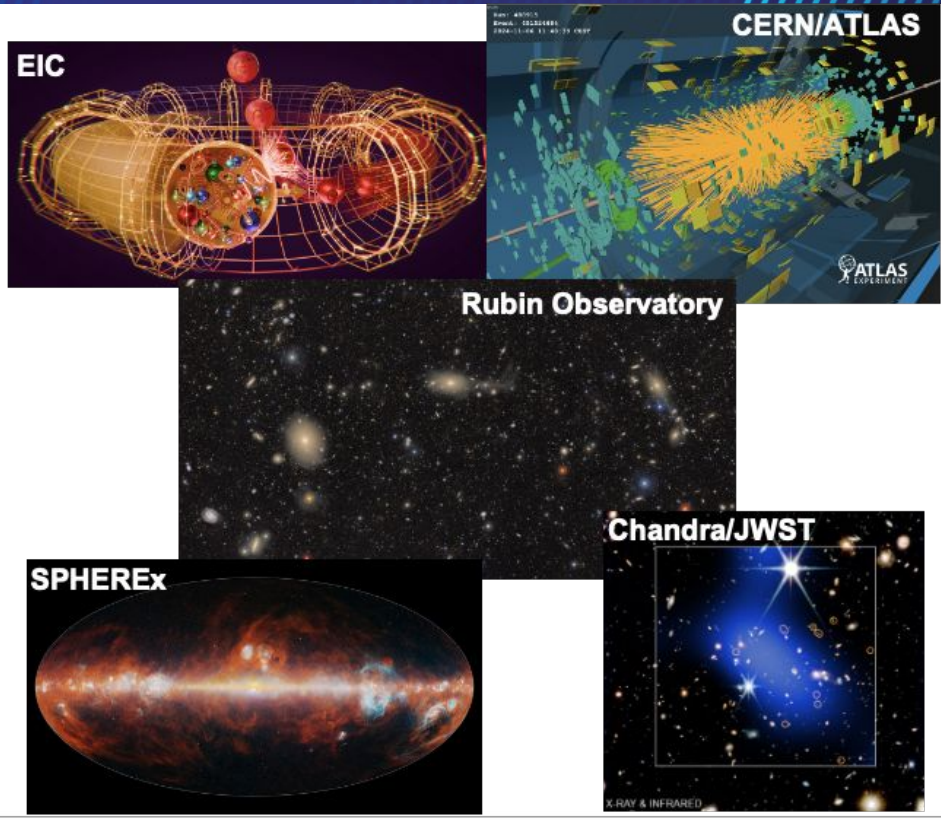
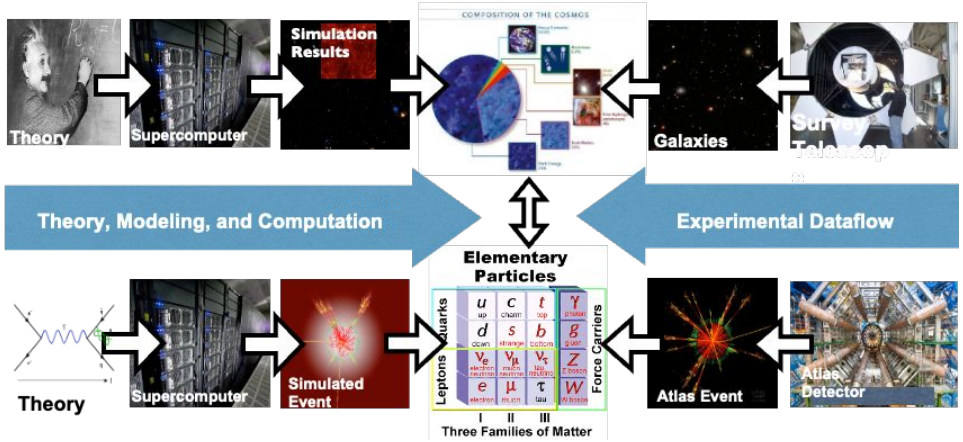
AGENTIC-AI SYSTEMS FOR HIGH ENERGY PHYSICS RESEARCH



Nesar Ramachandra
nramachandra@anl.gov
Argonne National Laboratory

Open Challenges in Data Driven Physics Research



Data Set Characteristics:

The data sets associated with large-scale experiments, observations and simulations:

- Well-calibrated, modern data management, large user base
- Petabyte scale, complex, multi-modal, **siloed**.

Key Challenges:

- How to combine the massive data sets in ways that allow harvesting of scientific insights?
- Deviate from simple combinations of individual analyses alone?
- **Lack of dedicated expertise and support for cross-cutting tasks.**
- Logistic issues: Data access policies, lack of compute/storage facilities for large-scale data analysis/transformation

AI Connections and Advantages

AI Connections for Discovery:

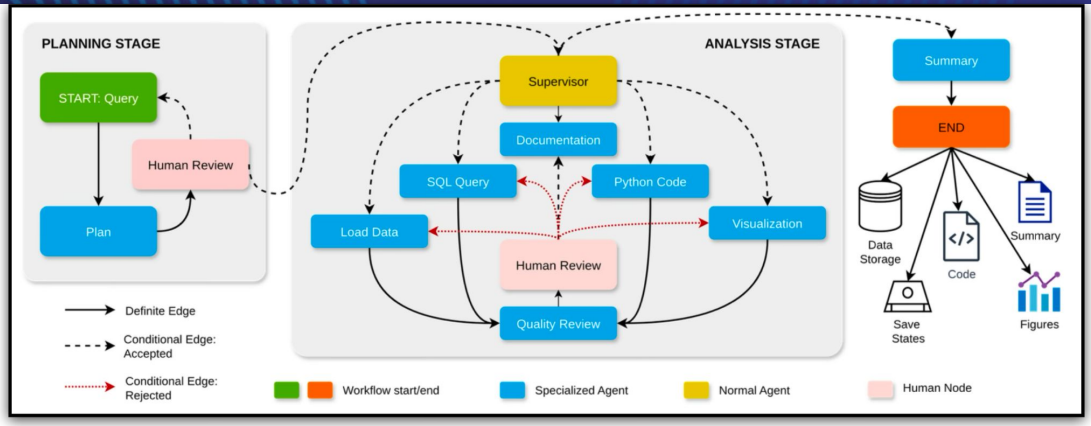
- Can integrative AI based on collections of disparate data sets analyzed within a unified AI-driven framework find **hidden connections for scientific discovery**?
- Can **cross-cutting knowledge enabled through AI** lead to new experiments and analyses that can get us to the desired results faster?

AI Advantage over Conventional methods:

- Complex data fusion across experiments and formats
- Cross-domain searches based on automated hypothesis/query generation
- Automated evaluation, optimization, and error tests
- Automated fusion of (digital twin) simulation and data sources

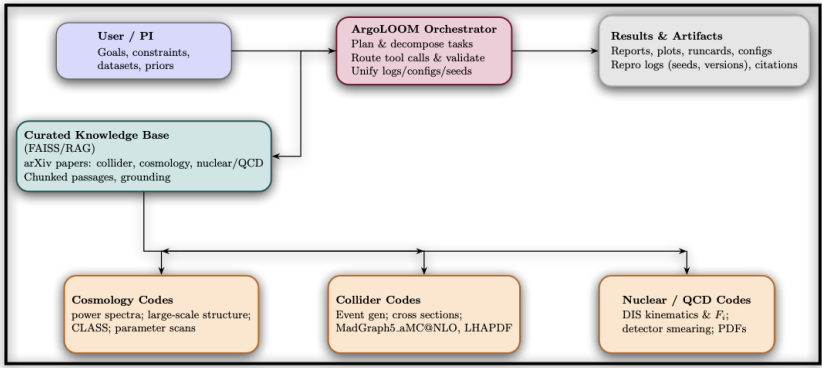


Agentic systems on laptops (first generation)



InferA: A Smart Assistant for Cosmological Ensemble Data, Tam et. al. (incl. Ramachandra, https://arxiv.org/abs/2510.12920

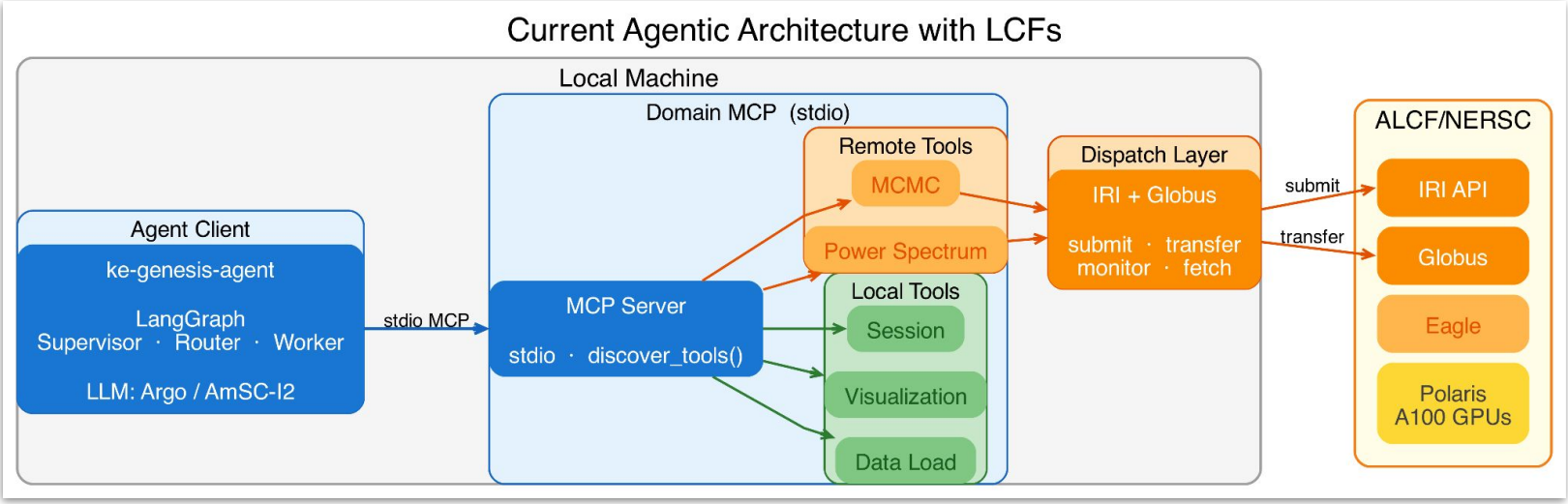
ArgoLOOM: agentic AI for fundamental physics from quarks to cosmos https://arxiv.org/abs/2510.02426



Ingredients of Multi-Agent systems – Tools, Servers, and databases:

- Large Reasoning models with generic tool access (web search, data base reading)
- Multiple expert interactions (Roles: Orchestrator, domain-expert, coding expert, critic)
- Local datasets (texts and numerical data) access, codes, tool interfaces.
- Python server to run codes, feed results back to LLMs

Agentic systems + supercomputing (second generation)



Multi-Agent systems with HPC scaling: Harnesses on local systems, Servers on Supercomputers, APIs for control and transfer.

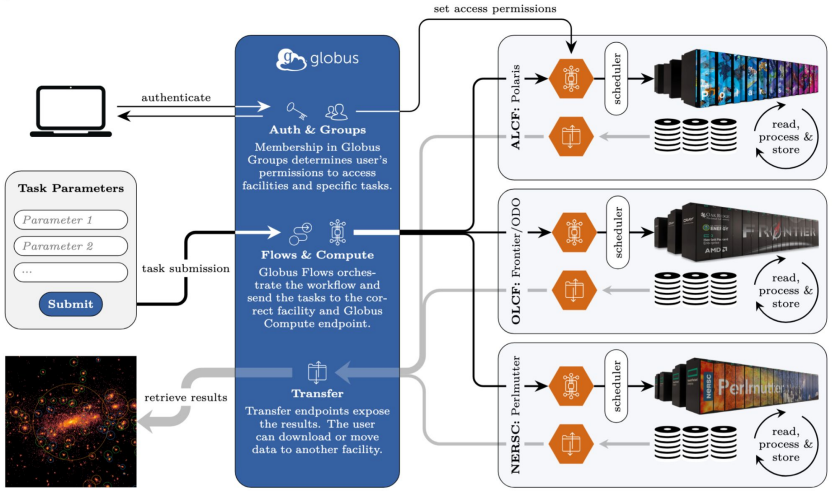
- Multiple persistently hosted servers for each sub-domain, linked to data on large storages.
- Repeated re-vibe-code some of the scripts is not needed – coding only when really needed, and within constraints.
- Interfaces to large scale codes, all accessible via local agentic systems.

Infrastructure for HPC-scale Agentic systems

- Data currently resides on large computing facilities
- HPC resources are needed for data analysis tasks and to support agentic systems
- Longterm vision: Static and on-demand integration of data from major experiments and observatories (EIC, LHC, Rubin LSST, …)
- Data and analysis and simulation integration points are currently at ALCF, NERSC, and OLCF
- Local models hosted on 1000s of GPU nodes.

Infrastructure products:

- ❑ API for LLM interfaces
- ❑ API for MCP servers
- ❑ Federated Compute
- ❑ Data Lakehouse
- ❑ Agent Skills
- ❑ Model training & tracking



Source: [OpenCosmo]

Agentic systems + Supercomputing systems

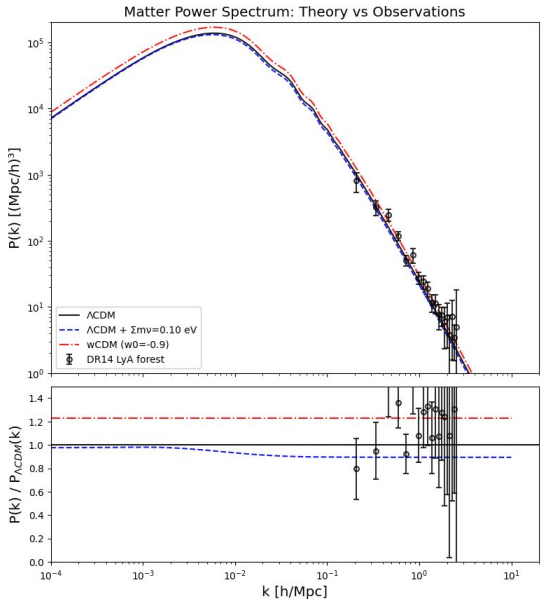
```
ROBLEMS 9  TERMINAL  PORTS                                   python
(cosmodev) nramachandra@roomba:/data/a/cpac/nramachandra/Projects/AmSC/ke_genesis_agent (master)$ python examples/05_agent_domain_test.py
  - arxiv_agent
  - download_arxiv_paper
  - download_full_arxiv_paper
  - list_files
  - read_text_file
  - search_arxiv

Query: Using the observational data from eBOSS, compare the linear matter power spectrum P(k) for LCDMand LCDM with massive neutrinos (sum_mnu=0.10 eV). Create a visualization showing both power spectra with the data.

Running agent...
[dispatch] Dispatching compute_power_spectrum to Polaris...
[dispatch] Local staging: /data/a/cpac/nramachandra/Projects/AmSC/ke_genesis_agent/.staging/compute_power_spectr_260409_090750_8qbjbhyz
[dispatch] Uploading 9 files to Eagle via Globus Transfer...
[dispatch] Globus upload task: 7ee0c6a4-341d-11f1-8e4c-02535127e3d7
[dispatch] Job submitted: 7066622.polaris-pbs-01.hsn.cm.polaris.alcf.anl.gov
[dispatch] Output ready: results.json (205B)

============================================================
HARD FAIL: DISPATCH_TO_POLARIS=true but Polaris dispatch failed.
Error: Error: CosmoSevereError:

Error in Class: Class did not read input parameter(s): m_ncdm
. Ensure params is dict and k_values is list of floats.

Fix: resolve the Globus Transfer permission issue, or set DISPATCH_TO_POLARIS=false for local execution.
See ISSUES.md for details.
============================================================

[dispatch] Processing request of type CallToolRequest
[dispatch] Dispatching compute_power_spectrum to Polaris...
[dispatch] Local staging: /data/a/cpac/nramachandra/Projects/AmSC/ke_genesis_agent/.staging/compute_power_spectr_260409_090844_8dv2b830
[dispatch] Uploading 9 files to Eagle via Globus Transfer...
[dispatch] Globus upload task: 9ee33307-341d-11f1-8144-02535127e3d7
[dispatch] Job submitted: 7066632.polaris-pbs-01.hsn.cm.polaris.alcf.anl.gov
```

README.md compute.py 1

jobs_staging > pk_test > compute.py > ...

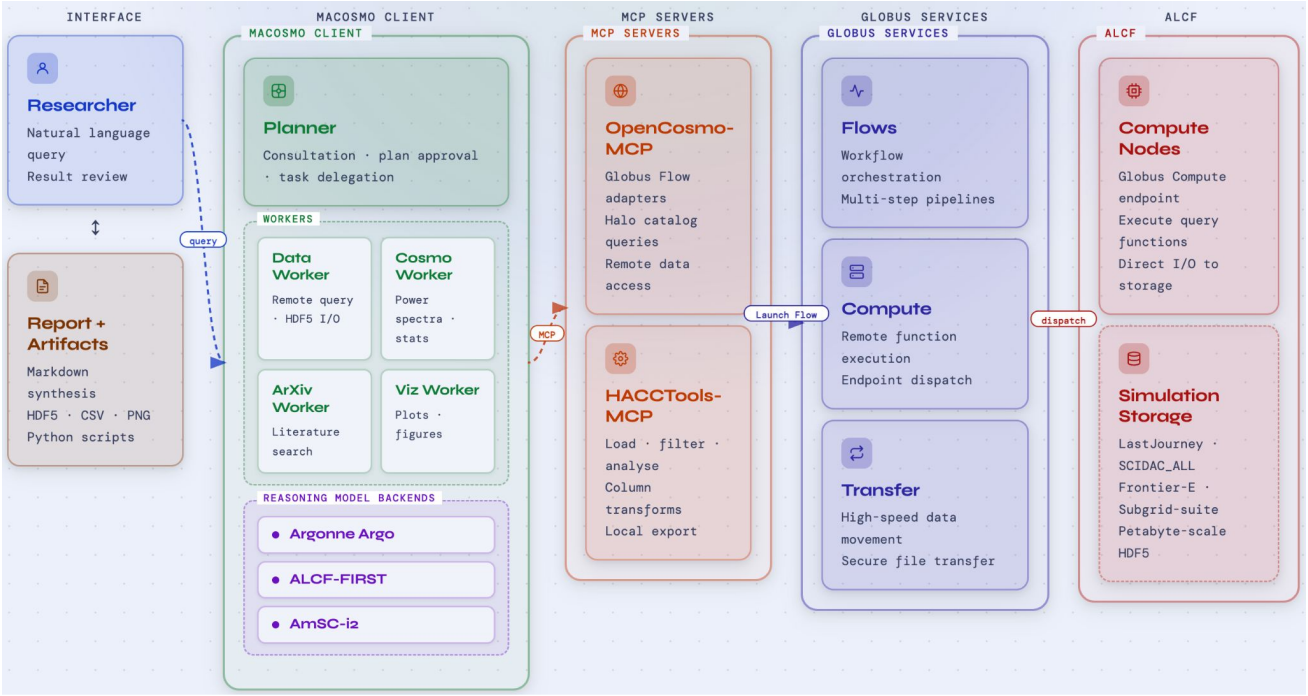
```python
"""Standalone CLASS power spectrum c
import json
import numpy as np

# Read parameters
with open("params.json") as f:
    params = json.load(f)

from classy import Class

cosmo = Class()
class_params = {
    "output": "mPk",
    "h": params["h"],
    "Omega_b": params["Omega_b"],
    "Omega_cdm": params["Omega_cdm"],
    "A_s": params["A_s"],
    "n_s": params["n_s"],
    "P_k_max_h/Mpc": params["P_k_ma
    "z_pk": params["z_pk"],
}

# Massive neutrinos
if params.get("m_ncdm") is not None:
    N = params.get("N_ncdm", 1)
    class_params["N_ncdm"] = N
    class_params["m_ncdm"] = ",".joi
    # Standard: 3 neutrinos total, N
    class_params["N_ur"] = max(0, 3.

# Dark energy
if params.get("w0_fld") is not None:
    class_params["Omega_fld"] = 1.0
    class_params["w0_fld"] = params
    class_params["wa_fld"] = 0.0
```

Update is ready, click to restart. Ln 24, Col 5 Spaces: 4 UTF-8 LF { } Python Finish Setup 3.10.12 Python 3.10.12



Plot generated by the power-spectrum agent. The power spectra lines are generated by CLASS tool with parameters decided by the agentic system.

A Powerful Agentic System to Explore exascale simulations



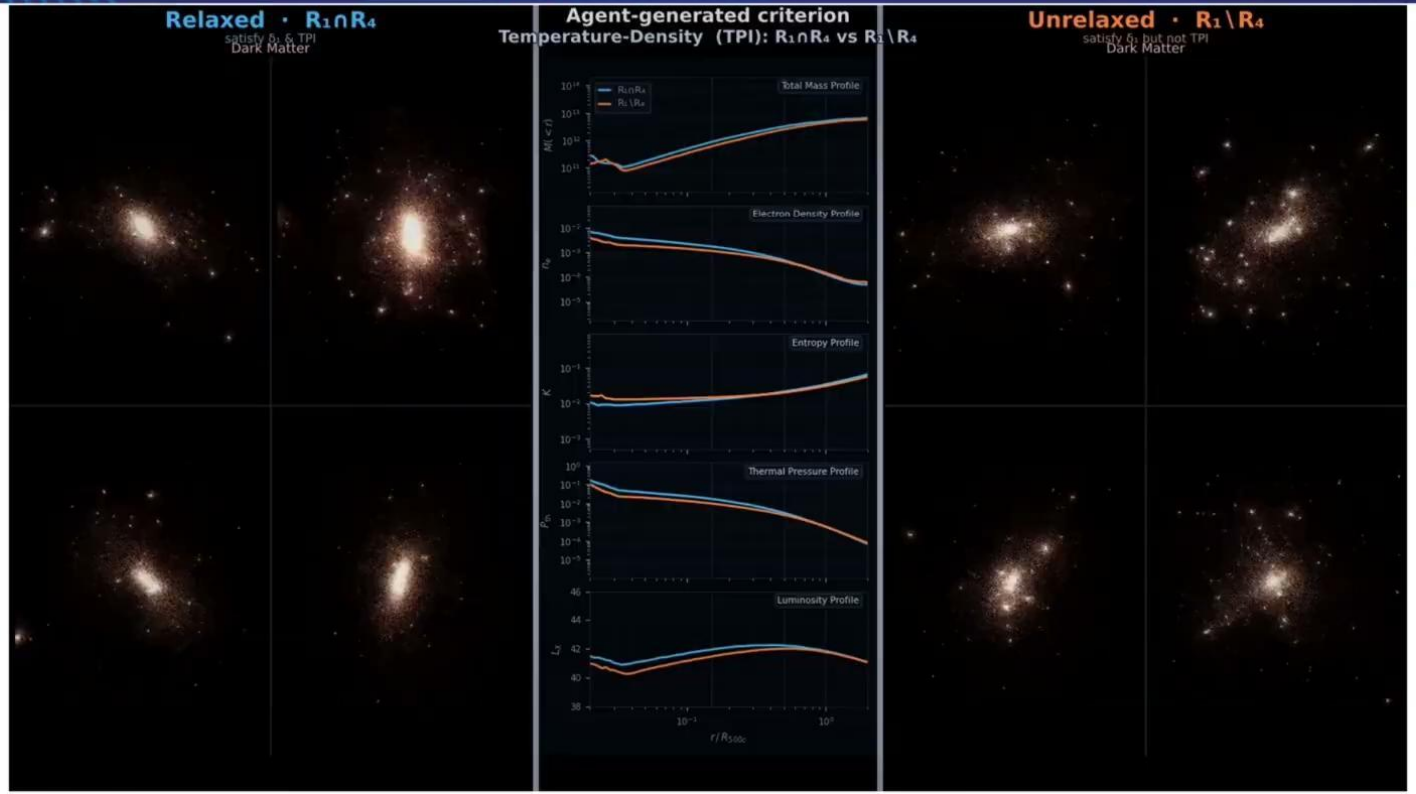
- Frontier-E simulation: largest hydrodynamical cosmology simulation ever carried out
- Rich physics, many scientific questions can be investigated
- Opportunity to compare agentic system performance against small team of expert researchers
- Tools for exploration readily available through OpenCosmo project

The AI Team Delivering New Results – Fast!

```
INFO     Registered tool: launch_galaxy_catalog_query   mcp.py:423
INFO     Registering tool for adapter: Halo-Particles   mcp.py:415
         Query (Gravity)
INFO     Registered tool:                               mcp.py:423
         launch_halo_particles_query_gravity
INFO     Registering tool for adapter: Halo-Particles   mcp.py:415
         Query (Hydro)
INFO     Registered tool:                               mcp.py:423
         launch_halo_particles_query_hydro
INFO     Registering tool for adapter: Halo Query       mcp.py:415
         (Gravity-Only)
INFO     Registered tool:                               mcp.py:423
         launch_halo_catalog_query_gravity
INFO     Registering tool for adapter: Halo Query       mcp.py:415
         (Hydro)
INFO     Registered tool:                               mcp.py:423
         launch_halo_catalog_query_hydro
INFO     Registering tool for adapter: Halo Lightcone   mcp.py:415
         Query (Gravity-Only)
INFO     Registered tool:                               mcp.py:423
         launch_halo_catalog_query_gravity_lc
INFO     Registering tool for adapter: Map Query        mcp.py:415
         (Gravity-Only)
INFO     Registered tool: launch_map_query_gravity      mcp.py:423
INFO     Registering tool for adapter: Diffsky          mcp.py:415
         Synthetic Galaxy Query
INFO     Registered tool:                               mcp.py:423
         launch_diffsky_galaxy_catalog_query
INFO     All tools registered. Starting MCP server.     mcp.py:425
INFO     Processing request of type                   server.py:713
         ListToolsRequest
Interactive mode enabled. Type 'exit' or 'quit' to exit.
Follow-up queries will maintain context from previous queries in this session.
Type 'new' to start a fresh session.

Please enter your query about cosmological data: Examine cluster-scale objects in the
 Frontier-E simulations.  Divide the population among cool-core, non cool-core, relax
ed, and unrelaxed halos.  Plot the populations to show how halo properties vary based
 on these classifications. Plot a visualization of the particles of the most unrelaxe
d halo you find in the simulation.
```

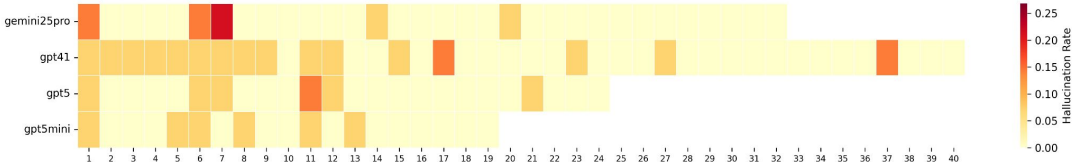
AGENTIC RUN: USER QUERY

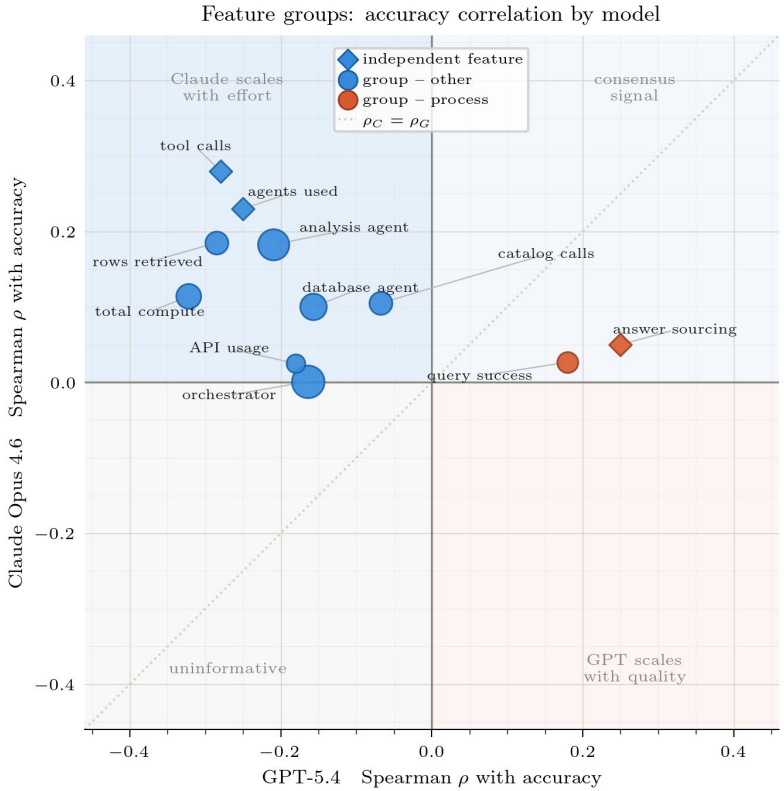
A New Subhalo Population Found in the Frontier-E Simulation



Validation and verification: Architecture checks

- Testing design choices – ablation studies.
  - Tests entire agentic system, with different LLM backends.
  - At the component level – i.e., correlating the features of the agentic systems with accuracies.
  - Initial results show the advantage of tool usage and multi-agent architectures.
- Tracking hallucination occurrences at different steps.
  - Mitigation strategies: working with files, provenance checks.

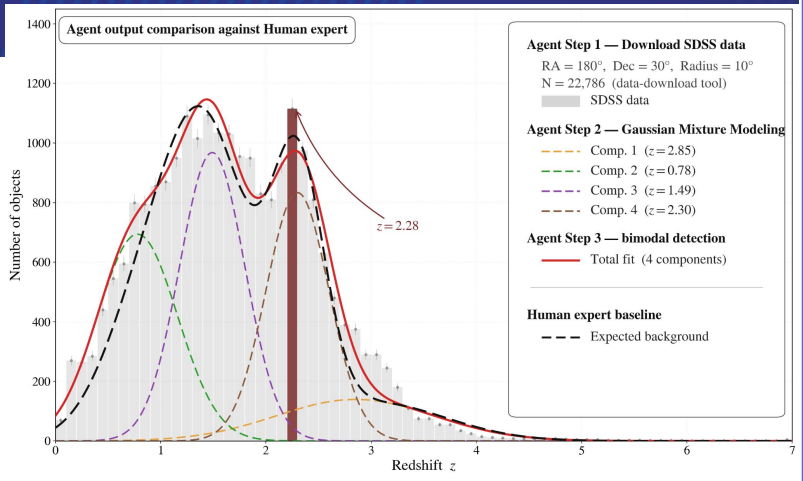


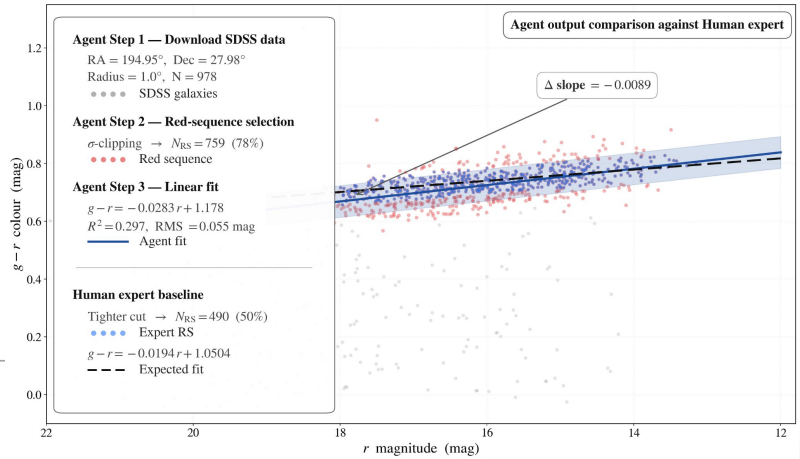


Validation and verification: Physics checks

One of the hardest things in Agentic-AI is confirming the systems provide useful science results.

- Benchmarking datasets close to real-world tests – quantitative metrics, ground truth steps, progressive hints with penalty.
  - Scalable with caveats.
  - Manageable with short queries
- A/B Tests: Experts with and without AI-companion tests:
  - Not scalable but useful
  - Comparisons of time taken in development vs verification, reasoning decisions, artifacts, final results.





Question Text:

Create a histogram of g-r color for SDSS galaxies with 0.02 < z < 0.22 to reveal the bimodal distribution separating the 'blue cloud' (star-forming galaxies) from the 'red sequence' (quiescent galaxies). Use extinction-corrected modelMag colors.

Difficulty & Complexity:

complexity_score: 5/10 query_type: "distribution_analysis"

Ground Truth:

blue_cloud_peak: 0.565 (star-forming galaxies)
red_sequence_peak: 0.943 (quiescent galaxies)
valley_location: 0.754, peak_separation: 0.378

Dataset: 199,818 galaxies, extinction-corrected modelMag colors

Expected Steps:

1. Query SpecObj + PhotoObj for galaxies with 0.02 < z < 0.22
2. Apply extinction corrections using SFD98 dust maps
3. Compute g-r color = modelMag_g - modelMag_r
4. Create histogram with appropriate binning
5. Identify blue cloud and red sequence peaks via peak detection

Progressive Hints:

Hint 1: Galaxy colors show bimodality - blue star-forming, red quiescent
Hint 2: Use histogram peak detection to find blue cloud and red sequence
Hint 3: Apply extinction corrections: g_corr = g - extinction_g
Hint 4: Query extinction_g, extinction_r from PhotoObj; use clean=1 filter
Hint 5: Histogram with bins=0.025 mag, scipy.signal.find_peaks for detection

Conclusions

Questions?

Backup slides

