Imitate-and-Evolve: A Multi-agent frameowrk to Bootstrap Long-Horizon Planning for Scientific Research Generation

ANONYMOUS AUTHOR(S)

TODO.

CCS Concepts: • **Computing methodologies** → **Multi-agent planning**; **Natural language generation**.

Additional Key Words and Phrases: Imitate-and-Evolve, Long-Horizon Planning, Research Paper Generation

ACM Reference Format:

Anonymous Author(s). 2018. Imitate-and-Evolve: A Multi-agent frameowrk to Bootstrap Long-Horizon Planning for Scientific Research Generation. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation email (Conference acronym 'XX)*. ACM, New York, NY, USA, 8 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Artificial Intelligence (AI) increasingly influences the landscape of scientific research [9, 23]. Automating general scientific discovery has been a long-standing ambition of the research community, dating back to the 1970s–1980s [15] with the advent of computer science, and, in AI, to early visions of automating science with AI itself [13]. Yet, for decades, day-to-day research has remained highly manual: researchers must read and synthesize overwhelming amounts of literature to formulate ideas, and then design and execute experiments to validate them [2]. Many promising ideas are thus buried under the rising costs of evidence gathering and verification.

Recent large language models (LLMs) [1] have made impressive progress in processing, organizing, and generating scientific text, enabling AI systems to assist across the full "idea generation—literature synthesis—experimental validation—manuscript writing" pipeline [23]. A growing body of work uses LLMs to propose research ideas [2, 25, 30], to assist in running or orchestrating experiments [5], or to act as AI scientists that generate open-ended scientific publications [17, 28]. Going further, end-to-end frameworks now close the loop by integrating ideation, code authoring, experiment execution, writing, and simulated peer review, making it possible to produce reviewable papers from scratch [27].

Despite these advances, fully automated research still faces a structural challenge: writing a research paper is a long-horizon process that requires multi-stage reasoning—from formulating testable hypotheses, to systematically aligning with the literature, to designing and revising feasible experiments. In practice, even expert researchers rarely "plan once and execute to the end." Instead, they proceed non-linearly—tracking frontier results, probing prototypes, and continuously re-aligning ideas with evidence. We refer to the failure mode of static, one-shot plans as *evidence drift*: scientific arguments must keep methods aligned with evidence, but static plans cannot absorb new facts surfaced by literature checks, pilot runs, or metric fluctuations. The result is a misalignment among method, evidence, and conclusion—manifesting as logical jumps between design and claims, mismatches between citations and implementations,

Manuscript submitted to ACM

or narrative statements that outrun their empirical support. Under this lens, static planning tends to enter a no-win corner: if one "reaches forward" for novelty without mid-course calibration, feasibility collapses; if one "steps back" for feasibility with early, conservative pruning, novelty collapses. These are not mere trade-offs by choice, but two facets of the same misalignment: a plan and its evolving evidence remain out of sync for too long, and the deviation becomes irrecoverable downstream.

This observation motivates a different question: can we generate expert-level papers that are both novel and feasible by treating planning as a policy constrained by evidence, rather than a fixed script? Concretely, the goal is not to "revise frequently" per se, but to convert weak, incremental signals into actionable small steps, to front-load calibration and error correction into the process, and to treat each step's inputs and outputs as state variables under control—thereby preserving soundness while pushing the frontier of innovation.

We propose an Imitation-and-Evolve approach that operationalizes this principle in both writing and revision cycles. During writing, given a research theme, we first identify its algorithmic paradigm and task lineage, then retrieve frontier papers via academic graphs and entity-centric knowledge, and construct an initial plan in the "candidate algorithm × target task" space. Instead of "plan once, execute to the end," we employ a dynamic re-planning loop: before each planned step, we retrieve evidence snippets from relevant documents to test feasibility; if the evidence is weak or conflicting, we revise that step or switch techniques; if the evidence supports, we execute the step, then parse execution logs and outputs to decide whether to re-plan subsequent steps. Thus, each step proceeds in a closed loop of "evidence retrieval—feasibility check—result review—re-plan," which suppresses evidence drift and prevents error cascades.

During revision, we again "imitate and evolve." We retrieve public reviews from papers on similar topics, align their human criteria (clarity, validity, novelty, reproducibility, etc.), and distill them into executable edit instructions. We then run a fine-grained "self-review—LLM reviewer—action" cycle: each review point is decomposed into atomic edits; the system decides to revise or to rebut based on current content; and only if the post-edit score, measured against human-aligned criteria, improves significantly do we accept the change—otherwise we roll back. This gated loop from "review signal → action plan → effect measurement" avoids blind compliance and prevents over-editing, ensuring that every change yields real quality gains.

In summary, our contributions are threefold. First, by matching and recombining "algorithm × task" under explicit evidence constraints, we expand the constructive search space and enable combinational innovation without sacrificing feasibility—resolving the aforementioned no-win corner. Second, by triggering dynamic re-planning through targeted evidence retrieval and result parsing, we decompose the long-horizon paper-generation problem into verifiable short-horizon decisions, continuously re-aligning plan, evidence, and conclusion to mitigate evidence drift. Third, by anchoring revision to human-aligned review criteria and enforcing a "revise-then-re-evaluate" gate, we convert external feedback into executable edits and guarantee measurable improvements rather than late-stage, costly overhauls. Compared to pipeline-style, fixed plans, our policy-like process better mirrors real scientific practice: it advances through uncertainty and feedback, and it evolves through alignment with evidence and standards—thereby increasing the likelihood of producing research outputs that are both genuinely novel and methodologically sound.

2 Related Work

2.1 LLMs for Research

In recent years, several studies have explored using language models for creative tasks in research, such as multi-agent collaborative writing [2] and multi-module retrieval [30] to improve research idea generation. These works aim to

boost the novelty and diversity of AI in creative tasks. Si et al. [20] conducted a comprehensive human evaluation of the task of idea generation by language models. Wang et al. [26] proposed using LLMs to automatically write survey papers. Additionally, LLMs have been used to automate the research process: Huang et al. [11] introduced a benchmark for evaluating LLMs in coding solutions for machine learning problems; Wang et al. [25] proposed a method leveraging LLMs for scientific literature retrieval. The AI Scientist project [17] introduced a fully automated, promptdriven research pipeline. The following work, AI Scientist-v2 project [28] further introduced a workshop-level automated scientific discovery via agentic tree search. More recently, Weng et al. [27] developed an iterative self-rewarding framework that enables the LLM to refine its ideas continuously, enhancing both diversity and practicality in research proposal generation. Guo et al. [10] proposed a benchmark for evaluating LLMs in research idea generation. Pu et al. [19] proposed a method for iterative research idea development through evolving and composing idea facets with literature-grounded feedback. Garikaparthi et al. [8] proposed a method for interactive research ideation system for accelerating scientific discovery.

Existing work have made progress in research paper generation. However, they rely on static planning, challenges of.

2.2 Planning of LLM Agents

LLMs have achievedremarkable success across various domains, showcasing sig-nificant intelligence in reasoning, tool usage, planning, andinstruction-following. The surprising intelligence of LLMssheds light on employing LLMs as the cognitive core ofagents, thereby offering the potential to improve planning ability [12]. For example, Plan-and-Solve [24] propose a two-step prompt instruction: “Let’s first de-vise a plan” and “Let’s carry out the plan”. This zero-shot approach has achieved improvements in mathematical rea-soning, common-sense reasoning, and symbolic reasoning. ProgPrompt [21] translates natural languagedescriptions of tasks into coding problems. It symbolizes theagent’s action space and objects in the environment throughcode, with each action formalized as a function and each ob-ject represented as a variable. Plan-Act [6] for HTML manipulation . LLMCompiler [14] break down QA tasks into parallel execution graphs.

Despite promising contributions in each direction,it is seen that LLMs still fall short in more complexscenarios, i.e., long-horizon planning [4]. There are several hierarchical planning frameworks. AgentOccam [29] incorporates planning into the action space with tree-like planning, WebPilot [31] uses six different agents, AdaPlanner [22] employs InPlan and Out-of-Plan Refiners for replanning, and ADaPT [18] uses recursive decomposition. PLAN-AND-ACT [7] provides a simpler two-agent framework with a systematic approach to generating high-quality training data for open-source LLMs.

However, lack of study on long-horizon planning for LLMs for research.

3 Methodology

In this section, we present our Imitation-and-Evolve approach in detail. The overall framework, illustrated in Fig. **??**, consists of two stages: drafting and revision. In the drafting stage, the imitation agent emulates relevant work to plan the manuscript outline, while the evolve agent iteratively drafts and refines the outline. In the revision stage, the imitation agent simulates human review to generate revision plan, and the evolve agent iteratively revises the manuscript and decides whether to adopt each point.

3.1 Imitation-and-Evolve for Drafting

Given a user request U describing a research topic, we first propose an imitation agent to imitate human expert planning. The imitation agent first prompts an LLM to analyze U and decompose it into an algorithm of interest U_A and an

application of interest U_T. Formally, we represent this as $\mu(U) = (U_A, U_T; r)$, where $\mu(.)$ is the decomposition function and r denotes any additional requirements or constraints.

Then, the imitation agent retrieves related work in terms of the algorithm of interest and the application of interest. The retrieval results are represented as $\phi(U_A) = \{L_A^{(1)}, \ldots, L_A^{(K)}\}$ and $\phi(U_T) = \{L_T^{(1)}, \ldots, L_T^{(K)}\}$, where $L_A^{(m)}$ and $L_T^{(n)}$ denote the m-th and n-th retrieved papers, respectively. The imitation agent is equipped with two tools for retrieval, i.e., a search tool and a rerank tool. The search tool receives a query and invokes HuggingFace's paper API[1] to search for candidate papers. The rerank tool uses the bge-reranker-v2-m3 [3] model to encode both the query and the titles and abstracts of the searched papers, and returns the top K most similar results. To find an optimal combination of retrieved algorithm and application, we prompt the LLM to evaluate all pairs of retrieved algorithms and applications, and identify the best-matched pair. This process derives $M_A \in L_A^{(m)}$ and $M_T \in L_T^{(n)}$, representing the matched algorithm and application papers, respectively.

The imitation agent refers to the section and subsection outlines of these selected papers as human expert plans for organizing the manuscript. We prompt the LLM to extract the sections and subsections along with their summaries from M_A and M_T, resulting in $P_A = \sigma(M_A)$ and $P_T = \sigma(M_T)$. Next, the imitation agent imitates these outlines to generate a new outline for the target manuscript, with the algorithm of interest U_A, application of interest U_T, and any constraints r as additional inputs. The derived outline P serves as the initial plan for drafting the manuscript.

Then, P is handed off to the evolve agent, which is responsible for executing P and updating it whenever discrepancies between the plan and observed results arise. The evolve agent follows an iterative paradigm, starting from step $n = 1$ and adjustment index $t = 1$, which corresponds to the initial execution before any adjustments. The plan at the t-th adjustment is denoted as $P^{(t)} = \{P_1^{(t)}, \ldots, P_N^{(t)}\}$, where N is the number of planned steps. For the plan $P_n^{(t)}$ at the n-th step after $t-1$ adjustments, the evolve agent first fetches observations relevant to $P_n^{(t)}$. The evolve agent is equipped with two tools, i.e., an external tool for fetching segments from external resources, and a contextual tool for fetching content within the existing draft. When using these tools, the evolve agent is provided with external resources, such as M_A and M_T, or the current draft as context. The LLM is prompted to identify the appropriate line ranges, and the content within these ranges is extracted and concatenated to facilitate efficient evidence gathering. The content retrieved from context for step n is denoted as C_n, while the content retrieved from external resources is denoted as E_n. Using these observations, the evolve agent updates the plan according to

$$P_i^{(t+1)} = \pi(P_i^{(t)} \mid C^{(t)}, E^{(t)}, P_i^{(t)}) \tag{1}$$

where $\pi(.)$ is a policy function that revises each step i of the current plan based on the available context $C^{(t)}$, external evidence $E^{(t)}$, and the i-th step of the current plan $P_i^{(t)}$. The policy function is implemented by prompting the LLM to update the plan according to the observations. Then, the evolve agent

The revised step $P_i^{(t+1)}$ is then executed to produce the draft D_i for the i-th section, using the contextual segments C_i and external segments E_i as context, and $P_i^{(t+1)}$ as the instruction. After deriving the draft D_i, the imitation agent further updates the remaining steps in the plan. The subsequent plan steps are modified as needed to be appropriate given the new results:

[1]https://huggingface.co/docs/hub/api

$$P_j^{(t+2)} = \begin{cases} P_j^{(t)}, & \text{if } j < i \\ P_j^{(t+1)}, & \text{if } j = i \\ \pi\left(P_j^{(t)} \mid \{D_p^{(t)}\}_{p \le i}, \{P_q^{(t)}\}_{q<j}\right), & \text{if } j > i \end{cases} \tag{2}$$

where $P_j^{(t+2)}$ denotes the j-th step in the updated plan after considering the collection of drafts $\{D_p^{(t)}\}_{p \le i}$ already produced and the collection of plan steps $\{P_q^{(t)}\}_{q<j}$ as context. In this paper, the LLM is prompted to revise the subsequent plan steps and return a revised version if the current steps are not appropriate; otherwise, the original steps are retained.

Next, the evolve agent moves to the next step in the plan and starts a new round of iteration. This iterative process continues until all N steps are completed. Finally, the evolve agent collects the section titles according to the last version of the plan and combines them with all the drafts $\{D_1, D_2, \ldots, D_N\}$ to form the main body of the manuscript. The evolve agent then prompts the LLM to generate the title and abstract based on the completed body as context. Supplemented with references, all the content is rendered into a LaTeX project according to the conference template, and the final PDF of the paper is generated.

3.2 Imitation-and-Evolve for Revision

After deriving draft sections $\{D_1, D_2, \ldots, D_{N_D}\}$, we design the revision stage, which aims to further improve the manuscript. To plan systematic revisions, we introduce an imitation agent that mimics the peer-review and revision process commonly practiced by human experts in scientific research.

We first design the imitation agent to generate review feedback and decompose it into actionable points, which then serve as the revision plan. Following the standard peer-review protocols adopted by leading conferences that make their review comments public on OpenReview[2] (e.g., ICLR), our imitation agent prompts the LLM to generate comprehensive review feedback across multiple dimensions. Formally, given the complete manuscript $\{D_1, D_2, \ldots, D_{N_D}\}$, the imitation agent applies a feedback generation function δ such that $F = \delta(\{D_1, D_2, \ldots, D_{N_D}\})$. The generated feedback F covers aspects such as summary, soundness, presentation, contribution, strengths, weaknesses, questions, and overall rating. Next, we decompose F into a set of review points $\{G_1, G_2, \ldots, G_L\} = \beta(F)$, where β is a decomposition function that prompts the LLM to enumerate feedback items from F. Each G_l represents a comment that can be addressed independently, forming the initial revision plan.

These review points $\{G_1, G_2, \ldots, G_L\}$ are then handed off to the evolve agent, which is responsible for executing and updating the revision plan in response to observed outcomes. The evolve agent operates in an iterative paradigm, starting from review point $l = 1$ and adjustment index $t = 1$, corresponding to the initial execution before any adjustments. The revision plan at the t-th adjustment is denoted as $\{G_1^{(t)}, \ldots, G_L^{(t)}\}$, where L is the total number of review points. For a given review point $G_l^{(t)}$ at step l after $t-1$ adjustments, the evolve agent first gathers relevant observations. Similar to imitation-and-evolve for drafting, it is equipped with two tools: an external tool for retrieving evidence from external resources, and a contextual tool for extracting content from the current manuscript draft. The content extracted from the current draft for step l is denoted as C_l, while content from external resources is denoted as E_l.

Not all review feedback should be followed unconditionally, as some points may arise from misunderstandings. Therefore, for each plan G_l, the evolve agent determines whether to revise or rebut using a decision function

[2] https://openreview.net/

$C(G_l, C_l, E_l) \in \{0, 1\}$, where 0 indicates rebuttal and 1 indicates revision. The update to the plan before execution can be formalized as:

$$G_l^{(t+1)} = \begin{cases} \varnothing, & \text{if } C(G_l, C_l, E_l) = 0 \\ G_l^{(t)}, & \text{if } C(G_l, C_l, E_l) = 1 \end{cases} \tag{3}$$

where $\varnothing$ denotes an empty plan that does not need to be executed, indicating that the review point is skipped if the decision is to rebut. If the decision is to rebut, the agent skips to the next point. Otherwise, the agent proceeds to modify the relevant section identified by C_l. The revised section is denoted as $\hat{D}_i = \eta(D_i, G_l)$, where η is a modification function that incorporates the feedback G_l into the original section draft D_i.

After executing the plan, we evaluate the revised draft to assess its quality. Here, we prompt the LLM to generate feedback on both versions, putting the revised section $\hat{D}_i$ and the original D_i back into the manuscript, respectively. Then we use $\delta()$ to generate feedback on the two versions of the manuscript and extract a quality score, with the derived score of the corresponding version denoted as $\gamma(\cdot)$. The acceptance or rollback of the update can be formalized as:

$$D_i^{(t+1)} = \begin{cases} \hat{D}_i, & \text{if } \gamma(\hat{D}_i) \geq \gamma(D_i) \\ D_i, & \text{if } \gamma(\hat{D}_i) < \gamma(D_i) \end{cases} \tag{4}$$

where the revision is accepted only if $\gamma(\hat{D}_i) \geq \gamma(D_i)$, indicating a genuine improvement. If the quality does not improve, we roll back to the previous version, keeping D_i unchanged.

This iterative process continues for all review points $\{G_1, G_2, \ldots, G_L\}$, resulting in a revised draft. The entire revision cycle can be repeated with new rounds of review feedback and revisions.

4 Experiments

4.1 Experimental Settings

Following the recenttrends in using LLMs to judge the quality of out-put texts (especially in the setting of reference-freeevaluations) [16, 32], we use GPT-4 to judge the qual-ity of research ideas. Note that each of the problem,method, and experiment design is evaluated withfive different criteria. We ask the LLM-based evaluation model toeither rate the generated idea on a 5-point Likertscale for each criterion or perform pairwise compar-isons between two ideas from different models.

We compare AutoSurvey with surveys authored by human experts (collected from Arxiv) and naive RAG-based LLMs across 20 different computer science topics across 20 different topics in the field of LLMs (see Table 6). For the naive RAG-based LLMs, we begin with a title and a survey length requirement, then iteratively prompt the model to write the content until completion. Note that we also provide the model with the same number of reference papers with AutoSurvey.

We mainly use the GPT-4 [1] releasefrom Nov 06, 2023, as the basis for all models,which is, notably, reported to be trained with dataup to Apr 2023 (meanwhile, the papers used foridea generation appear after May 2023).

4.2 Comparative Experiments

We extensively evaluate The AI Scientist on three templates (as described in Section 3) acrossdifferent publicly available LLMs: Claude Sonnet 3.5 (Anthropic, 2024), GPT-4o (OpenAI, 2023), DeepSeek Coder (Zhu et al., 2024), and Llama-3.1

405b (Llama Team, 2024). The first two modelsare only available by a public API, whilst the second two models are open-weight.

5 Conclusion

We have proposed a new method to solve the challenges in AI research paper generation.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *ArXiv Preprint* (2023).

[2] Jinheon Baek, Sujay Kumar Jauhar, Silviu Cucerzan, and Sung Ju Hwang. 2025. ResearchAgent: Iterative Research Idea Generation over Scientific Literature with Large Language Models. In *Nations of the Americas Chapter of the Association for Computational Linguistics*. 6709–6738.

[3] Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *ArXiv Preprint* (2024).

[4] Yanan Chen, Ali Pesaranghader, Tanmana Sadhu, and Dong Hoon Yi. 2024. Can We Rely on LLM Agents to Draft Long-Horizon Plans? Let's Take TravelPlanner as an Example. *ArXiv Preprint* (2024).

[5] Jiangshu Du, Yibo Wang, Wenting Zhao, Zhongfen Deng, Shuaiqi Liu, Renze Lou, Henry Zou, Pranav Narayanan Venkit, Nan Zhang, Mukund Srinath, et al. 2024. LLMs Assist NLP Researchers: Critique Paper (Meta-) Reviewing. In *Empirical Methods in Natural Language Processing*. 5081–5099.

[6] Lutfi Eren Erdogan, Hiroki Furuta, Sehoon Kim, Nicholas Lee, Suhong Moon, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. 2025. Plan-and-Act: Improving Planning of Agents for Long-Horizon Tasks. In *International Conference on Machine Learning*.

[7] Lutfi Eren Erdogan, Hiroki Furuta, Sehoon Kim, Nicholas Lee, Suhong Moon, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. 2025. Plan-and-Act: Improving Planning of Agents for Long-Horizon Tasks. In *International Conference on Machine Learning*.

[8] Aniketh Garikaparthi, Manasi Patwardhan, Lovekesh Vig, and Arman Cohan. 2025. Iris: Interactive research ideation system for accelerating scientific discovery. In *Annual Meeting of the Association for Computational Linguistics*.

[9] Yolanda Gil, Mark Greaves, James Hendler, and Haym Hirsh. 2014. Amplify scientific discovery with artificial intelligence. *Science* 346, 6206 (2014), 171–172.

[10] Sikun Guo, Amir Hassan Shariatmadari, Guangzhi Xiong, Albert Huang, Myles Kim, Corey M Williams, Stefan Bekiranov, and Aidong Zhang. 2025. Ideabench: Benchmarking large language models for research idea generation. In *Knowledge Discovery and Data Mining*. 5888–5899.

[11] Qian Huang, Jian Vora, Percy Liang, and Jure Leskovec. 2024. MLAgentBench: evaluating language agents on machine learning experimentation. In *International Conference on Machine Learning*. 20271–20309.

[12] Xu Huang, Weiwen Liu, Xiaolong Chen, Xingmei Wang, Hao Wang, Defu Lian, Yasheng Wang, Ruiming Tang, and Enhong Chen. 2024. Understanding the planning of LLM agents: A survey. *ArXiv Preprint* (2024).

[13] Marcus Hutter. 2001. Towards a universal theory of artificial intelligence based on algorithmic probability and sequential decisions. In *European conference on machine learning*. 226–238.

[14] Sehoon Kim, Suhong Moon, Ryan Tabrizi, Nicholas Lee, Michael W Mahoney, Kurt Keutzer, and Amir Gholami. 2024. An llm compiler for parallel function calling. In *International Conference on Machine Learning*.

[15] Pat Langley. 1987. *Scientific discovery: Computational explorations of the creative processes*.

[16] Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-Eval: NLG Evaluation using Gpt-4 with Better Human Alignment. In *Empirical Methods in Natural Language Processing*. 2511–2522.

[17] Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. 2024. The ai scientist: Towards fully automated open-ended scientific discovery. *ArXiv Preprint* (2024).

[18] Archiki Prasad, Alexander Koller, Mareike Hartmann, Peter Clark, Ashish Sabharwal, Mohit Bansal, and Tushar Khot. 2024. ADaPT: As-Needed Decomposition and Planning with Language Models. In *Findings of the Association for Computational Linguistics*. 4226–4252.

[19] Kevin Pu, KJ Kevin Feng, Tovi Grossman, Tom Hope, Bhavana Dalvi Mishra, Matt Latzke, Jonathan Bragg, Joseph Chee Chang, and Pao Siangliulue. 2025. Ideasynth: Iterative research idea development through evolving and composing idea facets with literature-grounded feedback. In *Conference on Human Factors in Computing Systems*. 1–31.

[20] Chenglei Si, Diyi Yang, and Tatsunori Hashimoto. 2024. Can LLMs Generate Novel Research Ideas? A Large-Scale Human Study with 100+ NLP Researchers. In *International Conference on Learning Representations*.

[21] Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. 2023. ProgPrompt: Generating Situated Robot Task Plans using Large Language Models. In *International Conference on Robotics and Automation*. 11523–11530.

[22] Haotian Sun, Yuchen Zhuang, Lingkai Kong, Bo Dai, and Chao Zhang. 2023. Adaplanner: Adaptive planning from feedback with language models. *Advances in Neural Information Processing Systems* 36 (2023), 58202–58245.

[23] Hanchen Wang, Tianfan Fu, Yuanqi Du, Wenhao Gao, Kexin Huang, Ziming Liu, Payal Chandak, Shengchao Liu, Peter Van Katwyk, Andreea Deac, et al. 2023. Scientific discovery in the age of artificial intelligence. *Nature* 620, 7972 (2023), 47–60.

[24] Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. 2023. Plan-and-Solve Prompting: Improving Zero-Shot Chain-of-Thought Reasoning by Large Language Models. In *Annual Meeting of the Association for Computational Linguistics*. 2609–2634.

[25] Qingyun Wang, Doug Downey, Heng Ji, and Tom Hope. 2023. Scimon: Scientific inspiration machines optimized for novelty. In *Annual Meeting of the Association for Computational Linguistics*. 279–299.

[26] Yidong Wang, Qi Guo, Wenjin Yao, Hongbo Zhang, Xin Zhang, Zhen Wu, Meishan Zhang, Xinyu Dai, Qingsong Wen, Wei Ye, et al. 2024. Autosurvey: Large language models can automatically write surveys. *Advances in Neural Information Processing Systems* 37 (2024), 115119–115145.

[27] Yixuan Weng, Minjun Zhu, Guangsheng Bao, Hongbo Zhang, Jindong Wang, Yue Zhang, and Linyi Yang. 2025. CycleResearcher: Improving Automated Research via Automated Review. In *International Conference on Learning Representations*.

[28] Yutaro Yamada, Robert Tjarko Lange, Cong Lu, Shengran Hu, Chris Lu, Jakob Foerster, Jeff Clune, and David Ha. 2025. The ai scientist-v2: Workshop-level automated scientific discovery via agentic tree search. *ArXiv Preprint* (2025).

[29] Ke Yang, Yao Liu, Sapana Chaudhary, Rasool Fakoor, Pratik Chaudhari, George Karypis, and Huzefa Rangwala. 2024. AgentOccam: A Simple Yet Strong Baseline for LLM-Based Web Agents. In *International Conference on Learning Representations*.

[30] Zonglin Yang, Xinya Du, Junxian Li, Jie Zheng, Soujanya Poria, and Erik Cambria. 2024. Large Language Models for Automated Open-domain Scientific Hypotheses Discovery. In *Findings of the Association for Computational Linguistics*. 13545–13565.

[31] Yao Zhang, Zijian Ma, Yunpu Ma, Zhen Han, Yu Wu, and Volker Tresp. 2025. Webpilot: A versatile and autonomous multi-agent system for web task execution with strategic exploration. In *AAAI Conference on Artificial Intelligence*, Vol. 39. 23378–23386.

[32] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems* 36 (2023), 46595–46623.