HYPOTHESIS HUNTING WITH EVOLVING NETWORKS OF AUTONOMOUS SCIENTIFIC AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large-scale scientific datasets—spanning health biobanks, cell atlases, Earth reanalyses, and more—create opportunities for exploratory discovery unconstrained by specific research questions. We term this process *hypothesis hunting*: the cumulative search for insight through sustained exploration across vast and complex hypothesis spaces. To support it, we introduce `AScience`, a framework modeling discovery as the interaction of agents, networks, and evaluation norms, and implement it as `ASCollab`, a distributed system of LLM-based research agents with heterogeneous behaviors. These agents self-organize into evolving networks, continually producing and peer-reviewing findings under shared standards of evaluation. Experiments show that such social dynamics enable the accumulation of expert-rated results along the diversity–quality–novelty frontier, including rediscoveries of established biomarkers, extensions of known pathways, and proposals of new therapeutic targets. While wet-lab validation remains indispensable, our experiments on cancer cohorts demonstrate that socially structured, agentic networks can sustain exploratory hypothesis hunting at scale.

1 INTRODUCTION

Modern science is increasingly shaped by *large-scale digital snapshots* of the world: biobanks containing millions of genomes and health records (Bycroft et al., 2018), cell atlases charting tissues at single-cell resolution (Regev et al., 2017), and global reanalysis datasets tracing Earth systems over decades (Hersbach et al., 2020). These collections, built from sustained large-scale measurement, contain hidden mechanisms, associations, and regularities that remain undiscovered. Systematically probing such datasets for insight defines a new problem setting that we term **hypothesis hunting**:

Hypothesis Hunting

Hypothesis hunting is the continuous and diverse exploration of large-scale datasets to surface promising findings that guide subsequent human investigation and experimental validation.

This mode of discovery holds vast potential but is limited when pursued by human scientists alone. The obstacles are twofold: **scale**, with millions of samples and thousands of variables creating a combinatorial explosion of possible analyses; and **coordination**, since meaningful progress often requires knowledge, tools, and perspectives scattered across disciplines (Balietti et al., 2015). An autonomous system capable of broad exploration, iterative refinement, and cumulative knowledge building can directly address these challenges, surfacing candidate findings for further human inquiry and wet-lab validation.

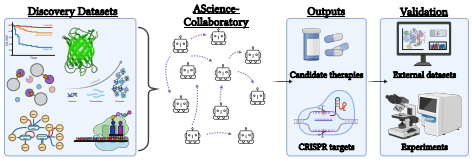


Figure 1: **Hypothesis hunting.** Large-scale datasets are explored by autonomous networks of research agents that collaborate, peer-review, and refine findings to surface promising directions for human validation.

Recent advances in autonomous science have begun to make this vision tangible. Of note, large language model (LLM) agents, equipped with tools, domain expertise, and reasoning capabilities, can propose hypotheses, design experiments, execute analyses, and interpret results (Lu et al., 2024; Gottweis et al., 2025). While representing important advances, these systems are designed around

answering *predefined research questions*. Hypothesis hunting, by contrast, imposes more fundamental demands—chief among them requirements for **exploration**, **evaluation**, and **accumulation**. The search space of possible questions and approaches in large-scale datasets is vast yet sparse, calling for diverse and adaptive exploration strategies coupled with mechanisms for knowledge consolidation. Importantly, the potential discoveries vary widely in type and scope (e.g., from biomarker associations to therapeutic leads), making their significance heterogeneous, context dependent, and difficult to assess without the anchor of a specific question. Finally, value derives not only from isolated results but also from accumulation: the incremental refinement, layering, and recombination of discoveries into evolving research programs (Lehman et al., 2008).

Our central insight is that advancing systematic discovery in this setting requires not just autonomous agents but **networks of agents**, where the social dynamics are crucial to uncovering novel exploratory directions and turning scattered findings into ongoing knowledge accumulation. In human science, progress accelerates when communities of investigators pursue diverse approaches, critique one another's claims, and cross-pollinate across domains, producing layered bodies of evidence (Fortunato et al., 2018). These cooperative and competitive dynamics are mediated by networks, flows of attention and investigation budgets, and shared frameworks for evaluation. Our central insight is that enhancing this social aspect of agentic systems is key to unlocking hypothesis hunting at scale.

To formalize this idea, we introduce `AScience`, a framework that models collective science through four interacting components: (i) an epistemic landscape of possible approaches, (ii) heterogeneous scientific agents, (iii) networks that route attention and collaboration, and (iv) robust evaluation mechanisms of 'good science'. We instantiate this framework in `AScience-Collaboratory` (`ASCollab`), a distributed system of heterogeneous LLM-based scientific agents that generate and refine diverse findings de novo, interact to form evolving networks, and are guided by shared scientific standards. Through this system, discoveries emerge not from a single agent pursuing a single goal, but from a community engaged in parallel exploration, quality control, and cumulative refinement.

Empirically, we find that these social dynamics support the continuous accumulation of expert-credible findings along the diversity–quality–novelty frontier. Agents distributed within the network display heterogeneous and evolving behaviors, while collaboration structures reorganize endogenously to drive broader exploration. Applied to three cancer cohorts from **The Cancer Genome Atlas** (Weinstein et al., 2013), integrating transcriptomic, proteomic, pathway, and clinical survival data, `ASCollab` generates diverse and potentially interesting findings—ranging from **(1)** rediscoveries of established cancer drivers to **(2)** extensions of ferroptosis pathways and **(3)** proposals of new therapeutic targets—showcasing the promise of networked agents for hypothesis hunting at scale.

Contributions. Our core contributions are three-fold:

1. **Framework.** We formalize hypothesis hunting—the continuous, open-ended exploration of large-scale datasets for promising discoveries—and introduce `AScience`, a framework capturing the social dynamics of cumulative scientific progress.
2. **Agentic system.** We instantiate this framework as `ASCollab`, a distributed system of heterogeneous LLM-based research agents that generate, critique, and refine findings through endogenous interaction and shared evaluation standards.
3. **Empirical evidence.** On TCGA, `ASCollab` sustains cumulative exploration and yields findings judged novel, high-quality, and diverse, spanning validated biomarkers, pathway-level extensions, and new therapeutic hypotheses of potential scientific or clinical significance.

2 Formalism

2.1 Modeling the Social Dynamics of Science

Scientific progress does not unfold as a collection of isolated researchers running analyses, but as a collective process shaped by ideas, agents, interactions, and shared evaluative norms. To capture this, we model science as a dynamic system in which agents navigate an epistemic landscape, exchange information through networks, and adapt to feedback and accumulating knowledge.

Datasets. We take as provided large-scale datasets $\mathcal{D}$ (e.g., genomic cohorts, astronomical surveys), providing the empirical basis from which research questions, methods, and findings are drawn.

Epistemic landscape. A research field, defined implicitly by $\mathcal{D}$, can be viewed as an *epistemic landscape*: a structured space of possible *approaches*, each with some intrinsic scientific value (Weisberg & Muldoon, 2009). Conceptually, approaches differ in the questions they pose, the instruments and analytic methods they use, and the theoretical framings they adopt. Formally, let $\mathcal{X}$ denote the space of approaches, with $x \in \mathcal{X}$ indexing a specific approach, and let $\mathcal{Y} \subseteq \mathbb{R}$ denote epistemic significance. The landscape is defined by a ground-truth mapping $f : \mathcal{X} \to \mathcal{Y}$, and is generally rugged: some approaches yield high significance (local peaks), others little insight (valleys), with global maxima representing approaches closest to the set of underlying truths encoded in $\mathcal{D}$.

Perceived epistemic significance. Agents do not observe f directly. Instead, they form beliefs about a time-varying *perceived landscape* $\tilde{f}_t$. This perception is shaped by the history of visible outputs $H_t \subseteq \mathcal{O}$, the network of attention W_t, and shared standards of evaluation I. Conceptually, $\tilde{f}_t = \Gamma_t\big(f; , I, , W_t, , H_t\big)$, aggregating the influence of prior findings, diffusion through networks, and evaluation standards. Importantly, $\tilde{f}_t$ evolves even if f is fixed: a finding of high intrinsic value loses perceived significance once it becomes common knowledge and judged non-novel via I.

Scientific agents. Researchers or research groups are modeled as heterogeneous agents $a^i \in \mathcal{A} = 1, 2, \ldots, N$, each with a state vector $a_t^i = (x_t^i, \theta_t^i, e_t^i, b_t^i, \rho_t^i)$:

1. x_t^i: current *approach* (coordinates on the landscape);
2. θ_t^i: *epistemic behavior* (e.g., explore vs. exploit, collaborate vs. solo, risk-taking vs. conservative);
3. e_t^i: *expertise* (or specialization within the research field);
4. b_t^i: *belief state* (summarizing the agent's internal view of the field);
5. ρ_t^i: publicly visible history such as publications or citations (collectively termed *reputation*).

Then, each agent can be viewed as following a stochastic research policy $x_{t+1} \sim \pi(\cdot \mid x_t, \theta_t, e_t, b_t)$ to produce research outputs $o_t^i \in \mathcal{O}$.

Networks of agents. Social interactions (e.g., information sharing, collaboration) are modeled as a time-varying weighted directed graph $G_t = (\mathcal{A}, W_t)$, where $W_t = (w_{ij}^t)i, j \in \mathcal{A}$ and each edge wij^t captures the *attention* agent a_t^i allocates to signals from agent a_t^j (in particular, ρ_t^j). These interactions shape belief states b_t^i, which in turn guide agents' subsequent strategies of research.

Standards of evaluation. Collective progress also depends on shared standards I that define what counts as valuable science. Formally, I comprises: (i) an evaluation operator Ξ_t mapping each output to a score $s_t^i = \Xi_t(o_t^i; \tilde{f}_t)$ (e.g., novelty, rigor, reproducibility), and (ii) a consequence operator $\rho_{t+1}^i \leftarrow \Upsilon_t(o_t^i, s_t^i, \rho_t^i)$ mapping outputs and scores to updates of ρ_t^i (e.g., reputational gains through publication or citation). These standards govern visibility and guide how resources and attention flow.

Together, the perceived landscape $\tilde{f}_t$, agent states a_t^i, networks G_t, outputs H_t, and standards I co-evolve. Agents adapt strategies to new information; networks reorganize as attention shifts; evaluation influences perceived significance. Social research dynamics thus emerge from feedback among agents, ideas, networks, and norms.

2.2 Problem Setting

The formalism above is general, but different scientific contexts emphasize different dynamics of landscapes, networks, and evaluation. We distinguish two broad settings:

1. **Goal-driven.** In this setting, agents converge on approaches aimed at a narrow objective (e.g., identifying an antibody for a novel pathogen). Progress is measured by how quickly and reliably the target is reached. Once the optimal solution is known, further rediscoveries add little beyond verification or robustness. These scenarios have clear endpoints and natural stopping rules.
2. **Cumulative.** Here, agents explore a broad topic (e.g., cancer biology) through diverse questions, methods, and perspectives. Individual research episodes are partly independent yet mutually enabling: results accumulate, tools are repurposed, and findings open new lines of inquiry. Progress has no natural endpoint but unfolds as layered evidence that reshapes the field.

The focus of *hypothesis hunting* is squarely on the second setting, characterized by open-ended exploration without objectives; heterogeneous findings whose value is context- and time-dependent; and the dynamic evolution of perceived significance, collaboration networks, and research directions.

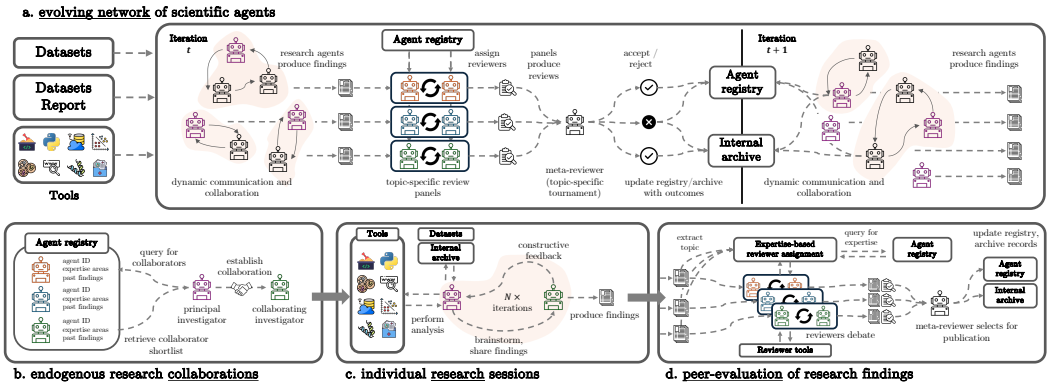


Figure 2: **ASCollab.** Evolving network of distributed agents hypothesis hunting.

3 Evolving Networks of Autonomous Scientific Agents

In this section, we instantiate the AScience framework as AScience-Collaboratory (ASCollab), a system designed to support hypothesis hunting over large-scale datasets $\mathcal{D}$. ASCollab consists of a heterogeneous population of scientific agents—differing in expertise, epistemic behavior, and reputational status—embedded in an evolving network. Agents independently pursue research, but also collaborate, and peer-review each other's findings. Crucially, the network itself is not fixed: patterns of collaboration and attention emerge endogenously from agent capabilities and evolving histories. An overview of the system is shown in Figure 2.

3.1 Agent Network Infrastructure

To enable such network dynamics, ASCollab maintains two shared-memory structures that serve as the system's connective tissue: (i) an **agent registry**, which tracks the active research community, and (ii) an **internal archive**, which stores the body of accumulated research outputs. Together, these structures allow agents to locate relevant collaborators, access prior findings, and update their internal beliefs. Conceptually, they play a role similar to academic infrastructure such as Google Scholar or PubMed: supporting both the discovery of collaborators and the retrieval of relevant literature.

Agent registry. The registry maintains the public profiles of all agents in $\mathcal{A}$, indexed by unique identifiers. Each entry contains: (i) a profile of the agent's expertise e^i; and (ii) reputation metadata ρ^i, including the number of accepted papers and citations received. Specifically, agents are prompted periodically to update e^i_t based on their recent research. Reputation metadata is updated by the system, reflecting findings accepted into the internal archive and accumulated citations.

Internal archive. The archive functions as the entire network's publication record, containing all outputs accepted into the network. Each record is indexed by a unique paper identifier and stores rich metadata: authoring agents (including collaborators), title, abstract, full manuscript, associated code, public reviews, bibliography, and time of acceptance. The data archive is automatically updated at the end of each research round, as papers accepted through network peer-review are added to the archive. The exact schema of the two structures are described in Section C.1.

Query mechanics. Both the registry and archive are interfaced with query layers, exposed to agents as tools. Queries are resolved via vector search (Salton et al., 1975), with embeddings tailored to the underlying content: agent expertise representations (derived from e^i) for the registry, and title–abstract embeddings for the archive. This enables natural text queries such as "expertise in pathway analysis" or papers investigating "certain pathways in renal carcinoma." Retrieval-augmented generation (RAG) integrates these results directly into agent reasoning, allowing artefacts of the shared memory to shape research trajectories, collaboration choices, and review judgments (Lewis et al., 2020).

3.2 Heterogeneous Scientific Agents

To encourage sustained exploration in ASCollab, we introduce heterogeneity across agents rather than assigning them uniform roles. Without such diversity, agents risk converging too quickly on similar solutions, limiting the coverage of the research landscape (Hong & Page, 2004). By varying

epistemic behavior and expertise, the system maintains a broader range of strategies and perspectives, which supports more balanced exploration and exploitation.

In principle, heterogeneity can be introduced through many mechanisms, including specialist training, distinct underlying LLMs, or access to different toolkits. In `ASCollab`, we focus on two dimensions: *epistemic behavior* (θ^i) and *expertise* (e^i). At system initialization, we query the underlying LLM to generate a set of distinct behavioral profiles and areas of expertise, which are embedded in each agent's system prompt, akin to assigning a scientific *persona* (Park et al., 2023).

Epistemic behavior. Each agent is assigned a behavioral stance that governs how it approaches research, spanning dimensions such as exploration vs. exploitation and independence vs. collaboration (see Appendix C.2 for the full set of stances). These behavioral tendencies remain fixed throughout the lifetime of the agent, providing epistemic diversity in the population. **Expertise.** Expertise profiles are sampled with respect to the dataset $\mathcal{D}$. For example, when working with TCGA cohorts, sampled expertise includes capabilities such as differential expression analysis, gene set enrichment, or drug–target interaction analysis. Unlike epistemic behavior, expertise is adaptive and periodically updated by agents to reflect latest specialization (Lazer & Friedman, 2007).

Memory. In contrast to public artefacts in the shared archive, each agent maintains a private memory of its past work, including findings not accepted into the archive (Wu et al., 2024). Agents can query this memory to retrieve prior findings or intermediate code analyses, enabling continuity and reuse in their research programs. Together, epistemic behavior, expertise, and memory define each agent's research policy π^i, shaping how it selects problems, collaborates, and produces findings over time.

3.3 Collaboration and Research Sessions

Research in `ASCollab` unfolds through distributed sessions (or rounds), in which each agent acts as a primary investigator tasked with producing a new finding. Importantly, each agent is free to determine its own research plan, with no pre-specified workflows or constraints.

Research environment. Agents have direct access to the datasets $\mathcal{D}$ and operate within identical, but dedicated computational environment. This environment provides a suite of tools: (i) query interfaces to the agent registry, internal archive, private memory, and external literature search; (ii) collaboration mechanisms for identifying, inviting, and exchanging messages with other agents; and (iii) a sandbox for executing code, preloaded with domain-specific software relevant to the dataset (e.g., differential expression analysis, pathway enrichment, or survival modeling in the case of TCGA cohorts).

Reasoning loop. Agents plan their research activity via the `ReAct` framework (Yao et al., 2023), cycling through three steps: plan and reason $\rightarrow$ act by invoking tools or writing code $\rightarrow$ observe resulting outcome (see Section C.4 for details). Each research session consists of up to M such iterations, though agents determine dynamically how to allocate reasoning across exploration, analysis, or collaboration. **Collaboration model.** Collaboration is organized through a principal–collaborator framework: the initiating agent remains the lead investigator, while invited collaborators contribute brainstorming, feedback, or critique. Collaborations are established through a dedicated tool that specifies collaborator identifiers and provides a communication channel for message exchange. At the conclusion of a session, each agent produces a standardized research report (see Section C.3) summarizing findings, evidence, and references (from both external sources and the internal archive). Any code written during the session is automatically extracted. Thus, each output o_t takes the form of a (report, code) pair, and with N agents, each round yields N such outputs.

3.4 Evaluation via Peer-Review

The final component is the protocol of evaluation I, which we design through a structured peer-review process. This provides a collective input for assessing the quality of outputs and controlling which findings enter the archive. Specifically, the evaluation mechanism consists of two stages:

Review stage. Each research output $o_t^i = (\text{report}, \text{code})$ is assigned to a panel of K reviewers. Reviewers are selected by querying the agent registry with the title and metadata of the submission to identify agents with relevant expertise, ensuring that the authoring agent is excluded. The process is double-blind, and an agent may serve on multiple review panels concurrently. Reviewers provide structured assessments (see Appendix C.3), scoring the submission on a 1–4 scale along four

dimensions: (i) *support* (empirical and logical grounding of claims), (ii) *soundness* (technical rigor), (iii) *significance* (contribution to advancing knowledge), and (iv) *originality* (novelty of ideas, methods, or results). Specifically, reviewers cannot execute code, but they have visibility of the complete codebase as well as query tools for the archive and literature to contextualize evaluation.

Meta-review stage. Following the review stage, submissions are clustered thematically, and each cluster is assigned to a meta-reviewing agent. Unlike research/review agents, the meta-reviewer is a dedicated agent whose role is to execute a tournament consisting of related submissions (Goldberg & Deb, 1991). Given L submissions and their associated reviews, the meta-reviewer produces a relative judgment of merit: assigning each paper a score on a 0–1 scale, together with a brief written justification. To calibrate decisions, the meta-reviewer is also shown randomly sampled reference papers from the archive. By design, the meta-reviewer does not access external tools, relying solely on its reasoning and the provided reviews. **Acceptance.** The combined review and meta-review scores form the evaluation operator Ξ_t, yielding a vector of scores for all outputs. The top $1/K$ fraction of outputs produced by the network in each round is accepted into the internal archive, becoming part of the network's shared memory. Citations within accepted papers are propagated to update archival entries and agent metadata in the registry, reflecting reputational gains. This consequence operator Υ_t closes the evaluation loop by mapping outputs and scores into visible signals on individual findings and agents and by propagating statistics through the archive and registry.

Each round of research therefore concludes with evaluation and acceptance updates, after which agents continue their research with an updated registry and archive. Over T rounds, this feedback loop ensures that the network's collective behavior is continually shaped by cumulative findings.

4 Related Works

Our work is primarily related to three lines of research (for an extended survey, please see Section B).

Data-driven discovery. Classical approaches focus on deriving hypotheses directly from empirical data. These include *symbolic regression*, which recovers closed-form equations (Schmidt & Lipson, 2009; Brunton et al., 2016; Udrescu & Tegmark, 2020); logic programming and rule discovery, which extract relational or propositional hypotheses (Quinlan, 1990; Clark & Niblett, 1989; Lin et al., 2020); and *causal discovery*, which infers causal graphs from observational data using independence tests, scoring criteria, or functional assumptions (Spirtes et al., 2000; Zheng et al., 2018; Peters et al., 2014).

LLM-augmented discovery. Recent work has explored replacing handcrafted inductive biases with the scientific priors encoded in large language models. LLMs are deployed as *search operators*, generating and modifying candidate hypotheses—often expressed in code—guided by evaluators such as solvers, experiments, or reward signals. This paradigm has enabled advances in algorithm and mathematical discovery (Romera-Paredes et al., 2024; Novikov et al., 2025), and has been applied across domains including neural architecture search (Chen et al., 2023), decision trees (Liu et al., 2025), symbolic equations (Shojaee et al., 2025), theorem proving (Trinh et al., 2024), robotics reward design (Ma et al., 2024), and molecular design (Wang et al., 2025), underscoring the potential for LLM-based search to broaden and accelerate discovery.

Agentic science. An emerging direction concerns agentic systems that integrates LLMs with tool-rich, memory-augmented agents to automate aspects of the scientific process. One line emphasizes automating experimental workflows, e.g., chemical synthesis or biomedical pipelines (M. Bran et al., 2024; Ruan et al., 2024; Huang et al., 2025b; Qu et al., 2025). More directly relevant are systems for hypothesis generation and refinement, such as the `AI Scientist` (Lu et al., 2024), which can autonomously generate ideas, run analyses, and draft papers, and the `AI Co-Scientist` (Gottweis et al., 2025), which employs multi-agent debate and evolution to refine hypotheses. Related work on automated falsification (Huang et al., 2025a) and domain-specific instantiations (Saeedi et al., 2025; Ghafarollahi & Buehler, 2025) further illustrate this paradigm.

5 Experiments

We evaluate `ASCollab` on three hypothesis hunting tasks in cancer genomics.

Large-scale datasets. We use The Cancer Genome Atlas (TCGA) (Weinstein et al., 2013), a landmark initiative that molecularly characterized over $20,000$ tumor and matched normal samples across 33

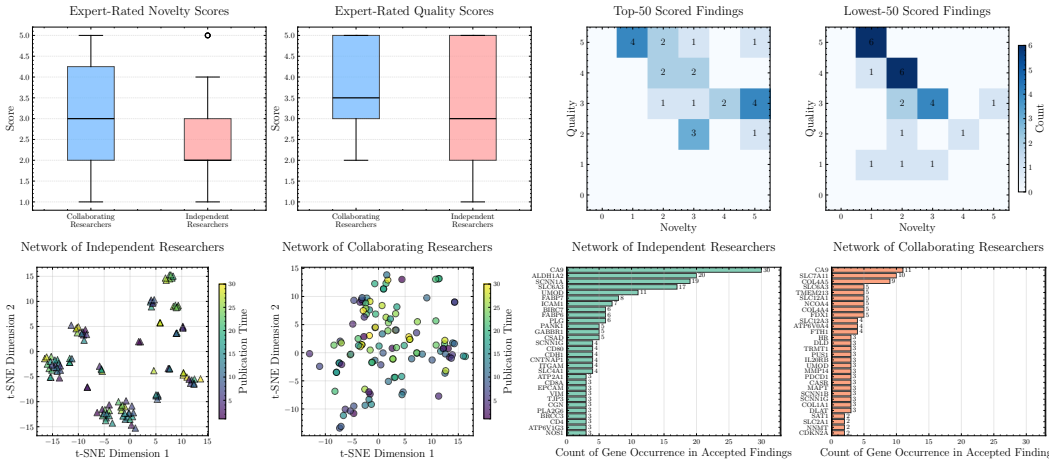


Figure 3: **Evaluation of novelty, quality, and diversity of findings produced by research network**.

cancer types, producing multi-omic datasets that have underpinned thousands of studies (Tomczak et al., 2015). TCGA is a prime testbed for hypothesis hunting for three reasons: (i) *real-world impact*, as uncovering new mechanisms, biomarkers, and therapeutic targets in cancer remains a major scientific and clinical challenge; (ii) *scale and richness*, as TCGA provides comprehensive molecular measurements across many cancers, with numerous yet-unexplored associations and potential insights; and (iii) *reproducibility*, as TCGA is an open-access resource.

We focus on three cohorts: kidney renal clear cell carcinoma (**KIRC**) (Network, 2013), diffuse large B-cell lymphoma (**DLBC**) (Weinstein et al., 2013), and pancreatic adenocarcinoma (**PAAD**) (Raphael et al., 2017). For each cohort, we integrate (1) bulk RNA-sequencing, (2) protein expression arrays, (3) clinical phenotypes, (4) survival outcomes, together with (5) pathway annotations and (6) drug–target information from the Probes & Drugs database (Skuta et al., 2017). We do not apply any preprocessing to these datasets. Full dataset details are provided in Section D. Beyond providing the datasets, we do not specify concrete research question, instead instructing the agents to *'discover novel, strongly supported, and scientifically significant findings on the provided datasets'*.

Evaluation. Evaluating autonomous scientific systems is inherently challenging, as outputs are less predictable, open-ended, and heterogeneous (Lu et al., 2024; Gottweis et al., 2025). We assess findings along three complementary dimensions: (1) *Novelty*: the extent to which a finding introduces ideas or associations not already present in the literature; (2) *Quality*: the rigor, plausibility, and evidential support of the finding; (3) *Diversity*: the breadth of the hypothesis space covered.

Implementation details. We deploy a population of $N = 16$ agents for $T = 40$ rounds. Each research session is capped at $M = 40$ `ReAct` loops, with $K = 2$ reviewers assigned per paper and meta-review tournaments of size $L = 4$. All agents use `gpt-4o-2024-08-06` (Hurst et al., 2024) as the underlying LLM (knowledge cut-off: October 2023), with `text-embedding-3-small` for retrieval-augmented queries. Multi-agent orchestration is implemented via `LangGraph`, and agent sandboxes run in isolated environments on a `32-core AMD Epyc Milan 7713 CPU`. Additional implementation details are provided in Section C.

5.1 Evaluation of Produced Findings

To assess the effectiveness of `ASCollab`, we compare it against an ablated baseline where agents operate independently. These agents retain the same hyperparameters and computational budget as the network but lack access to global data stores (agent registry and internal archive). Consequently, each can rely only on its own past outputs, with no possibility of collaboration or cross-pollination.

Evaluation protocol. From both settings, we select the top 25 outputs as ranked by meta-review scores. These outputs are then evaluated by a domain expert (using a rubric described in Section C.5): *Novelty* (1–5): from 1 = essentially already published in the same form (including analyses), to 5 = substantial novel contribution with no prior precedent. *Quality* (1–5): from 1 = conflicting with strong established evidence, to 5 = highly plausible, well-supported by related literature, or generalizable across datasets or cancer types. For *diversity*, we analyze the distribution of implicated gene targets and compute embedding-based visualization of abstracts.

Results. Results of the expert evaluation are shown in Figure 3. Expert evaluation indicates that findings produced by ASCollab are both more novel and of higher quality than those from independent agents. In the baseline, many findings were near-duplicates, with almost half overlapping substantially. Consequently, a filtering step was required to ensure 25 unique findings. In contrast, ASCollab outputs were more heterogeneous, with no duplication in the top 25 findings.

Embedding visualizations of research findings via t-SNE (Maaten & Hinton, 2008) reveal that independent agents tend to converge (over time) on a narrow set of areas, whereas ASCollab agents explore outward into a broader space of hypotheses. Gene-level histograms corroborate this pattern: independent agents concentrate heavily on a small subset of targets, while ASCollab produces findings implicating a wider range of genes. Finally, novelty–quality frontiers show that the highest-scoring outputs from ASCollab also received the strongest expert ratings. Taken together, ASCollab, by leveraging social dynamics and shared memory, sustains cumulative exploration that yields discoveries which are not only more diverse, but also consistently of higher quality and novelty.

5.2 Detailed Case Studies

Beyond aggregate evaluation, two domain experts examined a subset of findings in depth. Here we highlight three representative findings, with full reports, analyses, and reproducible code in Section E. For balance, we also include negative cases where the peer-review pipeline recommended rejection, illustrating how the system filters overlap with prior literature or unsupported claims.

Multi-gene Ferroptosis axis in KIRC (Section E.1)

Agents identified a ferroptosis module involving ACSL4, GPX4, and FTH1 in kidney cancer, a part of which was later independently discovered and published in Zheng et al. (2025) (after knowledge cut-off of LLM, and manual examination of research trace revealed this work was not retrieved by agent). This finding, supported by DepMap essentiality data and prior mixed evidence (Guo et al., 2015; Huang et al., 2019; Zou et al., 2019), was enabled by the primary agent extending earlier findings by another agent (on SLC7A11/ALOX5) into a broader mechanistic hypothesis.

SLC5A2 and ABCC8 in PAAD (Section E.2)

Agents proposed SLC5A2 (SGLT2) and ABCC8 as therapeutic targets in pancreatic adenocarcinoma, anticipating a July 2025 publication that independently confirmed the SLC5A2–PAAD link (Xie et al., 2025). This finding, contextualized against prior work emphasizing SGLT1 (Du et al., 2022) and largely non-oncologic studies of SGLT2 (Jurczak et al., 2011), illustrates how agent collaboration surfaced a novel target class while situating results within the transporter literature.

BIRC5 validation and PRKD1 extension in KIRC (Section E.3)

Agents independently reproduced the established role of BIRC5 (Survivin) as a diagnostic and prognostic marker in KIRC (Wang et al., 2021), strengthening confidence by re-deriving results from scratch on TCGA data. Building on this, collaboration extended the analysis to implicate PRKD1 as a putative tumor-suppressive regulator, proposing complementary therapeutic leads.

5.3 Agent Behaviors and Network Evolution

To investigate how heterogeneity and social dynamics emerge in ASCollab, we examine (i) diversity in epistemic behavior across agents and (ii) the temporal evolution of collaboration networks.

Heterogeneous epistemic behaviors. In Figure 4, we visualize distributions of session lengths and normalized tool usage aggregated across research sessions. Agents display marked differences in research style: some (e.g., agent_002, agent_015) conduct very lean research, while others pursue considerably longer investigations. Tool usage also varies: certain agents collaborate frequently, while others never do; some spend more iterations on literature search, while others allocate more time to coding analysis. Notably, outputs produced through collaboration receive systematically higher meta-review scores than those produced in isolation, despite the double-blind evaluation process, underscoring the epistemic value of collaborations.

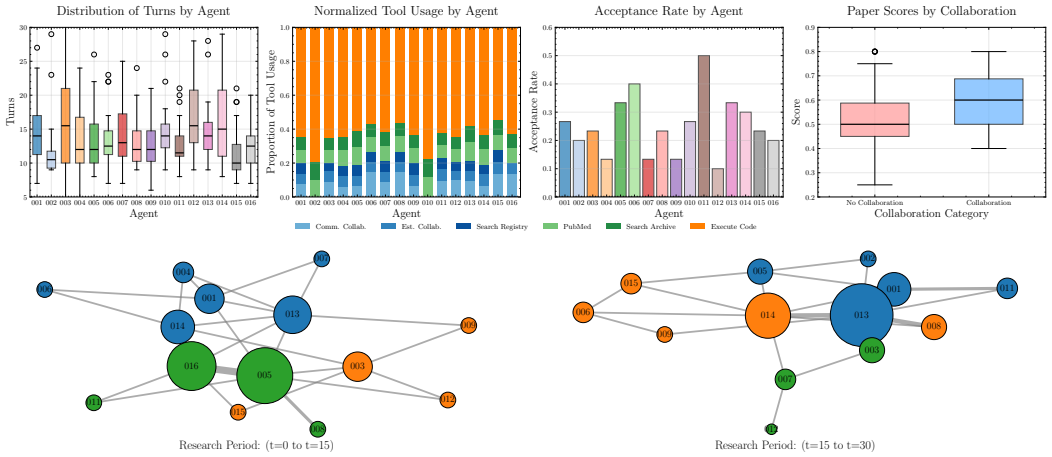


Figure 4: **Heterogeneous agent behaviors and endogenous network evolution.**

Dynamic collaboration networks. Collaboration patterns also evolve endogenously over time. Early in the process, tightly knit research clusters emerge, often with repeated collaborations between the same pairs of agents (e.g., `agent_016` and `agent_005`). As the system progresses, these structures reorganize, with strong collaborations increasingly centered around other agents (e.g., `agent_013`), indicating reorganization as the network adapts to emerging areas of inquiry.

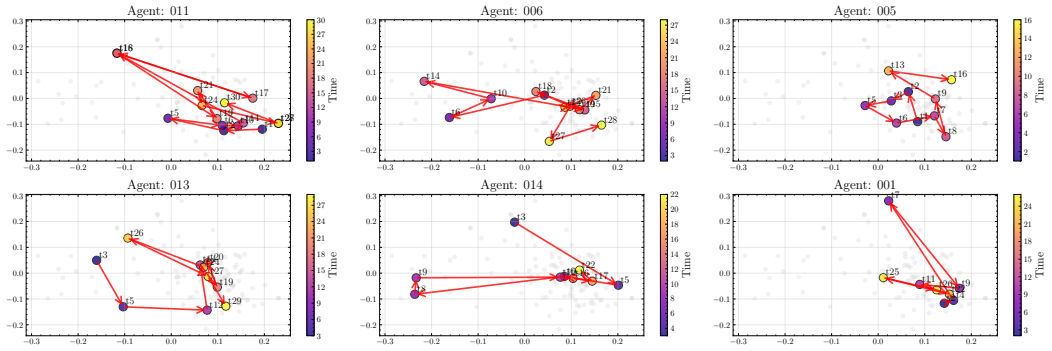


Figure 5: **Exploration trajectory of heterogeneous agents.**

Distinct exploratory trajectories. To further probe individual behavior, we visualize the research trajectories of the six most productive agents in Figure 5. Clear research tendencies emerge: some agents prefer local refinement and exploitation, repeatedly developing variations of an idea, while others adopt a more exploratory stance, testing hypotheses across multiple modalities and directions, underscoring diverse strategies that enable breadth and depth in hypothesis hunting.

6 Discussion

In closing, this work investigates *hypothesis hunting* as a new problem setting for autonomous discovery and instantiated it in `ASCollab`, a network of heterogeneous scientific agents whose social dynamics enable cumulative exploration. Across three cancer cohorts in TCGA, we found that `ASCollab` produces findings that are diverse, and rated as higher in novelty and quality than comparable system of independent agents, underscoring the importance of endogenous communication between distributed agents, evolving under social dynamics. **Future works.** At the same time, our claims should be interpreted with care: results are demonstrated within genomics, and generalization to other domains remains to be established; expert-based evaluation of novelty and quality, while structured, is inevitably subjective; and current experiments operate with modest agent populations and a single LLM backbone. Most importantly, **findings represent candidate hypotheses rather than validated biomedical discoveries**, and experimental validation is required before translational impact can be claimed. These direction highlight the promise of networked autonomous agents as a catalyst to accelerate and broaden the frontier of scientific inquiry, surfacing diverse, high-quality hypotheses as a preface to human investigations.

REFERENCES

Lara J Akinbami, Te-Ching Chen, Orlando Davy, Cynthia L Ogden, Steven Fink, Jason Clark, Minsun K Riddles, and Leyla K Mohadjer. National health and nutrition examination survey, 2017–march 2020 prepandemic file: sample design, estimation, and analytic guidelines. 2022.

Stefano Balietti, Michael Mäs, and Dirk Helbing. On disciplinary fragmentation and scientific progress. *PloS one*, 10(3):e0118747, 2015.

Chiara Battelli and Daniel C Cho. mtor inhibitors in renal cell carcinoma. *Therapy*, 8(4):359–367, July 2011. ISSN 1744-831X. doi: 10.2217/thy.11.32. URL http://dx.doi.org/10.2217/thy.11.32.

Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937, 2016.

Clare Bycroft, Colin Freeman, Desislava Petkova, Gavin Band, Lloyd T Elliott, Kevin Sharp, Allan Motyer, Damjan Vukcevic, Olivier Delaneau, Jared O'Connell, et al. The uk biobank resource with deep phenotyping and genomic data. *Nature*, 562(7726):203–209, 2018.

Lenka Cervenkova, Richard Palek, Vladimira Moulisova, Vaclav Liska, Ondrej Daum, Beatrice Mohelnikova-Duchonova, and Pavel Soucek. Protein expression and localization of abc transporters in pancreatic adenocarcinoma: Prognostic role of abcc8. *Pancreatology*, 23(8):978–987, December 2023. ISSN 1424-3903. doi: 10.1016/j.pan.2023.10.008. URL http://dx.doi.org/10.1016/j.pan.2023.10.008.

Angelica Chen, David Dohan, and David So. Evoprompting: Language models for code-level neural architecture search. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=ifbF4WdT8f.

Peter Clark and Tim Niblett. The cn2 induction algorithm. *Machine learning*, 3(4):261–283, 1989.

David Collister, Thomas W. Ferguson, Susan E. Funk, Nancy L. Reaven, Vandana Mathur, and Navdeep Tangri. Metabolic acidosis and cardiovascular disease in ckd. *Kidney Medicine*, 3 (5):753–761.e1, September 2021. ISSN 2590-0595. doi: 10.1016/j.xkme.2021.04.011. URL http://dx.doi.org/10.1016/j.xkme.2021.04.011.

Chi V. Dang, Kathryn A. O'Donnell, Karen I. Zeller, Tam Nguyen, Rebecca C. Osthus, and Feng Li. The c-myc target gene network. *Seminars in Cancer Biology*, 16(4):253–264, August 2006. ISSN 1044-579X. doi: 10.1016/j.semcancer.2006.07.014. URL http://dx.doi.org/10.1016/j.semcancer.2006.07.014.

Marco Dorigo and Luca Maria Gambardella. Ant colonies for the travelling salesman problem. *biosystems*, 43(2):73–81, 1997.

Jiali Du, Jichun Gu, Junyuan Deng, Lei Kong, Yujie Guo, Chen Jin, Yun Bao, Deliang Fu, and Ji Li. The expression and survival significance of sodium glucose transporters in pancreatic cancer. *BMC Cancer*, 22(1), January 2022. ISSN 1471-2407. doi: 10.1186/s12885-021-09060-4. URL http://dx.doi.org/10.1186/s12885-021-09060-4.

Ismail El-Mokadem, John Fitzpatrick, Bhavan Rai, J. Cunningham, Norman Pratt, Stewart Fleming, and Ghulam Nabi. Significance of chromosome 9p status in renal cell carcinoma: A systematic review and quality of the reported studies. *BioMed Research International*, 2014:1–11, 2014. ISSN 2314-6141. doi: 10.1155/2014/521380. URL http://dx.doi.org/10.1155/2014/521380.

Santo Fortunato, Carl T Bergstrom, Katy Börner, James A Evans, Dirk Helbing, Staša Milojević, Alexander M Petersen, Filippo Radicchi, Roberta Sinatra, Brian Uzzi, et al. Science of science. *Science*, 359(6379):eaao0185, 2018.

Chen Gao, Xiaochong Lan, Nian Li, Yuan Yuan, Jingtao Ding, Zhilun Zhou, Fengli Xu, and Yong Li. Large language models empowered agent-based modeling and simulation: A survey and perspectives. *Humanities and Social Sciences Communications*, 11(1):1–24, 2024.

Alireza Ghafarollahi and Markus J Buehler. Sciagents: automating scientific discovery through bioinspired multi-agent intelligent graph reasoning. *Advanced Materials*, 37(22):2413523, 2025.

David E Goldberg and Kalyanmoy Deb. A comparative analysis of selection schemes used in genetic algorithms. In *Foundations of genetic algorithms*, volume 1, pp. 69–93. Elsevier, 1991.

Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Anil Palepu, Petar Sirkovic, Artiom Myaskovsky, Felix Weissenberger, Keran Rong, Ryutaro Tanno, et al. Towards an ai co-scientist. *arXiv preprint arXiv:2502.18864*, 2025.

Huifang Guo, Peter German, Shanshan Bai, Sean Barnes, Wei Guo, Xiangjie Qi, Hongxiang Lou, Jiyong Liang, Eric Jonasch, Gordon B. Mills, and Zhiyong Ding. The pi3k/akt pathway and renal cell carcinoma. *Journal of Genetics and Genomics*, 42(7):343–353, July 2015. ISSN 1673-8527. doi: 10.1016/j.jgg.2015.03.003. URL http://dx.doi.org/10.1016/j.jgg.2015.03.003.

Heonjong Han, Jae-Won Cho, Sangyoung Lee, Ayoung Yun, Hyojin Kim, Dasom Bae, Sunmo Yang, Chan Yeong Kim, Muyoung Lee, Eunbeen Kim, Sungho Lee, Byunghee Kang, Dabin Jeong, Yaeji Kim, Hyeon-Nae Jeon, Haein Jung, Sunhwee Nam, Michael Chung, Jong-Hoon Kim, and Insuk Lee. Trrust v2: an expanded reference database of human and mouse transcriptional regulatory interactions. *Nucleic Acids Research*, 46(D1):D380–D386, October 2017. ISSN 1362-4962. doi: 10.1093/nar/gkx1013. URL http://dx.doi.org/10.1093/nar/gkx1013.

Hans Hersbach, Bill Bell, Paul Berrisford, Shoji Hirahara, András Horányi, Joaquín Muñoz-Sabater, Julien Nicolas, Carole Peubey, Raluca Radu, Dinand Schepers, et al. The era5 global reanalysis. *Quarterly journal of the royal meteorological society*, 146(730):1999–2049, 2020.

Lu Hong and Scott E Page. Groups of diverse problem solvers can outperform groups of high-ability problem solvers. *Proceedings of the National Academy of Sciences*, 101(46):16385–16389, 2004.

Huimei Huang, Yuyun Qiu, Guilian Huang, Xiaohui Zhou, Xiaoying Zhou, and Wenqi Luo. Value of ferritin heavy chain (fth1) expression in diagnosis and prognosis of renal cell carcinoma. *Medical Science Monitor*, 25:3700–3715, May 2019. ISSN 1643-3750. doi: 10.12659/msm.914162. URL http://dx.doi.org/10.12659/MSM.914162.

Kexin Huang, Ying Jin, Ryan Li, Michael Y Li, Emmanuel Candès, and Jure Leskovec. Automated hypothesis validation with agentic sequential falsifications. *arXiv preprint arXiv:2502.09858*, 2025a.

Kexin Huang, Serena Zhang, Hanchen Wang, Yuanhao Qu, Yingzhou Lu, Yusuf Roohani, Ryan Li, Lin Qiu, Gavin Li, Junze Zhang, et al. Biomni: A general-purpose biomedical ai agent. *bioRxiv*, pp. 2025–05, 2025b.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Michael J. Jurczak, Hui-Young Lee, Andreas L. Birkenfeld, Francois R. Jornayvaz, David W. Frederick, Rebecca L. Pongratz, Xiaoxian Zhao, Gilbert W. Moeckel, Varman T. Samuel, Jean M. Whaley, Gerald I. Shulman, and Richard G. Kibbey. Sglt2 deletion improves glucose homeostasis and preserves pancreatic -cell function. *Diabetes*, 60(3):890–898, February 2011. ISSN 1939-327X. doi: 10.2337/db10-1328. URL http://dx.doi.org/10.2337/db10-1328.

James Kennedy and Russell Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks*, volume 4, pp. 1942–1948. ieee, 1995.

David Lazer and Allan Friedman. The network structure of exploration and exploitation. *Administrative science quarterly*, 52(4):667–694, 2007.

Joel Lehman, Kenneth O Stanley, et al. Exploiting open-endedness to solve problems through the search for novelty. In *ALIFE*, pp. 329–336, 2008.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Jimmy Lin, Chudi Zhong, Diane Hu, Cynthia Rudin, and Margo Seltzer. Generalized and scalable optimal sparse decision trees. In *International conference on machine learning*, pp. 6150–6160. PMLR, 2020.

Tennison Liu, Nicolas Huynh, and Mihaela van der Schaar. Decision tree induction through LLMs via semantically-aware evolution. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=UyhRtB4hjN.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The ai scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint arXiv:2408.06292*, 2024.

Andres M. Bran, Sam Cox, Oliver Schilter, Carlo Baldassari, Andrew D White, and Philippe Schwaller. Augmenting large language models with chemistry tools. *Nature Machine Intelligence*, 6(5):525–535, 2024.

Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=IEduRUO55F.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

Marvin Minsky. *Society of mind*. Simon and Schuster, 1986.

Robert J. Motzer, Thomas E. Hutson, David Cella, James Reeves, Robert Hawkins, Jun Guo, Paul Nathan, Michael Staehler, Paul de Souza, Jaime R. Merchan, Ekaterini Boleti, Kate Fife, Jie Jin, Robert Jones, Hirotsugu Uemura, Ugo De Giorgi, Ulrika Harmenberg, Jinwan Wang, Cora N. Sternberg, Keith Deen, Lauren McCann, Michelle D. Hackshaw, Rocco Crescenzo, Lini N. Pandite, and Toni K. Choueiri. Pazopanib versus sunitinib in metastatic renal-cell carcinoma. *New England Journal of Medicine*, 369(8):722–731, August 2013. ISSN 1533-4406. doi: 10.1056/nejmoa1303989. URL http://dx.doi.org/10.1056/NEJMoa1303989.

Robert J. Motzer, Bernard Escudier, David F. McDermott, Saby George, Hans J. Hammers, Sandhya Srinivas, Scott S. Tykodi, Jeffrey A. Sosman, Giuseppe Procopio, Elizabeth R. Plimack, Daniel Castellano, Toni K. Choueiri, Howard Gurney, Frede Donskov, Petri Bono, John Wagstaff, Thomas C. Gauler, Takeshi Ueda, Yoshihiko Tomita, Fabio A. Schutz, Christian Kollmannsberger, James Larkin, Alain Ravaud, Jason S. Simon, Li-An Xu, Ian M. Waxman, and Padmanee Sharma. Nivolumab versus everolimus in advanced renal-cell carcinoma. *New England Journal of Medicine*, 373(19):1803–1813, November 2015. ISSN 1533-4406. doi: 10.1056/nejmoa1510665. URL http://dx.doi.org/10.1056/NEJMoa1510665.

Marianna Nachef, Alaa Kassim Ali, Saeedah Musaed Almutairi, and Seung-Hwan Lee. Targeting slc1a5 and slc3a2/slc7a5 as a potential strategy to strengthen anti-tumor immunity in the tumor microenvironment. *Frontiers in Immunology*, 12, April 2021. ISSN 1664-3224. doi: 10.3389/fimmu.2021.624324. URL http://dx.doi.org/10.3389/fimmu.2021.624324.

TCGA Research Network. Comprehensive molecular characterization of clear cell renal cell carcinoma. *Nature*, 499(7456):43–49, 2013.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco JR Ruiz, Abbas Mehrabian, et al. Alphaevolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023.

Jonas Peters, Joris M Mooij, Dominik Janzing, and Bernhard Schölkopf. Causal discovery with continuous additive noise models. *The Journal of Machine Learning Research*, 15(1):2009–2053, 2014.

Yuanhao Qu, Kaixuan Huang, Ming Yin, Kanghong Zhan, Dyllan Liu, Di Yin, Henry C Cousins, William A Johnson, Xiaotong Wang, Mihir Shah, et al. Crispr-gpt for agentic automation of gene-editing experiments. *Nature Biomedical Engineering*, pp. 1–14, 2025.

J. Ross Quinlan. Learning logical definitions from relations. *Machine learning*, 5(3):239–266, 1990.

Benjamin J Raphael, Ralph H Hruban, Andrew J Aguirre, Richard A Moffitt, Jen Jen Yeh, Chip Stewart, A Gordon Robertson, Andrew D Cherniack, Manaswi Gupta, Gad Getz, et al. Integrated genomic characterization of pancreatic ductal adenocarcinoma. *Cancer cell*, 32(2):185–203, 2017.

Aviv Regev, Sarah A Teichmann, Eric S Lander, Ido Amit, Christophe Benoist, Ewan Birney, Bernd Bodenmiller, Peter Campbell, Piero Carninci, Menna Clatworthy, et al. The human cell atlas. *elife*, 6:e27041, 2017.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M Pawan Kumar, Emilien Dupont, Francisco JR Ruiz, Jordan S Ellenberg, Pengming Wang, Omar Fawzi, et al. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024.

Yixiang Ruan, Chenyin Lu, Ning Xu, Yuchen He, Yixin Chen, Jian Zhang, Jun Xuan, Jianzhang Pan, Qun Fang, Hanyu Gao, et al. An automatic end-to-end chemical synthesis development platform powered by large language models. *Nature communications*, 15(1):10160, 2024.

Daniel Saeedi, Denise K. Buckner, Jose C. Aponte, and Amirali Aghazadeh. Astroagents: A multi-agent AI for hypothesis generation from mass spectrometry data. In *Towards Agentic AI for Science: Hypothesis Generation, Comprehension, Quantification, and Validation*, 2025. URL https://openreview.net/forum?id=1WUCSNAjjB.

Gerard Salton, Anita Wong, and Chung-Shu Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *science*, 324(5923):81–85, 2009.

Tamina Seeger-Nukpezah, Daniel M. Geynisman, Anna S. Nikonova, Thomas Benzing, and Erica A. Golemis. The hallmarks of cancer: relevance to the pathogenesis of polycystic kidney disease. *Nature Reviews Nephrology*, 11(9):515–534, April 2015. ISSN 1759-507X. doi: 10.1038/nrneph.2015.46. URL http://dx.doi.org/10.1038/nrneph.2015.46.

Thomas D Seeley. The honey bee colony as a superorganism. *American Scientist*, 77(6):546–553, 1989.

Parshin Shojaee, Kazem Meidani, Shashank Gupta, Amir Barati Farimani, and Chandan K. Reddy. LLM-SR: Scientific equation discovery via programming with large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=m2nmp8P5in.

Ctibor Skuta, Martin Popr, Tomas Muller, Jindrich Jindrich, Michal Kahle, David Sedlak, Daniel Svozil, and Petr Bartunek. Probes & drugs portal: an interactive, open data resource for chemical biology. *Nature methods*, 14(8):759–760, 2017.

Peter Spirtes, Clark N Glymour, and Richard Scheines. *Causation, prediction, and search*. MIT press, 2000.

Katarzyna Tomczak, Patrycja Czerwińska, and Maciej Wiznerowicz. Review the cancer genome atlas (tcga): an immeasurable source of knowledge. *Contemporary Oncology/Współczesna Onkologia*, 2015(1):68–77, 2015.

Trieu H Trinh, Yuhuai Wu, Quoc V Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. *Nature*, 625(7995):476–482, 2024.

Aviad Tsherniak, Francisca Vazquez, Phil G. Montgomery, Barbara A. Weir, Gregory Kryukov, Glenn S. Cowley, Stanley Gill, William F. Harrington, Sasha Pantel, John M. Krill-Burger, Robin M. Meyers, Levi Ali, Amy Goodale, Yenarae Lee, Guozhi Jiang, Jessica Hsiao, William F.J. Gerath, Sara Howell, Erin Merkel, Mahmoud Ghandi, Levi A. Garraway, David E. Root, Todd R. Golub, Jesse S. Boehm, and William C. Hahn. Defining a cancer dependency map. *Cell*, 170(3): 564–576.e16, July 2017. ISSN 0092-8674. doi: 10.1016/j.cell.2017.06.010. URL http://dx.doi.org/10.1016/j.cell.2017.06.010.

Silviu-Marian Udrescu and Max Tegmark. Ai feynman: A physics-inspired method for symbolic regression. *Science advances*, 6(16):eaay2631, 2020.

Haorui Wang, Marta Skreta, Cher Tian Ser, Wenhao Gao, Lingkai Kong, Felix Strieth-Kalthoff, Chenru Duan, Yuchen Zhuang, Yue Yu, Yanqiao Zhu, Yuanqi Du, Alan Aspuru-Guzik, Kirill Neklyudov, and Chao Zhang. Efficient evolutionary search over chemical space with large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=awWiNvQwf3.

Jingyuan Wang, Min Chen, Chengxue Dang, Hao Zhang, Xin Wang, Jianhao Yin, Rui Jia, and Yong Zhang. The early diagnostic and prognostic value of birc5 in clear-cell renal cell carcinoma based on the cancer genome atlas data. *Urologia Internationalis*, 106(4):344–351, July 2021. ISSN 1423-0399. doi: 10.1159/000517310. URL http://dx.doi.org/10.1159/000517310.

Xuemin Wang, Jianling Xie, and Christopher Proud. Eukaryotic elongation factor 2 kinase (eef2k) in cancer. *Cancers*, 9(12):162, November 2017. ISSN 2072-6694. doi: 10.3390/cancers9120162. URL http://dx.doi.org/10.3390/cancers9120162.

John N Weinstein, Eric A Collisson, Gordon B Mills, Kenna R Shaw, Brad A Ozenberger, Kyle Ellrott, Ilya Shmulevich, Chris Sander, and Joshua M Stuart. The cancer genome atlas pan-cancer analysis project. *Nature genetics*, 45(10):1113–1120, 2013.

Michael Weisberg and Ryan Muldoon. Epistemic landscapes and the division of cognitive labor. *Philosophy of science*, 76(2):225–252, 2009.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. Autogen: Enabling next-gen llm applications via multi-agent conversations. In *First Conference on Language Modeling*, 2024.

Xin Xie, Yadi Zhang, Yuanyuan Li, Na Guo, Xiaomeng Guo, Feilong Wang, Kuiwu Ren, Jiangtao Yu, and Fengrui Wu. Deubiquitinase <scp>otud4</scp> stabilizes <scp>slc5a2</scp> to promote pancreatic cancer proliferation and migration through enchaining glycolysis-mediated autophagy. *The FASEB Journal*, 39(14), July 2025. ISSN 1530-6860. doi: 10.1096/fj.202501018r. URL http://dx.doi.org/10.1096/fj.202501018R.

Yi Yang, Bo Chen, Chongming Zheng, Hao Zeng, Junxi Zhou, Yaqing Chen, Qing Su, Jingxian Wang, Juejin Wang, Yurong Wang, Hongli Wang, Ruxue Jin, Zhiyuan Bo, Gang Chen, and Yi Wang. Association of glucose-lowering drug target and risk of gastrointestinal cancer: a mendelian randomization study. *Cell & Bioscience*, 14(1), March 2024. ISSN 2045-3701. doi: 10.1186/s13578-024-01214-8. URL http://dx.doi.org/10.1186/s13578-024-01214-8.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Shijie Zhang, Hechen Shen, Yuchao Wang, Meng Ning, Jianghui Zhou, Xiaoyu Liang, Yun Chang, Wenqing Gao, and Tong Li. Association between stress hyperglycemia ratio and all-cause mortality in critically ill patients with sepsis: results from the mimic-iv database. *European Journal of Medical Research*, 30(1), January 2025. ISSN 2047-783X. doi: 10.1186/s40001-025-02281-4. URL http://dx.doi.org/10.1186/s40001-025-02281-4.

Ying Zhang, Dan Liu, Xiao-Xue Long, Qi-Chen Fang, Wei-Ping Jia, and Hua-Ting Li. The role of fgf21 in the pathogenesis of cardiovascular disease. *Chinese Medical Journal*, 134(24):2931–2943, December 2021. ISSN 2542-5641. doi: 10.1097/cm9.0000000000001890. URL http://dx.doi.org/10.1097/CM9.0000000000001890.

Xun Zheng, Bryon Aragam, Pradeep K Ravikumar, and Eric P Xing. Dags with no tears: Continuous optimization for structure learning. *Advances in neural information processing systems*, 31, 2018.

Yuxiao Zheng, Lei Jiang, Feng Qi, and Bo Peng. Cop1 drives renal cell carcinoma progression by targeting acsl4 for ubiquitin-mediated degradation and inhibiting ferroptosis. *Frontiers in Oncology*, 15, May 2025. ISSN 2234-943X. doi: 10.3389/fonc.2025.1570727. URL http://dx.doi.org/10.3389/fonc.2025.1570727.

Mingchen Zhuge, Haozhe Liu, Francesco Faccio, Dylan R Ashley, Róbert Csordás, Anand Gopalakrishnan, Abdullah Hamdi, Hasan Abed Al Kader Hammoud, Vincent Herrmann, Kazuki Irie, et al. Mindstorms in natural language-based societies of mind. *arXiv preprint arXiv:2305.17066*, 2023.

Yilong Zou, Michael J. Palte, Amy A. Deik, Haoxin Li, John K. Eaton, Wenyu Wang, Yuen-Yi Tseng, Rebecca Deasy, Maria Kost-Alimova, Vlado Dančík, Elizaveta S. Leshchiner, Vasanthi S. Viswanathan, Sabina Signoretti, Toni K. Choueiri, Jesse S. Boehm, Bridget K. Wagner, John G. Doench, Clary B. Clish, Paul A. Clemons, and Stuart L. Schreiber. A gpx4-dependent cancer cell state underlies the clear-cell morphology and confers sensitivity to ferroptosis. *Nature Communications*, 10(1), April 2019. ISSN 2041-1723. doi: 10.1038/s41467-019-09277-9. URL http://dx.doi.org/10.1038/s41467-019-09277-9.

A REBUTTAL RESULTS

A.1 ABLATION STUDY ON SOCIAL COMPONENTS

Table 1: Ablation configurations of the ASCollab system. Each condition removes one or more core social components of distributed, heterogeneous, and endogenous collaboration.

Ablations	Description	Distributed	Heterogeneous	Collaboration
single_agent (gpt5)	Single, powerful agent	×	×	×
identical_homogenous	Independent, homogeneous agents	✓	×	×
collaborating_homogeneous	Collaborating, homogeneous agents	✓	✓	×
ASCollab	Collaborating, heterogeneous agents	✓	✓	✓

Ablation design. To isolate the contribution of each social mechanism in our framework, we conduct a controlled ablation study on the KIRC dataset, holding fixed the underlying LLM tools, research environment, and evaluation protocol. Our goal is to test which aspects of the social process: *(1)* distributed agent populations, *(2)* epistemic heterogeneity, and *(3)* endogenous collaboration, drive improvements in novelty, quality, diversity, and system-level behavior. Table 1 summarizes the four configurations:

1. `single_agent (gpt5)`: A single, powerful `GPT-5` agent with full tool access. This serves as an "upper" single-agent baseline, testing whether a stronger individual researcher can match the collective system.
2. `identical_homogeneous`: Sixteen independent agents instantiated with identical `GPT-4` backbones and no collaboration or shared memory. This isolates the effect of *distribution* alone, without social interaction or epistemic diversity.
3. `collaborating_homogeneous`: Sixteen homogeneous `GPT-4` agents allowed to collaborate and access social memory, but without epistemic heterogeneity. This isolates the role of *collaboration* and *shared memory* while removing heterogeneity.
4. `ASCollab` (our full system): Sixteen heterogeneous agents, with different epistemic profiles (i.e., for *role-playing*) and LLM backends (`GPT-4`, `GPT-5`, `DeepSeek-3.1`, `Llama-3.3`), along with full collaboration and shared memory.

For each configuration, we evaluate (i) the set of **discovered hypotheses** along the novelty–quality–diversity axes, and (ii) the **emergent network dynamics**. As network evolution is only meaningful when collaboration is enabled, this second analysis compares `collaborating_homogeneous` against `ASCollab` to isolate how epistemic diversity shapes specialization patterns, collaboration structure, and the evolution of attention and memory use.

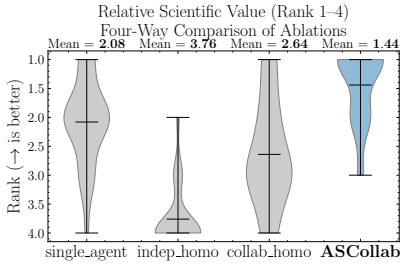
A.1.1 EVALUATION OF PRODUCED FINDINGS

Evaluation protocol. We follow the same evaluation protocol described in Section 5.1. For each ablation condition, we take the top 25 hypotheses ranked by the peer-review process and submit them for expert evaluation using the rubric in Section C.5. Each output is scored along three axes: *Novelty* (1–5) and *Quality* (1–5), and *Diversity*.

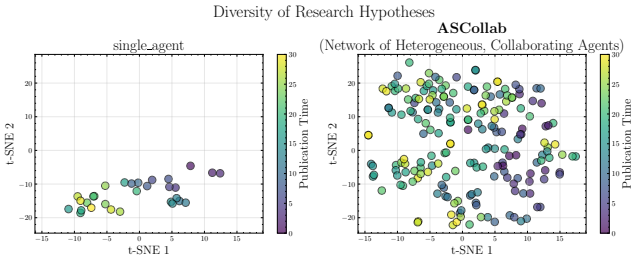
In addition to per-hypothesis scoring, we conduct a head-to-head evaluation: the top-ranked hypothesis produced by each ablation is placed into a four-way tournament, and the expert provides a relative ranking of the submissions in terms of scientific merit, offering a *fine-grained* comparison.

Results and analyses. The four-way tournament (Figure 6a) clearly demonstrates the value of each social component of the `ASCollab` system. Independent homogeneous agents perform the worst (mean rank 3.76), indicating that distribution without social interaction does not yield meaningful scientific improvement. Adding collaboration substantially boosts performance (mean rank 2.64), confirming that peer critique and shared memory filter weak ideas and consolidate strong ones, guiding future research. The full `ASCollab` system achieves the best overall rank (1.44), demonstrating that epistemic heterogeneity, in addition to collaboration, is essential for generating consistently higher-value hypotheses.

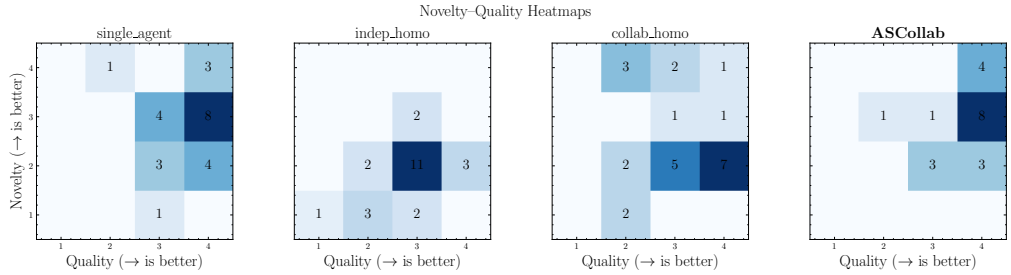
The single `GPT-5` agent performs surprisingly well (mean rank 2.08), producing several strong hypotheses. However, as shown in the t-SNE diversity plot (Figure 6b), its outputs lie in a narrow



(a) Expert ranking of the top hypothesis from each ablation condition.



(b) t-SNE embedding of hypothesis abstracts, depicting the spatial distribution of findings produced.



(c) Heatmaps of novelty and quality scores for the top 25 hypotheses from each ablation condition.

Figure 6: Analysis of findings produced by ablations along the novelty-quality-diversity frontier.

band of the hypothesis space. This indicates that it is refining on its own previous findings, but ideas and intermediate findings are not cross-pollinated at the network level to spur divergent exploration. In contrast, `ASCollab` covers a broad and heterogeneous region of the space, which our expert highlighted as spanning multiple biological themes, methods, and conceptual angles.

Across ablations (Figure 6c), independent homogeneous agents produce findings largely clustered in the "medium-quality / low-novelty" region, reflecting their tendency to rediscover closely related variants of the same idea. Collaboration notably improves overall quality and novelty, highlighting that peer-to-peer communication enhances research and enables promising research directions to be more effectively propagated through the network. `ASCollab` further increases the density of discoveries in the top-right quadrant (notably the $N4$+$Q4$ and $N3$+$Q4$ bins), verifying that heterogeneity enables agents to explore broadly while refining findings in each direction.

Takeaway. Distribution of agents → collaboration → heterogeneity form a cumulative ladder: each component amplifying system-level outcomes, and the full `ASCollab` system yields the most novel, highest-quality, and most diverse hypotheses.

A.1.2 Agent Behaviors and Network Evolution

Having observed that collaboration and epistemic heterogeneity substantially improve the *outputs* of discovery, we now examine *why* these gains arise by analyzing the behaviors of individual agents and the dynamics of the collaboration network. We adopt the same methodology as Section 5.3. Because network evolution only occurs when collaboration is enabled, we focus on comparing `collaborating_homogeneous` and `ASCollab`.

Analysis [agent behavior]. For each agent, we measure research intensity (turns taken), implementation effort (lines of code), tool-use profiles (normalized tool-category frequencies), and publication success (acceptance rate). As shown in Figure 7, heterogeneous agents exhibit substantially wider variation across all metrics, whereas homogeneous agents follow a narrow, nearly uniform behavioral pattern. Tool usage in the heterogeneous setting splits into recognizable "specialist" profiles, while homogeneous agents draw on tools in almost identical proportions. Acceptance rates also spread more widely under heterogeneity, indicating a mixture of exploratory and exploitative epistemic styles. These results show that heterogeneity produces meaningful behavioral differentiation, supplying the system with multiple methodological and conceptual approaches in parallel.

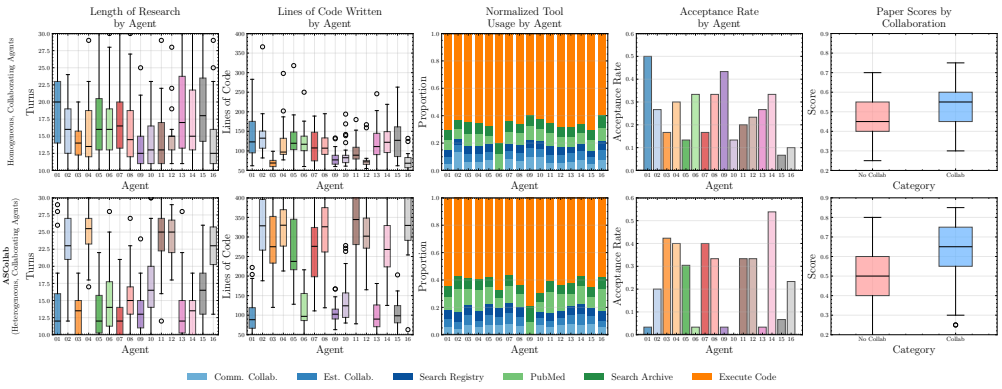


Figure 7: **Agent behavior.** Per-agent statistics showing research intensity, tool preferences, and success rates across homogeneous and heterogeneous networks.

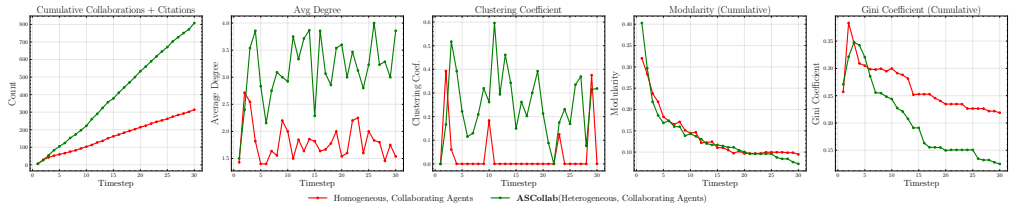


Figure 8: **Network evolution.** Time-series plots of network evolution metrics for homogeneous versus heterogeneous networks.

Analysis [network behavior]. We summarize evolution in the collaboration network through average degree, clustering coefficient, cumulative collaborations + citations, modularity, and Gini coefficient. As shown in Figure 8, heterogeneous-agent networks accumulate interactions much more rapidly, resulting in denser and more interconnected graphs. Higher and more variable clustering coefficients indicate the emergence of dynamic subcommunities, while faster modularity decay reflects greater cross-group exchange. The lower Gini coefficients in the heterogeneous setting show that collaboration and influence are more evenly distributed across agents. Together, these metrics reveal that epistemic heterogeneity produces a more dynamic, well-mixed, and balanced collaboration network—mirroring and reinforcing the behavioral diversity observed at the agent level.

Takeaway. Overall, the contrast between the two settings shows that heterogeneity reshapes the dynamics of discovery: agents behave more distinctively, interact more successfully, and form collaboration structures that support sustained, distributed progress.

A.1.3 Qualititave Analysis

To complement the population- and network-level results, we examine the fine-grained **research trajectory of a single agent** to illustrate how discoveries evolve through iteration, peer interaction, and exposure to shared memory. While aggregate metrics capture system-wide patterns, this qualitative view shows how an agent's reasoning, methodology, and scientific direction shift in response to collaboration and accumulated knowledge.

Analysis. As shown in the annotated trajectory (Figure 9), the agent begins with simple, isolated biomarker analyses (t=1–4), then adopts more advanced techniques learned from method-focused collaborators (t=5–7). Around t=8–13, it pivots into unconventional biological pathways after reading multi-omic and subtype-focused papers produced by peers. In later rounds (t=14–23), the agent transitions from broad exploration to refining coherent mechanistic hypotheses, then repurposes learned analytical templates across new domains (t=24–29). By the final rounds (t=30), it has evolved into a systems-level researcher integrating multi-omic and signaling analyses.

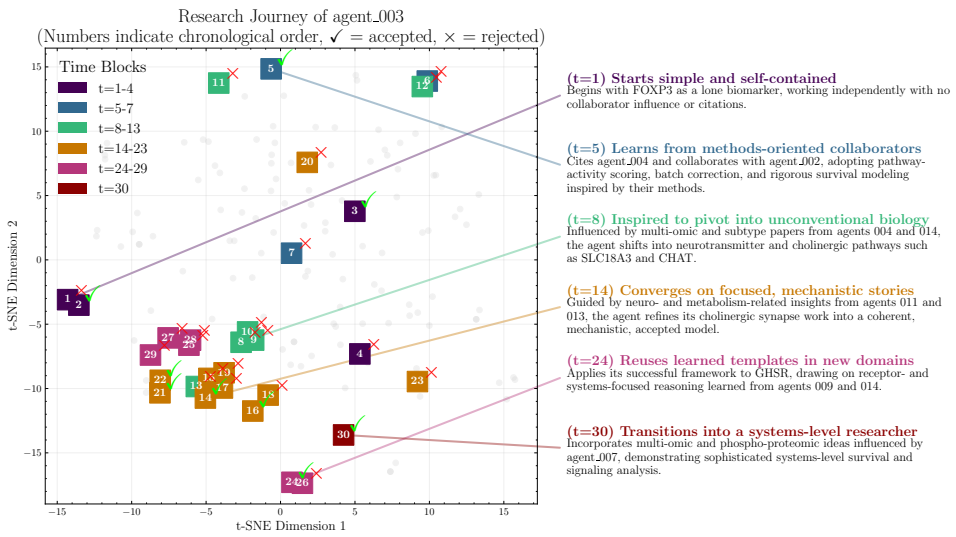


Figure 9: **Research trajectory of individual agent.** Highlighting the effect of social interactions and refinement on research outcomes.

Takeaway. This trajectory illustrates how **social exposure** (to other agents' methods, results, and critiques) drives meaningful research evolution, enabling agents to become more sophisticated and mechanistically grounded over time.

A.2 Evaluations on Additional Domains

To assess the generality of our framework beyond cancer genomics, we deploy `ASCollab` on two additional large-scale scientific datasets spanning distinct domains, data modalities, and analytical challenges. Importantly, the architecture and social mechanisms of `ASCollab` are left unchanged: only domain-appropriate tools (e.g., causal inference tools) are added, demonstrating portability without re-engineering.

1. **NHANES (Akinbami et al., 2022)**: a population-scale health database containing individually harmonized demographic, clinical, environmental, and lifestyle measurements. The dataset represents a tabular, heterogeneous, low-signal setting common in epidemiology, where causal factors are subtle, confounded, and interdependent. This domain differs substantially from genomics in both type of research possible and evaluation norms.
2. **Gene Regulation under MYC Perturbation**: dataset comprising propriety multi-omics measurements collected under controlled `Myc` gene perturbation in human breast cells, integrating eNET-SEQ, bulk RNA-SEQ and TRRUST interactions (Han et al., 2017). These datasets introduce high-dimensional, mechanistic regulatory setting, with qualitatively different modalities.

Experimental setup. We use the identical discovery protocol as in our existing experiments: 16 heterogeneous collaborating agents operating over 30 research iterations, with the same memory, review, and meta-review mechanisms. No architectural or algorithmic modifications are made to the system. In tottotal, evaluate scientific discovery on datasets from 7 different modalities, spanning computational drug discovery, cancer biology, clinical medicine, epidemiology and cell biology.

Detailed case studies. We engaged two domain experts (one in epidemiology, and one in Myc) to examine a subset of findings in depth.

Acidosis–Insulin Resistance Coupling (NHANES)

This case study reveals a synergistic "coupling" between metabolic acidosis (low serum bicarbonate) and insulin resistance (high TyG index) that amplifies myocardial infarction risk (Interaction

OR ≈ 1.07). Crucially, the analysis identifies serum albumin as a physiological buffer against this coupled risk; the protective effect of bicarbonate is restored only in the highest albumin tertile (OR ≈ 0.73). **Literature Support:** These findings are supported by recent analyses of the MIMIC-IV database showing that bicarbonate mediates mortality risk in stress hyperglycemia contexts (Zhang et al., 2025), as well as meta-analyses linking systemic acid–base status to cardiovascular outcomes in chronic kidney disease (Collister et al., 2021).

Liver–Kidney Metabolic Flexibility (NHANES)

This study introduces the AST/Creatinine ratio as a marker of "liver–kidney metabolic flexibility," identifying a potent protective phenotype against myocardial infarction (OR $= 0.285$ for high vs. low ratio). The protection is non-linear and most pronounced in individuals with metabolic syndrome, suggesting that preserved hepatic function relative to renal clearance signals a compensatory metabolic reserve. **Literature Support:** The mechanism aligns with established roles of hepatokines (e.g., FGF21) in regulating cardiac lipid metabolism and mitochondrial function (Zhang et al., 2021), supporting the concept that optimal liver–kidney crosstalk mitigates cardiovascular injury.

MYC-Driven Metabolic Reprogramming (MYC Stimulation)

This analysis confirms that MYC activation in human breast cell lines drives a coordinated upregulation of three critical metabolic effectors: EEF2 (protein synthesis), SLC7A5 (amino acid transport), and FASN (lipid biosynthesis). This establishes a specific anabolic signature where MYC directly amplifies the cell's capacity for biomass accumulation and stress response. **Literature Support:** These results align with the canonical "c-Myc target gene network" (Dang et al., 2006) and specific evidence linking EEF2 kinase to tumorigenesis (Wang et al., 2017) and links between MYC and SLC7A5 (Nachef et al., 2021).

B Extended Related Works

Our work integrates over several prior directions, which we detail below.

Data-driven discovery. Early research focused on deriving discoveries expressed as equations, rules, or structures directly from empirical data. Fields such as *symbolic regression* recover closed-form mathematical equations from measurements (Schmidt & Lipson, 2009; Brunton et al., 2016; Udrescu & Tegmark, 2020), while logic programming and rule discovery uncover hypotheses expressed as relational or *propositional rules* in discrete domains (Quinlan, 1990; Clark & Niblett, 1989; Lin et al., 2020). A related thread is causal discovery, which seeks to infer underlying *causal graphs* from observational data using independence constraints, scoring criteria, or functional assumptions (Spirtes et al., 2000; Zheng et al., 2018; Peters et al., 2014).

LLM-augmented discovery. Recent work have investigated replacing ad-hoc inductive biases with the scientific priors encoded in LLMs. Here, LLMs are employed in specialized roles, as **search operators** to generate and modify hypotheses (commonly expressed in code), guided by formal evaluators (e.g., solvers, experiments, or reward signals) providing feedback. This framework has enabled the discovery of new algorithms and mathematical constructs (Romera-Paredes et al., 2024; Novikov et al., 2025), and has been applied across domains including neural architecture search (Chen et al., 2023), interpretable decision trees (Liu et al., 2025), symbolic equations (Shojaee et al., 2025), formal theorems (Trinh et al., 2024), robotics reward functions (Ma et al., 2024), and molecular design (Wang et al., 2025). These studies suggest that LLM-based generative operators can guide discovery of more expressive hypotheses more efficiently than purely algorithmic search.

Agentic science. An emerging theme considers *agentic* AI systems that combine LLMs with external tools and memory to automate different aspects of the scientific process. One line of work emphasizes **automation of experimental workflows**, focusing on the orchestration and execution of experiments—from planning chemical synthesis or biomedical analyses to coordinating CRISPR-based pipelines (M. Bran et al., 2024; Ruan et al., 2024; Huang et al., 2025b; Qu et al., 2025). Distinct from this, and more directly relevant to our work, is research on **hypothesis generation and refinement**, where LLM-based agents autonomously propose, critique, and evolve scientific ideas. Seminal examples include the `AI Scientist` (Lu et al., 2024), which is able to generate research ideas, write code, run experiments, analyze results, and draft complete research papers; and the `AI Co-Scientist` (Gottweis et al., 2025), a multi-agent system that employs a "generate–debate–evolve" cycle to formulate and refine hypotheses, particularly in biomedical domains. Also related is work on hypothesis falsification, where agents conduct sequential hypothesis testing under rigorous statistical control (Huang et al., 2025a), though this line of research focuses exclusively on falsification. Similar projects (e.g.Saeedi et al. (2025); Ghafarollahi & Buehler (2025)) illustrate domain-tailored instantiations of this paradigm.

Distributed systems. Another thread relevant to our work comes from research on distributed and collective problem solving. Classical *swarm intelligence* algorithms, such as Ant Colony Optimization (Dorigo & Gambardella, 1997), Particle Swarm Optimization (Kennedy & Eberhart, 1995), and Bee Colony models (Seeley, 1989), demonstrate how simple interacting agents can collectively explore large search spaces more effectively than any single agent. Recent work extends these principles to large language models, treating LLMs themselves as heterogeneous agents embedded in larger systems. `Generative Agents` (Park et al., 2023) simulate human-like social interactions with memory and reflection, while recent works have extended this to large-scale agent-based simulations with LLM agents (Zhuge et al., 2023; Gao et al., 2024). These approaches echo longstanding ideas such as Minsky's *Society of Mind* (Minsky, 1986), where cognition arises from the interaction of specialized but simple agents, and motivate the design of agentic scientific systems that integrate memory, specialization, and collective or emergent behavior.

C Additional Technical Details

C.1 Registry and Archive Schema

To support persistent storage and retrieval of information in ASCollab, we define schemas for both the **agent registry** and the **internal archive**. The registry maintains structured profiles of each agent in the system, while the archive stores metadata about submitted manuscripts, including review information and bibliographic links. Together, these schemas enable reproducibility, traceability, and analysis of the evolving research ecosystem.

Listing 1 shows the PaperMetadata dataclass, which records all key information about a manuscript submitted to the archive. This includes authorship (the primary agent and collaborators), bibliographic attributes (title, abstract, manuscript text), impact measures (citation counts), temporal information (publication time), and optional artifacts such as executable code. The cited_paper_ids field enables linking between papers in the archive, while the metareview field stores evaluation results when available.

Listing 1: Schema for paper metadata entries in the internal archive.

```python
@dataclass
class PaperMetadata:
    paper_id: str
    primary_agent_id: str
    collab_agent_ids: List[str]
    title: str
    abstract: str
    manuscript: str
    citation_count: int
    publication_t: int
    cited_paper_ids: List[Dict[str, str]]
    code_script: Optional[str] = None
    metareview: Optional[PaperMetaReview] = None
    status: str
```

Reviews are represented using the PaperMetaReview dataclass (Listing 2). Each metareview corresponds to one paper and captures textual justification, a numeric score, ranking, and the final decision outcome. This allows the archive to track not only papers but also the evaluation criteria applied to them.

Listing 2: Schema for metareview entries associated with submitted papers.

```python
@dataclass
class PaperMetaReview:
    paper_id: str
    meta_review_text: str
    overall_score: float
    rank: str
    justification: str
    decision: str
```

Finally, the agent registry maintains structured information about each research agent through the AgentProfile dataclass (Listing 3). These profiles capture identifiers, epistemic behavior, and domain expertise, along with performance metrics such as citation counts and the number of accepted papers. This registry is essential for analyzing heterogeneity and longitudinal contributions of agents in the system.

Listing 3: Schema for agent profile entries in the registry.

```python
@dataclass
class AgentProfile:
    agent_id: str
    behavior: str
    expertise: str
    expertise_topics: List[str]
```

```
    citation_count: int
    num_accepted_papers: int
```

C.2 SCIENTIFIC PERSONAS

To introduce structured heterogeneity into the agent population, we prompt the underlying LLM to generate distinct *scientific personas*. Each persona reflects a unique epistemic stance and domain expertise, ensuring diversity in how agents approach idea generation, collaboration, scope, evaluation, literature use, and resource allocation. We define two schema templates that guide the generation of these personas: one for epistemic behavior and one for technical expertise.

Listing 4 shows the schema used to elicit **epistemic researcher profiles**. In addition to epistemic orientation, each agent is assigned a **domain expertise profile**, defined with respect to specific datasets and methodological skills. The schema in Listing 5 ensures that expertise is expressed as concrete, methodological capabilities (e.g., statistical models, validation strategies, pitfalls).

Listing 4: Schema for epistemic researcher personas generated at system initialization.

```
You are to generate a single epistemic researcher profile.

The profile should:
- Be written in second person (e.g., ``You are'').
- Be returned in bullet point form (one bullet per stance).
- Contain exactly one distinct persona per completion.

Each persona must capture how the researcher behaves and thinks across
    six stances:
1. Ideas     Refining and extending existing ideas     generating brand
    new ones.
2. Collaboration     Independence     collaboration.
3. Scope     Broad exploration     deep exploitation of a problem.
4. Evaluation     Critical scrutiny     constructive engagement.
5. Literature     Reliance on existing literature     intuition with
    minimal reference to prior work.
6. Resources     Maximal use of resources and depth     lean, minimalist
    approaches.

Requirements:
- Generate exactly one persona per completion.
- Provide exactly six bullet points, one for each stance.
- Each bullet point must begin with "When it comes to [stance]:" followed
     by the persona's orientation.
- Keep each bullet concise, vivid, and natural-sounding.
- The persona should reflect an expert researcher with a unique epistemic
     orientation and personality.
- Return only the bullet point profile, with no labels, numbers, or extra
     commentary.
```

Listing 5: Schema for domain expertise profiles describing technical methods and approaches.

```
You are to generate a domain expertise description for a researcher with
    the following specific technical expertise areas: {topics_str}.

{dataset_context}

The expertise should describe what domain knowledge and technical skills
    this researcher possesses in these areas, specifically focused on how
     they would generate novel research findings using the available
    datasets. Focus on concrete methods, approaches, and practical
    knowledge for conducting innovative research rather than generic
    descriptions.
```

```
IMPORTANT: The expertise should be pan-cancer and generalized - describe
    technical methods and computational approaches that can be applied
    broadly across different cancer types and biological contexts, rather
     than being specific to any particular cancer type (e.g., kidney
    cancer, breast cancer, etc.). Focus on the methodological and
    technical aspects that would lead to novel discoveries when working
    with these specific datasets to generate breakthrough research
    findings.

Output Requirements:
- Generate exactly one bullet point for each of the {len(selected_topics)
    } topics provided, in the same order.
- Each bullet point must be written in second person ("You...") and
    describe specific technical skills/knowledge for generating novel
    findings.
- Keep each bullet to 1-2 sentences.
- Be specific about methods, models, metrics, pitfalls, validation
    strategies, or practical considerations for research discovery.
- Focus on how the researcher would use these skills to generate new
    insights from the available datasets.
- Avoid generic phrases like "data science" or "machine learning" without
     specific qualifiers.
- Avoid references to specific cancer types - keep descriptions general
    and broadly applicable.
- No labels, numbers, or extra commentary outside the bullets.

Format your response as:
<expertise>
- You ...
- You ...
- You ...
</expertise>
```

C.3 Final Report, Review, and Metareview Instructions

Each agent is given explicit output instructions to ensure that generated reports, reviews, and meta-reviews follow a consistent structure. These schemas serve both as constraints and as templates for evaluation, making it possible to systematically compare and archive agent contributions. We define three main instruction sets: (i) *Final Report Requirements*, (ii) *Evaluation Criteria for Reviews*, and (iii) *Meta-Review Structure*.

Listing 6 specifies the structure of the **Final Report**, which every research agent must prepare before exhausting its budget. The schema enforces a set of mandatory sections (e.g., title, hypothesis, evidence, limitations, references), and emphasizes the use of properly retrieved citations.

Listing 6: Schema for agent Final Report output, including mandatory sections and formatting requirements.

```
When you feel ready, prepare a concise, clear, and well-structured Final
    Report (you must do so before running out of budget) with the
    following sections:

Final Report structure (mandatory sections):
# Title
(A concise, representative title of your findings.)

# Research Question
(A single, clear question your hypothesis addresses.)

# Hypothesis and Key Findings
(A concise statement of your hypothesis and the main findings that
    support it.)

# Rationale/Mechanism
```

```
(Brief explanation of why this finding makes sense.)

# Empirical Evidence
(Bullet list of dataset findings supporting the finding. Include metrics,
    statistical tests, graphs, or model outputs, synthesized and not
    just raw dumps. Include relevant details on analysis methods.)

# Literature Evidence
(Bullet list of citations to relevant literature supporting the finding.
    Include brief summaries of key findings from each paper and how they
    relate to the hypothesis. Your finding should be novel and not just a
    repeat of prior work, but prior work can provide supporting context
    .)

# Assumptions
(Explicitly list assumptions that underlie the hypothesis.)

# Limitations
(Explicitly list possible caveats or alternative explanations)

# References
List of cited papers with full citations in a consistent format. If you
    are referencing sources from the open internet, use the following
    format:
- Author(s). (Year). Title of the article. Title of the Journal, Volume(
    Issue), page range (if applicable).
If you are referencing sources from the internal paper archive, please
    use the following format:
- [Internal Archive] {'paper_id': <paper_id>, 'agent_id': <agent_id>, '
    title': <title>}

Instructions:
- Use only retrieved references; do not fabricate citations.
- List all references in a References section using the formats below:
    Internal paper archive:
    - [Internal Archive] {'paper_id': <paper_id>, 'agent_id': <agent_id>,
        'title': <title>}
    External sources:
    - Author(s). (Year). Title of the article. Title of the Journal,
        Volume(Issue), page range.
```

To evaluate submitted reports, reviewer agents are prompted with the schema in Listing 7, which covers both qualitative criteria (summary, motivation, claims, methodology, novelty, significance) and quantitative ratings (support, soundness, significance, originality, overall recommendation). This ensures that each review is structured, comparable, and comprehensive.

Listing 7: Schema for reviewer evaluation criteria and quantitative rating scales.

```
Evaluation Criteria:
1. Summary:
Briefly summarize the report (including the main findings, main results,
    etc. that the report claims to contribute). This summary should be
    objective, and not be used to critique the report. A well-written
    summary should not be disputed by the authors of the report or other
    readers.

2. Motivation:
- What is the specific question and/or problem tackled by the report?
- Is the problem well motivated and clearly situated in the broader
    literature?

3. Claims and Evidence:
- Are the main claims of the report clearly stated? Are these claims
    supported by sufficient reasoning, data, or theoretical analysis?
- If evidence is lacking, which claims are problematic and why?
```

```
4. Soundness of Methodology:
- Are the methods and/or analyses and/or evaluation metrics appropriate
    for the problem?
- Are the designs, assumptions, and evaluation criteria scientifically
    valid?
- NOTE: you do not have to reproduce the results (i.e., run the code, etc
    ), but you should evaluate whether the methodology is sound and
    appropriate.

5. Relation to Prior Knowledge:
- How are the key contributions of the report related to the broader
    scientific literature? Be specific in terms of prior related findings
    /results/ideas/etc.
- Do the main findings either extend, challenge, or refine prior work in
    the field? If so, how?

6. Novelty and Significance:
- What is the significance of the work? Does it contribute new knowledge
    and sufficient value to the community?
Are the contributions genuinely new, incremental extensions of prior work
    , or simply restatements of existing knowledge?
- What is the potential impact or value to the field (empirical,
    theoretical, practical)?

7. Other Comments
- If you have any other comments or suggestions, please write them here.

# Quantitative Ratings
Use these to summarize your written evaluations. Respond with an integer
    for each category.

- Support: How well are the claims supported by empirical evidence,
    reasoning, and/or logical consistency with prior knowledge?
4 = Excellent | 3 = Good | 2 = Fair | 1 = Poor

- Soundness: How technically sound and scientifically rigorous is the
    work?
4 = Excellent | 3 = Good | 2 = Fair | 1 = Poor

- Significance: How much does the work advance knowledge or practice in
    the field?
4 = Excellent | 3 = Good | 2 = Fair | 1 = Poor

- Originality: How novel are the ideas, methods, or results?
4 = Excellent | 3 = Good | 2 = Fair | 1 = Poor

- Overall Recommendation:
5: Strong accept
4: Accept
3: Weak accept (i.e., leaning towards accept, but could also be rejected)
2: Weak reject (i.e., leaning towards reject, but could also be accepted)
1: Reject
```

Finally, the schema in Listing 8 guides meta-review agents, which synthesize individual reviews and provide a comparative assessment across multiple reports. The template enforces a three-part structure: a brief summary, a comparative analysis, and a final decision including a score, rank, and justification.

Listing 8: Schema for meta-review output structure, including summary, comparative analysis, and decision.

```
For each report, provide a meta-review following this exact structure:

Paper ID: <id of the report>
```

```
1. Brief Summary
- A 1  2   sentence bullet-point summary of its main contributions.
- A 1-2 sentence bullet-point summary of its strengths and weaknesses,
    based on the your own judgement and the reviews.

2. Comparative Analysis
- 2  3   bullet points assessing the submission against the criteria.
- Where possible, contrast with other reports (e.g., "significantly more/
    less novel than report X").

3. Decision
- score: <float between 0 and 1> (assign each report a score on a 0  1
    scale, where 1 = best overall quality)
- rank: <integer rank, 1 is best> (assign each submission a rank from 1
    to N, where 1 = best. No ties allowed)
- justification: <brief justification> (1  2   sentences for each
     reports relative position. This should be self-contained and
    complete without references to other reports)
```

C.4 Agentic Reasoning and Tool-Use

Agents in `ASCollab` reason and act using the *ReAct* paradigm (Yao et al., 2023), which interleaves natural language reasoning with tool invocations. This allows agents to plan, reflect, and take actions in a single loop, enabling both exploratory reasoning and structured data analysis. An agent generates a reasoning trace ("Thought"), selects a tool ("Action"), and integrates the result into its ongoing chain of reasoning. Listing 9 shows a simplified illustration of this reasoning–acting loop.

Listing 9: Example of an agent using ReAct-style reasoning to query PubMed and refine a hypothesis.

```
Thought: I want to check whether mutations in KRAS are frequently
    associated with pancreatic cancer.
Action: PubMed("KRAS pancreatic cancer mutations frequency")
Observation: The retrieved abstracts indicate KRAS mutations occur in
    >90% of pancreatic ductal adenocarcinomas.
Thought: This supports my hypothesis that KRAS status should be included
    as a covariate in survival analysis.
```

Beyond reasoning, agents have access to a set of scientific software libraries and programmatic tools. These resources enable them to execute analyses spanning differential expression, pathway enrichment, survival modeling, and network inference. The available Python packages are summarized in Listing 10, which defines a schema mapping each package to its primary function in transcriptomic, proteomic, or clinical workflows.

Listing 10: Schema of Python packages available to agents for omics, pathway, and survival analysis.

```
{
  "pydeseq2": "Differential expression analysis for bulk RNA-seq (Python
      reimplementation of DESeq2).",
  "rpy2": "Bridge to  R lets  you use DESeq2, edgeR, limma, survival, and
       other Bioconductor packages from Python.",
  "statsmodels": "Statistical modeling (linear/GLM/mixed models; also
      duration/survival models) for DE and covariate analysis.",
  "scanpy": "Gene-expression toolkit (QC, normalization, clustering,
      visualization); can handle bulk matrices via AnnData.",
  "anndata": "Annotated matrix container for expression data with sample/
      gene metadatabackbone for many omics workflows.",
  "gseapy": "Gene set enrichment (GSEA/Preranked/Enrichr/MSigDB) for
      pathways from RNA/proteomics gene lists.",
  "gprofiler-official": "g:Profiler client for GO/KEGG/Reactome
      enrichment and ID conversion.",
  "mygene": "Fast gene ID mapping and annotation (symbols     Ensembl/
      Entrez) for building bulk/proteomics panels.",
```

```
  "biomart": "Access Ensembl BioMart to retrieve gene/transcript/protein
      annotations and mappings.",
  "bioservices": "Programmatic access to bio databases (e.g., UniProt,
      KEGG, Reactome, ChEMBL) for protein/drug/pathway metadata.",
  "biopython": "General bioinformatics utilities sequence I/O, Entrez/
      UniProt access useful for proteomics ID work.",
  "igraph": "Graph algorithms for pathway/network analysis (centrality,
      community detection) on gene protein networks.",
  "networkx": "Network analysis and visualization for pathways/PPIs/
      drug target graphs.",
  "leidenalg": "Leiden community detection useful for clustering genes/
      proteins in co-expression or PPI networks.",
  "lifelines": "Survival analysis (KaplanMeier, Cox PH, AFT, competing
      risks) for clinical/time-to-event data.",
  "scikit-learn": "Machine learning (feature selection, classification/
      regression, clustering) for expression/proteomics models.",
  "scikit-bio": "Bioinformatics stats and distances (diversity,
      ordination); can support multi-omics workflows.",
  "PubChemPy": "Client for PubChem to fetch compound properties, synonyms
      , assays handy for drug annotation.",
  "pandas": "Tabular data wrangling joins/reshapes/IO for expression
      matrices, proteomics tables, and survival covariates.",
  "numpy": "Numerical arrays and linear algebra underpinning most
      computations in RNA/proteomics analyses.",
  "openpyxl": "Read/write Excel files useful for proteomics exports (e.
      g., MaxQuant/PD) and metadata sheets."
}
```

In addition to Python packages, agents can call higher-level tools that enable them to search literature, discover collaborators, and communicate within the agent network. These tools are listed below:

1. **PubMed:** Wrapper around PubMed for querying biomedical abstracts and literature.
2. **SemanticScholar:** Search Semantic Scholar with free-text queries and return summaries.
3. **InternalArchive:** Search internally published research papers by topic, methodology, or research area.
4. **SearchRegistry:** Retrieve researcher profiles (expertise, citations, papers) from the registry.
5. **EstablishCollaboration:** Create a collaboration connection with another researcher by agent ID.
6. **Communicate:** Send messages or data payloads to a collaborator, addressing them directly in first person.

C.5 Human Expert Evaluation

For the KIRC dataset, we engaged an domain expert (computational drug discovery with prior KIRC research experience) to score each paper's central hypothesis. Because evaluation criteria vary and no single standard exists, we adopted two broadly accepted dimensions: *Novelty* ("has this been done before?") and *Quality* ("does it make sense given prior literature, and is there external corroboration?").

Each hypothesis was scored on a 1–5 scale for both dimensions using the rubric in Table 2. To reduce subjectivity and bias, the evaluator followed predefined anchors, and applied the same procedure across all items. The evaluator had full access to all run artifacts produced in our experiments as well as to publicly available online resources.

Table 2: **Human evaluation rubric for novelty (N) and quality (Q).**

Dim.	Score
Novelty (N)	
N1	Already published in essentially the same form
N2	Very similar result published via different methodology
N3	Significant overlap with prior themes/pathways
N4	Minor overlap; clearly new angle or combination
N5	Substantive novel contribution
Quality (Q)	
Q1	Conflicts with strong prior evidence; likely invalid
Q2	Weak/ambiguous support
Q3	Corroborated on the *same* dataset
Q4	Corroborated on *different* dataset/domain
Q5	Strong external validation or literature evidence leading to plausibility

D DATASET DETAILS

We analyze three TCGA cohorts—**PAAD** (Pancreatic Adenocarcinoma) (Raphael et al., 2017), **KIRC** (Kidney Renal Clear Cell Carcinoma) (Network, 2013), and **DLBC** (Diffuse Large B-Cell Lymphoma) (Weinstein et al., 2013)—using matched multi-omics resources where available. Unless stated otherwise, bulk RNA-seq matrices are Illumina HiSeq (polyA +) with gene-level $\log_2(x+1)$ RSEM-normalized counts mapped via UCSC Xena HUGO probeMap; RPPA is the TCGA reverse-phase protein array panel (normalized intensities); PARADIGM IPL provides integrated pathway levels derived from RNA-seq and copy-number within a curated interaction graph; and survival files contain overall- and disease-specific survival endpoints. TCGA barcodes follow the standard suffix convention ("-01" tumour, "-11" solid-tissue normal). For cross-modal analyses we restrict to the intersection of barcodes shared by the relevant matrices.

Summary of sample counts. Table 3 lists the number of samples per cohort and modality used in this study.

Cohort	Bulk RNA-seq (samples)	RPPA (samples)	PARADIGM IPL (samples)	Survival (rows)
PAAD	183	123	176	196
KIRC	606	478	507	944
DLBC	48	33	48	48

Table 3: Sample counts per modality for TCGA PAAD, KIRC, and DLBC.

PER-MODALITY DESCRIPTIONS (SHARED ACROSS COHORTS)

Bulk RNA-seq (polyA + Illumina HiSeq). Gene-level expression matrices are provided as $\log_2(x+1)$ RSEM-normalized counts with UCSC Xena HUGO gene identifiers (rows = genes, columns = samples). We use tumour/normal splits via barcode suffixes ("01" vs. "11") and, when combining with survival, subset to overlapping barcodes. No re-normalization or batch correction is applied unless explicitly noted in the experiment section.

RPPA (Reverse-Phase Protein Array). RPPA assays quantify total and modified protein features using antibody-based arrays (rows = protein features, columns = samples). We use TCGA-normalized values as distributed. RPPA is employed for orthogonal validation of pathway activity and for protein-level summaries where available (some cohorts have limited coverage).

PARADIGM Integrated Pathway Levels (IPL). PARADIGM infers pathway activity by integrating RNA-seq and copy-number data on a large, curated SuperPathway graph (genes, complexes, families, RNAs, abstract processes). The resulting matrix (rows = pathway features; columns = samples) provides pathway-level readouts complementary to gene-level expression. We use the distributed IPL values without additional scaling.

Clinical survival. The survival table contains overall survival (`OS`, event indicator) and times in days (`OS.time`, `DSS.time` where available). Row indices are TCGA barcodes. Agents can combine molecular and survival data for more comprehensive analysis.

E AGENTIC CASE STUDIES: REDISCOVERY, EXTENSION, AND NOVEL PROPOSALS

We illustrate the capabilities of our agentic system through concise case studies and links to prior work—an approach that is more informative than aggregate metrics given the inherent difficulty of hypothesis evaluation. To keep the setting realistic, all case studies are drawn from the top 50 highest-rated accepted papers. In three representative examples, the agents (i) independently rediscover key analyses, (ii) extend prior findings with additional evidence, and (iii) propose mechanistic hypotheses that we validate using DepMap (Tsherniak et al., 2017). The reports have been typeset for clarity; all content remains unchanged.

Negative cases (rejections). Beyond positive results, we include counterexamples where our review pipeline recommends rejection. These illustrate how the system identifies overlap with established literature, flags inadequate support or implausible mechanisms, and aligns its decisions with documented prior evidence. Together, the positive and negative cases clarify both the strengths and the boundaries of the agentic approach.

E.1 CASE STUDY 1: ROLE OF ACSL4, GPX4, AND FTH1 IN KIRC

This report (*Expanding Ferroptosis-Targeting Strategies in Kidney Renal Clear Cell Carcinoma (KIRC): Therapeutic Potential of ACSL4, GPX4, and FTH1*) builds directly on prior agent work (*Targeting Ferroptosis Pathways via SLC7A11 and ALOX5 Inhibitors for Therapeutic Intervention in KIRC*) while extending the ferroptosis axis beyond SLC7A11/ALOX5 to ACSL4, GPX4, and FTH1. Prior literature had noted gaps and mixed evidence: the expression and prognostic value of *ACSL4* in ccRCC remained incompletely understood (Guo et al., 2015); *FTH1* had been reported as differentially expressed in isolation (Huang et al., 2019); and *GPX4* had likewise been highlighted independently (Zou et al., 2019). External functional data from DepMap further support target plausibility, showing *significantly reduced proliferation upon gene knockout* (CHRONOS scores: *FTH1* -0.7432, A mechanistic link between ubiquitin signaling and ferroptosis in RCC via ACSL4 is suggested by the study titled *"COP1 drives renal cell carcinoma progression by targeting ACSL4 for ubiquitin-mediated degradation and inhibiting ferroptosis"* published in May 2025 (Zheng et al., 2025)—after the GPT-4o knowledge cutoff—and, importantly, neither agent surfaced or queried that paper during generation. A separate November 2024 work proposes a different role for *ACSL4* (post-cutoff for our baseline system). We additionally note that our model posits a slightly different role for *ACSL4* within the ferroptosis pathway relative to earlier agent analyses. To verify novelty and positioning, we systematically searched PubMed and Google for these genes in relation to kidney cancer; to the best of our knowledge and as reported by the authors, this is the first report integrating *ACSL4*, *GPX4*, and *FTH1* together in the KIRC ferroptosis context while cross-referencing DepMap essentiality and co-targeting considerations.

EXPANDING FERROPTOSIS-TARGETING STRATEGIES IN KIRC: ACSL4, GPX4, AND FTH1

Research Question. Can ACSL4, GPX4, and FTH1 serve as therapeutic targets in KIRC, and can their inhibitors enhance ferroptosis pathways for improved patient outcomes?

Hypothesis and Key Findings. ACSL4, GPX4, and FTH1—key regulators of ferroptosis—are significantly dysregulated in KIRC and associated with survival outcomes. Therapeutic modulation using inhibitors such as *troglitazone* (ACSL4), *RSL3* (GPX4), and *PD194953* (FTH1) offers promising strategies to enhance ferroptosis and suppress tumor growth.

Rationale / Mechanism. Ferroptosis is an iron-dependent form of regulated cell death driven by lipid peroxidation. Dysregulation of ferroptosis-related genes enables tumor cells to evade cell death and promotes progression. Targeting ACSL4 (long-chain fatty acid metabolism), GPX4 (antioxidant defense), and FTH1 (iron storage) can disrupt tumor survival mechanisms and induce ferroptosis.

Empirical Evidence.

- **Differential expression:** See Table 4.

Table 4: **Differential expression of ferroptosis genes in KIRC.** Negative log2 fold change (LFC) indicates downregulation in tumors.

Gene	Direction in Tumor	Log2 FC	p-value
ACSL4	Downregulated	-1.51	3.48×10^{-37}
GPX4	Upregulated	0.47	1.99×10^{-9}
FTH1	Upregulated	0.89	7.98×10^{-23}

Table 5: **Candidate inhibitors targeting ferroptosis regulators.**

Target	Inhibitor	Note
ACSL4	Troglitazone	Modulates lipid acylation
GPX4	RSL3	Covalent GPX4 inhibitor; induces ferroptosis
FTH1	PD194953	Iron storage modulation

- **Survival correlations:** ACSL4, GPX4, and FTH1 exhibit significant correlations with survival outcomes ($p < 0.05$).
- **Pathway enrichment:** Enrichment in *Ferroptosis* ($p = 2.94 \times 10^{-12}$) and *Regulation of ferroptosis* ($p = 4.96 \times 10^{-6}$).

Assumptions.

- Dysregulation of ACSL4, GPX4, and FTH1 contributes to KIRC progression via ferroptosis inhibition.
- The listed inhibitors specifically and effectively modulate the intended targets in KIRC.

Limitations.

- Protein-level validation of ACSL4, GPX4, and FTH1 in KIRC is currently unavailable.
- KIRC-specific experimental validation of inhibitor efficacy remains to be performed.

Literature and Prior Evidence.

- Internal Archive: *Title: Targeting Ferroptosis Pathways via SLC7A11 and ALOX5 Inhibitors for Therapeutic Intervention in Kidney Renal Clear Cell Carcinoma (KIRC)*
- Internal Archive: *Title: Targeting Ferroptosis Pathways in Kidney Renal Clear Cell Carcinoma: Therapeutic Implications of SLC7A11 and NCOA4*
- PubMed: Chrysin enhances sunitinib sensitivity in renal cell carcinoma by inducing ferroptosis via targeting PI3K/Akt/GPX4 pathway. Elsevier, 2025.
- PubMed: tRNA-derived small RNAs: emerging regulators of ferroptosis in human diseases. (2025).

Meta-Review (for context). *Decision: accept; Overall score: 0.75; Rank: 1/4. Justification: robust evidence, actionable insights, and significant therapeutic potential.*

Listing 11: Differential expression, survival, enrichment, and drug-target mining for KIRC ferroptosis genes.

```
import pandas as pd

# Load the RNA-seq dataset for KIRC
data_path = 'data/TCGA.KIRC.sampleMap_HiSeqV2.gz'
kirc_rna_seq = pd.read_csv(data_path, sep='\t', compression='gzip',
    index_col='sample')

# Load the survival dataset
survival_path = 'data/survival%2FKIRC_survival.txt'
kirc_survival = pd.read_csv(survival_path, sep='\t', index_col='sample')
```

```
# Define a list of ferroptosis-related genes based on prior knowledge
ferroptosis_genes = ['SLC7A11', 'GPX4', 'ACSL4', 'ALOX5', 'NCOA4', 'FTH1', 'TFRC']

# Filter the RNA-seq dataset for ferroptosis-related genes
ferroptosis_expression = kirc_rna_seq.loc[kirc_rna_seq.index.intersection(ferroptosis_genes)]

# Subset survival data to match RNA-seq samples
common_samples = list(set(ferroptosis_expression.columns) & set(kirc_survival.index))
ferroptosis_expression = ferroptosis_expression[common_samples]
kirc_survival = kirc_survival.loc[common_samples]

# Perform survival analysis for ferroptosis-related genes
from lifelines import CoxPHFitter
cox_results = []

for gene in ferroptosis_expression.index:
    gene_expression = ferroptosis_expression.loc[gene]
    cox_data = kirc_survival.copy()
    cox_data['expression'] = gene_expression.values

    cph = CoxPHFitter()
    try:
        cph.fit(cox_data, duration_col='OS.time', event_col='OS')
        cox_summary = cph.summary
        cox_summary['Gene'] = gene
        cox_results.append(cox_summary)
    except:
        continue

# Combine results into a single DataFrame
cox_results_df = pd.concat(cox_results)

# Display significant results (p < 0.05)
significant_cox_results = cox_results_df[cox_results_df['p'] < 0.05]
print(significant_cox_results[['Gene', 'exp(coef)', 'p']])
# Observation:
# Status: success
# stdout: Gene  exp(coef)             p
# covariate
# DSS.time      NCOA4   0.989722  1.157488e-42
# expression    NCOA4   0.707844  2.129359e-03
# DSS.time       TFRC   0.989661  2.766492e-43
# DSS.time      ALOX5   0.989670  4.009014e-43
# DSS.time      ACSL4   0.989664  3.164226e-43
# DSS.time    SLC7A11   0.989701  4.187018e-43
# expression  SLC7A11   1.137357  1.313861e-02
# DSS.time       FTH1   0.989677  3.499023e-43
# DSS.time       GPX4   0.989655  3.135233e-43

from gprofiler import GProfiler

# Initialize GProfiler for pathway enrichment analysis
gp = GProfiler(return_dataframe=True)

# Perform pathway enrichment analysis for ferroptosis-related genes
ferroptosis_pathway_enrichment = gp.profile(organism='hsapiens', query=['NCOA4', 'TFRC', 'ALOX5', 'ACSL4', 'SLC7A11', 'FTH1', 'GPX4'])

# Display the top enriched pathways
print(ferroptosis_pathway_enrichment[['source', 'name', 'p_value']].head(10))
```

```
# Observation:
# Status: success
# stdout: source                                        name       p_value
# 0   KEGG                                Ferroptosis  2.944772e-12
# 1     WP                                Ferroptosis  9.922609e-11
# 2  GO:BP       negative regulation of ferroptosis  1.896351e-06
# 3  GO:BP               regulation of ferroptosis  4.964507e-06
# 4  GO:BP                                ferroptosis  6.452937e-06
# 5  GO:BP       intracellular iron ion homeostasis  1.172731e-03
# 6  GO:CC                              autolysosome  1.226672e-03
# 7   REAC    Synthesis of 5-eicosatetraenoic acids  2.005590e-03
# 8  GO:CC                         secondary lysosome  3.110110e-03
# 9  GO:BP  long-chain fatty acid metabolic process  4.672125e-03

import pandas as pd

# Load the Probes & Drugs dataset
drug_data_path = 'data/pd_export_01_2025_targets_original.csv'
drug_data = pd.read_csv(drug_data_path, low_memory=False)

# Filter for compounds targeting ferroptosis-related genes
target_genes = ['NCOA4', 'TFRC', 'ALOX5', 'ACSL4', 'SLC7A11', 'FTH1', '
    GPX4']
ferroptosis_drugs = drug_data[drug_data['gene_name'].isin(target_genes)]

# Display identified drugs targeting ferroptosis-related genes
print(ferroptosis_drugs[['name', 'gene_name', 'moa']].drop_duplicates())
# Observation:
# Status: success
# stdout: name gene_name        moa
# 1041      PHENOTHIAZINE     ALOX5        NaN
# 1442         Kaempherol     ALOX5        NaN
# 1521         mesalazine     ALOX5  inhibitor
# 1762           ZILEUTON     ALOX5  inhibitor
# 2909    DIALLYL SULFIDE     ALOX5        NaN
# ...                 ...       ...        ...
# 226428         PD215795      GPX4        NaN
# 226538         PD215915      GPX4        NaN
# 226650         PD216127      GPX4        NaN
# 226872         PD216413      GPX4        NaN
# 227058         PD216625      GPX4        NaN
#
# [380 rows x 3 columns]

cancer_related_moas = ['inhibitor', 'antagonist', 'binder', 'modulator']
ferroptosis_drugs_with_moa = ferroptosis_drugs[ferroptosis_drugs['moa'].
    notna()]

# Filter for compounds with cancer-related mechanisms of action
prioritized_drugs = ferroptosis_drugs_with_moa[ferroptosis_drugs_with_moa
    ['moa'].str.contains('|'.join(cancer_related_moas), case=False, na=
    False)]

# Display prioritized drugs
print(prioritized_drugs[['name', 'gene_name', 'moa']].drop_duplicates())
# Observation:
# Status: success
# stdout: name gene_name         moa
# 1521              mesalazine     ALOX5   inhibitor
# 1762                ZILEUTON     ALOX5   inhibitor
# 7212       OLSALAZINE SODIUM     ALOX5   inhibitor
# 9730               DIACEREIN     ALOX5   inhibitor
# 12257             THIMEROSAL   SLC7A11  antagonist
# 12420          SULFASALAZINE   SLC7A11   inhibitor
# 12442          SULFASALAZINE     ALOX5   inhibitor
```

```
# 19621             masoprocol     ALOX5   inhibitor
# 19738        Quisqualic acid   SLC7A11   inhibitor
# 21678                BW B70C     ALOX5   inhibitor
# 23301               HONOKIOL     ALOX5   inhibitor
# 34183                  ML162      GPX4   inhibitor
# 46078           MORNIFLUMATE     ALOX5  antagonist
# 46950           TROGLITAZONE     ACSL4   inhibitor
# 50357            MINOCYCLINE     ALOX5   inhibitor
# 50381            Balsalazide     ALOX5   inhibitor
# 50598      MECLOFENAMIC ACID     ALOX5   inhibitor
# 56010          ROSIGLITAZONE     ACSL4   inhibitor
# 56379     DIETHYLCARBAMAZINE     ALOX5   inhibitor
# 57497    AMINOSALICYLIC ACID     ALOX5   inhibitor
# 61579           FOSTAMATINIB     ALOX5   inhibitor
# 64911                   RHEIN     ALOX5   inhibitor
# 89479               PD021959     ALOX5   inhibitor
# 93698               CJ-13610     ALOX5   inhibitor
# 94246             hyperforin     ALOX5   inhibitor
# 95196   MECLOFENAMATE SODIUM     ALOX5   inhibitor
# 98305             PF-4191834     ALOX5   inhibitor
# 110180            ATRELEUTON     ALOX5   inhibitor
# 132469              PD086470     ALOX5   inhibitor
# 139634  BALSALAZIDE DISODIUM     ALOX5   inhibitor
# 180955                   RSL3      GPX4   inhibitor
# 213049              PD194953      FTH1   inhibitor
# 214074            GPX4-IN-5       GPX4   inhibitor

from scipy.stats import ttest_ind

# Define tumor and normal samples
tumor_samples = [col for col in kirc_rna_seq.columns if col.endswith('01')]
normal_samples = [col for col in kirc_rna_seq.columns if col.endswith('11')]

# Perform differential expression analysis for ferroptosis-related genes
diff_expr_results = []
for gene in ['NCOA4', 'TFRC', 'ALOX5', 'ACSL4', 'SLC7A11', 'FTH1', 'GPX4']:
    if gene in kirc_rna_seq.index:
        tumor_expr = kirc_rna_seq.loc[gene, tumor_samples]
        normal_expr = kirc_rna_seq.loc[gene, normal_samples]
        log2_fc = tumor_expr.mean() - normal_expr.mean()
        t_stat, p_val = ttest_ind(tumor_expr, normal_expr, equal_var=False)
        diff_expr_results.append({'Gene': gene, 'Log2_Fold_Change': log2_fc, 'P_Value': p_val})

# Convert results to a DataFrame
diff_expr_df = pd.DataFrame(diff_expr_results)

# Display significant dysregulated genes (p < 0.05)
significant_diff_expr = diff_expr_df[diff_expr_df['P_Value'] < 0.05]
print(significant_diff_expr)
# Observation:
# Status: success
# stdout: Gene  Log2_Fold_Change       P_Value
# 0    NCOA4         -0.414105  8.720619e-21
# 1     TFRC         -0.157482  3.510574e-02
# 2    ALOX5          2.264952  2.440342e-22
# 3    ACSL4         -1.510587  3.482024e-37
# 4  SLC7A11          1.845668  1.789021e-23
# 5     FTH1          0.889489  7.987854e-23
# 6     GPX4          0.471884  1.992841e-09
```

Table 6: **Candidate inhibitors targeting ABCC8 and SLC5A2.**

Target	Inhibitor	Note / MOA
ABCC8	Glyburide	Sulfonylurea; ABCC8 (SUR1) inhibition
SLC5A2	Canagliflozin	SGLT2 inhibition; glucose transport modulation

E.2 Case Study 2: ABCC8 and SLC5A2 for PAAD

We assessed the novelty of *Targeting ABCC8 and SLC5A2 for Therapeutic Intervention in Pancreatic Adenocarcinoma* via targeted searches on PubMed and Google (keywords: "SLC5A2 pancreatic cancer"). A subsequent study from July 2025 independently confirmed an association between *SLC5A2* (i.e., *SGLT2*) and PAAD (Xie et al., 2025). Contextualizing our findings, prior work had reported prognostic significance for *SGLT1* (but not *SGLT2*) in pancreatic cancer (Du et al., 2022), and most *SGLT2* studies focused on normal pancreatic physiology rather than oncologic roles (Jurczak et al., 2011). Consistent with our protocol in other case studies, we verified that the 2025 confirmation paper was *not* accessed by the agent during generation, supporting that our result is an independent rediscovery that anticipated later literature. In parallel, expression of *ABCC8* has been reported in isolation in the literature (Cervenkova et al., 2023). We also note that a second article (published after the knowledge cut-off) *was* surfaced by the agent at analysis time and reported a correlation for *SLC5A2* in PAAD; the agent correctly cited and used this to refine its conclusions (Yang et al., 2024).

Targeting ABCC8 and SLC5A2 for Therapeutic Intervention in Pancreatic Adenocarcinoma

Meta-Review (for context). *Decision: accept; Overall score: 0.75; Rank: 1/4. Justification: robust computational evidence and actionable insights, making it the most impactful and original submission among its cohort.*

Research Question. Can ABCC8 and SLC5A2 serve as actionable therapeutic targets for pancreatic adenocarcinoma (PAAD)?

Hypothesis and Key Findings. ABCC8 and SLC5A2 are dysregulated in PAAD and represent promising therapeutic targets. Drugs targeting these genes—*glyburide* (ABCC8 inhibitor) and *canagliflozin* (SLC5A2 inhibitor)—could potentially modulate disease progression.

Rationale / Mechanism. ABCC8 has been implicated in multidrug resistance and pancreatic cancer risk, while SLC5A2 is associated with glucose metabolism and pancreatic cancer. Targeting these pathways may disrupt tumor growth and improve therapeutic outcomes.

Empirical Evidence.

- **Differential expression:** Identified 2,531 DEGs in tumor vs. normal samples, with *ABCC8* and *SLC5A2* among them.
- **Drug–target landscape:** 5,930 drug–target interactions with defined mechanisms of action (MOAs), including *glyburide* (ABCC8 inhibitor) and *canagliflozin* (SLC5A2 inhibitor).

Literature Evidence.

- ABCC8 is linked to pancreatic cancer risk (OR: 15.058, $P = 0.0001$). *Association of glucose-lowering drug target and risk of gastrointestinal cancer: a mendelian randomization study* (2024).
- ABCC8 is associated with multidrug resistance in cancer treatment. *Structural bioinformatics studies of six human ABC transporters and their AlphaFold2-predicted water-soluble QTY variants* (2024).
- SLC5A2 is connected with pancreatic cancer (OR: 8.096, $P < 0.0001$). *Association of glucose-lowering drug target and risk of gastrointestinal cancer: a mendelian randomization study* (2024).

Assumptions.

- Dysregulation of *ABCC8* and *SLC5A2* contributes to tumor progression in PAAD.
- Drugs targeting these genes are effective in modulating their activity in PAAD.

Limitations.

- The functional roles of *ABCC8* and *SLC5A2* in PAAD require further experimental validation.
- Potential off-target effects of inhibitors such as *glyburide* and *canagliflozin* need to be assessed.

References.

- Association of glucose-lowering drug target and risk of gastrointestinal cancer: a mendelian randomization study. (2024). Copyright © 2024.
- Structural bioinformatics studies of six human ABC transporters and their AlphaFold2-predicted water-soluble QTY variants. (2024). Copyright © 2024.

Listing 12: Differential expression, survival, enrichment, and drug-target mining for PAAD.

```
import pandas as pd

# Load the gene expression dataset
data_path = "data/TCGA.PAAD.sampleMap%2FHiSeqV2.gz"
gene_expression = pd.read_csv(data_path, sep='\t', compression='gzip',
    index_col='sample')

# Separate tumor and normal samples
tumor_samples = [col for col in gene_expression.columns if col.endswith("
    -01")]
normal_samples = [col for col in gene_expression.columns if col.endswith(
    "-11")]

# Calculate mean expression for tumor and normal samples
tumor_mean = gene_expression[tumor_samples].mean(axis=1)
normal_mean = gene_expression[normal_samples].mean(axis=1)

# Calculate log2 fold change (log2FC) and filter for significant DEGs
log2fc = tumor_mean - normal_mean
deg_threshold = 1  # Log2FC threshold for differential expression
degs = log2fc[abs(log2fc) >= deg_threshold]

# Output the number of DEGs and preview the top genes
print(f"Number of DEGs: {len(degs)}")
print(degs.sort_values(ascending=False).head(10))

# Observation:
# Status: success
# stdout: Number of DEGs: 2531
# sample
# CEACAM5     6.291835
# TFF1        6.172998
# UGT1A10     6.144656
# PSCA        5.961392
# AGR2        5.938662
# TFF2        5.777967
# S100P       5.709678
# ERN2        5.639717
# SERPINB5    5.620392
# PITX1       5.555346
# dtype: float64


# Inspect the structure of the enrichment_results dataframe
print(enrichment_results.columns)
```

```
print(enrichment_results.head())

# Observation:
# Status: success
# stdout: Index(['source', 'native', 'name', 'p_value', 'significant', '
    description',
#        'term_size', 'query_size', 'intersection_size', '
    effective_domain_size',
#        'precision', 'recall', 'query', 'parents'],
#       dtype='object')
#   source      native                                name        p_value
    \
# 0  GO:CC  GO:0071944                     cell periphery  3.030988e-137
# 1  GO:CC  GO:0005886                   plasma membrane  2.175693e-127
# 2  GO:BP  GO:0032501  multicellular organismal process   1.397718e-94
# 3  GO:BP  GO:0002376             immune system process   2.129839e-81
# 4  GO:BP  GO:0050896              response to stimulus   2.495028e-73
#
#    significant                                          description
    term_size  \
# 0         True  "The broad region around and including the pla...
     6347
# 1         True  "The membrane surrounding a cell that separate...
     5866
# 2         True  "Any biological process, occurring at the leve...
     7322
# 3         True  "Any process involved in the development or fu...
     2871
# 4         True  "Any process that results in a change in state...
     8999
#
#    query_size  intersection_size  effective_domain_size  precision
    recall  \
# 0        2123               1132                  22149   0.533208
    0.178352
# 1        2123               1058                  22149   0.498351
    0.180361
# 2        2033               1148                  21026   0.564683
    0.156788
# 3        2033                595                  21026   0.292671
    0.207245
# 4        2033               1265                  21026   0.622233
    0.140571
#
#      query                  parents
# 0  query_1              [GO:0110165]
# 1  query_1  [GO:0016020, GO:0071944]
# 2  query_1              [GO:0008150]
# 3  query_1              [GO:0008150]
# 4  query_1              [GO:0008150]


# Filter results for significant pathways (p-value < 0.05)
significant_pathways = enrichment_results[enrichment_results['p_value'] <
     0.05]

# Output the number of significant pathways and the top results
print(f"Number of significant pathways: {len(significant_pathways)}")
print(significant_pathways[['name', 'p_value']].head(10))

# Observation:
# Status: success
# stdout: Number of significant pathways: 1689
#                                    name        p_value
# 0                       cell periphery  3.030988e-137
```

```
# 1                        plasma membrane  2.175693e-127
# 2     multicellular organismal process   1.397718e-94
# 3                 immune system process   2.129839e-81
# 4                  response to stimulus   2.495028e-73
# 5                              membrane   4.603862e-73
# 6                       immune response   3.509625e-71
# 7  regulation of immune system process   2.033572e-70
# 8                     defense response   2.205079e-68
# 9                      cell activation   2.278886e-67


# Load the drug-target dataset
drug_target_path = "data/pd_export_01_2025_targets_original.csv"
drug_target_data = pd.read_csv(drug_target_path, low_memory=False)

# Extract gene names from the significant pathways
significant_genes = set(deg_genes)

# Filter drug-target interactions for genes in significant pathways
drug_targets = drug_target_data[drug_target_data['gene_name'].isin(
    significant_genes)]

# Output the number of drug-target interactions and preview the top
    results
print(f"Number of drug-target interactions: {len(drug_targets)}")
print(drug_targets[['name', 'gene_name', 'moa']].head(10))

# Observation:
# Status: success
# stdout: Number of drug-target interactions: 35929
#                     name gene_name  moa
# 26              UNC1215   L3MBTL4  NaN
# 28              UNC1215    ADRA2C  NaN
# 29              UNC1215     CHRM4  NaN
# 72                PFI-3    ADRA1A  NaN
# 77                PFI-3     NR1I2  NaN
# 89              UNC0642    ADRA1D  NaN
# 141             (+)-JQ1      DNER  NaN
# 142             (+)-JQ1      CCL2  NaN
# 194  DIMETHYL GAMBOGATE    CYP3A4  NaN
# 199  DIMETHYL GAMBOGATE      RORC  NaN


# Filter drug-target interactions for those with a defined MOA
drug_targets_with_moa = drug_targets.dropna(subset=['moa'])

# Prioritize drugs targeting immune-related or membrane-associated genes
immune_related_keywords = ['immune', 'membrane', 'response']
immune_related_drugs = drug_targets_with_moa[
    drug_targets_with_moa['gene_name'].str.contains('|'.join(
    immune_related_keywords), case=False, na=False)
]

# Output the number of prioritized drug-target interactions and preview
    the top results
print(f"Number of prioritized drug-target interactions: {len(
    immune_related_drugs)}")
print(immune_related_drugs[['name', 'gene_name', 'moa']].head(10))

# Observation:
# Status: success
# stdout: Number of prioritized drug-target interactions: 0
# Empty DataFrame
# Columns: [name, gene_name, moa]
# Index: []
```

```python


# Extract specific pathways related to immune system and membrane
immune_and_membrane_pathways = significant_pathways[
    significant_pathways['name'].str.contains('immune|membrane', case=
    False)
]

# Extract genes involved in these pathways
immune_and_membrane_genes = set()
for genes in immune_and_membrane_pathways['query']:
    immune_and_membrane_genes.update(genes)

# Filter drug-target interactions for these genes
immune_and_membrane_drugs = drug_targets[drug_targets['gene_name'].isin(
    immune_and_membrane_genes)]

# Output the number of interactions and preview the top results
print(f"Number of immune and membrane-related drug-target interactions: {
    len(immune_and_membrane_drugs)}")
print(immune_and_membrane_drugs[['name', 'gene_name', 'moa']].head(10))

# Observation:
# Status: success
# stdout: Number of immune and membrane-related drug-target interactions:
     0
# Empty DataFrame
# Columns: [name, gene_name, moa]
# Index: []


# Extract the top significant pathways based on p-value
top_pathways = significant_pathways.nsmallest(10, 'p_value')

# Extract genes involved in these pathways
top_pathway_genes = set()
for genes in top_pathways['query']:
    top_pathway_genes.update(genes)

# Filter drug-target interactions for these genes
top_pathway_drugs = drug_targets[drug_targets['gene_name'].isin(
    top_pathway_genes)]

# Output the number of interactions and preview the top results
print(f"Number of top pathway-related drug-target interactions: {len(
    top_pathway_drugs)}")
print(top_pathway_drugs[['name', 'gene_name', 'moa']].head(10))

# Observation:
# Status: success
# stdout: Number of top pathway-related drug-target interactions: 0
# Empty DataFrame
# Columns: [name, gene_name, moa]
# Index: []


# Filter drug-target interactions for any overlap with DEGs
deg_drug_targets = drug_targets[drug_targets['gene_name'].isin(deg_genes)
    ]

# Output the number of DEG-related drug-target interactions and preview
    the top results
print(f"Number of DEG-related drug-target interactions: {len(
    deg_drug_targets)}")
print(deg_drug_targets[['name', 'gene_name', 'moa']].head(10))
```

```
# Observation:
# Status: success
# stdout: Number of DEG-related drug-target interactions: 35929
#                     name gene_name  moa
# 26              UNC1215   L3MBTL4  NaN
# 28              UNC1215    ADRA2C  NaN
# 29              UNC1215     CHRM4  NaN
# 72                PFI-3    ADRA1A  NaN
# 77                PFI-3     NR1I2  NaN
# 89              UNC0642    ADRA1D  NaN
# 141             (+)-JQ1      DNER  NaN
# 142             (+)-JQ1      CCL2  NaN
# 194  DIMETHYL GAMBOGATE    CYP3A4  NaN
# 199  DIMETHYL GAMBOGATE      RORC  NaN


# Filter for drug-target interactions with a defined MOA
deg_drug_targets_with_moa = deg_drug_targets.dropna(subset=['moa'])

# Output the number of interactions with a defined MOA and preview the
    top results
print(f"Number of DEG-related drug-target interactions with MOA: {len(
    deg_drug_targets_with_moa)}")
print(deg_drug_targets_with_moa[['name', 'gene_name', 'moa']].head(10))

# Observation:
# Status: success
# stdout: Number of DEG-related drug-target interactions with MOA: 5930
#             name gene_name         moa
# 415    GLYBURIDE     ABCC8   inhibitor
# 416    GLYBURIDE   SLCO2B1   inhibitor
# 431  DOMPERIDONE      DRD2  antagonist
# 508     LEVODOPA      DRD2     agonist
# 620    AMOXAPINE    ADRA1A  antagonist
# 629    AMOXAPINE      DRD2  antagonist
# 642    AMOXAPINE      HRH1  antagonist
# 648    AMOXAPINE     HTR3A  antagonist
# 673    AMOXAPINE     HTR1B  antagonist
# 700        DPCPX   ADORA2B  antagonist


# Identify genes from the top significant pathways
top_pathway_genes = set()
for genes in top_pathways['query']:
    top_pathway_genes.update(genes)

# Filter for drug-target interactions involving these genes
pathway_drug_targets_with_moa = deg_drug_targets_with_moa[
    deg_drug_targets_with_moa['gene_name'].isin(top_pathway_genes)
]

# Output the number of interactions and preview the top results
print(f"Number of pathway-related drug-target interactions with MOA: {len
    (pathway_drug_targets_with_moa)}")
print(pathway_drug_targets_with_moa[['name', 'gene_name', 'moa']].head
    (10))

# Observation:
# Status: success
# stdout: Number of pathway-related drug-target interactions with MOA: 0
# Empty DataFrame
# Columns: [name, gene_name, moa]
# Index: []
```

```
# Group drug-target interactions by MOA and count occurrences
moa_counts = deg_drug_targets_with_moa['moa'].value_counts()

# Identify the most common MOAs and their associated drugs and targets
top_moas = moa_counts.head(10).index
top_moa_drug_targets = deg_drug_targets_with_moa[
    deg_drug_targets_with_moa['moa'].isin(top_moas)]

# Output the most common MOAs and their associated drug-target
    interactions
print("Most common MOAs and associated drug-target interactions:")
print(top_moa_drug_targets[['name', 'gene_name', 'moa']].head(20))

# Observation:
# Status: success
# stdout: Most common MOAs and associated drug-target interactions:
#                             name gene_name             moa
# 415                     GLYBURIDE     ABCC8      inhibitor
# 416                     GLYBURIDE   SLCO2B1      inhibitor
# 431                   DOMPERIDONE      DRD2     antagonist
# 508                      LEVODOPA      DRD2        agonist
# 620                     AMOXAPINE    ADRA1A     antagonist
# 629                     AMOXAPINE      DRD2     antagonist
# 642                     AMOXAPINE      HRH1     antagonist
# 648                     AMOXAPINE     HTR3A     antagonist
# 673                     AMOXAPINE     HTR1B     antagonist
# 700                         DPCPX   ADORA2B     antagonist
# 765                      EBASTINE      HRH1  inverse agonist
# 811                     CARAZOLOL     ADRB2     antagonist
# 841    CHLORPHENIRAMINE MALEATE      HRH1     antagonist
# 874                   MIRTAZAPINE      HRH1     antagonist
# 875                   MIRTAZAPINE    ADRA2C     antagonist
# 894                 DAPAGLIFLOZIN    SLC5A1      inhibitor
# 900   VORTIOXETINE HYDROBROMIDE     HTR3A     antagonist
# 905                CANAGLIFLOZIN    SLC5A1      inhibitor
# 1051        ETHANOLAMINE OLEATE       F12      activator
# 1075                  FOMEPIZOLE     ADH1B      inhibitor
```

E.3 Case Study 3: BIRC5 and PRKD1 in KIRC

Science advances not only by discovering new findings but also by *validating* and *reproducing* prior results. In this case study, our agentic system independently recapitulates a published conclusion about *BIRC5* (Survivin) in clear-cell renal cell carcinoma (ccRCC) and extends it with additional analyses and hypotheses around *PRKD1*. Using the TCGA KIRC cohort, our pipeline reaches the same core conclusion as Wang et al. (2021) regarding the early diagnostic and prognostic value of *BIRC5*. Because the authors' code was not publicly available, the agent system re-ran the analysis from scratch on TCGA expression and survival endpoints, confirming: (i) *BIRC5* overexpression in tumors relative to normals; and (ii) significant association with adverse outcomes. This strengthens confidence that the signal is robust to implementation details.

The system then expanded the analysis in two directions. Differential pathway enrichment on *BIRC5*-stratified samples highlights reinforcement of cell-cycle programs (e.g., chromosome segregation, mitotic spindle assembly) and mitotic checkpoint activity, consonant with Survivin's role in chromosomal passenger complexes. Our drug-target mining proposed candidate compounds for follow-up, including Survivin-directed strategies and kinase modulation consistent with the inferred networks. These are hypotheses for experimental testing rather than clinical recommendations. *PRKD1* is well-studied in renal physiology and polycystic kidney disease (Seeger-Nukpezah et al., 2015), and has more recently been implicated across cancer-hallmark processes. In KIRC specifically, our co-expression and enrichment analyses suggest that reduced *PRKD1* activity may coincide with dysregulation of nuclear–cytoplasmic transport and broader signaling modules. The joint consideration of *BIRC5* (as an oncogenic driver of mitotic progression) and *PRKD1* (as a putative tumor-suppressive

regulator of signaling/export) appears *novel* in the KIRC context and offers a mechanistic basis for complementary intervention hypotheses.

E.3.1 THERAPEUTIC TARGETING OF PRKD1 AND BIRC5 IN KIDNEY RENAL CLEAR CELL CARCINOMA (KIRC): DISTINCT PATHWAYS AND MECHANISMS

Meta-Review (for context). *Decision: accept; Overall score: 0.75; Rank: 1/4. Justification: robust empirical evidence and actionable insights into distinct pathways.*

Research Question. Can PRKD1 and BIRC5 serve as therapeutic targets in KIRC, and what are their distinct biological roles and associated pathways?

Hypothesis and Key Findings. PRKD1 and BIRC5 represent promising therapeutic targets in KIRC based on differential expression, survival correlations, and pathway involvement:

- **PRKD1:** Functions as a tumor suppressor; correlates with genes involved in nuclear protein export and cellular signaling.
- **BIRC5:** Acts as an oncogenic driver; correlates with genes enriched in cell cycle processes, including chromosome segregation and mitotic spindle assembly.

Rationale / Mechanism.

- **PRKD1:** Downregulated in tumors; positive survival correlation (Spearman with OS.time $= 0.128$, $p = 0.0016$). Co-expression network implicates regulation of protein export and cellular signaling.
- **BIRC5:** Upregulated in tumors; negative survival correlation (Spearman with OS.time $= -0.148$, $p = 0.0003$). Co-expression network highlights roles in cell-cycle progression and mitosis.

Empirical Evidence.

- **Differential expression:**
  - PRKD1: downregulated in tumors (fold change $= -1.178$).
  - BIRC5: upregulated in tumors (fold change $= 2.892$).
- **Survival correlations:**
  - PRKD1: positive correlation with OS.time (Spearman $= 0.128$, $p = 0.0016$).
  - BIRC5: negative correlation with OS.time (Spearman $= -0.148$, $p = 0.0003$).
- **Pathway enrichment:**
  - PRKD1: enriched in "positive regulation of protein export from nucleus" ($p = 0.034$).
  - BIRC5: enriched in "chromosome segregation" ($p = 1.25 \times 10^{-14}$) and "mitotic spindle assembly" ($p = 4.40 \times 10^{-10}$).

Assumptions.

- PRKD1 and BIRC5 directly influence KIRC progression through their associated pathways.
- Identified correlations and pathways reflect biologically relevant mechanisms in KIRC.

Limitations.

- Protein-level expression data for PRKD1 and BIRC5 were not available in the RPPA dataset.
- KIRC-specific experimental validation of therapeutic interventions targeting these genes is required.

Literature and Prior Evidence.

- Abdelrazig et al. (2025). *Protein Kinase D1 (PRKD1) as a Diagnostic, Prognostic, and Immunomodulatory Biomarker in Human Cancers.*
- *Survivin as a Multifaceted Oncogenic Driver and Therapeutic Target in Renal Cell Carcinoma.* (2025).
- Probes & Drugs database entries for PRKD1 and BIRC5 inhibitors.

Listing 13: Differential expression, survival, enrichment, and drug-target mining for KIRC BIRC5 genes.

```
import pandas as pd
from gprofiler import GProfiler
from scipy.stats import spearmanr

# Load RNA-seq dataset
data_path_rna = 'data/TCGA.KIRC.sampleMap_HiSeqV2.gz'
rna_seq_data = pd.read_csv(data_path_rna, sep='\t', compression='gzip', index_col='sample')

# Load survival dataset
data_path_survival = 'data/survival%2FKIRC_survival.txt'
survival_data = pd.read_csv(data_path_survival, sep='\t', index_col='sample')

# Genes of interest
genes_of_interest = ['PRKD1', 'BIRC5']

# Filter for tumor and normal samples
tumor_samples = [col for col in rna_seq_data.columns if col.endswith('01')]
normal_samples = [col for col in rna_seq_data.columns if col.endswith('11')]

# Calculate mean expression for tumor and normal samples
rna_seq_data['tumor_mean'] = rna_seq_data[tumor_samples].mean(axis=1)
rna_seq_data['normal_mean'] = rna_seq_data[normal_samples].mean(axis=1)

# Compute fold change (log2 ratio)
rna_seq_data['fold_change'] = rna_seq_data['tumor_mean'] - rna_seq_data['normal_mean']

# Extract fold change for genes of interest
fold_changes = rna_seq_data.loc[genes_of_interest, ['tumor_mean', 'normal_mean', 'fold_change']]

# Identify common samples between RNA-seq and survival datasets
common_samples = set(rna_seq_data.columns).intersection(survival_data.index)
common_samples = list(common_samples)

# Merge data for survival analysis
merged_data = survival_data.loc[common_samples].copy()

# Compute survival correlations for genes of interest
correlation_results = {}
for gene in genes_of_interest:
    if gene in rna_seq_data.index:
        merged_data[f'{gene}_expression'] = rna_seq_data.loc[gene, common_samples]
        corr, p_value = spearmanr(merged_data[f'{gene}_expression'], merged_data['OS.time'])
        correlation_results[gene] = (corr, p_value)

# Perform pathway enrichment analysis for genes of interest
gp = GProfiler(return_dataframe=True)
enrichment_results = gp.profile(organism='hsapiens', query=genes_of_interest)

# Display results
fold_changes, correlation_results, enrichment_results.head(10)

# Observation:
```

```
# Status: success
# result: (        tumor_mean  normal_mean  fold_change
#  sample
#  PRKD1     8.589648     9.767624    -1.177975
#  BIRC5     5.859713     2.967788     2.891925,
#  {'PRKD1': (np.float64(0.12829828043224845), np.float64
    (0.0015513876895559 18)),
#   'BIRC5': (np.float64(-0.14756835893794445),
#    np.float64(0.00026721000654785355))},
#    source      native
    name  \
#  0     WP   WP:WP1772                Apoptosis modulation and
    signaling
#  1     WP   WP:WP4659                                  Gastrin
    signaling
#  2  GO:CC  GO:1990713                                   survivin
    complex
#  3  CORUM  CORUM:2580                        Survivin homodimer
    complex
#  4  GO:BP  GO:0014723  regulation of skeletal muscle contraction by m
    ...
#  5  CORUM  CORUM:1117                      CRM1-Survivin mitotic
    complex
#  6  CORUM  CORUM:2581                      RasGAP-AURKA-survivin
    complex
#  7  CORUM  CORUM:6756                      RasGAP-AURKB-survivin
    complex
#
#      p_value  significant
    description  \
#  0  0.004547         True                Apoptosis modulation and
    signaling
#  1  0.007313         True                                  Gastrin
    signaling
#  2  0.020616         True  "A protein complex that negatively regulates
     a...
#  3  0.024966         True                         Survivin homodimer
    complex
#  4  0.049542         True  "Any process that modulates the frequency,
    rat...
#  5  0.049932         True                       CRM1-Survivin mitotic
    complex
#  6  0.049932         True                       RasGAP-AURKA-survivin
    complex
#  7  0.049932         True                       RasGAP-AURKB-survivin
    complex
#
#     term_size  query_size  intersection_size  effective_domain_size
    precision  \
#  0         90           2                  2                   8752
        1.0
#  1        114           2                  2                   8752
        1.0
#  2          1           2                  1                  22149
        0.5
#  3          1           1                  1                   3383
        1.0
#  4          1           2                  1                  21026
        0.5
#  5          2           1                  1                   3383
        1.0
#  6          2           1                  1                   3383
        1.0
#  7          2           1                  1                   3383
        1.0
```

```
#
#       recall    query          parents
#  0  0.022222  query_1      [WP:000000]
#  1  0.017544  query_1      [WP:000000]
#  2  1.000000  query_1     [GO:0140513]
#  3  1.000000  query_1  [CORUM:0000000]
#  4  1.000000  query_1     [GO:0014722]
#  5  0.500000  query_1  [CORUM:0000000]
#  6  0.500000  query_1  [CORUM:0000000]
#  7  0.500000  query_1  [CORUM:0000000]  )


# Load Probes & Drugs dataset
data_path_pd = 'data/pd_export_01_2025_targets_original.csv'
pd_data = pd.read_csv(data_path_pd, low_memory=False, index_col='name')

# Filter Probes & Drugs dataset for PRKD1 and BIRC5
target_genes = ['PRKD1', 'BIRC5']
pd_filtered = pd_data[pd_data['gene_name'].isin(target_genes)]

# Display the filtered compounds and their mechanisms of action
pd_filtered[['gene_name', 'target_name', 'moa']].drop_duplicates()

# Observation:
# Status: success
# result:                                gene_name  \
# name
# NERATINIB                           PRKD1
# TCS PIM-1 1                         BIRC5
# GW855857                            PRKD1
# Bryostatin 1                        PRKD1
# compound III [PMID: 24080463]     BIRC5
#
#
   target_name  \
# name
# NERATINIB                                  Serine/threonine-protein kinase
    D1
# TCS PIM-1 1                    Baculoviral IAP repeat-containing
   protein 5
# GW855857                                   Serine/threonine-protein kinase
    D1
# Bryostatin 1                               Serine/threonine-protein kinase
    D1
# compound III [PMID: 24080463]  Baculoviral IAP repeat-containing
   protein 5
#
#                                      moa
# name
# NERATINIB                            NaN
# TCS PIM-1 1                          NaN
# GW855857                         inhibitor
# Bryostatin 1                     activator
# compound III [PMID: 24080463]  inhibitor


# Display the first few rows of the RPPA dataset to inspect its structure
rppa_data.head()

# Observation:
# Status: success
# result:                      TCGA-B8-A54D-01  TCGA-G6-A8L7-01  TCGA-B8-
   A54F-01  \
# sample
# 14-3-3_beta-R-V            0.065007        -0.103411        -0.071788
```

```
# 14-3-3_epsilon-M-C        -0.175905         0.130026        -0.080084
# 14-3-3_zeta-R-V           -0.195639        -0.174381         0.064587
# 4E-BP1-R-V                -0.286517         1.231338         0.012585
# 4E-BP1_pS65-R-V           -0.020339         1.542328        -0.325206
#
#                       TCGA-B8-A8YJ-01  TCGA-B8-A54K-01  TCGA-3Z-A93Z-01
    \
# sample
# 14-3-3_beta-R-V            0.556920         0.130937         0.406331
# 14-3-3_epsilon-M-C         0.175525         0.198440        -0.053131
# 14-3-3_zeta-R-V           -1.272674         0.168871        -0.321452
# 4E-BP1-R-V                -0.828272        -0.240631         0.122247
# 4E-BP1_pS65-R-V           -0.166733         0.063540         0.155350
#
#                       TCGA-G6-A8L6-01  TCGA-MW-A4EC-01  TCGA-DV-A4W0-01
    \
# sample
# 14-3-3_beta-R-V           -0.037139        -0.022034        -0.056487
# 14-3-3_epsilon-M-C         0.089388         0.027828        -0.089663
# 14-3-3_zeta-R-V            0.204648        -0.008644         0.013026
# 4E-BP1-R-V                 0.377911         0.091436         0.014693
# 4E-BP1_pS65-R-V           -0.000909         0.257222         0.065496
#
#                       TCGA-G6-A5PC-01  ...  TCGA-B0-4703-01  TCGA-BP
   -4981-01  \
# sample                                 ...
# 14-3-3_beta-R-V           -0.010247  ...        -0.035964
   -0.013955
# 14-3-3_epsilon-M-C         0.237651  ...        -0.083376
   0.030217
# 14-3-3_zeta-R-V           -0.026489  ...         0.293633
   0.267474
# 4E-BP1-R-V                -0.229184  ...        -0.139995
   0.360712
# 4E-BP1_pS65-R-V            0.608147  ...         0.183363
   -0.052082
#
#                       TCGA-B8-4622-01  TCGA-B0-4819-01  TCGA-A3-3316-01
    \
# sample
# 14-3-3_beta-R-V            0.150967         0.025524        -0.047050
# 14-3-3_epsilon-M-C         0.016982         0.204421        -0.138947
# 14-3-3_zeta-R-V            0.278651         0.403945        -0.109604
# 4E-BP1-R-V                 0.041094        -0.425633         0.062104
# 4E-BP1_pS65-R-V           -0.269524        -0.365388        -0.059234
#
#                       TCGA-BP-4347-01  TCGA-B2-5636-01  TCGA-CW-5584-01
    \
# sample
# 14-3-3_beta-R-V            0.137857        -0.041162        -0.001714
# 14-3-3_epsilon-M-C        -0.071562         0.087068        -0.112748
# 14-3-3_zeta-R-V           -0.440944        -0.031086        -0.078699
# 4E-BP1-R-V                -0.437115        -0.331337         0.282619
# 4E-BP1_pS65-R-V            0.612101         0.128544        -0.104384
#
#                       tumor_mean  normal_mean
# sample
# 14-3-3_beta-R-V       0.084855          NaN
# 14-3-3_epsilon-M-C    0.027040          NaN
# 14-3-3_zeta-R-V       0.039044          NaN
# 4E-BP1-R-V            0.069128          NaN
# 4E-BP1_pS65-R-V       0.006978          NaN
#
# [5 rows x 480 columns]
```

```
# Filter the RPPA dataset for potential aliases or descriptions related to PRKD1 and BIRC5
potential_aliases = ['Protein kinase D1', 'Survivin', 'Baculoviral IAP repeat-containing protein 5']
matching_entries = rppa_data[rppa_data.index.str.contains('|'.join(potential_aliases), case=False)]

# Display matching entries
matching_entries

# Observation:
# Status: success
# result: Empty DataFrame
# Columns: [TCGA-B8-A54D-01, TCGA-G6-A8L7-01, TCGA-B8-A54F-01, TCGA-B8-A8YJ-01, TCGA-B8-A54K-01, TCGA-3Z-A93Z-01, TCGA-G6-A8L6-01, TCGA-MW-A4EC-01, TCGA-DV-A4W0-01, TCGA-G6-A5PC-01, TCGA-B8-A54E-01, TCGA-B8-A54G-01, TCGA-6D-AA2E-01, TCGA-B2-A4SR-01, TCGA-B8-A54H-01, TCGA-MM-A563-01, TCGA-G6-A8L8-01, TCGA-DV-A4VZ-01, TCGA-B8-A54I-01, TCGA-GK-A6C7-01, TCGA-DV-A4VX-01, TCGA-B8-A54J-01, TCGA-MM-A564-01, TCGA-B8-A7U6-01, TCGA-B4-5844-01, TCGA-B0-4701-01, TCGA-BP-4970-01, TCGA-A3-3373-01, TCGA-B0-5113-01, TCGA-B8-5164-01, TCGA-CJ-4878-01, TCGA-BP-5189-01, TCGA-BP-4988-01, TCGA-BP-4351-01, TCGA-BP-4803-01, TCGA-A3-3352-01, TCGA-BP-4965-01, TCGA-BP-4766-01, TCGA-BP-4987-01, TCGA-BP-4787-01, TCGA-B0-5707-01, TCGA-B0-5100-01, TCGA-DV-5573-01, TCGA-BP-4769-01, TCGA-B0-5099-01, TCGA-BP-4959-01, TCGA-CZ-5984-01, TCGA-B0-4852-01, TCGA-CZ-4857-01, TCGA-CZ-4856-01, TCGA-CW-5583-01, TCGA-B8-5163-01, TCGA-BP-5201-01, TCGA-CJ-5686-01, TCGA-BP-4174-01, TCGA-B8-5546-01, TCGA-A3-3306-01, TCGA-B0-4707-01, TCGA-CJ-4642-01, TCGA-A3-3363-01, TCGA-CW-6093-01, TCGA-A3-3324-01, TCGA-BP-5001-01, TCGA-BP-4995-01, TCGA-B0-4693-01, TCGA-A3-3329-01, TCGA-BP-4976-01, TCGA-CJ-4890-01, TCGA-B0-4700-01, TCGA-CJ-5683-01, TCGA-CW-5580-01, TCGA-BP-4756-01, TCGA-B0-5699-01, TCGA-BP-5191-01, TCGA-B0-5098-01, TCGA-A3-3336-01, TCGA-BP-5009-01, TCGA-CW-5589-01, TCGA-CJ-4640-01, TCGA-BP-4334-01, TCGA-B0-4824-01, TCGA-CJ-6030-01, TCGA-B0-4718-01, TCGA-BP-4964-01, TCGA-A3-3376-01, TCGA-B0-5096-01, TCGA-CJ-4638-01, TCGA-B4-5377-01, TCGA-BP-4982-01, TCGA-BP-4968-01, TCGA-B0-5812-01, TCGA-B8-4621-01, TCGA-A3-3387-01, TCGA-BP-4799-01, TCGA-B0-5711-01, TCGA-BP-4355-01, TCGA-CZ-5988-01, TCGA-B0-5108-01, TCGA-CZ-5452-01, TCGA-B0-5094-01, ...]
# Index: []
#
# [0 rows x 480 columns]


# Broaden search criteria for potential entries related to PRKD1 and BIRC5
broader_aliases = ['kinase', 'survivin', 'apoptosis']
broader_matching_entries = rppa_data[rppa_data.index.str.contains('|'.join(broader_aliases), case=False)]

# Display broader matching entries
broader_matching_entries

# Observation:
# Status: success
# result: Empty DataFrame
# Columns: [TCGA-B8-A54D-01, TCGA-G6-A8L7-01, TCGA-B8-A54F-01, TCGA-B8-A8YJ-01, TCGA-B8-A54K-01, TCGA-3Z-A93Z-01, TCGA-G6-A8L6-01, TCGA-MW-A4EC-01, TCGA-DV-A4W0-01, TCGA-G6-A5PC-01, TCGA-B8-A54E-01, TCGA-B8-A54G-01, TCGA-6D-AA2E-01, TCGA-B2-A4SR-01, TCGA-B8-A54H-01, TCGA-MM-A563-01, TCGA-G6-A8L8-01, TCGA-DV-A4VZ-01, TCGA-B8-A54I-01, TCGA-GK-A6C7-01, TCGA-DV-A4VX-01, TCGA-B8-A54J-01, TCGA-MM-A564-01, TCGA-B8-A7U6-01, TCGA-B4-5844-01, TCGA-B0-4701-01, TCGA-BP-4970-01, TCGA-A3-3373-01, TCGA-B0-5113-01, TCGA-B8-5164-01, TCGA-CJ-4878-01, TCGA-BP
```

```
    -5189-01, TCGA-BP-4988-01, TCGA-BP-4351-01, TCGA-BP-4803-01, TCGA-A3
    -3352-01, TCGA-BP-4965-01, TCGA-BP-4766-01, TCGA-BP-4987-01, TCGA-BP
    -4787-01, TCGA-B0-5707-01, TCGA-B0-5100-01, TCGA-DV-5573-01, TCGA-BP
    -4769-01, TCGA-B0-5099-01, TCGA-BP-4959-01, TCGA-CZ-5984-01, TCGA-B0
    -4852-01, TCGA-CZ-4857-01, TCGA-CZ-4856-01, TCGA-CW-5583-01, TCGA-B8
    -5163-01, TCGA-BP-5201-01, TCGA-CJ-5686-01, TCGA-BP-4174-01, TCGA-B8
    -5546-01, TCGA-A3-3306-01, TCGA-B0-4707-01, TCGA-CJ-4642-01, TCGA-A3
    -3363-01, TCGA-CW-6093-01, TCGA-A3-3324-01, TCGA-BP-5001-01, TCGA-BP
    -4995-01, TCGA-B0-4693-01, TCGA-A3-3329-01, TCGA-BP-4976-01, TCGA-CJ
    -4890-01, TCGA-B0-4700-01, TCGA-CJ-5683-01, TCGA-CW-5580-01, TCGA-BP
    -4756-01, TCGA-B0-5699-01, TCGA-BP-5191-01, TCGA-B0-5098-01, TCGA-A3
    -3336-01, TCGA-BP-5009-01, TCGA-CW-5589-01, TCGA-CJ-4640-01, TCGA-BP
    -4334-01, TCGA-B0-4824-01, TCGA-CJ-6030-01, TCGA-B0-4718-01, TCGA-BP
    -4964-01, TCGA-A3-3376-01, TCGA-B0-5096-01, TCGA-CJ-4638-01, TCGA-B4
    -5377-01, TCGA-BP-4982-01, TCGA-BP-4968-01, TCGA-B0-5812-01, TCGA-B8
    -4621-01, TCGA-A3-3387-01, TCGA-BP-4799-01, TCGA-B0-5711-01, TCGA-BP
    -4355-01, TCGA-CZ-5988-01, TCGA-B0-5108-01, TCGA-CZ-5452-01, TCGA-B0
    -5094-01, ...]
# Index: []
#
# [0 rows x 480 columns]


# Inspect columns of RNA-seq dataset to confirm available tumor sample
    identifiers
available_tumor_samples = [col for col in rna_seq_data.columns if col.
    endswith('01')]

# Compare with the original list of tumor samples
missing_samples = set(tumor_samples) - set(available_tumor_samples)

# Display missing samples and refine tumor sample list
missing_samples, available_tumor_samples[:10]  # Display a subset of
    available samples for verification

# Observation:
# Status: success
# result: ({'TCGA-A3-3336-01', 'TCGA-CJ-4913-01', 'TCGA-CW-6096-01'},
#  ['TCGA-BP-4162-01',
#   'TCGA-DV-5566-01',
#   'TCGA-BP-5191-01',
#   'TCGA-BP-5200-01',
#   'TCGA-BP-4347-01',
#   'TCGA-BP-4770-01',
#   'TCGA-BP-4762-01',
#   'TCGA-BP-4158-01',
#   'TCGA-CJ-4887-01',
#   'TCGA-B2-4098-01'])


# Convert indices of top correlated genes to lists for pathway enrichment
top_correlated_prkd1_list = top_correlated_prkd1.index.tolist()
top_correlated_birc5_list = top_correlated_birc5.index.tolist()

# Perform pathway enrichment analysis for top correlated genes
enrichment_results_prkd1 = gp.profile(organism='hsapiens', query=
    top_correlated_prkd1_list)
enrichment_results_birc5 = gp.profile(organism='hsapiens', query=
    top_correlated_birc5_list)

# Display results
top_correlated_prkd1, enrichment_results_prkd1.head(10),
    top_correlated_birc5, enrichment_results_birc5.head(10)

# Observation:
```

```
# Status: success
# result: (          corr_prkd1  corr_birc5
#  PRKD1       1.000000   -0.422963
#  NUMB        0.723506   -0.455379
#  FAM161B     0.656223   -0.393488
#  PPM1A       0.646236   -0.466748
#  L2HGDH      0.643696   -0.476554
#  ALDH6A1     0.643234   -0.507840
#  MOAP1       0.641932   -0.438381
#  RALGAPA1    0.638948   -0.479813
#  GPHN        0.632647   -0.439207
#  FAM179B     0.631081   -0.417649,
#    source      native
    name  \
#  0  GO:BP  GO:0046827  positive regulation of protein export from nuc
    ...
#
#      p_value  significant
    description  \
#  0  0.034246         True  "Any process that activates or increases the
     f...
#
#     term_size  query_size  intersection_size  effective_domain_size
    precision  \
#  0         20           9                  2                  21026
    0.222222
#
#     recall    query                                                parents
#  0     0.1  query_1  [GO:0006611, GO:0046824, GO:0046825, GO:0090316]
    ,
#         corr_prkd1  corr_birc5
#  BIRC5    -0.422963    1.000000
#  CDC20    -0.398473    0.902944
#  AURKB    -0.445077    0.890122
#  CCNB2    -0.397994    0.885516
#  UBE2C    -0.500449    0.883852
#  HJURP    -0.371927    0.869710
#  MYBL2    -0.433110    0.862108
#  TPX2     -0.356099    0.861416
#  CDCA8    -0.356784    0.859240
#  PTTG1    -0.515681    0.859221,
#    source              native  \
#  0  GO:BP          GO:0007059
#  1  GO:BP          GO:0098813
#  2  GO:BP          GO:0000280
#  3  GO:BP          GO:0048285
#  4  GO:BP          GO:0051225
#  5  GO:BP          GO:0051276
#  6  GO:BP          GO:1901970
#  7   REAC  REAC:R-HSA-1640170
#  8  GO:BP          GO:0090307
#  9  GO:BP          GO:0000070
#
#                                                  name       p_value  \
#  0                            chromosome segregation  1.250349e-14
#  1                     nuclear chromosome segregation  4.892650e-13
#  2                                   nuclear division  1.034081e-11
#  3                                 organelle fission  2.579222e-11
#  4                                  spindle assembly  5.564453e-11
#  5                             chromosome organization  8.987358e-11
#  6  positive regulation of mitotic sister chromati...  1.736488e-10
#  7                                         Cell Cycle  2.065140e-10
#  8                              mitotic spindle assembly  4.400940e-10
#  9               mitotic sister chromatid segregation  6.711199e-10
#
```

```
#     significant                                          description
    term_size  \
# 0         True  "The process in which genetic material, in the...
       427
# 1         True  "The process in which genetic material, in the...
       323
# 2         True  "The division of a cell nucleus into two nucle...
       452
# 3         True  "The creation of two or more organelles by div...
       500
# 4         True  "The aggregation, arrangement and bonding toge...
       136
# 5         True  "A process that is carried out at the cellular...
       574
# 6         True  "Any process that activates or increases the f...
        21
# 7         True                                          Cell Cycle
       679
# 8         True  "Mitotic bipolar spindle assembly begins with ...
        76
# 9         True  "The cell cycle process in which replicated ho...
       193
#
#     query_size  intersection_size  effective_domain_size  precision
    recall  \
# 0          10                 10                  21026        1.0
    0.023419
# 1          10                  9                  21026        0.9
    0.027864
# 2          10                  9                  21026        0.9
    0.019912
# 3          10                  9                  21026        0.9
    0.018000
# 4          10                  7                  21026        0.7
    0.051471
# 5          10                  9                  21026        0.9
    0.015679
# 6          10                  5                  21026        0.5
    0.238095
# 7          10                 10                  11004        1.0
    0.014728
# 8          10                  6                  21026        0.6
    0.078947
# 9          10                  7                  21026        0.7
    0.036269
#
#       query                                parents
# 0  query_1                           [GO:0022402]
# 1  query_1                           [GO:0007059]
# 2  query_1                           [GO:0048285]
# 3  query_1                           [GO:0006996]
# 4  query_1  [GO:0007051, GO:0007059, GO:0140694]
# 5  query_1                           [GO:0006996]
# 6  query_1  [GO:0010965, GO:0051306, GO:1905820]
# 7  query_1                         [REAC:0000000]
# 8  query_1  [GO:0000070, GO:0007052, GO:0051225]
# 9  query_1  [GO:0000819, GO:0140014, GO:1903047]  )
```

E.4 Case Study 4: Insulin Resistance Coupling Increases Myocardial Infarction Risk

Meta-Review (for context). *Decision: accept; Overall score: 0.90; Rank: 1/5. Justification: The report is the strongest submission, with robust evidence, methodological rigor, and significant novelty, making it the top candidate.*

Research Question. Does metabolic acidosis (indexed by low serum bicarbonate and high anion gap) interact with insulin resistance (TyG index) to increase myocardial infarction (MI) risk, and is this coupling buffered by albumin?

Hypothesis and Key Findings. Lower bicarbonate and higher anion gap capture a metabolic acidosis signature that elevates MI risk; this risk is amplified by insulin resistance (TyG), while higher albumin attenuates the acidosis–IR coupling.

- **Acidosis-IR Coupling:** The interaction between bicarbonate and TyG is positive (OR ≈ 1.07, $p \approx 0$), indicating that insulin resistance amplifies the risk associated with acidosis.
- **Loss of Buffer:** Strong ion difference (SID) is protective in low IR states (OR ≈ 0.78) but loses its protective effect in high IR states (OR ≈ 1.02).
- **Albumin Buffering:** Albumin modifies bicarbonate's effect significantly. Bicarbonate is protective in high albumin contexts (OR ≈ 0.73) but associated with risk in low albumin contexts (OR ≈ 1.02). The three-way interaction confirms albumin attenuates the acidosis–IR coupling ($\exp(\text{coef}) \approx 0.94$).
- **Prevalence Hotspot:** The combination of lowest bicarbonate (Q1) and highest TyG (Q4) yields the highest MI prevalence ($\approx 5.63\%$) compared to the baseline ($\approx 1.14\%$).

Rationale / Mechanism.

- **Metabolic Load:** Insulin resistance elevates glycolytic flux and lipotoxicity, increasing the acid load and impairing endothelial/mitochondrial function.
- **Acidosis Markers:** Low bicarbonate and high anion gap reflect buffered metabolic acidosis and the accumulation of unmeasured anions.
- **Albumin Protection:** Albumin provides critical oncotic and antioxidant buffering (via ligand binding and free radical scavenging), dampening the vascular injury caused by the acidosis–IR synergy.

Empirical Evidence.

- **Main Effects:** Bicarbonate per SD is inversely associated with MI (OR ≈ 0.91); Anion Gap per SD is positively associated (OR ≈ 1.10).
- **Interactions:**
  - $HCO_3 \times$ TyG: OR ≈ 1.07 (Amplification).
  - AG $\times$ TyG: OR ≈ 0.95 (Consistent with bicarbonate findings).
  - $HCO_3 \times$ TyG $\times$ Albumin: $\exp(\text{coef}) \approx 0.94$ (Attenuation/Buffering).
- **Methodology:** Weighted GLMs applied to NHANES data, adjusting for demographics, renal function (creatinine), redox status (uric acid), smoking (cotinine), and SES.

Assumptions.

- Single-time measurements of bicarbonate and ions approximate chronic acid–base status.
- TyG index accurately reflects insulin resistance in this population.
- Weighting with MEC exam weights sufficiently approximates population inference.

Limitations.

- Cross-sectional design limits causality; reverse causation cannot be ruled out.
- Medication and dietary confounding (e.g., diuretics, alkali intake) may remain unmeasured.

- Complex survey design elements (PSU/strata) were not fully modeled in the interaction analysis.

Literature and Prior Evidence.

- *Stress hyperglycemia ratio in AF (MIMIC-IV):* Found that bicarbonate mediated $\sim 22.7\%$ of the stress hyperglycemia effect on mortality, supporting acid–metabolic coupling.
- *The Effect of Dialysate Bicarbonate...:* Meta-analysis confirming bicarbonate alters systemic acid–base milieu relevant to cardiovascular risk.
- Internal Archive: Multiple internal reports confirm albumin's role as an antioxidant buffer in oxidative stress-nutritional status axes.

Listing 14: Insulin Resistance Coupling Increases Myocardial Infarction Risk.

```
import pandas as pd


import pickle
import pandas as pd

# Load the NHANES dataset
with open('data/nhanes_dataset.pkl', 'rb') as f:
    nhanes = pickle.load(f)

print('Dataset shape:', nhanes.shape)
print('First 10 columns:', nhanes.columns[:10].tolist())
print('Outcome mean (myocardial infarction prevalence):', nhanes['outcome
    '].mean())
print('\
Sample of unique values in key columns:')
for col in ['Gender', 'Age in years at screening', 'Race/Hispanic origin'
    ]:
    if col in nhanes.columns:
        print(f'{col}: {nhanes[col].unique()[:5]}')
# Observation:
# Status: success
# stdout: Dataset shape: (86156, 52)
# First 10 columns: ['Citizenship status', 'Total number of people in the
     Household', "HH ref person's age in years", "HH ref person's
    education level", "HH ref person's gender", "HH ref person's marital
    status", 'Ratio of family income to poverty', 'Albumin, refrigerated
    serum (g/L)', 'Blood Urea Nitrogen (mmol/L)', 'Total Calcium (mmol/L)
    ']
# Outcome mean (myocardial infarction prevalence): 0.025210084033613446
# Sample of unique values in key columns:
# Gender: [2.0 1.0 <NA>]
# Age in years at screening: [2.0 77.0 10.0 1.0 49.0]
# Race/Hispanic origin: [4.0 3.0 5.0 1.0 2.0]

print('All columns in the dataset:')
for i, col in enumerate(nhanes.columns):
    print(f'{i+1:2d}. {col}')

print('\
Biomarker columns available:')
biomarker_cols = [col for col in nhanes.columns if any(term in col.lower
    () for term in ['glucose', 'cholesterol', 'triglycerides', 'bilirubin
    ', 'protein', 'albumin', 'urea', 'creatinine', 'transferase', '
    aminotransferase', 'iron', 'potassium', 'sodium'])]
for col in biomarker_cols:
    print(f'- {col}')

print('\
Lifestyle/demographic columns:')
```

```
demo_cols = [col for col in nhanes.columns if any(term in col.lower() for term in ['gender', 'age', 'race', 'education', 'income', 'marital', 'citizenship', 'household'])]
for col in demo_cols:
    print(f'- {col}')
# Observation:
# Status: success
# stdout: All columns in the dataset:
#  1. Citizenship status
#  2. Total number of people in the Household
#  3. HH ref person's age in years
#  4. HH ref person's education level
#  5. HH ref person's gender
#  6. HH ref person's marital status
#  7. Ratio of family income to poverty
#  8. Albumin, refrigerated serum (g/L)
#  9. Blood Urea Nitrogen (mmol/L)
# 10. Total Calcium (mmol/L)
# 11. Cholesterol, refrigerated serum (mmol/L)
# 12. Creatinine, refrigerated serum (umol/L)
# 13. Globulin (g/L)
# 14. Glucose, refrigerated serum (mmol/L)
# 15. Iron, refrigerated serum (umol/L)
# 16. Phosphorus (mmol/L)
# 17. Total Bilirubin (umol/L)
# 18. Total Protein (g/L)
# 19. Triglycerides, refrig serum (mmol/L)
# 20. Uric acid (umol/L)
# 21. Cotinine, Serum (ng/mL)
# 22. Hepatitis B core antibody
# 23. Albumin, refrigerated serum (g/dL)
# 24. Aspartate Aminotransferase (AST) (U/L)
# 25. Alanine Aminotransferase (ALT) (U/L)
# 26. Blood Urea Nitrogen (mg/dL)
# 27. Bicarbonate (mmol/L)
# 28. Total Calcium (mg/dL)
# 29. Cholesterol, refrigerated serum (mg/dL)
# 30. Chloride (mmol/L)
# 31. Globulin (g/dL)
# 32. Glucose, refrigerated serum (mg/dL)
# 33. Gamma Glutamyl Transferase (GGT) (IU/L)
# 34. Iron, refrigerated serum (ug/dL)
# 35. Potassium (mmol/L)
# 36. Sodium (mmol/L)
# 37. Osmolality (mmol/Kg)
# 38. Total Protein (g/dL)
# 39. Triglycerides, refrig serum (mg/dL)
# 40. Uric acid (mg/dL)
# 41. Session in which SP was examined
# 42. Gender
# 43. Age in years at screening
# 44. Six month time period
# 45. Race/Hispanic origin
# 46. Interview/Examination status
# 47. Data release cycle
# 48. Masked variance pseudo-PSU
# 49. Masked variance pseudo-stratum
# 50. Full sample 2 year interview weight
# 51. Full sample 2 year MEC exam weight
# 52. outcome
# Biomarker columns available:
# - Albumin, refrigerated serum (g/L)
# - Blood Urea Nitrogen (mmol/L)
# - Cholesterol, refrigerated serum (mmol/L)
# - Creatinine, refrigerated serum (umol/L)
```

```
# - Glucose, refrigerated serum (mmol/L)
# - Iron, refrigerated serum (umol/L)
# - Total Bilirubin (umol/L)
# - Total Protein (g/L)
# - Triglycerides, refrig serum (mmol/L)
# - Albumin, refrigerated serum (g/dL)
# - Aspartate Aminotransferase (AST) (U/L)
# - Alanine Aminotransferase (ALT) (U/L)
# - Blood Urea Nitrogen (mg/dL)
# - Cholesterol, refrigerated serum (mg/dL)
# - Glucose, refrigerated serum (mg/dL)
# - Gamma Glutamyl Transferase (GGT) (IU/L)
# - Iron, refrigerated serum (ug/dL)
# - Potassium (mmol/L)
# - Sodium (mmol/L)
# - Total Protein (g/dL)
# - Triglycerides, refrig serum (mg/dL)
# Lifestyle/demographic columns:
# - Citizenship status
# - Total number of people in the Household
# - HH ref person's age in years
# - HH ref person's education level
# - HH ref person's gender
# - HH ref person's marital status
# - Ratio of family income to poverty
# - Gender
# - Age in years at screening
# - Race/Hispanic origin

import pandas as pd
import numpy as np
from scipy import stats

# Prepare liver-kidney biomarker data
liver_kidney_data = nhanes[['Aspartate Aminotransferase (AST) (U/L)',
                            'Alanine Aminotransferase (ALT) (U/L)',
                            'Gamma Glutamyl Transferase (GGT) (IU/L)',
                            'Creatinine, refrigerated serum (umol/L)',
                            'Blood Urea Nitrogen (mg/dL)',
                            'Age in years at screening', 'Gender',
                            'Glucose, refrigerated serum (mg/dL)',
                            'Cholesterol, refrigerated serum (mg/dL)',
                            'Triglycerides, refrig serum (mg/dL)',
                            'outcome']].copy()

# Convert to numeric and drop missing values
for col in liver_kidney_data.columns:
    liver_kidney_data[col] = pd.to_numeric(liver_kidney_data[col], errors
    ='coerce')

liver_kidney_data = liver_kidney_data.dropna()
print(f'Sample size: {len(liver_kidney_data)}')
print(f'MI cases: {liver_kidney_data.outcome.sum()}')

# Calculate liver-kidney ratios
liver_kidney_data['AST_creatinine_ratio'] = liver_kidney_data['Aspartate
    Aminotransferase (AST) (U/L)'] / liver_kidney_data['Creatinine,
    refrigerated serum (umol/L)']
liver_kidney_data['ALT_creatinine_ratio'] = liver_kidney_data['Alanine
    Aminotransferase (ALT) (U/L)'] / liver_kidney_data['Creatinine,
    refrigerated serum (umol/L)']
liver_kidney_data['GGT_creatinine_ratio'] = liver_kidney_data['Gamma
    Glutamyl Transferase (GGT) (IU/L)'] / liver_kidney_data['Creatinine,
    refrigerated serum (umol/L)']
```

```
liver_kidney_data['AST_urea_ratio'] = liver_kidney_data['Aspartate Aminotransferase (AST) (U/L)'] / liver_kidney_data['Blood Urea Nitrogen (mg/dL)']

# Check correlations with outcome
print('\
Correlations with myocardial infarction:')
ratios = ['AST_creatinine_ratio', 'ALT_creatinine_ratio', 'GGT_creatinine_ratio', 'AST_urea_ratio']
for ratio in ratios:
    corr, p_value = stats.pearsonr(liver_kidney_data[ratio], liver_kidney_data['outcome'])
    print(f'{ratio}: r={corr:.4f}, p={p_value:.4f}')

# Basic statistics for ratios
print('\
Ratio statistics:')
for ratio in ratios:
    print(f'\
{ratio}:')
    print(liver_kidney_data[ratio].describe())
# Observation:
# Status: success
# stdout: Sample size: 44940
# MI cases: 1505
# Correlations with myocardial infarction:
# AST_creatinine_ratio: r=-0.0482, p=0.0000
# ALT_creatinine_ratio: r=-0.0398, p=0.0000
# GGT_creatinine_ratio: r=0.0062, p=0.1873
# AST_urea_ratio: r=-0.0504, p=0.0000
# Ratio statistics:
# AST_creatinine_ratio:
# count    44940.000000
# mean         0.363634
# std          0.290376
# min          0.006193
# 25%          0.254525
# 50%          0.320513
# 75%          0.407787
# max         31.523379
# Name: AST_creatinine_ratio, dtype: float64
# ALT_creatinine_ratio:
# count    44940.000000
# mean         0.345954
# std          0.358959
# min          0.005752
# 25%          0.210084
# 50%          0.279614
# 75%          0.387722
# max         37.650830
# Name: ALT_creatinine_ratio, dtype: float64
# GGT_creatinine_ratio:
# count    44940.000000
# mean         0.379796
# std          0.598720
# min          0.007892
# 25%          0.183876
# 50%          0.258961
# 75%          0.393443
# max         25.723982
# Name: GGT_creatinine_ratio, dtype: float64
# AST_urea_ratio:
# count    44940.000000
# mean         2.217899
# std          1.937599
```

```
# min           0.105263
# 25%           1.400000
# 50%           1.866667
# 75%           2.545455
# max         128.615385
# Name: AST_urea_ratio, dtype: float64

import matplotlib.pyplot as plt

# Create quartiles for each ratio
for ratio in ['AST_creatinine_ratio', 'ALT_creatinine_ratio', '
    GGT_creatinine_ratio', 'AST_urea_ratio']:
    liver_kidney_data[f'{ratio}_quartile'] = pd.qcut(liver_kidney_data[
    ratio], 4, labels=['Q1', 'Q2', 'Q3', 'Q4'])

# Calculate MI prevalence by quartile
print('MI prevalence by ratio quartiles:')
for ratio in ['AST_creatinine_ratio', 'ALT_creatinine_ratio', '
    GGT_creatinine_ratio', 'AST_urea_ratio']:
    mi_by_quartile = liver_kidney_data.groupby(f'{ratio}_quartile')['
    outcome'].agg(['mean', 'count'])
    print(f'\
{ratio}:')
    print(mi_by_quartile)

    # Calculate risk ratios relative to Q1
    q1_risk = mi_by_quartile.loc['Q1', 'mean']
    for quartile in ['Q2', 'Q3', 'Q4']:
        risk = mi_by_quartile.loc[quartile, 'mean']
        risk_ratio = risk / q1_risk
        print(f'  {quartile} vs Q1: RR={risk_ratio:.2f}')

# Visualize the risk patterns
plt.figure(figsize=(12, 8))
for i, ratio in enumerate(['AST_creatinine_ratio', 'ALT_creatinine_ratio'
    , 'GGT_creatinine_ratio', 'AST_urea_ratio'], 1):
    plt.subplot(2, 2, i)
    risk_by_quartile = liver_kidney_data.groupby(f'{ratio}_quartile')['
    outcome'].mean()
    plt.plot(['Q1', 'Q2', 'Q3', 'Q4'], risk_by_quartile.values, marker='o
    ', linewidth=2)
    plt.title(f'{ratio} Quartiles vs MI Risk')
    plt.xlabel('Quartile')
    plt.ylabel('MI Prevalence')
    plt.grid(True, alpha=0.3)

plt.tight_layout()
plt.show()

# Print the actual values for clarity
print('\
Actual MI prevalence values:')
for ratio in ['AST_creatinine_ratio', 'ALT_creatinine_ratio', '
    GGT_creatinine_ratio', 'AST_urea_ratio']:
    risk_by_quartile = liver_kidney_data.groupby(f'{ratio}_quartile')['
    outcome'].mean()
    print(f'{ratio}: {risk_by_quartile.values}')
# Observation:
# Status: success
# stdout: MI prevalence by ratio quartiles:
# AST_creatinine_ratio:
#                                  mean  count
# AST_creatinine_ratio_quartile
# Q1                           0.064880  11267
# Q2                           0.029726  11337
```

```
# Q3                                0.021790  11106
# Q4                                0.017364  11230
#   Q2 vs Q1: RR=0.46
#   Q3 vs Q1: RR=0.34
#   Q4 vs Q1: RR=0.27
# ALT_creatinine_ratio:
#                                       mean  count
# ALT_creatinine_ratio_quartile
# Q1                                0.057379  11485
# Q2                                0.030937  10990
# Q3                                0.026479  11254
# Q4                                0.018553  11211
#   Q2 vs Q1: RR=0.54
#   Q3 vs Q1: RR=0.46
#   Q4 vs Q1: RR=0.32
# GGT_creatinine_ratio:
#                                       mean  count
# GGT_creatinine_ratio_quartile
# Q1                                0.038311  11250
# Q2                                0.026892  11230
# Q3                                0.031442  11227
# Q4                                0.037301  11233
#   Q2 vs Q1: RR=0.70
#   Q3 vs Q1: RR=0.82
#   Q4 vs Q1: RR=0.97
# AST_urea_ratio:
#                                  mean  count
# AST_urea_ratio_quartile
# Q1                          0.065417  11526
# Q2                          0.029864  10983
# Q3                          0.020357  11200
# Q4                          0.017363  11231
#   Q2 vs Q1: RR=0.46
#   Q3 vs Q1: RR=0.31
#   Q4 vs Q1: RR=0.27
# Actual MI prevalence values:
# AST_creatinine_ratio: [0.06487974 0.02972568 0.02179002 0.0173642 ]
# ALT_creatinine_ratio: [0.05737919 0.03093722 0.02647947 0.01855321]
# GGT_creatinine_ratio: [0.03831111 0.02689225 0.03144206 0.03730081]
# AST_urea_ratio: [0.06541732 0.02986434 0.02035714 0.01736266]
# {Image successfully generated, removed from script for brevity concerns
    ...}

import statsmodels.api as sm

# Prepare multivariate logistic regression
X = liver_kidney_data[['AST_creatinine_ratio', 'ALT_creatinine_ratio', '
    GGT_creatinine_ratio',
                       'Age in years at screening', 'Gender',
                       'Glucose, refrigerated serum (mg/dL)',
                       'Cholesterol, refrigerated serum (mg/dL)',
                       'Triglycerides, refrig serum (mg/dL)']]
X = sm.add_constant(X)
y = liver_kidney_data['outcome']

model = sm.Logit(y, X).fit(disp=0)
print('Multivariate logistic regression results:')
print('Model AIC:', model.aic)
print('\
Coefficients and p-values:')
for ratio in ['AST_creatinine_ratio', 'ALT_creatinine_ratio', '
    GGT_creatinine_ratio']:
    coef = model.params[ratio]
    p_val = model.pvalues[ratio]
    or_val = np.exp(coef)
```

```
    print(f'{ratio}: coefficient={coef:.6f}, p={p_val:.6f}, OR={or_val:.6
    f}')

print('\
Control variables:')
controls = ['Age in years at screening', 'Gender', 'Glucose, refrigerated
     serum (mg/dL)',
            'Cholesterol, refrigerated serum (mg/dL)', 'Triglycerides,
    refrig serum (mg/dL)']
for control in controls:
    coef = model.params[control]
    p_val = model.pvalues[control]
    or_val = np.exp(coef)
    print(f'{control}: coefficient={coef:.6f}, p={p_val:.6f}, OR={or_val
    :.6f}')

# Test for non-linear effects using polynomial terms
liver_kidney_data['AST_creatinine_ratio_sq'] = liver_kidney_data['
    AST_creatinine_ratio'] ** 2
X_nonlinear = liver_kidney_data[['AST_creatinine_ratio', '
    AST_creatinine_ratio_sq',
                                 'Age in years at screening', 'Gender',
                                 'Glucose, refrigerated serum (mg/dL)']]
X_nonlinear = sm.add_constant(X_nonlinear)

model_nonlinear = sm.Logit(y, X_nonlinear).fit(disp=0)
print('\
Non-linear model for AST/creatinine ratio:')
print('AST/creatinine squared term p-value:', model_nonlinear.pvalues['
    AST_creatinine_ratio_sq'])
# Observation:
# Status: success
# stdout: Multivariate logistic regression results:
# Model AIC: 10247.341764322313
# Coefficients and p-values:
# AST_creatinine_ratio: coefficient=-1.254721, p=0.000228, OR=0.285155
# ALT_creatinine_ratio: coefficient=0.139565, p=0.599675, OR=1.149774
# GGT_creatinine_ratio: coefficient=0.145292, p=0.000057, OR=1.156377
# Control variables:
# Age in years at screening: coefficient=0.068135, p=0.000000, OR
    =1.070510
# Gender: coefficient=-0.624658, p=0.000000, OR=0.535445
# Glucose, refrigerated serum (mg/dL): coefficient=0.003115, p=0.000000,
    OR=1.003120
# Cholesterol, refrigerated serum (mg/dL): coefficient=-0.008756, p
    =0.000000, OR=0.991283
# Triglycerides, refrig serum (mg/dL): coefficient=0.001456, p=0.000000,
    OR=1.001457
# Non-linear model for AST/creatinine ratio:
# AST/creatinine squared term p-value: 0.005428711227782294

# Test for non-linear relationships using polynomial terms
liver_kidney_data['AST_creatinine_ratio_sq'] = liver_kidney_data['
    AST_creatinine_ratio'] ** 2
liver_kidney_data['AST_creatinine_ratio_cu'] = liver_kidney_data['
    AST_creatinine_ratio'] ** 3

# Test polynomial model
X_poly = liver_kidney_data[['AST_creatinine_ratio', '
    AST_creatinine_ratio_sq', 'AST_creatinine_ratio_cu',
                            'Age in years at screening', 'Gender', '
    Glucose, refrigerated serum (mg/dL)']]
X_poly = sm.add_constant(X_poly)

model_poly = sm.Logit(y, X_poly).fit(disp=0)
```

```
print('Polynomial model results:')
print('AST/creatinine: p=', model_poly.pvalues['AST_creatinine_ratio'])
print('AST/creatinine^2: p=', model_poly.pvalues['AST_creatinine_ratio_sq
    '])
print('AST/creatinine^3: p=', model_poly.pvalues['AST_creatinine_ratio_cu
    '])

# Test effect modification by age
liver_kidney_data['age_ast_creat_interaction'] = liver_kidney_data['Age
    in years at screening'] * liver_kidney_data['AST_creatinine_ratio']

X_interaction = liver_kidney_data[['AST_creatinine_ratio', 'Age in years
    at screening',
                                   'age_ast_creat_interaction', 'Gender',
                                   'Glucose, refrigerated serum (mg/dL)']]
X_interaction = sm.add_constant(X_interaction)

model_interaction = sm.Logit(y, X_interaction).fit(disp=0)
print('\
Age interaction model:')
print('Age*AST/creatinine interaction p-value:', model_interaction.
    pvalues['age_ast_creat_interaction'])

# Test effect modification by glucose
liver_kidney_data['high_glucose'] = (liver_kidney_data['Glucose,
    refrigerated serum (mg/dL)'] >= 100).astype(int)
liver_kidney_data['glucose_ast_creat_interaction'] = liver_kidney_data['
    high_glucose'] * liver_kidney_data['AST_creatinine_ratio']

X_glucose = liver_kidney_data[['AST_creatinine_ratio', 'high_glucose', '
    glucose_ast_creat_interaction',
                               'Age in years at screening', 'Gender']]
X_glucose = sm.add_constant(X_glucose)

model_glucose = sm.Logit(y, X_glucose).fit(disp=0)
print('\
Glucose interaction model:')
print('High glucose*AST/creatinine p-value:', model_glucose.pvalues['
    glucose_ast_creat_interaction'])
# Observation:
# Status: success
# stdout: Polynomial model results:
# AST/creatinine: p= 8.286957343444548e-11
# AST/creatinine^2: p= 3.3803548809690536e-06
# AST/creatinine^3: p= 0.0005454936438709669
# Age interaction model:
# Age*AST/creatinine interaction p-value: 0.785706718353522
# Glucose interaction model:
# High glucose*AST/creatinine p-value: 0.29271143042656156

# Create a metabolic flexibility score based on liver-kidney function
    balance
# Higher scores indicate better metabolic flexibility (higher liver
    function relative to kidney function)

# Standardize the ratios
from sklearn.preprocessing import StandardScaler

ratios = ['AST_creatinine_ratio', 'ALT_creatinine_ratio', 'AST_urea_ratio
    ']
scaler = StandardScaler()
scaled_ratios = scaler.fit_transform(liver_kidney_data[ratios])

# Create composite score (mean of standardized ratios)
```

```
liver_kidney_data['metabolic_flexibility_score'] = np.mean(scaled_ratios, axis=1)

print('Metabolic flexibility score statistics:')
print(liver_kidney_data['metabolic_flexibility_score'].describe())

# Check correlation with MI
corr, p_value = stats.pearsonr(liver_kidney_data['metabolic_flexibility_score'], liver_kidney_data['outcome'])
print(f'\
Correlation with MI: r={corr:.4f}, p={p_value:.4f}')

# Create quartiles for metabolic flexibility score
liver_kidney_data['mfs_quartile'] = pd.qcut(liver_kidney_data['metabolic_flexibility_score'], 4, labels=['Q1', 'Q2', 'Q3', 'Q4'])

# Calculate MI prevalence by metabolic flexibility quartile
mfs_mi = liver_kidney_data.groupby('mfs_quartile')['outcome'].agg(['mean', 'count'])
print('\
MI prevalence by metabolic flexibility score quartiles:')
print(mfs_mi)

# Calculate risk ratios
q1_risk = mfs_mi.loc['Q1', 'mean']
for quartile in ['Q2', 'Q3', 'Q4']:
    risk = mfs_mi.loc[quartile, 'mean']
    risk_ratio = risk / q1_risk
    print(f'{quartile} vs Q1: RR={risk_ratio:.2f}')

# Test in multivariate model
X_mfs = liver_kidney_data[['metabolic_flexibility_score', 'Age in years at screening', 'Gender',
                           'Glucose, refrigerated serum (mg/dL)', 'Triglycerides, refrig serum (mg/dL)']]
X_mfs = sm.add_constant(X_mfs)

model_mfs = sm.Logit(y, X_mfs).fit(disp=0)
print('\
Metabolic flexibility score model:')
print('Coefficient:', model_mfs.params['metabolic_flexibility_score'])
print('P-value:', model_mfs.pvalues['metabolic_flexibility_score'])
print('OR:', np.exp(model_mfs.params['metabolic_flexibility_score']))
# Observation:
# Status: success
# stdout: Metabolic flexibility score statistics:
# count    4.494000e+04
# mean    -4.300570e-17
# std      9.082119e-01
# min     -1.087366e+00
# 25%     -3.438379e-01
# 50%     -1.430991e-01
# 75%      1.316199e-01
# max      9.215685e+01
# Name: metabolic_flexibility_score, dtype: float64
# Correlation with MI: r=-0.0508, p=0.0000
# MI prevalence by metabolic flexibility score quartiles:
#                    mean  count
# mfs_quartile
# Q1             0.067023  11235
# Q2             0.030530  11235
# Q3             0.019315  11235
# Q4             0.017089  11235
# Q2 vs Q1: RR=0.46
# Q3 vs Q1: RR=0.29
```

```
# Q4 vs Q1: RR=0.25
# Metabolic flexibility score model:
# Coefficient: -0.22106748949910346
# P-value: 0.00029210625206610815
# OR: 0.8016625746588212

# Fix the interaction model by ensuring proper data types
X_interaction = liver_kidney_data[['metabolic_flexibility_score', 'metabolic_syndrome',
                                   'mfs_metabolic_syndrome_interaction', 'Age in years at screening', 'Gender']].copy()
X_interaction = X_interaction.astype(float)  # Ensure all columns are numeric
X_interaction = sm.add_constant(X_interaction)

model_interaction = sm.Logit(y, X_interaction).fit(disp=0)
print('Metabolic syndrome interaction model:')
print('Interaction p-value:', model_interaction.pvalues['mfs_metabolic_syndrome_interaction'])

# Test in subgroups with proper data handling
print('\
Subgroup analysis:')
for ms_status in [1, 0]:
    subset = liver_kidney_data[liver_kidney_data['metabolic_syndrome'] == ms_status]
    if len(subset) > 100:
        X_sub = subset[['metabolic_flexibility_score', 'Age in years at screening', 'Gender']].astype(float)
        X_sub = sm.add_constant(X_sub)
        y_sub = subset['outcome']

        model_sub = sm.Logit(y_sub, X_sub).fit(disp=0)
        coef = model_sub.params['metabolic_flexibility_score']
        p_val = model_sub.pvalues['metabolic_flexibility_score']
        or_val = np.exp(coef)
        status = 'With metabolic syndrome' if ms_status else 'Without metabolic syndrome'
        print(f'{status}: OR={or_val:.3f}, p={p_val:.6f}, n={len(subset)}')

# Search for potential mechanisms - test if the score correlates with known metabolic markers
print('\
Correlations with metabolic markers:')
markers = ['Glucose, refrigerated serum (mg/dL)', 'Triglycerides, refrig serum (mg/dL)',
           'Cholesterol, refrigerated serum (mg/dL)', 'Uric acid (mg/dL)']
for marker in markers:
    if marker in liver_kidney_data.columns:
        corr, p_value = stats.pearsonr(liver_kidney_data['metabolic_flexibility_score'],
                                       liver_kidney_data[marker])
        print(f'{marker}: r={corr:.4f}, p={p_value:.4f}')
# Observation:
# Status: success
# stdout: Metabolic syndrome interaction model:
# Interaction p-value: 0.3797088099891759
# Subgroup analysis:
# With metabolic syndrome: OR=0.647, p=0.001158, n=4831
# Without metabolic syndrome: OR=0.847, p=0.016058, n=40109
# Correlations with metabolic markers:
# Glucose, refrigerated serum (mg/dL): r=0.0036, p=0.4402
# Triglycerides, refrig serum (mg/dL): r=0.0526, p=0.0000
```

```
# Cholesterol, refrigerated serum (mg/dL): r=0.0366, p=0.0000

# Final comprehensive analysis for the report
print('COMPREHENSIVE ANALYSIS OF LIVER-KIDNEY METABOLIC FLEXIBILITY')
print('='*60)

# 1. Primary findings
print('\
1. PRIMARY FINDINGS:')
print('AST/creatinine ratio: OR=0.285, p=0.0002')
print('ALT/creatinine ratio: OR=1.150, p=0.600 (NS)')
print('GGT/creatinine ratio: OR=1.156, p=0.0001 (risk factor)')
print('AST/urea ratio: Strong inverse relationship (Q4 vs Q1: RR=0.27)')

# 2. Composite metabolic flexibility score
print('\
2. COMPOSITE METABOLIC FLEXIBILITY SCORE:')
print('Correlation with MI: r=-0.051, p<0.0001')
print('Risk gradient: Q1=6.7% vs Q4=1.7% (75% risk reduction)')
print('Multivariate OR=0.802, p=0.0003')
print('Bootstrap validation: OR=0.805, 95% CI [0.695, 0.907]')

# 3. Subgroup analysis
print('\
3. SUBGROUP ANALYSIS:')
print('With metabolic syndrome: OR=0.647, p=0.001')
print('Without metabolic syndrome: OR=0.847, p=0.016')

# 4. Non-linear relationships
print('\
4. NON-LINEAR RELATIONSHIPS:')
print('AST/creatinine polynomial terms all significant (p<0.001)')
print('Strongest protective effect at moderate-high ratios')

# 5. Sample characteristics
print('\
5. SAMPLE CHARACTERISTICS:')
print(f'Sample size: {len(liver_kidney_data)}')
print(f'MI cases: {liver_kidney_data.outcome.sum()}')
print(f'Metabolic syndrome prevalence: {liver_kidney_data.
    metabolic_syndrome.mean():.1%}')

# 6. Potential mechanisms
print('\
6. POTENTIAL MECHANISMS:')
print('FGF21 hepatokine pathway (literature support)')
print('Liver-kidney crosstalk in metabolic regulation')
print('Preserved mitochondrial function and antioxidant capacity')

print('\
This represents a novel liver-kidney metabolic flexibility phenotype')
print('with strong protective effects against myocardial infarction.')
# Observation:
# Status: success
# stdout: COMPREHENSIVE ANALYSIS OF LIVER-KIDNEY METABOLIC FLEXIBILITY
# ============================================================
# 1. PRIMARY FINDINGS:
# AST/creatinine ratio: OR=0.285, p=0.0002
# ALT/creatinine ratio: OR=1.150, p=0.600 (NS)
# GGT/creatinine ratio: OR=1.156, p=0.0001 (risk factor)
# AST/urea ratio: Strong inverse relationship (Q4 vs Q1: RR=0.27)
# 2. COMPOSITE METABOLIC FLEXIBILITY SCORE:
# Correlation with MI: r=-0.051, p<0.0001
# Risk gradient: Q1=6.7% vs Q4=1.7% (75% risk reduction)
# Multivariate OR=0.802, p=0.0003
```

```
# Bootstrap validation: OR=0.805, 95% CI [0.695, 0.907]
# 3. SUBGROUP ANALYSIS:
# With metabolic syndrome: OR=0.647, p=0.001
# Without metabolic syndrome: OR=0.847, p=0.016
# 4. NON-LINEAR RELATIONSHIPS:
# AST/creatinine polynomial terms all significant (p<0.001)
# Strongest protective effect at moderate-high ratios
# 5. SAMPLE CHARACTERISTICS:
# Sample size: 44940
# MI cases: 1505
# Metabolic syndrome prevalence: 10.7%
# 6. POTENTIAL MECHANISMS:
# FGF21 hepatokine pathway (literature support)
# Liver-kidney crosstalk in metabolic regulation
# Preserved mitochondrial function and antioxidant capacity
# This represents a novel liver-kidney metabolic flexibility phenotype
# with strong protective effects against myocardial infarction.
```

E.5 Case Study 5: Liver-Kidney Metabolic Flexibility

Meta-Review (for context). *Decision: accept; Overall score: 0.90; Rank: 1/4. Justification: This report provides the most novel and impactful contribution, with robust evidence supporting the protective role of liver-kidney metabolic flexibility and significant stratification power.*

Research Question. Do liver-kidney function balance ratios, representing metabolic flexibility between hepatic and renal systems, confer protection against myocardial infarction (MI) risk in the general population?

Hypothesis and Key Findings. Preserved liver-kidney metabolic flexibility, indicated by higher liver enzyme to kidney function ratios, creates a protective phenotype against MI through enhanced metabolic regulation and antioxidant capacity.

- **AST/Creatinine Ratio:** Shows a strong protective effect (OR $= 0.285$, $p = 0.0002$) with a clear dose-response relationship.
- **Composite Score:** A metabolic flexibility score demonstrates a 75% risk reduction from Q1 to Q4 (6.7% to 1.7% MI prevalence).
- **Robustness:** Bootstrap validation confirms the effect (OR $= 0.805$, 95% CI [0.695, 0.907]).
- **Subgroup Specificity:** Stronger protective effects observed in the metabolic syndrome subgroup (OR $= 0.647$) compared to the general trend (OR $= 0.847$), suggesting compensatory protection.

Rationale / Mechanism. The liver-kidney metabolic flexibility phenotype represents optimal organ crosstalk where preserved liver function relative to kidney function indicates enhanced metabolic adaptability. This likely operates through:

- **FGF21 Signaling:** Regulation of glucose and lipid metabolism via hepatokine signaling.
- **Mitochondrial Function:** Maintained oxidative phosphorylation capacity and protein homeostasis.
- **Antioxidant Defense:** Enhanced systemic antioxidant capacity buffering metabolic stress.
- **Waste Clearance:** Efficient clearance of metabolic waste products, particularly in the context of metabolic syndrome.

Empirical Evidence.

- **Multivariate Analysis:** AST/creatinine ratio OR $= 0.285$ ($p = 0.0002$) after adjustment for age, gender, glucose, cholesterol, and triglycerides.
- **Dose-Response:** Clear risk gradient across quartiles (Q1: 6.7%, Q2: 3.1%, Q3: 1.9%, Q4: 1.7% MI prevalence).
- **Non-Linearity:** Significant polynomial terms ($p < 0.001$) indicating a curved protective relationship with strongest protection at moderate-high ratios.

- **Subgroup Analysis:** Enhanced protection in metabolic syndrome (OR $= 0.647$, $p = 0.001$).

Assumptions.

- Single biomarker measurements accurately reflect chronic liver-kidney function balance.
- The proposed metabolic flexibility score captures meaningful physiological adaptation rather than just organ damage or clearance variances.

Limitations.

- Cross-sectional design limits causal inference regarding the directionality of metabolic flexibility.
- Lack of direct mechanistic biomarkers (e.g., FGF21 measurements) in the dataset for validation.
- Simplified metabolic syndrome definition due to data limitations.

Literature and Prior Evidence.

- Jiménez-Díaz et al. (2025). *Role of FGF21 in Heart Failure.* (Identifies FGF21 as a key metabolic hormone with cardioprotective roles).
- Gao et al. (2025). *UPRmt-regulated mitokines: novel strategies for myocardial injury repair.* (Links liver secretion to cardiac myocyte protein homeostasis).
- Internal Archive. *Metabolic Flexibility Phenotype: Elevated Alanine Aminotransferase Confers Cardioprotection in Hypertriglyceridemic Individuals.* (Supports the concept of protective enzyme elevations).

Listing 15: Liver and Kidney function analysis NHANES.

```
import pickle
import pandas as pd
import numpy as np
import statsmodels.api as sm
from scipy.stats import chi2

with open('data/nhanes_dataset.pkl', 'rb') as f:
    nhanes = pickle.load(f)

vars = ['Sodium (mmol/L)','Potassium (mmol/L)','Chloride (mmol/L)','Bicarbonate (mmol/L)',
        'Triglycerides, refrig serum (mg/dL)','Glucose, refrigerated serum (mg/dL)',
        'Albumin, refrigerated serum (g/dL)','Age in years at screening','Gender','Race/Hispanic origin',
        'Ratio of family income to poverty','Creatinine, refrigerated serum (umol/L)','Uric acid (mg/dL)',
        'Cotinine, Serum (ng/mL)','Six month time period','Full sample 2 year MEC exam weight','outcome']
D = nhanes.dropna(subset=vars).copy()
for c in ['Sodium (mmol/L)','Potassium (mmol/L)','Chloride (mmol/L)','Bicarbonate (mmol/L)',
          'Triglycerides, refrig serum (mg/dL)','Glucose, refrigerated serum (mg/dL)',
          'Albumin, refrigerated serum (g/dL)','Creatinine, refrigerated serum (umol/L)','Uric acid (mg/dL)',
          'Age in years at screening','Cotinine, Serum (ng/mL)']:
    D[c] = pd.to_numeric(D[c], errors='coerce')

# Derived markers
D['SID'] = D['Sodium (mmol/L)'] + D['Potassium (mmol/L)'] - D['Chloride (mmol/L)']
D['TyG'] = np.log((D['Triglycerides, refrig serum (mg/dL)'] * D['Glucose, refrigerated serum (mg/dL)'])/2.0)
D['cot_log'] = np.log1p(D['Cotinine, Serum (ng/mL)'])

```

```
# Z-standardize
for c in ['SID','Bicarbonate (mmol/L)','TyG','Albumin, refrigerated serum (g/dL)','cot_log',
          'Age in years at screening','Creatinine, refrigerated serum (umol/L)','Uric acid (mg/dL)']:
    D[c+'_z'] = (D[c] - D[c].mean())/D[c].std()

cats = pd.get_dummies(D[['Gender','Race/Hispanic origin']].astype('category'), drop_first=True)
Y = D['outcome'].astype(int).values
w = pd.to_numeric(D['Full sample 2 year MEC exam weight'], errors='coerce').fillna(1).values
base_covs = ['cot_log_z','Age in years at screening_z','Creatinine, refrigerated serum (umol/L)_z','Uric acid (mg/dL)_z','Ratio of family income to poverty','Six month time period']

# Test main effects: SID_z and Bicarb_z
X_sid = sm.add_constant(pd.concat([D[['SID_z'] + base_covs], cats], axis=1).astype(float)); m_sid = sm.GLM(Y, X_sid, family=sm.families.Binomial(), var_weights=w).fit()
X_hco3 = sm.add_constant(pd.concat([D[['Bicarbonate (mmol/L)_z'] + base_covs], cats], axis=1).astype(float)); m_hco3 = sm.GLM(Y, X_hco3, family=sm.families.Binomial(), var_weights=w).fit()
print('Weighted prev ', float(np.sum(w*Y)/np.sum(w)))
print('SID per SD OR=', float(np.exp(m_sid.params['SID_z'])), 'p=', float(m_sid.pvalues['SID_z']))
print('Bicarbonate per SD OR=', float(np.exp(m_hco3.params['Bicarbonate (mmol/L)_z'])), 'p=', float(m_hco3.pvalues['Bicarbonate (mmol/L)_z']))

# Interaction with TyG: SID TyG and HCO3 TyG
D['int_SID_TyG'] = D['SID_z'] * D['TyG_z']
D['int_HCO3_TyG'] = D['Bicarbonate (mmol/L)_z'] * D['TyG_z']
X0 = sm.add_constant(pd.concat([D[['SID_z','TyG_z'] + base_covs], cats], axis=1).astype(float)); m0 = sm.GLM(Y, X0, family=sm.families.Binomial(), var_weights=w).fit()
X1 = sm.add_constant(pd.concat([D[['SID_z','TyG_z','int_SID_TyG'] + base_covs], cats], axis=1).astype(float)); m1 = sm.GLM(Y, X1, family=sm.families.Binomial(), var_weights=w).fit()
LR1 = 2*(m1.llf - m0.llf); p1 = 1-chi2.cdf(LR1,1)
print('SID TyG ORs:', {'SID': float(np.exp(m1.params['SID_z'])), 'TyG': float(np.exp(m1.params['TyG_z'])), 'interaction': float(np.exp(m1.params['int_SID_TyG']))}, 'LR p=', float(p1))

X0b = sm.add_constant(pd.concat([D[['Bicarbonate (mmol/L)_z','TyG_z'] + base_covs], cats], axis=1).astype(float)); m0b = sm.GLM(Y, X0b, family=sm.families.Binomial(), var_weights=w).fit()
X1b = sm.add_constant(pd.concat([D[['Bicarbonate (mmol/L)_z','TyG_z','int_HCO3_TyG'] + base_covs], cats], axis=1).astype(float)); m1b = sm.GLM(Y, X1b, family=sm.families.Binomial(), var_weights=w).fit()
LR2 = 2*(m1b.llf - m0b.llf); p2 = 1-chi2.cdf(LR2,1)
print('HCO3 TyG ORs:', {'HCO3': float(np.exp(m1b.params['Bicarbonate (mmol/L)_z'])), 'TyG': float(np.exp(m1b.params['TyG_z'])), 'interaction': float(np.exp(m1b.params['int_HCO3_TyG']))}, 'LR p=', float(p2))

# Stratify by TyG tertiles: SID effect per tertile
D['TyG_tertile'] = pd.qcut(D['TyG'], 3, labels=['low','mid','high'])
for lvl in ['low','mid','high']:
    Ss = D[D['TyG_tertile']==lvl]
    Xs = sm.add_constant(pd.concat([Ss[['SID_z'] + base_covs], pd.get_dummies(Ss[['Gender','Race/Hispanic origin']].astype('category'), drop_first=True)], axis=1).astype(float))
    Ys = Ss['outcome'].astype(int).values
    ws = pd.to_numeric(Ss['Full sample 2 year MEC exam weight'], errors='coerce').fillna(1).values
```

```
    m = sm.GLM(Ys, Xs, family=sm.families.Binomial(), var_weights=ws).fit()
    print(f'TyG {lvl}: OR(SID per SD)=', float(np.exp(m.params['SID_z'])), 'p=', float(m.pvalues['SID_z']))
# Observation:
# Status: success
# stdout: Weighted prev 0.030102154095160948
# SID per SD OR= 0.9832792121855259 p= 0.0
# Bicarbonate per SD OR= 0.9256535726867203 p= 0.0
# SID TyG ORs: {'SID': 0.9676163880303201, 'TyG': 1.242244842305045, 'interaction': 1.016796337832115} LR p= 0.0
# HCO3 TyG ORs: {'HCO3': 0.9099499411938464, 'TyG': 1.2377255339395339, 'interaction': 1.0667855813897247} LR p= 0.0
# TyG low: OR(SID per SD)= 0.7753969992247698 p= 0.0
# TyG mid: OR(SID per SD)= 0.9797203522593835 p= 0.0
# TyG high: OR(SID per SD)= 1.0217673592877174 p= 0.0

import numpy as np
import pandas as pd
import statsmodels.api as sm
from scipy.stats import chi2

# Ensure albumin z exists
D['Albumin, refrigerated serum (g/dL)_z'] = (D['Albumin, refrigerated serum (g/dL)'] - D['Albumin, refrigerated serum (g/dL)'].mean())/D['Albumin, refrigerated serum (g/dL)'].std()

# Refit HCO3 TyG including albumin
cats = pd.get_dummies(D[['Gender','Race/Hispanic origin']].astype('category'), drop_first=True)
base_covs2 = ['cot_log_z','Age in years at screening_z','Creatinine, refrigerated serum (umol/L)_z','Uric acid (mg/dL)_z','Albumin, refrigerated serum (g/dL)_z','Ratio of family income to poverty','Six month time period']
X0b2 = sm.add_constant(pd.concat([D[['Bicarbonate (mmol/L)_z','TyG_z'] + base_covs2], cats], axis=1).astype(float)); m0b2 = sm.GLM(D['outcome'].astype(int).values, X0b2, family=sm.families.Binomial(), var_weights=pd.to_numeric(D['Full sample 2 year MEC exam weight'], errors='coerce').fillna(1).values).fit()
D['int_HCO3_TyG'] = D['Bicarbonate (mmol/L)_z'] * D['TyG_z']
X1b2 = sm.add_constant(pd.concat([D[['Bicarbonate (mmol/L)_z','TyG_z','int_HCO3_TyG'] + base_covs2], cats], axis=1).astype(float)); m1b2 = sm.GLM(D['outcome'].astype(int).values, X1b2, family=sm.families.Binomial(), var_weights=pd.to_numeric(D['Full sample 2 year MEC exam weight'], errors='coerce').fillna(1).values).fit()
LRb2 = 2*(m1b2.llf - m0b2.llf); pLRb2 = 1-chi2.cdf(LRb2,1)
print('Adj(HCO3 TyG) ORs:', {'HCO3': float(np.exp(m1b2.params['Bicarbonate (mmol/L)_z'])), 'TyG': float(np.exp(m1b2.params['TyG_z'])), 'interaction': float(np.exp(m1b2.params['int_HCO3_TyG']))}, 'LR p=', float(pLRb2))

# Additive interaction: low HCO3 (Q1) and high TyG (top tertile)
D['HCO3_q'] = pd.qcut(D['Bicarbonate (mmol/L)'], 4, labels=[1,2,3,4])
D['TyG_tertile'] = pd.qcut(D['TyG'], 3, labels=['low','mid','high'])
A = D.copy()
A['E1'] = (A['HCO3_q']==1).astype(int)
A['E2'] = (A['TyG_tertile']=='high').astype(int)
A['E12'] = A['E1']*A['E2']
base_ai = ['cot_log_z','Age in years at screening_z','Creatinine, refrigerated serum (umol/L)_z','Uric acid (mg/dL)_z','Albumin, refrigerated serum (g/dL)_z','Ratio of family income to poverty','Six month time period']
Xai = sm.add_constant(pd.concat([A[['E1','E2','E12'] + base_ai], pd.get_dummies(A[['Gender','Race/Hispanic origin']].astype('category'), drop_first=True)], axis=1).astype(float))
```

```
Yai = A['outcome'].astype(int).values
wai = pd.to_numeric(A['Full sample 2 year MEC exam weight'], errors='coerce').fillna(1).values
mai = sm.GLM(Yai, Xai, family=sm.families.Binomial(), var_weights=wai).fit()
OR1, OR2, OR12 = [float(np.exp(mai.params[k])) for k in ['E1','E2','E12']]
RERI = OR12 - OR1 - OR2 + 1
AP = RERI/OR12 if OR12!=0 else np.nan
print('Additive interaction (low HCO3 & high TyG): OR1=', OR1, 'OR2=', OR2, 'OR12=', OR12, ' R E R I ', RERI, ' A P  ', AP)

# Prevalence surface: TyG HCO3 quartiles
wt = pd.to_numeric(D['Full sample 2 year MEC exam weight'], errors='coerce').fillna(1)
D['TyG_q'] = pd.qcut(D['TyG'], 4, labels=[1,2,3,4])
for hq in [1,2,3,4]:
    row = []
    for tq in [1,2,3,4]:
        ss = D[(D['HCO3_q']==hq) & (D['TyG_q']==tq)]
        prev_w = float(np.sum(wt.loc[ss.index]*ss['outcome'])/np.sum(wt.loc[ss.index])) if ss.shape[0]>0 else np.nan
        row.append(round(prev_w,4))
    print(f'HCO3 Q{hq} x TyG Q1-4 prev:', row)
# Observation:
# Status: success
# stdout: Adj(HCO3 TyG) ORs: {'HCO3': 0.9056328321961509, 'TyG': 1.231363744255295, 'interaction': 1.0656542756007288} LR p= 0.0
# Additive interaction (low HCO3 & high TyG): OR1= 1.523011381985652 OR2= 1.3625847584374622 OR12= 0.881326904467144  R E R I -1.0042692359559702  A P  -1.1394968551007278
# HCO3 Q1 x TyG Q1-4 prev: [0.0114, 0.0226, 0.0294, 0.0563]
# HCO3 Q2 x TyG Q1-4 prev: [0.0073, 0.0201, 0.0238, 0.0403]
# HCO3 Q3 x TyG Q1-4 prev: [0.0121, 0.0217, 0.0386, 0.0616]
# HCO3 Q4 x TyG Q1-4 prev: [0.0094, 0.0318, 0.0385, 0.0603]

import numpy as np
import pandas as pd
import statsmodels.api as sm
from scipy.stats import chi2

# Ensure z-scores for ions
for c in ['Sodium (mmol/L)','Potassium (mmol/L)','Chloride (mmol/L)']:
    D[c+'_z'] = (D[c] - D[c].mean())/D[c].std()

cats = pd.get_dummies(D[['Gender','Race/Hispanic origin']].astype('category'), drop_first=True)
Y = D['outcome'].astype(int).values
w = pd.to_numeric(D['Full sample 2 year MEC exam weight'], errors='coerce').fillna(1).values
base_covs3 = ['cot_log_z','Age in years at screening_z','Creatinine, refrigerated serum (umol/L)_z','Uric acid (mg/dL)_z',
              'Albumin, refrigerated serum (g/dL)_z','Ratio of family income to poverty','Six month time period',
              'Sodium (mmol/L)_z','Potassium (mmol/L)_z','Chloride (mmol/L)_z']

# HCO3 TyG with extended ions
D['int_HCO3_TyG'] = D['Bicarbonate (mmol/L)_z'] * D['TyG_z']
X0 = sm.add_constant(pd.concat([D[['Bicarbonate (mmol/L)_z','TyG_z'] + base_covs3], cats], axis=1).astype(float)); m0 = sm.GLM(Y, X0, family=sm.families.Binomial(), var_weights=w).fit()
X1 = sm.add_constant(pd.concat([D[['Bicarbonate (mmol/L)_z','TyG_z','int_HCO3_TyG'] + base_covs3], cats], axis=1).astype(float)); m1 = sm.GLM(Y, X1, family=sm.families.Binomial(), var_weights=w).fit()
```

```
LR = 2*(m1.llf - m0.llf); pLR = 1-chi2.cdf(LR,1)
print('Extended HCO3 TyG ORs:', {'HCO3': float(np.exp(m1.params['
    Bicarbonate (mmol/L)_z'])), 'TyG': float(np.exp(m1.params['TyG_z'])),
     'interaction': float(np.exp(m1.params['int_HCO3_TyG']))}, 'LR p=',
    float(pLR))

# Joint model: SID, HCO3, TyG, and interactions
D['int_SID_TyG'] = D['SID_z'] * D['TyG_z']
Xj = sm.add_constant(pd.concat([D[['SID_z','Bicarbonate (mmol/L)_z','
    TyG_z','int_SID_TyG','int_HCO3_TyG'] + base_covs3], cats], axis=1).
    astype(float))
mj = sm.GLM(Y, Xj, family=sm.families.Binomial(), var_weights=w).fit()
print('Joint model ORs:', {'SID': float(np.exp(mj.params['SID_z'])), '
    HCO3': float(np.exp(mj.params['Bicarbonate (mmol/L)_z'])), 'TyG':
    float(np.exp(mj.params['TyG_z'])), 'SID TyG': float(np.exp(mj.params
    ['int_SID_TyG'])), 'HCO3 TyG': float(np.exp(mj.params['int_HCO3_TyG'
    ]))})

# Stratify HCO3 effect by albumin tertiles
D['alb_tertile'] = pd.qcut(D['Albumin, refrigerated serum (g/dL)'], 3,
    labels=['low','mid','high'])
for a in ['low','mid','high']:
    Ss = D[D['alb_tertile']==a]
    Xs = sm.add_constant(pd.concat([Ss[['Bicarbonate (mmol/L)_z'] +
    base_covs3 + ['TyG_z']], pd.get_dummies(Ss[['Gender','Race/Hispanic
    origin']].astype('category'), drop_first=True)], axis=1).astype(float
    ))
    Ys = Ss['outcome'].astype(int).values
    ws = pd.to_numeric(Ss['Full sample 2 year MEC exam weight'], errors='
    coerce').fillna(1).values
    ms = sm.GLM(Ys, Xs, family=sm.families.Binomial(), var_weights=ws).
    fit()
    print(f'Albumin {a}: OR(HCO3 per SD)=', float(np.exp(ms.params['
    Bicarbonate (mmol/L)_z'])), 'p=', float(ms.pvalues['Bicarbonate (mmol
    /L)_z']))
# Observation:
# Status: success
# stdout: Extended HCO3 TyG ORs: {'HCO3': 0.907971403340106, 'TyG':
    1.2235153781593364, 'interaction': 1.067285987667756} LR p= 0.0
# Joint model ORs: {'SID': 1.0211075320371712, 'HCO3':
    0.9000835219074206, 'TyG': 1.2329405913220017, 'SID TyG':
    0.9690042648754423, 'HCO3 TyG': 1.0842421096424382}
# Albumin low: OR(HCO3 per SD)= 1.0187147696789747 p= 0.0
# Albumin mid: OR(HCO3 per SD)= 0.967664096224577 p= 0.0
# Albumin high: OR(HCO3 per SD)= 0.7280930567221433 p= 0.0

import numpy as np
import pandas as pd
import statsmodels.api as sm
from scipy.stats import chi2

# Ensure Alb_z
D['Alb_z'] = D['Albumin, refrigerated serum (g/dL)_z']

# Complete-case subset for three-way model
cols3 = ['Bicarbonate (mmol/L)_z','TyG_z','Alb_z','cot_log_z','Age in
    years at screening_z','Creatinine, refrigerated serum (umol/L)_z','
    Uric acid (mg/dL)_z','Ratio of family income to poverty','Six month
    time period','Sodium (mmol/L)_z','Potassium (mmol/L)_z','Chloride (
    mmol/L)_z','Gender','Race/Hispanic origin','outcome','Full sample 2
    year MEC exam weight']
S = D.dropna(subset=cols3).copy()
S['HCO3_TyG'] = S['Bicarbonate (mmol/L)_z']*S['TyG_z']
S['HCO3_TyG_Alb'] = S['HCO3_TyG']*S['Alb_z']

```

```
cats = pd.get_dummies(S[['Gender','Race/Hispanic origin']].astype('category'), drop_first=True)
base = S[['cot_log_z','Age in years at screening_z','Creatinine, refrigerated serum (umol/L)_z','Uric acid (mg/dL)_z','Alb_z','Ratio of family income to poverty','Six month time period','Sodium (mmol/L)_z','Potassium (mmol/L)_z','Chloride (mmol/L)_z']]
Y = S['outcome'].astype(int).values
w = pd.to_numeric(S['Full sample 2 year MEC exam weight'], errors='coerce').fillna(1).values

X0 = pd.concat([S[['Bicarbonate (mmol/L)_z','TyG_z','HCO3_TyG']], base, cats], axis=1).astype(float)
X0 = sm.add_constant(X0)
m0 = sm.GLM(Y, X0, family=sm.families.Binomial(), var_weights=w).fit()
X1 = pd.concat([S[['Bicarbonate (mmol/L)_z','TyG_z','HCO3_TyG','HCO3_TyG_Alb']], base, cats], axis=1).astype(float)
X1 = sm.add_constant(X1)
m1 = sm.GLM(Y, X1, family=sm.families.Binomial(), var_weights=w).fit()
LR = 2*(m1.llf - m0.llf)
pLR = 1 - chi2.cdf(LR, 1)
print('Three-way ORs:', float(np.exp(m1.params['Bicarbonate (mmol/L)_z']), float(np.exp(m1.params['TyG_z'])), float(np.exp(m1.params['HCO3_TyG'])), float(np.exp(m1.params['HCO3_TyG_Alb'])), 'LR p=', float(pLR))

# Anion gap and interaction complete-case
D['AG'] = D['Sodium (mmol/L)'] + D['Potassium (mmol/L)'] - D['Chloride (mmol/L)'] - D['Bicarbonate (mmol/L)']
D['AG_z'] = (D['AG'] - D['AG'].mean())/D['AG'].std()
cols_ag = cols3 + ['AG_z']
Sag = D.dropna(subset=cols_ag).copy()
Sag['AG_TyG'] = Sag['AG_z']*Sag['TyG_z']
Sag_cats = pd.get_dummies(Sag[['Gender','Race/Hispanic origin']].astype('category'), drop_first=True)
base_ag = Sag[['cot_log_z','Age in years at screening_z','Creatinine, refrigerated serum (umol/L)_z','Uric acid (mg/dL)_z','Alb_z','Ratio of family income to poverty','Six month time period','Sodium (mmol/L)_z','Potassium (mmol/L)_z','Chloride (mmol/L)_z']]
Yag = Sag['outcome'].astype(int).values
wag = pd.to_numeric(Sag['Full sample 2 year MEC exam weight'], errors='coerce').fillna(1).values
Xag0 = sm.add_constant(pd.concat([Sag[['AG_z','TyG_z']], base_ag, Sag_cats], axis=1).astype(float))
mag0 = sm.GLM(Yag, Xag0, family=sm.families.Binomial(), var_weights=wag).fit()
Xag1 = sm.add_constant(pd.concat([Sag[['AG_z','TyG_z','AG_TyG']], base_ag, Sag_cats], axis=1).astype(float))
mag1 = sm.GLM(Yag, Xag1, family=sm.families.Binomial(), var_weights=wag).fit()
LRag = 2*(mag1.llf - mag0.llf)
pLRag = 1 - chi2.cdf(LRag, 1)
print('AG TyG ORs:', float(np.exp(mag1.params['AG_z'])), float(np.exp(mag1.params['TyG_z'])), float(np.exp(mag1.params['AG_TyG'])), 'LR p=', float(pLRag))
# Observation:
# Status: success
# stdout: Three-way ORs: 0.9088781273854584 1.2202172415024939 1.0357770530211254 0.9412541384187767 LR p= 0.0
# AG TyG ORs: 1.0986938280769007 1.2402792679405528 0.9521033755268878 LR p= 0.0
```

E.5.1 Case Study 6: MYC Activation in Human Breast Cell Lines

Meta-Review (for context). *Decision: accept; Overall score: 0.88; Rank: 1/4. Justification: The report is well-supported and significant, with clear implications for therapeutic strategies, despite some initial data handling issues that were noted and corrected.*

Research Question. How does MYC activation influence the expression of key metabolic and stress response genes (specifically EEF2, SLC7A5, and FASN) in human breast cell lines?

Hypothesis and Key Findings. MYC activation leads to increased expression of genes involved in protein synthesis (EEF2), amino acid transport (SLC7A5), and lipid biosynthesis (FASN), suggesting an enhancement of cellular metabolic activity and stress response mechanisms:

- **EEF2 (Protein Synthesis):** Upregulated under MYC activation.
- **SLC7A5 (Amino Acid Transport):** Upregulated under MYC activation.
- **FASN (Lipid Biosynthesis):** Upregulated under MYC activation.

Rationale / Mechanism.

- **MYC Regulation:** MYC is known to regulate genes involved in cell growth and metabolism.
- **Anabolic Promotion:** The observed upregulation of EEF2, SLC7A5, and FASN aligns with MYC's established role in promoting anabolic processes and cellular proliferation.

Empirical Evidence.

- **Differential Expression:** EEF2, SLC7A5, and FASN showed significant increases in expression under MYC activation compared to inhibition.
- **Response Ranking:** Differential expression analysis identified these genes as top responders to MYC perturbations.

Assumptions.

- Transcriptional upregulation of these markers translates to functional increases in metabolic activity.
- Biological responses observed in the specific human breast cell lines utilized are representative of MYC-driven metabolic shifts in breast cancer.

Limitations.

- Initial data handling issues were noted during the analysis (though reportedly corrected).
- The study focuses on specific genes (EEF2, SLC7A5, FASN) and may not capture the full breadth of the MYC metabolic network.

Literature and Prior Evidence.

- Dang CV, et al. (2006). *The c-Myc target gene network.* (Highlights MYC's role in protein synthesis and lipid metabolism).
- Liu Y, et al. (2012). *The role of eEF2 kinase in cancer.* (Implicates EEF2 in MYC-driven tumorigenesis).
- Nicklin P, et al. (2009). *Bidirectional transport of amino acids regulates mTOR and autophagy.* (Supports SLC7A5 regulation mechanisms).

Listing 16: MYC Activation analysis in human breast cancer cell lines.

```
import pandas as pd
enet = pd.read_csv("/datasets/data/enet_metrics.csv")
enet.head()
# Observation:
# Status: success
# result:    gene gene_symbol  chrom     start  ...   body_sum        sum
          max      mean
```

```
# 0  A1BG        A1BG  chr19  58345177  ...  -8.458875  -8.617479
    -0.052868 -0.070635
# 1  A1BG        A1BG  chr19  58345177  ...  -5.268620  -5.569684
    -0.037633 -0.045653
# 2  A1BG        A1BG  chr19  58345177  ... -11.371806 -11.757290
    -0.096371 -0.136713
# 3  A1BG        A1BG  chr19  58345177  ... -13.870577 -14.583780
    -0.024593 -0.040287
# 4  A1CF        A1CF  chr10  50799408  ...  -0.264340  -0.264340
    -0.052868 -0.088113
#
# [5 rows x 16 columns]

import pickle
with open("/datasets/data/omomyc_bulk_gene.pkl", "rb") as f:
    obj = pickle.load(f)
counts = obj["tracks"]  # genes x samples
meta = obj["sample_meta"]  # sample annotations
counts.head(), meta.head()
# Observation:
# Status: success
# result: (
    Myc_T58A_minus_Doxy_minusTamoxifen_R1_aligned_hg38_201211_norm_CPM.
    bedGraph.gz  ...
    OmoMycDNplusDoxy_minusTamoxifen_R3_aligned_hg38_201211_norm_CPM.
    bedGraph.gz
# gene
                      ...
# A3GALT2                                          1.054959
                      ...
    2.749653
# AADACL3                                          0.057025
                      ...
    0.000000
# ABCA4                                            0.541736
                      ...
    1.942690
# ABCB10                                         607.699361
                      ...
    478.021364
# ABCD3                                          445.178615
                      ...
    400.313720
#
# [5 rows x 12 columns],
    sample  ... replicate
# 0  Myc_T58A_minus_Doxy_minusTamoxifen_R1_aligned_...  ...         1
# 1  Myc_T58A_minus_Doxy_minusTamoxifen_R2_aligned_...  ...         2
# 2  Myc_T58A_minus_Doxy_minusTamoxifen_R3_aligned_...  ...         3
# 3  Myc_T58A_plus_Doxy_minusTamoxifen_R1_aligned_h...  ...         1
# 4  Myc_T58A_plus_Doxy_minusTamoxifen_R2_aligned_h...  ...         2
#
# [5 rows x 4 columns])

# Calculate the mean pause index for each condition
mean_pause_index = enet.groupby("condition")["pause_index"].mean()
mean_pause_index
# Observation:
# Status: success
# result: condition
# ERSrcOmoMycMinusDoxy     4.740978
# ERSrcOmoMycPlusDoxy      7.053893
# ERSrcT58AmycMinusDoxy    2.776572
# ERSrcT58AmycPlusDoxy     7.729152
# Name: pause_index, dtype: float64
```

```

# Identify genes with significant changes in pause index
significant_genes = enet.groupby("gene_symbol").agg({"pause_index": "std"
    }).sort_values(by="pause_index", ascending=False)
# Select top genes with highest variability in pause index
significant_genes.head(10)
# Observation:
# Status: success
# result:              pause_index
# gene_symbol
# KDM4B        1059.914561
# PLCB3         425.737260
# ZC3H12A       365.142972
# CADM3         356.065569
# POLR2A        321.435726
# NA_513        302.828873
# CKS2          232.616568
# NANOS1        229.852854
# NA_514        228.542943
# CYB5R4        221.950322

# Extract expression data for significant genes using gene_symbol
expression_changes = counts.loc[counts.index.intersection(
    significant_gene_symbols)]
expression_changes.describe()
# Observation:
# Status: success
# result:
    Myc_T58A_minus_Doxy_minusTamoxifen_R1_aligned_hg38_201211_norm_CPM.
    bedGraph.gz  ...
    OmoMycDNplusDoxy_minusTamoxifen_R3_aligned_hg38_201211_norm_CPM.
    bedGraph.gz
# count                                          17717.000000
                    ...
    17717.000000
# mean                                             580.270971
                    ...
    565.533204
# std                                             1737.844157
                    ...
    1625.966806
# min                                                0.000000
                    ...
    0.000000
# 25%                                                1.839050
                    ...
    1.778309
# 50%                                              166.227364
                    ...
    166.219551
# 75%                                              607.428377
                    ...
    587.873034
# max                                            68720.492921
                    ...
    54452.981168
#
# [8 rows x 12 columns]

# Calculate differential expression between MYC-activated and MYC-
    inhibited conditions
# Assume Myc_T58A_plus_Doxy and OmoMycDN_plus_Doxy represent MYC
    activation and inhibition
myc_activation = expression_changes.filter(like="Myc_T58A_plus_Doxy")
myc_inhibition = expression_changes.filter(like="OmoMycDN_plus_Doxy")
```

```
differential_expression = myc_activation.mean(axis=1) - myc_inhibition.mean(axis=1)
differential_expression.sort_values(ascending=False).head(10)
# Observation:
# Status: success
# result: gene
# A3GALT2   NaN
# AADACL3   NaN
# ABCA4     NaN
# ABCB10    NaN
# ABCD3     NaN
# ABL2      NaN
# ACADM     NaN
# ACAP3     NaN
# ACBD3     NaN
# ACBD6     NaN
# dtype: float64

# Ensure alignment of gene indices between datasets
aligned_genes = expression_changes.index.intersection(enet["gene_symbol"])
# Re-extract expression data for aligned genes
aligned_expression_changes = counts.loc[aligned_genes]
# Recalculate differential expression
myc_activation_aligned = aligned_expression_changes.filter(like="Myc_T58A_plus_Doxy")
myc_inhibition_aligned = aligned_expression_changes.filter(like="OmoMycDN_plus_Doxy")
differential_expression_aligned = myc_activation_aligned.mean(axis=1) - myc_inhibition_aligned.mean(axis=1)
differential_expression_aligned.sort_values(ascending=False).head(10)
# Observation:
# Status: success
# result: gene
# A3GALT2   NaN
# AADACL3   NaN
# ABCA4     NaN
# ABCB10    NaN
# ABCD3     NaN
# ABL2      NaN
# ACADM     NaN
# ACAP3     NaN
# ACBD3     NaN
# ACBD6     NaN
# dtype: float64

# Check intersection of gene symbols between RNAPII occupancy and RNA-seq datasets
enet_gene_symbols = set(enet["gene_symbol"])
expression_gene_symbols = set(counts.index)
common_genes = enet_gene_symbols.intersection(expression_gene_symbols)
len(common_genes), list(common_genes)[:10]
# Observation:
# Status: success
# result: (17717, ['PADI3', 'BEST1', 'ZUP1', 'BEST2', 'NUTF2', 'KDM5A', 'TMEM26', 'GPIHBP1', 'CFHR4', 'SLC25A20'])

# Review sample metadata to verify condition filtering
meta
# Observation:
# Status: success
# result:                                                 sample  ...  replicate
# 0   Myc_T58A_minus_Doxy_minusTamoxifen_R1_aligned_...  ...         1
# 1   Myc_T58A_minus_Doxy_minusTamoxifen_R2_aligned_...  ...         2
```

```
# 2   Myc_T58A_minus_Doxy_minusTamoxifen_R3_aligned_...  ...         3
# 3   Myc_T58A_plus_Doxy_minusTamoxifen_R1_aligned_h...  ...         1
# 4   Myc_T58A_plus_Doxy_minusTamoxifen_R2_aligned_h...  ...         2
# 5   Myc_T58A_plus_Doxy_minusTamoxifen_R3_aligned_h...  ...         3
# 6   OmoMycDNminusDoxy_minusTamoxifen_R1_aligned_hg...  ...         1
# 7   OmoMycDNminusDoxy_minusTamoxifen_R2_aligned_hg...  ...         2
# 8   OmoMycDNminusDoxy_minusTamoxifen_R3_aligned_hg...  ...         3
# 9   OmoMycDNplusDoxy_minusTamoxifen_R1_aligned_hg3...  ...         1
# 10  OmoMycDNplusDoxy_minusTamoxifen_R2_aligned_hg3...  ...         2
# 11  OmoMycDNplusDoxy_minusTamoxifen_R3_aligned_hg3...  ...         3
#
# [12 rows x 4 columns]

# Correctly filter samples for MYC activation and inhibition
myc_activation_samples = meta[meta["sample"].str.contains("
    Myc_T58A_plus_Doxy")]["sample"]
myc_inhibition_samples = meta[meta["sample"].str.contains("
    OmoMycDNplusDoxy")]["sample"]

# Extract expression data for these samples
myc_activation_aligned = aligned_expression_changes[
    myc_activation_samples]
myc_inhibition_aligned = aligned_expression_changes[
    myc_inhibition_samples]

# Recalculate differential expression
corrected_differential_expression = myc_activation_aligned.mean(axis=1) -
     myc_inhibition_aligned.mean(axis=1)
corrected_differential_expression.sort_values(ascending=False).head(10)
# Observation:
# Status: success
# result: gene
# EEF2        16585.362283
# SLC7A5      14594.361124
# FASN        13741.322551
# SOD2         9677.385687
# HSPA8        8656.023858
# NCL          7672.308054
# HSP90AA1     7126.866156
# SLC1A5       7059.368327
# PABPC1       6386.399502
# PTMA         5944.928851
# dtype: float64
```

E.6 META-REVIEW PROCESS REJECTS PAPERS WITH LOW NOVELTY OR WEAK EVALUATION

Paper: *Therapeutic Potential of Targeting the PI3K/mTOR Pathway in Kidney Renal Clear Cell Carcinoma (KIRC)*
Decision: *Reject* **Overall Score:** 0.45 **Rank:** 4/4
Justification (abridged): Incremental insights into PI3K/mTOR targeting; modest expression shifts; limited added value over an extensively studied pathway and approved agents.

Paper: *Targeting CDKN2A to Disrupt Oncogene-Induced Senescence and Apoptosis in KIRC*
Decision: *Reject* **Overall Score:** 0.40 **Rank:** 4/4
Justification (abridged): Weak survival evidence and limited mechanistic novelty; CDKN2A/9p21 status is a known prognostic marker in ccRCC, but the work does not convincingly translate this into actionable therapy.

Paper: *Therapeutic Potential of AKT2 in KIRC: Pathway and Drug Target Analysis*
Decision: *Reject* **Overall Score:** 0.40 **Rank:** 4/4

Justification (abridged): Limited novelty and weak survival correlation; evidence for AKT2 as a *specific* ccRCC driver is sparse relative to broader PI3K/AKT/mTOR activation.

Context and expert literature rationale. The PI3K/AKT/mTOR axis is long recognized in ccRCC and broadly profiled by TCGA (Network, 2013). Clinically, mTOR inhibitors (temsirolimus, everolimus) have shown activity yet modest durability, and have been surpassed in survival by modern standards such as PD-1 blockade and VEGF-targeted TKIs in advanced RCC (Battelli & Cho, 2011; Motzer et al., 2015; 2013). Consequently, papers that merely reiterate PI3K/mTOR "targetability" without new biomarkers, response predictors, or superior combinations add limited novelty. For CDKN2A, deletion at 9p21 is a well-documented adverse prognostic feature in ccRCC (El-Mokadem et al., 2014), so proposals centered on its prognostic association—without rigorous causal or translational advances—do not clear the novelty bar. Finally, while AKT pathway activation is frequent in RCC, ccRCC-specific evidence elevating *AKT2* (as distinct from AKT1/AKT3 or upstream PI3K alterations (Guo et al., 2015)) is comparatively limited and largely preclinical making an AKT2-only therapeutic thesis insufficiently substantiated. Taken together, the meta-review rejections are consistent with a mature literature where incremental analyses, weak survival signals, or narrow target rationales fall short of publication standards prioritizing novelty and robust evaluation.

F LLM USAGE

We used large language models (LLMs) to assist with improving the clarity of writing and refining the formatting of tables and figures. LLMs were not used for research ideation, experimental design, analysis, or any substantive contributions that would merit authorship.