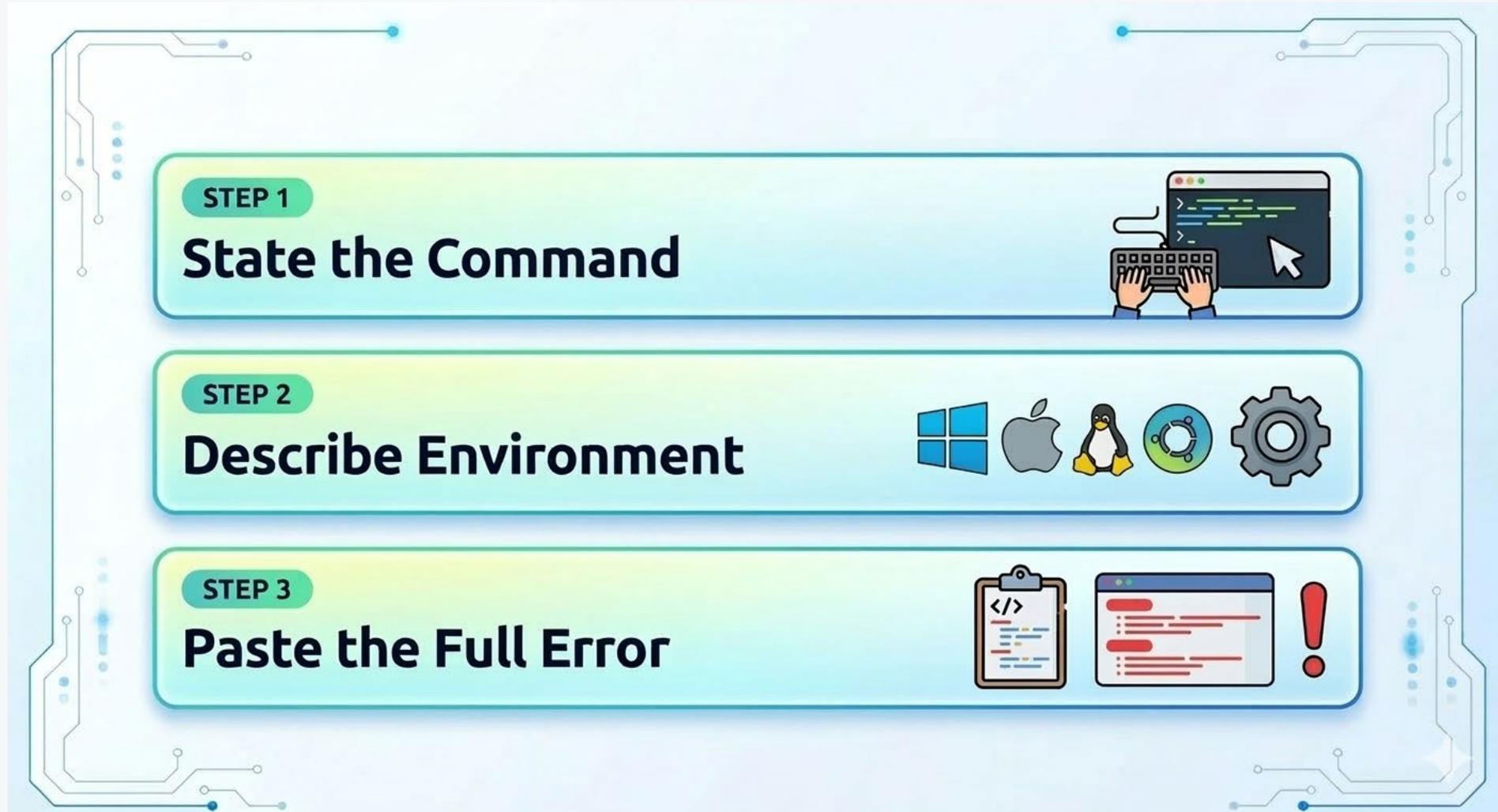
I want to change all the sequencing files in my raw data directory to be read only, but not the other files. This is on my university HPC cluster

How can I combine a bunch of phylogenetic models (*.mod files) produced by phyloFit in the PHAST package into a single average mod file?

I need to remove UMIs from about 20 bam files I made by aligning RNA-seq data to my reference genome with STAR. This is on my university HPC slurm cluster.

I have 10000 fasta alignment files in a bunch of directories, called things like gene1/alignment.fasta, gene2/alignment.fasta, gene3/alignment.fasta. I need to copy them all to the alignments directory with names like gene1.fasta, gene2.fasta, gene3.fasta.

Scenario 2: What does this error message mean?



Scenario 2: What does this error message mean?

```
(bioinf) tsackton@holybioinf testing % bedtools merge -i bedtest.bed
```

```
ERROR: file bedtest.bed has non positional records, which are only valid  
for the groupBy tool.
```

```
...
```

Scenario 2: What does this error message mean?

I ran a long pipeline with many steps and it failed somewhere in the middle.

The error log is long and I don't know where to start...

Other Chatbot Comments

Good for helping generate a one-liner, explaining code snippets, Bash scripting, finding the right tool for a task.

Accuracy is not 100%, though at least the frontier models with reasoning effort hallucinate much less than they did even a few months ago. You need to develop an intuition for when the AI is on the wrong track.

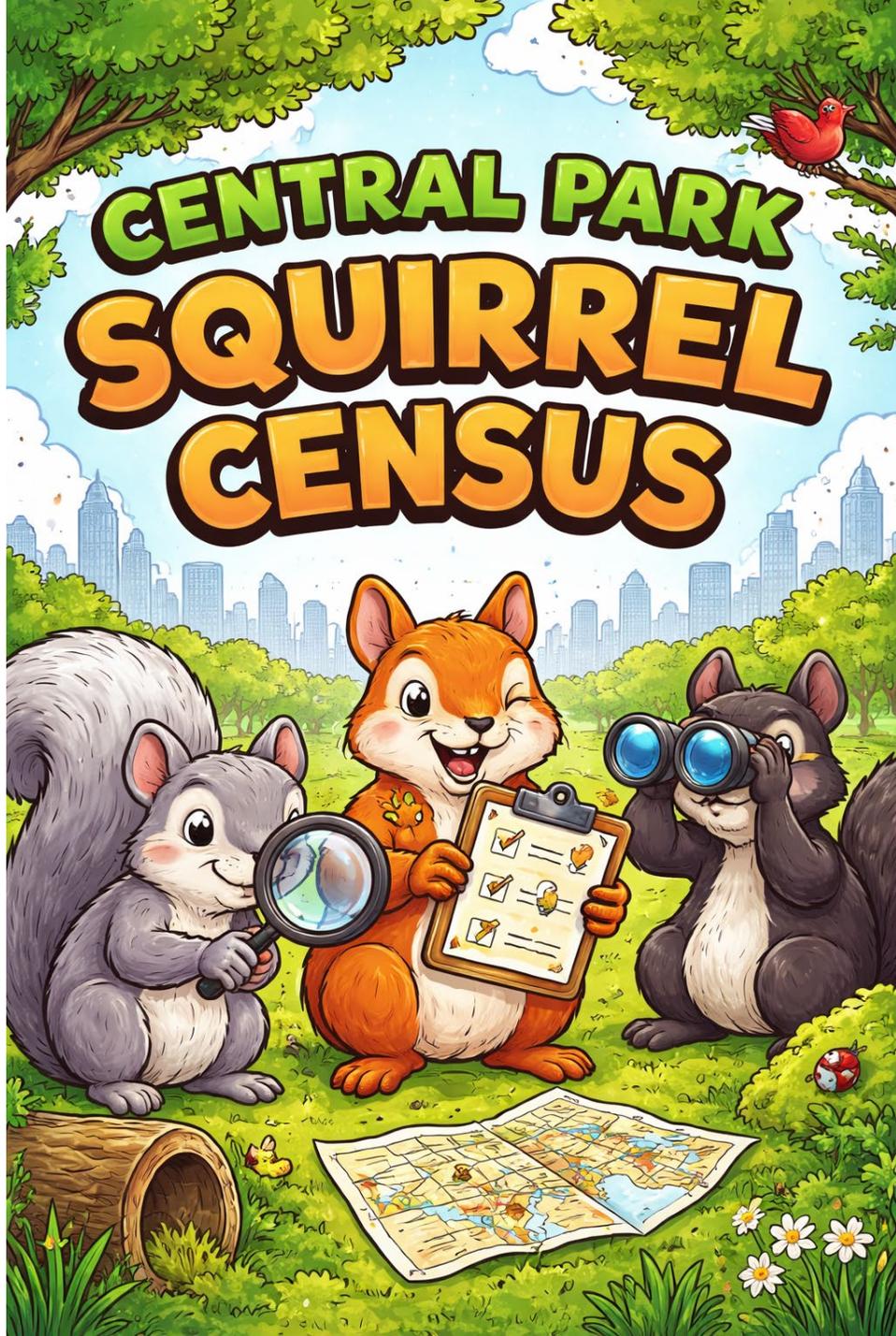
LLMs can be weirdly “human-like” in their responses, and weirdly “not-computer-like”. For example, sometimes things like “can you double-check that”, or “please show me your sources for that” will fix mistakes. We are not used to having to tell computers to double-check their work.

For complex data analysis or coding tasks, it is better to use tools that can run code, either in a sandbox or on your computer.

(1) “Debugging bioinformatics” with chatbots

(2) Interactive data analysis

(3) Agentic coding for tool and workflow development



Location, color, and many other observations on squirrel sightings in Central Park in 2018

(1) “Debugging bioinformatics” with chatbots

(2) Interactive data analysis

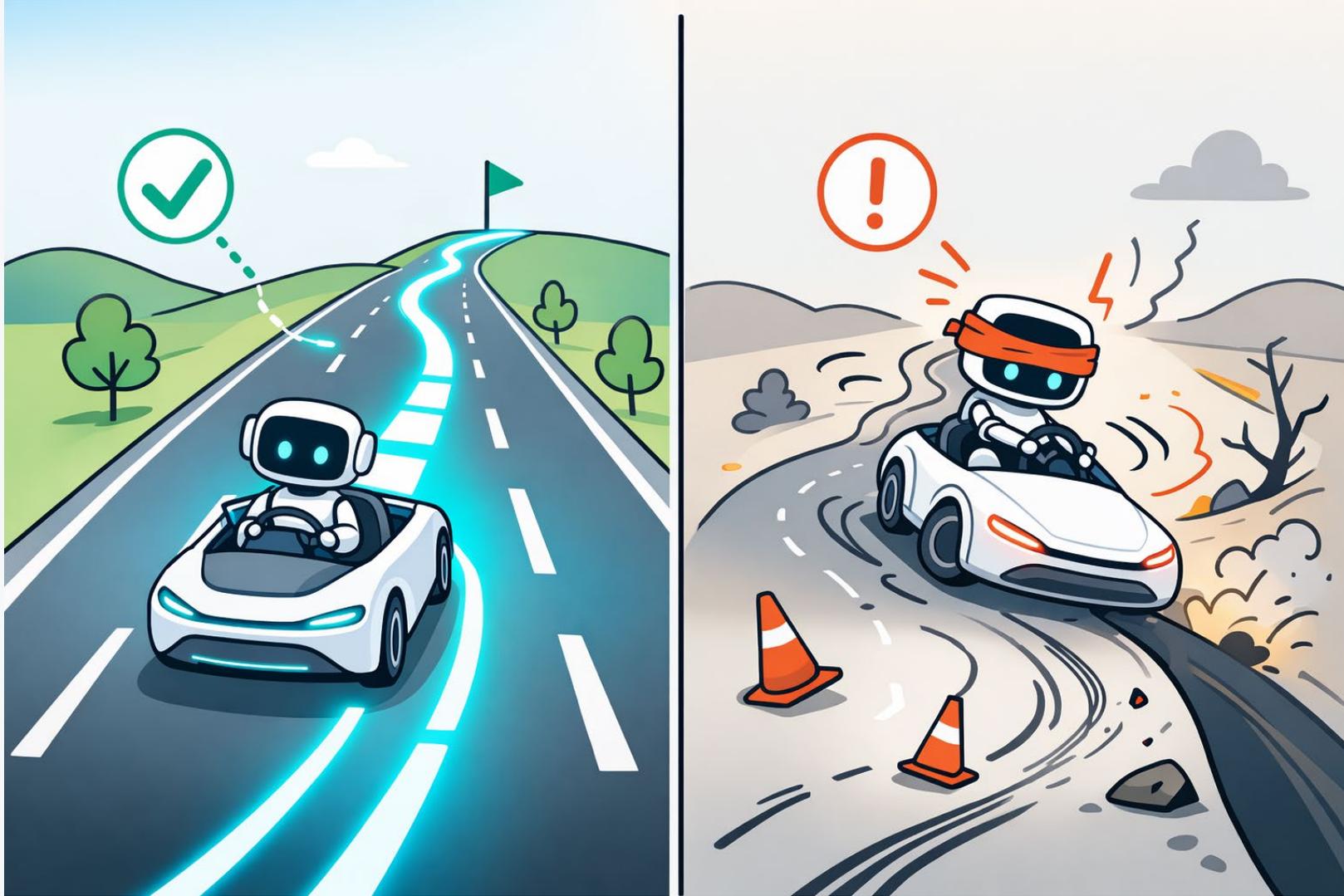
(3) Agentic coding for tool and workflow development



If you spend much time in certain corners of the internet, or know any software developers, the current AI moment can feel like a rocket taking off.

Perhaps we are blasting off for the Moon and beyond, but it it also feels like the rocket is still half-finished.

Point #1: Test-Driven Development



Agentic tools do better when they can verify their outputs.

Point #1: Test-Driven Development

A simple test: an input file and an output file that you know are correct.

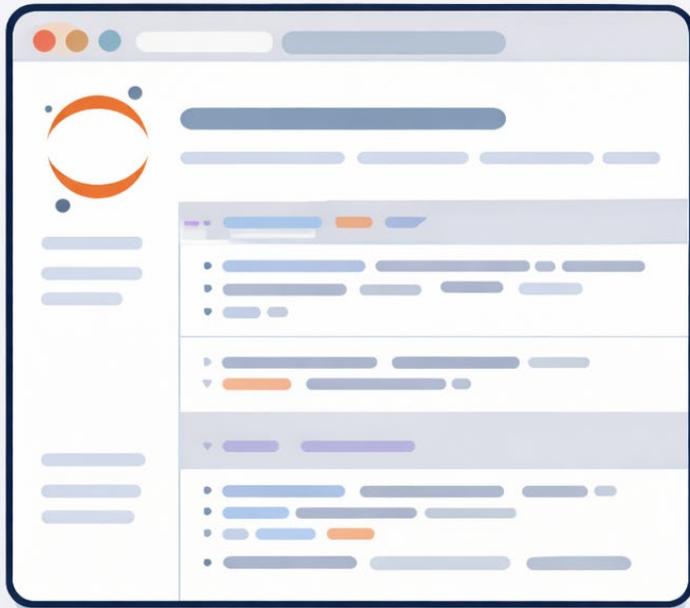
More complicated: specific validated input/output pairs for specific functions.

Agents can bootstrap tests!

If you are fixing a bug: make a reproducible example first!

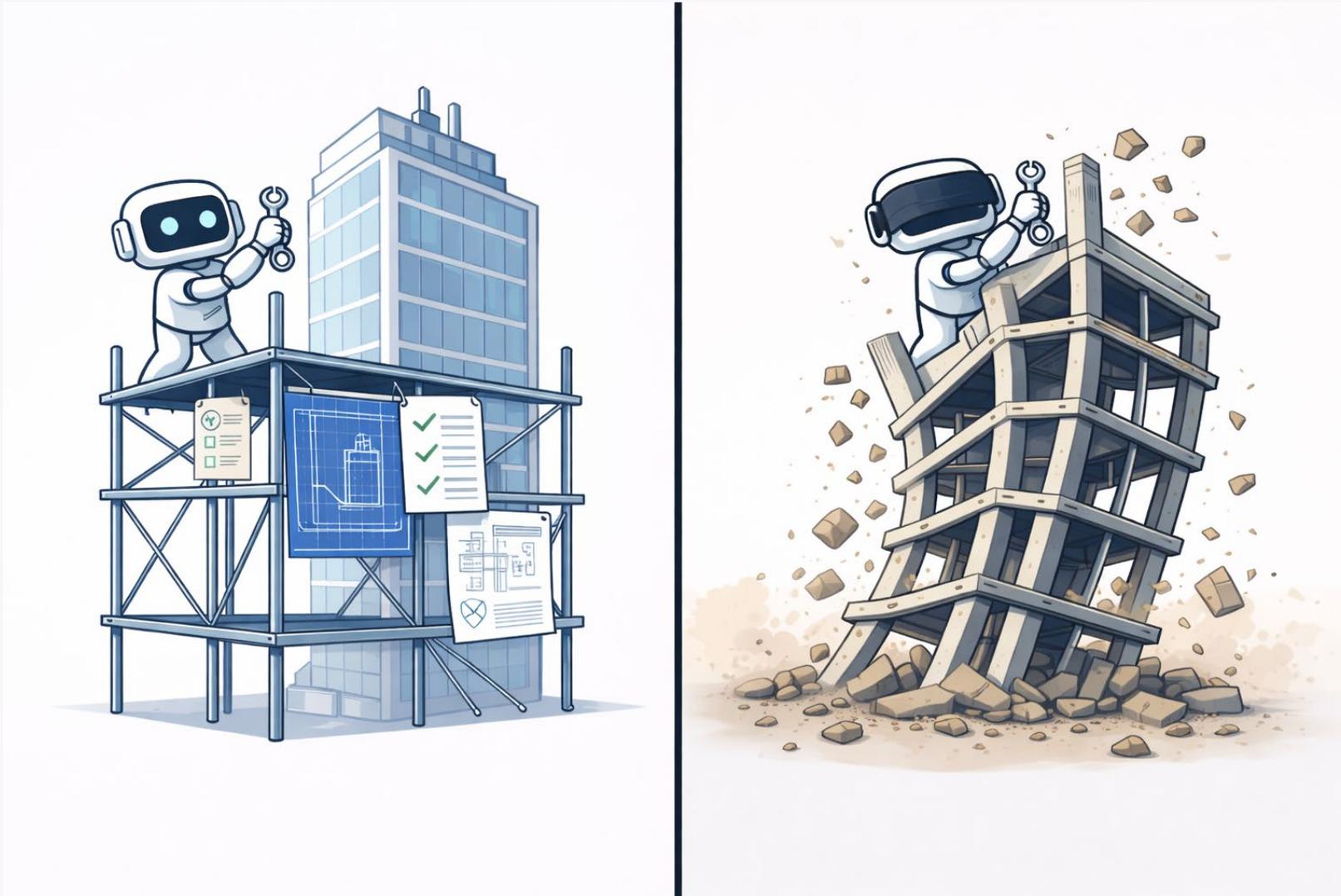
Verify outputs! Code that runs can still be wrong.

Example: Refactor a Jupyter Notebook into a script



```
$ ./program --option1 value1  
--option2 value2  
-
```

Point #2: Plan First, Then Implement



Ask an AI to explain what it is going to do before it does it.

Check the logic.

Ask it to check the logic.

Have a different model check the logic.

Example: Convert Loose Notes to a Series of Scripts

My RNA-seq analysis:

- Trim reads
- Map to transcriptome. Maybe with salmon or STAR?
- Compute count matrix
- Run DESeq2 in R



My RNA-seq analysis:

```
trim_reads.sh  
map_reads.sh  
compute_counts.sh  
analyze_matrix.R
```

[Though for this specific use case, consider `nf-core/RNA-seq!`]

Point #3: Experiment

can you build a tool that pulls preprints published today on biorxiv about bioinformatics and generates a nicely formatted html page with titles, authors, and abstracts for each one?

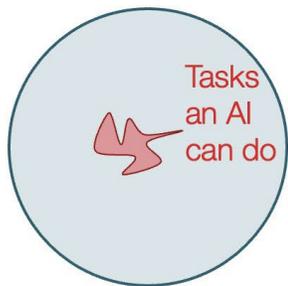


I'll implement this directly in the repo: first I'm going to inspect the current project layout and pick the best place for a small CLI script that fetches today's bioRxiv bioinformatics preprints and renders an HTML report.

can you make a game that pulls a bird image from the Macauly Library, gives the user the image, the location, and the season it was seen, and then runs an identification quiz? easy mode should give the user four possible choices, hard mode should force the user to type

General Limits, Caveat, Challenges

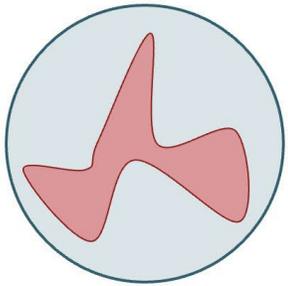
"The AI is a fun toy."



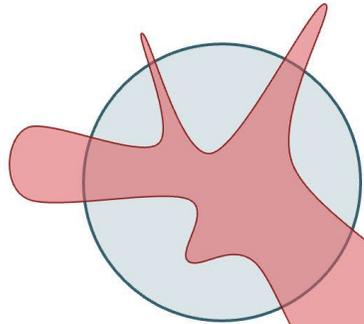
Tasks of a human job



"The AI is helping me in some tasks."

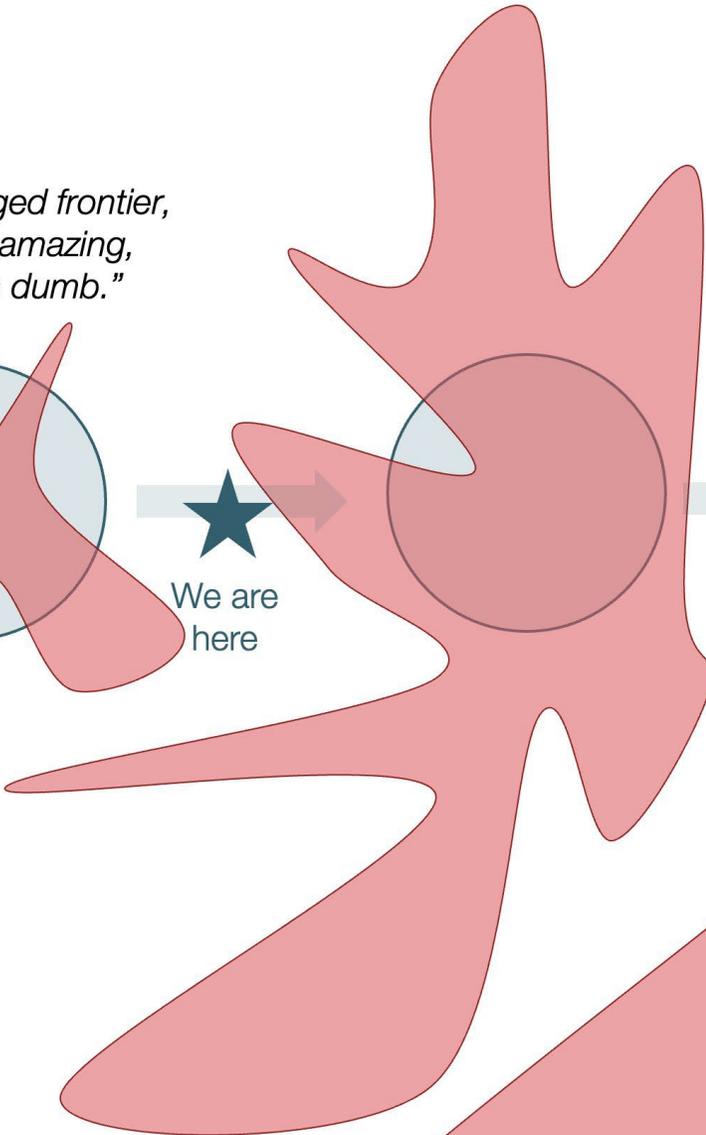


"The AI has a jagged frontier, sometimes it's amazing, sometimes it's dumb."

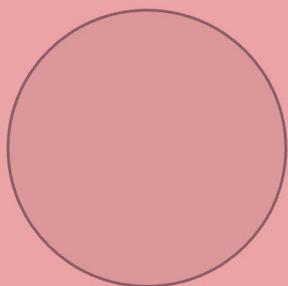


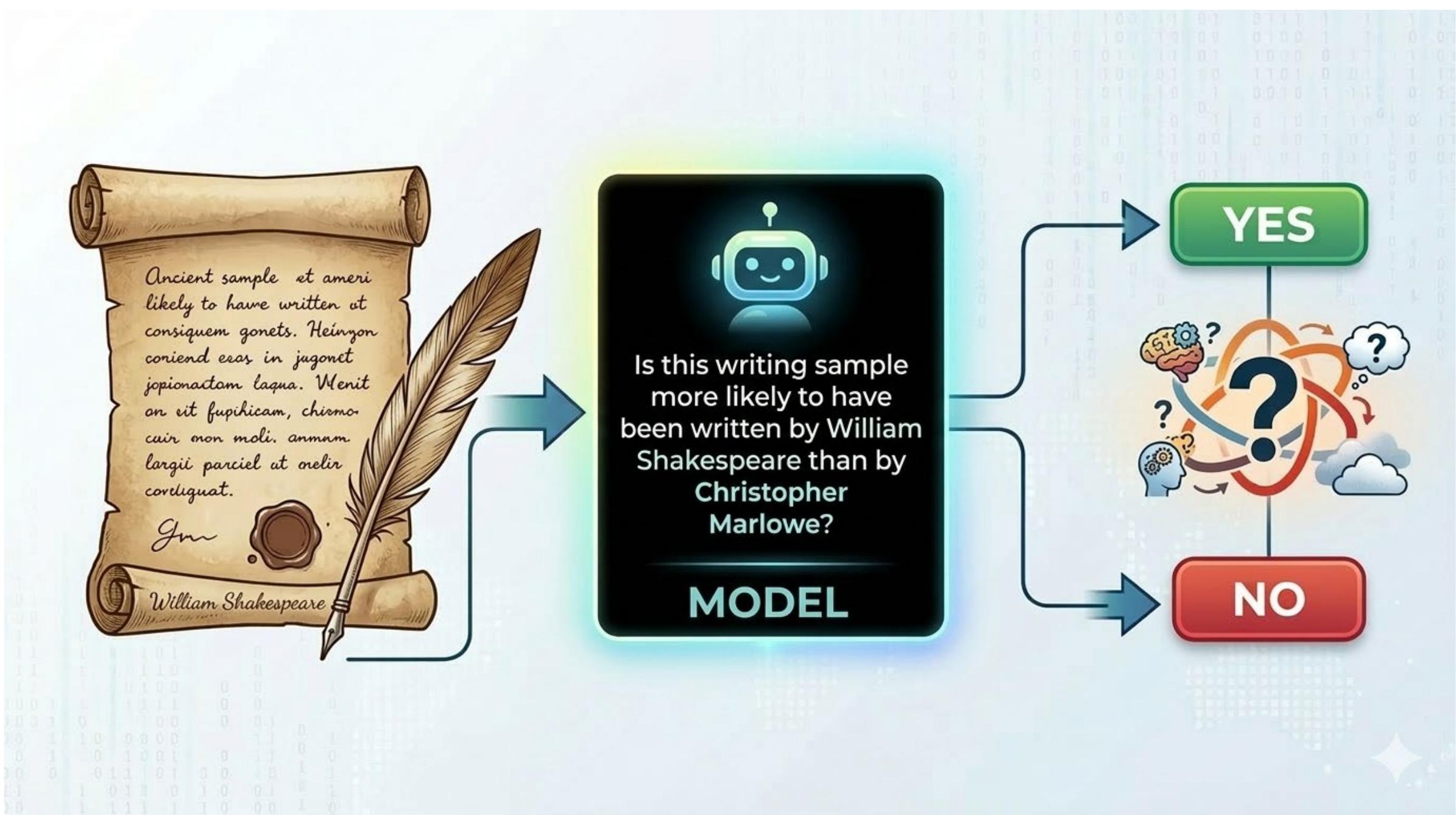
We are here

"The AI is unbelievably intelligent but for some reason it fails at X."

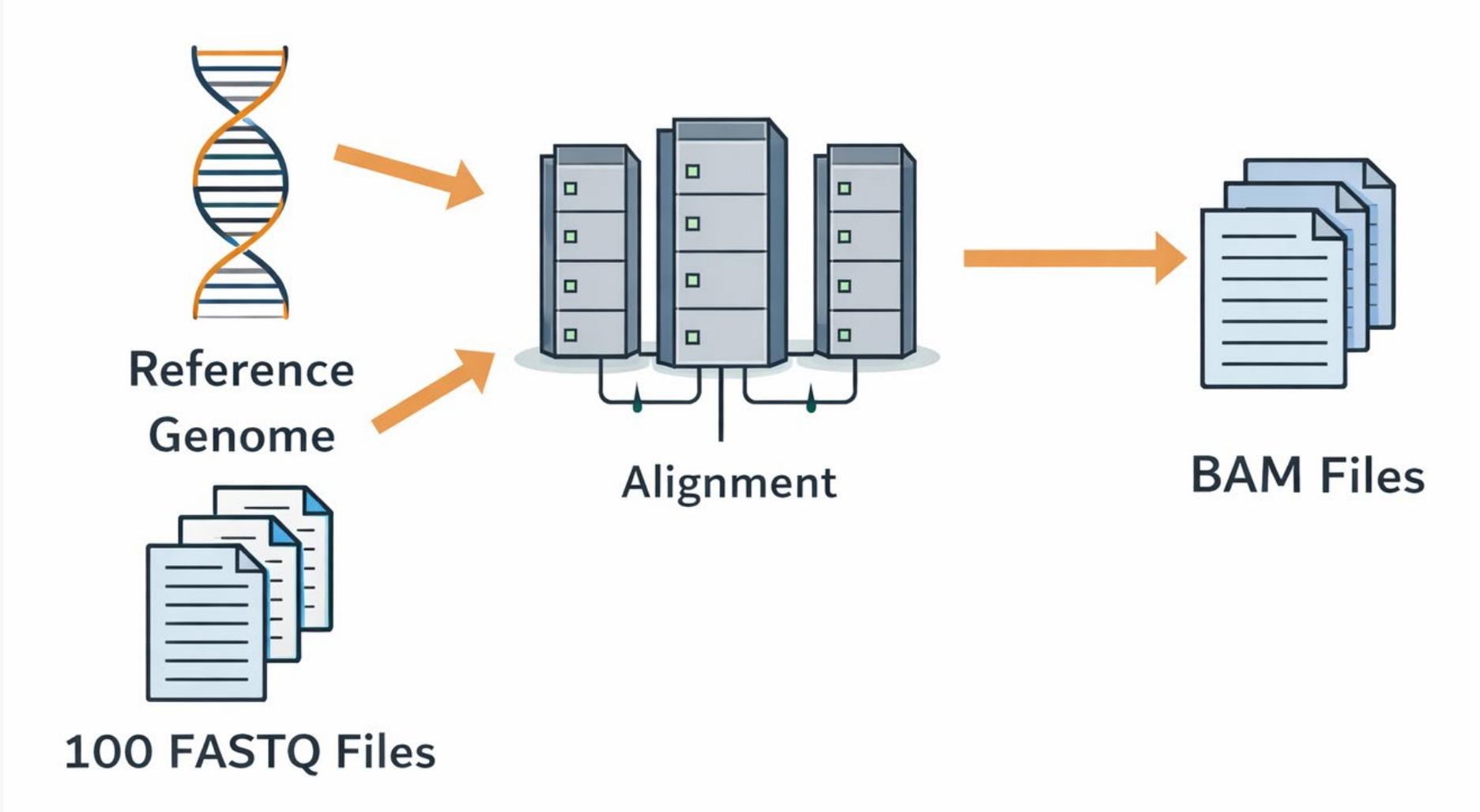


AGI

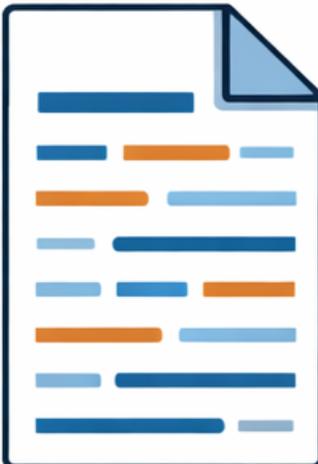




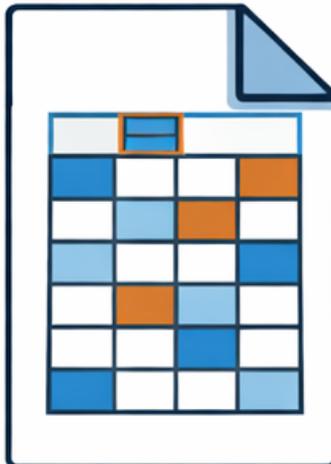
Bioinformatics-Specific Challenges: Heavy Compute



Bioinformatics-Specific Challenges: Big Files



BAM



VCF



FASTA



FASTQ

Bioinformatics-Specific Challenges: Big Files



Workshop – In Person

Genomics on the command line

April 2, 9, 16, 23

9:30 AM - 12:30 PM

informatics.fas.harvard.edu



**Using OMERO for Imaging Workflows: A Full-Day Virtual
Hands-On Workshop**

**March 27 9 AM - 3 PM
Virtual**

informatics.fas.harvard.edu





Intro to scRNA-seq part 1 zoom reg
March 19 11 AM - 12 PM



Intro to scRNA-seq part 2 zoom reg
March 26 11 AM - 12 PM

Conclusions and Looking Forward