

From Orchestration to Capability

Building Framework-Free Scientific Agent Systems with Portable Units of Knowledge

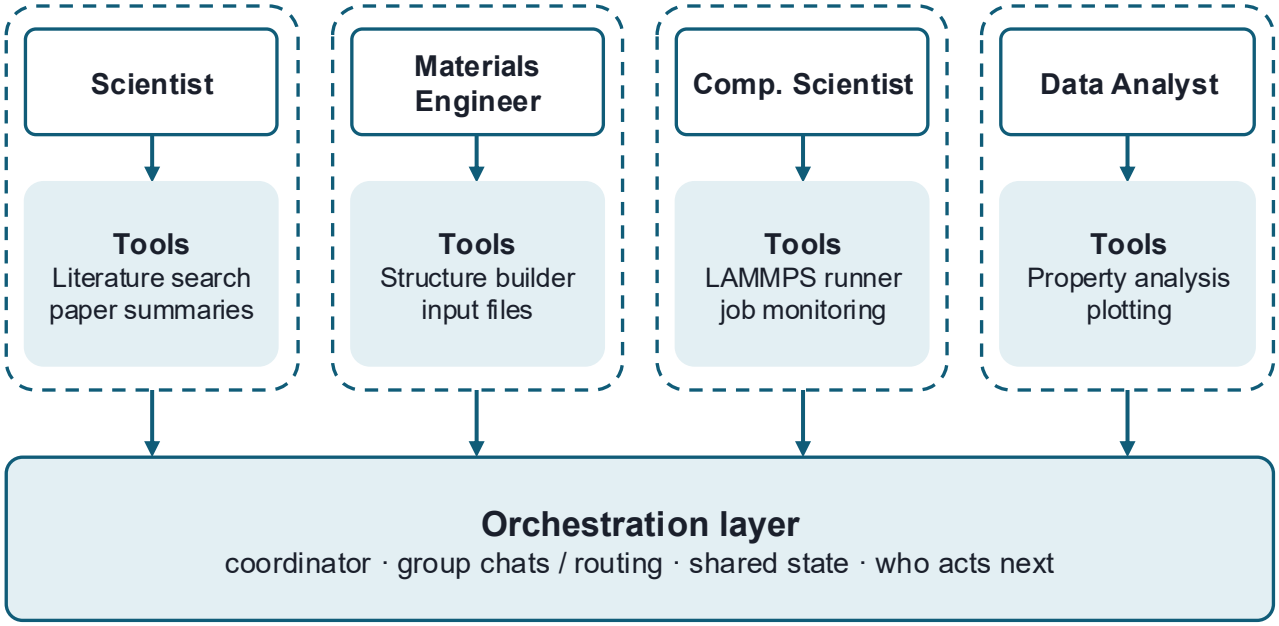
Shray Mathur

Center for Functional Nanomaterials, Brookhaven National Laboratory

New York Scientific Data Summit 2026

@BrookhavenLab

Personas, tools and orchestration

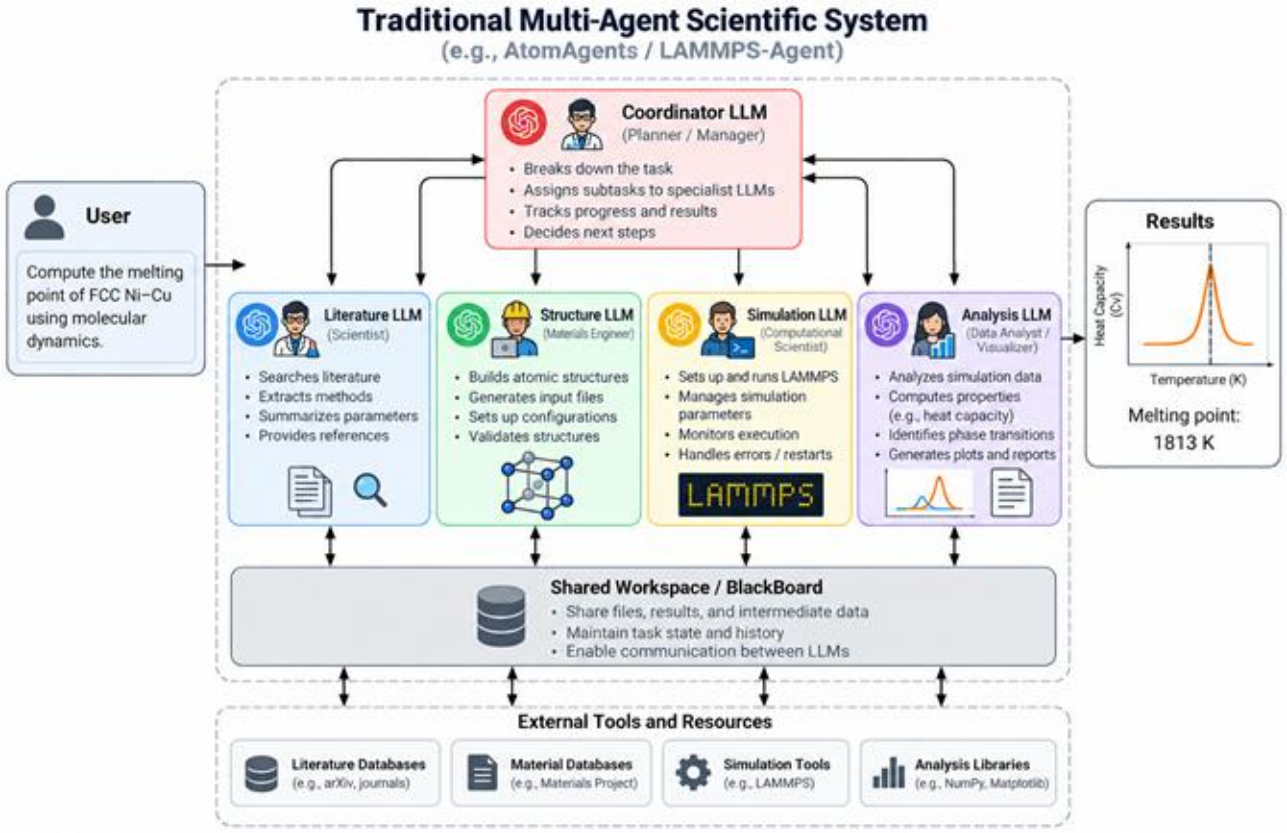


Personas: one narrow role per agent

Each agent gets a few tools for its narrow task

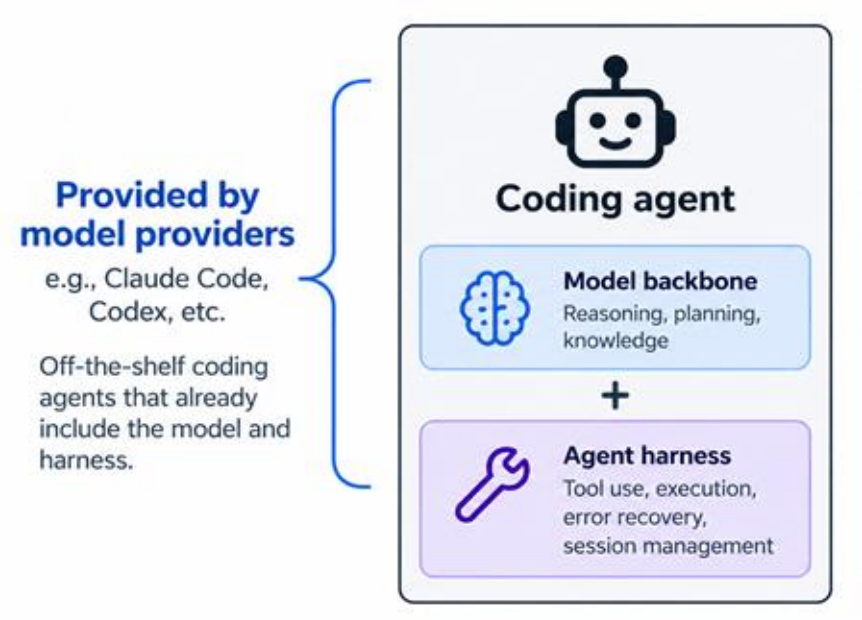
Combines the narrow agents and their tools into one system

How scientific multi-agent systems are built today



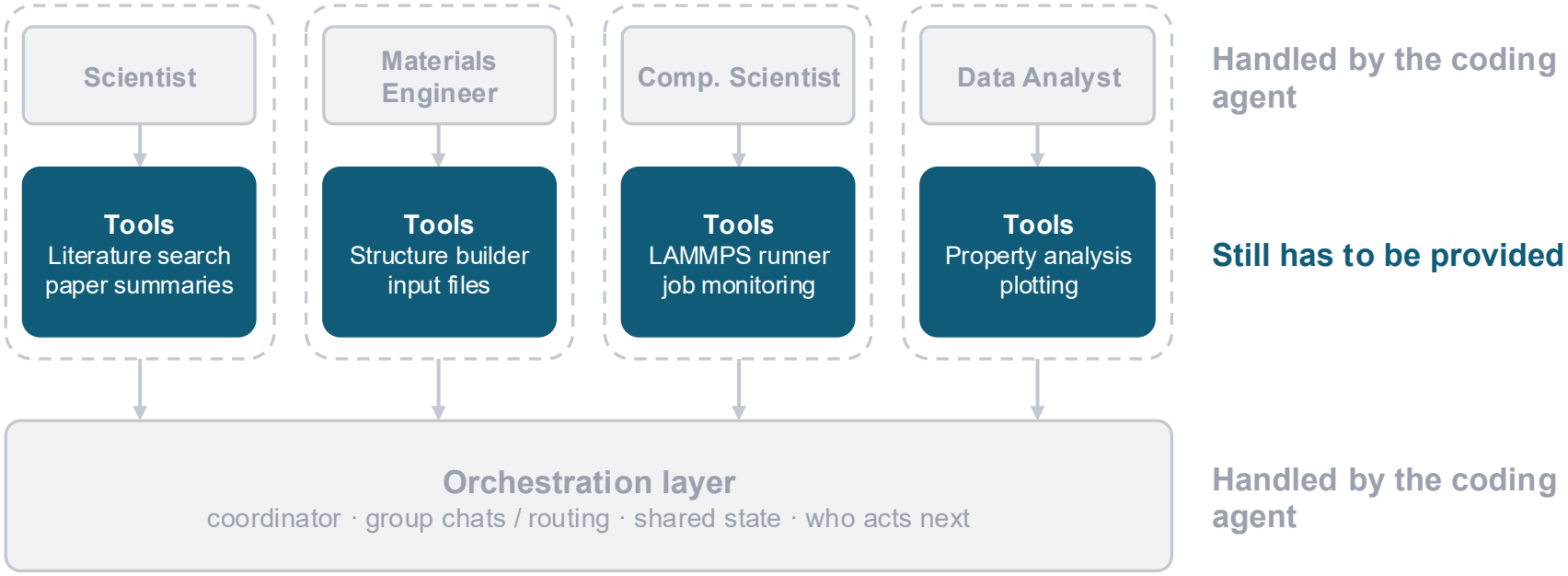
- Specialized LLM agents, each given a persona: Scientist, Materials Engineer, Data Analyst
- A coordinator and shared workspace route work between them
- Built by hand or with frameworks (AutoGen, LangGraph)
- **Much of the effort goes into coordination, not science**

Coding agents make orchestration and personas redundant



- Claude Code, Codex and Hermes Agent combine a model with a harness that plans multi-step tasks, selects and sequences tools, recovers from errors and manages long sessions
- That is what the orchestration layer was built to do
- Personas are no longer needed: one session holds reasoning that used to be split across roles

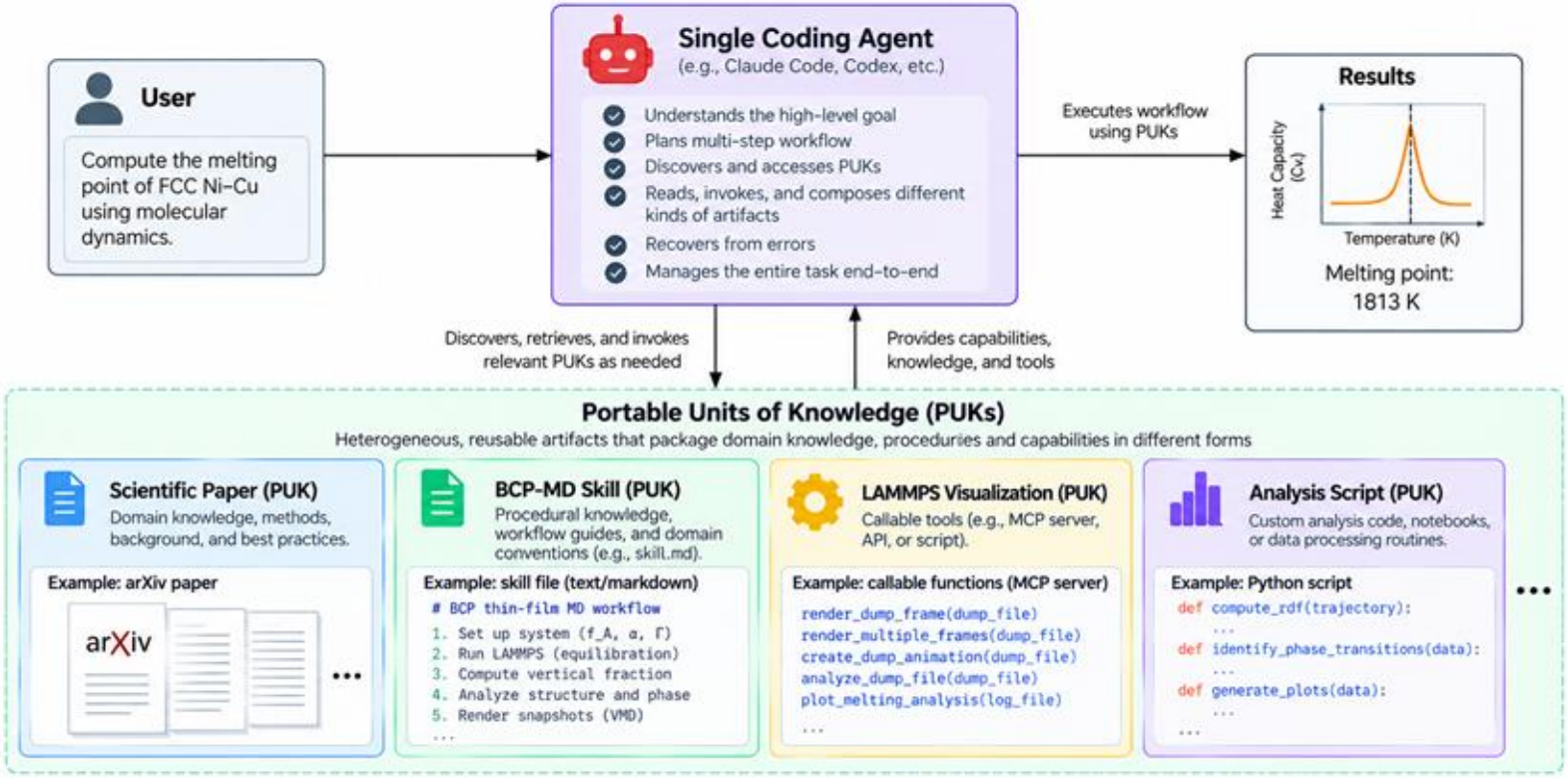
But the scientific capabilities still have to be supplied



- **Domain-specific context:** procedures, conventions and publications that describe how the science is done
- **Task-specific capabilities:** simulation routines, analysis scripts, instrument-control functions

Portable Units of Knowledge (PUKs) bridge the gap

Framework-agnostic, directly invocable packages of scientific knowledge. **No fixed form:** anything the agent can't be expected to work out on its own.



The claim, and what this talk shows

Modern coding agents make hand-built orchestration largely redundant.
What they still need is the science, packaged as Portable Units of Knowledge (PUKs).

Part I: port existing systems
Strip the orchestration from four published multi-agent systems; one coding agent with only their scientific capabilities reproduces every result

Part II: build from scratch
A coding agent builds PUKs from a researcher's paper and data; we compare them to the researcher's procedure, then evaluate the outputs

Key takeaways

- **Build PUKs, not orchestration**
- **Delegate more, not less**
- **Spend supervision on grounding**

Part I: Porting existing systems

Extract the scientific capabilities as PUKs, connect them to a coding agent, reproduce the results

Four systems, orchestration removed

Result up front: with all orchestration removed, one coding agent reproduces every original result. Only the scientific capabilities were kept, as MCP tools; one prompt per experiment.

System	Domain	Original coordination	Agents	Tools in → out
AtomAgents	Alloy design, LAMMPS	AutoGen, nested group chats	14	22 → 12
LAMMPS-Agents	Classical MD, LAMMPS	AutoGen, dynamic speaker selection	11	47 → 31
DREAMS	DFT, Quantum ESPRESSO	LangGraph, supervisor state graph	4	21 → 20
VISION	Synchrotron beamline	Custom classifier-routed pipeline	6	37 → 37

Three frameworks, four coordination topologies. In AtomAgents, half of the 22 functions only routed work between agents.

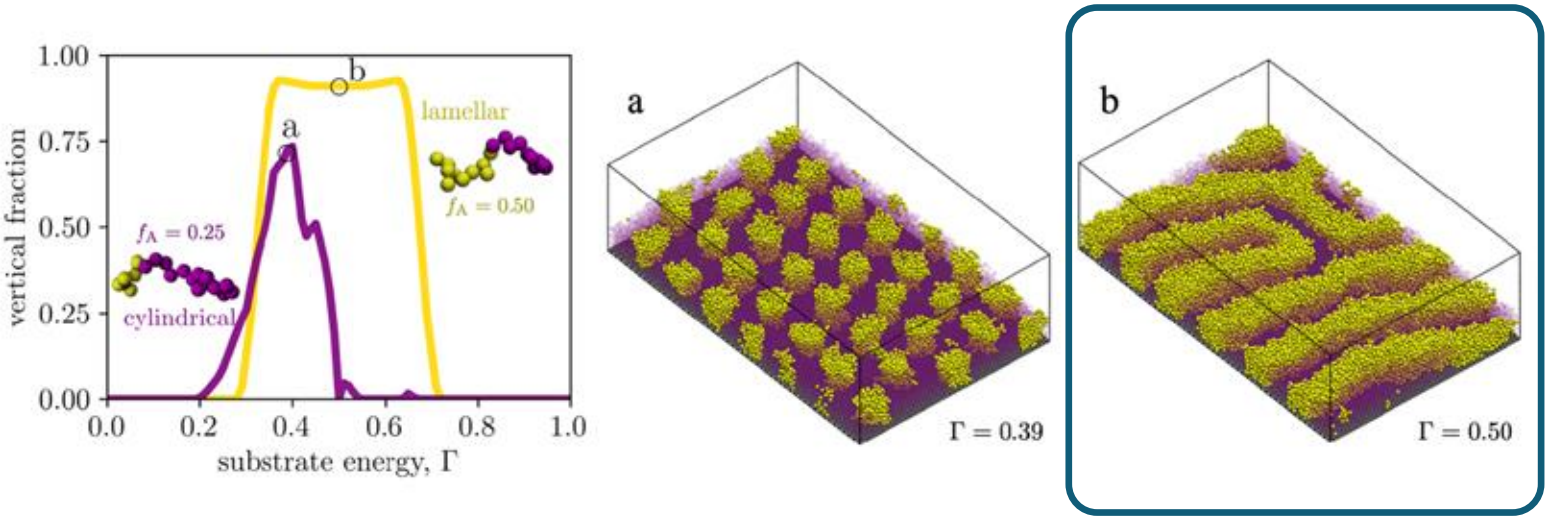
One agent reproduces all four original results

System	Reproduced experiment	Original	Single coding agent
AtomAgents	Screw dislocation core in W, two EAM potentials	Compact vs. polarized core	Same classification
LAMMPS-Agents	Melting point of FCC Ni–Cu by solid–liquid coexistence	1814 K (expert 1830 K)	1813 K
DREAMS	Lattice constant of BCC Li, with convergence tests	3.44 Å, 40 Ry, 12×12×12	3.4398 Å, same parameters
VISION	Beamline command from natural language	Published control code	Identical code

Planning, tool sequencing, error recovery and image interpretation all come from the agent itself.

Part II: can the agent build its own PUKs?

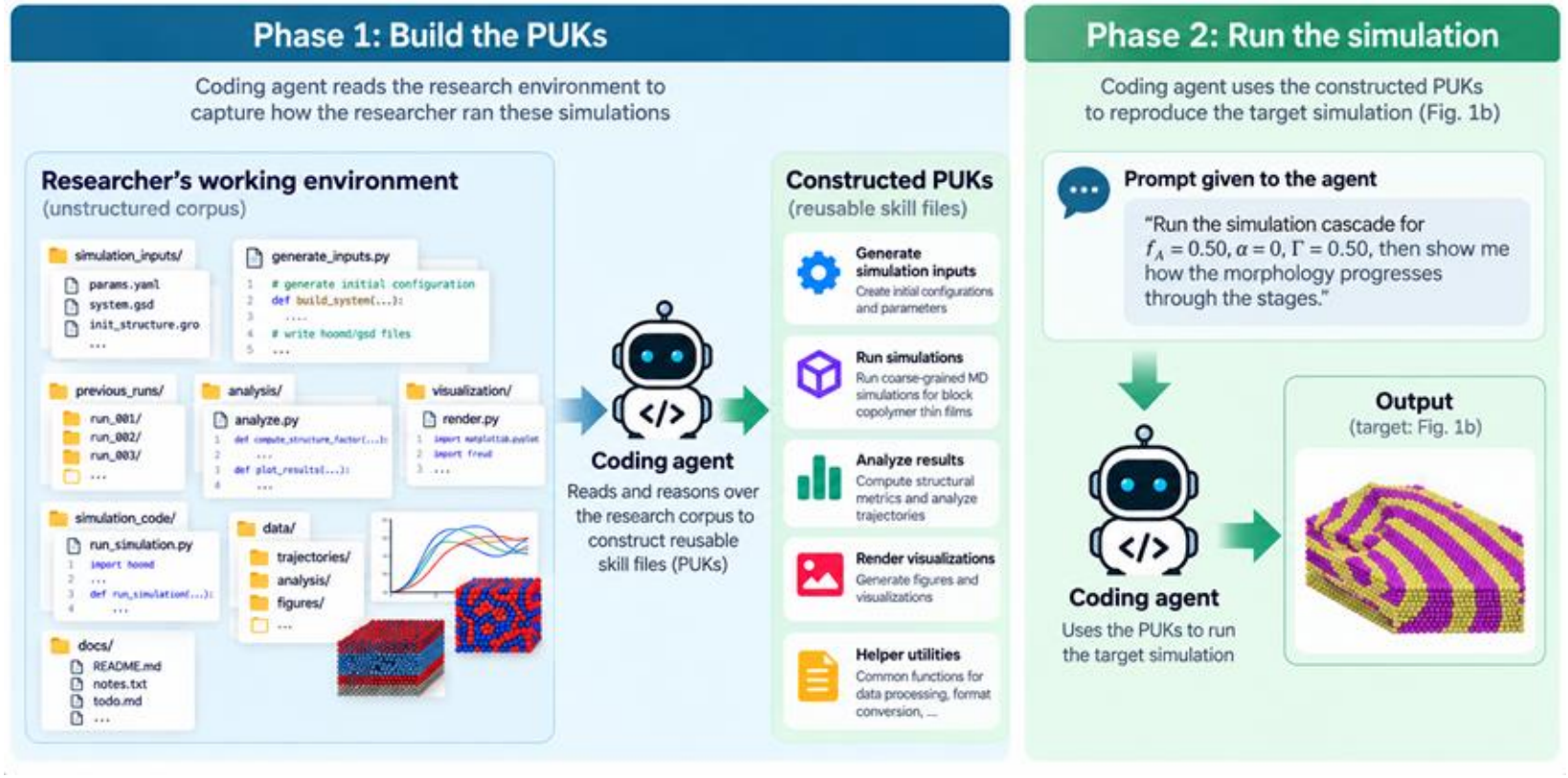
Surface enrichment dictates block copolymer orientation (Nanoscale, 2023): coarse-grained MD of block-copolymer thin films, measuring whether the morphology orients vertically (vertical fraction f_V)



- **Target:** lamellar film, $f_A = 0.50$, $\alpha = 0$, $\Gamma = 0.50$, reported as vertical
- The agent first builds the PUKs it needs to run these simulations, then uses them to reproduce this result

Figure adapted from S. Bae, M. M. Noack and K. G. Yager, Nanoscale 15, 6901 (2023).

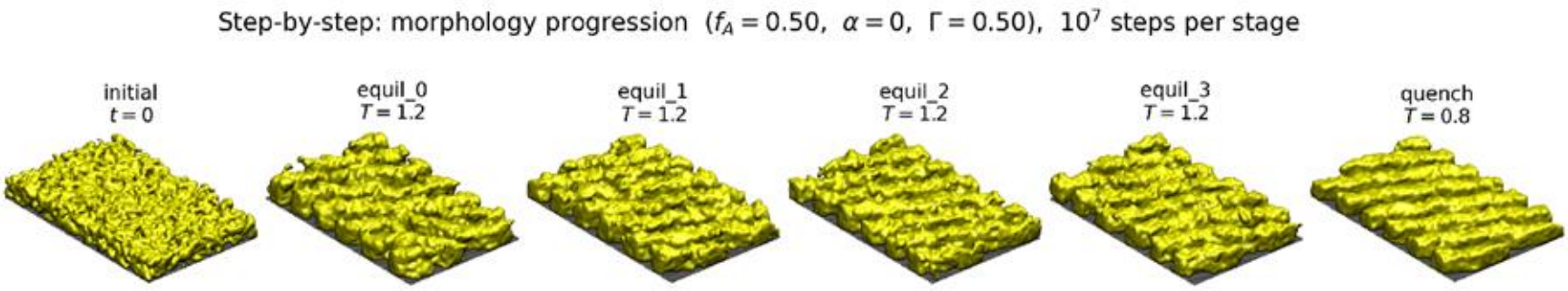
Building PUKs: the coding agent distills a researcher's data



Attempt 1: step by step

We split the workflow into small pieces; the agent built one skill at a time, and we checked each one.

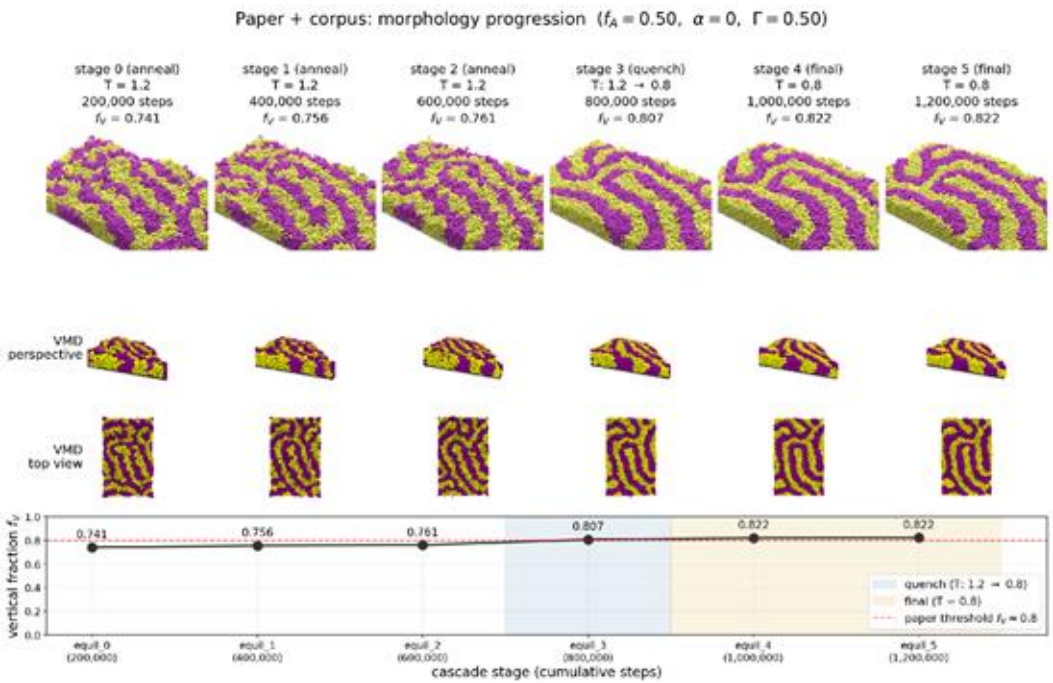
Time to build	Match to researcher's procedure	What it missed
Weeks	**~100%**	**Nothing**



Correct: vertical lamellae, as in the paper. And the agent did everything we asked of it.

Attempt 2: the whole task, with paper + corpus

No decomposition: the whole task in one prompt, with the paper and the researcher's files.



Time to build
30 min

Match to researcher's procedure
~90%

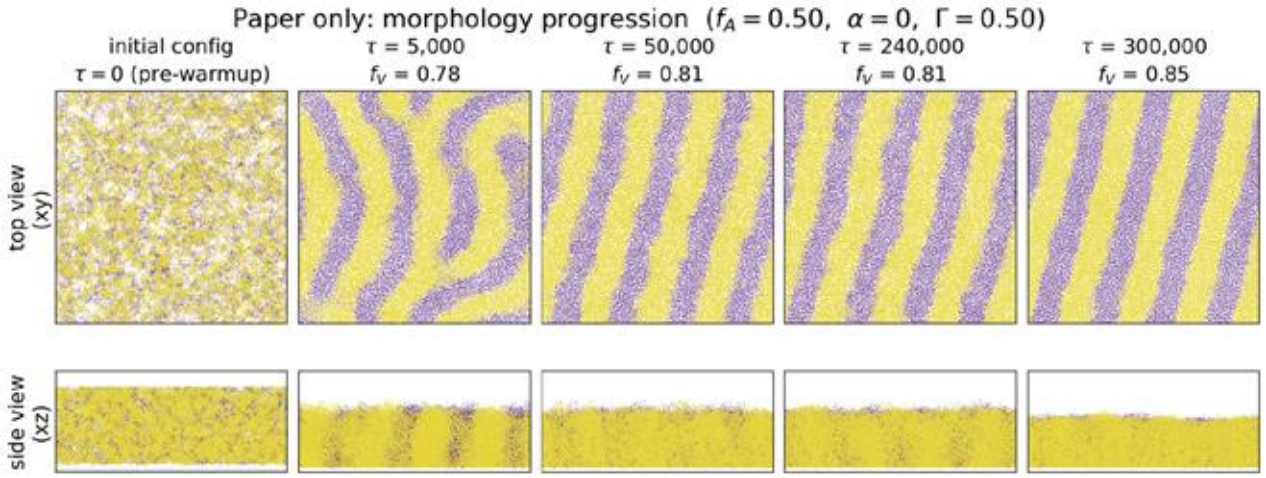
What it missed
Temperature schedule

Correct, and richer than step by step. It explored 96 of the researcher's files and reused their rendering and analysis code.

Attempt 3: the whole task, with the paper only

No decomposition and no files: the whole task with just the paper.

Time to build	Match to researcher's procedure	What it missed
2 hours	**~80%**	**Box size, temperature schedule**



Correct morphology, but not the researcher's visualization: that could only come from the corpus.

Lessons for working with coding agents

- Distilling how a researcher ran a simulation into PUKs is one representative task; the lessons carry over to other work with coding agents
- **Pre-decomposing the task gives control,** but builds in an assumption about the agent's limits and underuses its ability to break down complex work itself
- **Delegate more, not less:** single-pass builds recovered most of the procedure at a fraction of the time and effort
- **Delegation works best with good grounding:** with the researcher's corpus, 30 minutes and one minor error; with the paper alone, 2 hours and more mistakes
- **Spend supervision on grounding,** not on breaking the task down for the agent

Takeaway

Recent advances in coding agents warrant a change in how we build agentic scientific systems.

- Extend existing coding agents with PUKs, rather than engineering new agent roles and workflows around them
- Orchestration and personas were workarounds for earlier model limits; the lasting effort is in building good PUKs
- Hand-built orchestration still fits high-stakes workflows that need a guaranteed, verifiable sequence of steps, or tasks too large for one agent

Thank you