ADAPT FOR SCIENCE: A MULTI-AGENT SYSTEM FOR SELF-EVOLVING SCIENTIFIC OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Scientific discovery often requires identifying a small number of high-performing designs in vast, constrained search spaces. This is difficult when evaluations are costly, feasible solutions are sparse, parameters are strongly coupled, and physical constraints are strict. Recent AI for AI methods can automatically improve algorithms and optimization procedures. However, general purpose approaches are not designed around scientific priors, physical constraints, and limited evaluation budgets. We introduce **ADAPT for Science**, a multi-agent framework that treats scientific optimization as two coupled decisions: *where to search* and *how to search*. ADAPT uses historical designs to identify promising and diverse initial conditions, progressively concentrates computation on favorable search directions, and evolves the optimization procedure from execution feedback. We evaluate ADAPT on the design of gravitational wave detectors using a differentiable physics simulator and eight unseen detector topologies. Under matched scientific evaluation budgets, ADAPT achieves a mean loss of 0.204 ± 0.101, reducing loss by 54.9% relative to the best classical optimizer and by 62.2% relative to the best general purpose agentic evolution baseline. ADAPT achieves the lowest loss on all unseen topologies among the evaluated baselines. ADAPT provides a step toward scientific optimization systems that continuously learn from prior experience and their own search outcomes.

1 INTRODUCTION

Scientific progress often depends on finding a small number of useful solutions in vast and constrained design spaces. From drug discovery to gravitational-wave detector design, each candidate can be costly to evaluate, many parameters interact strongly, and feasible solutions may occupy only a small fraction of the search space. The central challenge is therefore not only to evaluate a candidate, but to decide what is worth trying next. This challenge becomes acute when optimization must proceed under a limited experimental or computational budget. Search performance depends not only on local parameter updates, but also on where the search begins, which prior experience is reused, which optimization procedure is applied, and how resources are allocated. These higher-level decisions are often fixed in advance or made manually, despite their substantial influence on the final solution. Gravitational-wave detector design provides a representative example. Detector sensitivity depends on many interacting continuous parameters subject to strict physical constraints. A poor initialization or an unsuitable optimization strategy can consume substantial computation before reaching a useful solution. At the same time, previous detector designs provide valuable experience that can guide future search. The problem is therefore not optimization from scratch, but how to reuse prior scientific experience, choose where to begin, and adapt the search strategy as new evidence becomes available.

AI has begun to automate parts of this process. Bayesian optimization and autonomous laboratories use previous evaluations to select promising candidates (Shahriari et al., 2015; Szymanski et al., 2023), while scientific agents automate planning and experimental workflows (Boiko et al., 2023). These approaches primarily adapt the scientific candidate while the higher-level search procedure remains largely specified in advance. In parallel, recent program-search methods show that language models can improve algorithms through evaluator feedback (Romera-Paredes et al., 2024; Shojaee et al., 2025). These methods adapt the procedure itself, but do not explicitly model reusable scien-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

tific experience, target-dependent initialization, physical feasibility, and limited scientific evaluation budgets.

We address this gap with **ADAPT for Science**, a multi-agent framework for self-evolving scientific optimization. ADAPT treats the complete scientific search policy as the object of optimization and decomposes it into two coupled decisions: *where to search* and *how to search*. Historical scientific designs provide offline experience for selecting promising and diverse starting points and allocating evaluation effort. Online execution feedback is then used to evolve the optimization procedure itself. The resulting closed loop jointly adapts initialization, resource allocation, and optimization strategy to the scientific target. We evaluate ADAPT on gravitational-wave detector design using a differentiable physics simulator and eight unseen detector topologies. Under matched scientific evaluation budgets, ADAPT achieves a mean loss of 0.204 ± 0.101, compared with 0.452 ± 0.150 for the strongest classical optimizer and 0.540 ± 0.263 for the strongest general-purpose agentic evolution baseline. These results correspond to reductions in mean loss of **54.9%** and **62.2%**, respectively. ADAPT also achieves the lowest loss on all eight held-out topologies among the evaluated baselines. The results indicate that scientific optimization benefits from adapting the search policy itself rather than optimizing scientific parameters or optimization programs in isolation.

Our contributions are threefold:

- We formulate scientific optimization at the level of the complete search policy rather than the scientific parameters alone, and decompose this policy into two coupled decisions: *where to search* and *how to search*.
- We develop **ADAPT for Science**, which combines reusable scientific experience with online execution feedback to jointly adapt initialization, resource allocation, and optimization procedures under physical and evaluation-budget constraints.
- We establish a held-out evaluation on eight unseen gravitational-wave detector topologies. ADAPT reduces mean loss by **54.9%** relative to the strongest classical optimizer and by **62.2%** relative to the strongest general-purpose agentic evolution baseline under matched scientific evaluation budgets, while achieving the lowest loss on all eight topologies among the evaluated baselines.

2 Related Work

Scientific optimization is often formulated as sequential decision-making under expensive evaluations and limited budgets. Bayesian optimization and active learning leverage previous observations to select informative evaluations (Shahriari et al., 2015; Ren et al., 2023; Zhang et al., 2023; Jones et al., 1998; Snoek et al., 2012; Gardner et al., 2014; Eriksson et al., 2019). These methods have been widely applied to scientific discovery, including chemical reaction optimization and autonomous experimentation (Hase et al., 2018; Shields et al., 2021; Granda et al., 2018; Coley et al., 2019; Burger et al., 2020; MacLeod et al., 2020). Recent scientific foundation models and autonomous agents further exploit historical data, planning, and tool use for complex scientific workflows (Merchant et al., 2023; Wang et al., 2023; Boiko et al., 2023; M. Bran et al., 2024; Chen et al., 2026). However, initialization, optimizer choice, and resource allocation are usually specified beforehand.

A complementary direction treats the optimization algorithm itself as the search object. Learned optimizers and symbolic program search demonstrate that optimization rules can be discovered from experience (Andrychowicz et al., 2016; Chen et al., 2023; Feng et al., 2026; Zhou et al., 2026). Algorithm discovery has further achieved improved computational procedures for tasks such as matrix multiplication and sorting (Fawzi et al., 2022; Mankowitz et al., 2023). More recently, large language models enable program and heuristic evolution through evaluator feedback, including FunSearch, EoH, ReEvo, and MCTS-AHD (Romera-Paredes et al., 2024; Liu et al., 2024; Ye et al., 2024; Zheng et al., 2025). Related systems extend this idea to scientific equations, computational workflows, and automated optimizer evolution (Shojaee et al., 2025; Cheng et al., 2024; Agrawal et al., 2026; Lange et al., 2026; Shi et al., 2026; Huang et al., 2026; Sharma, 2025). These approaches show that algorithms themselves can be optimized, but usually without target-specific scientific priors.

ADAPT connects these directions by treating the complete scientific search policy as the object of adaptation. Historical designs provide priors for *where to search*, while execution feedback evolves

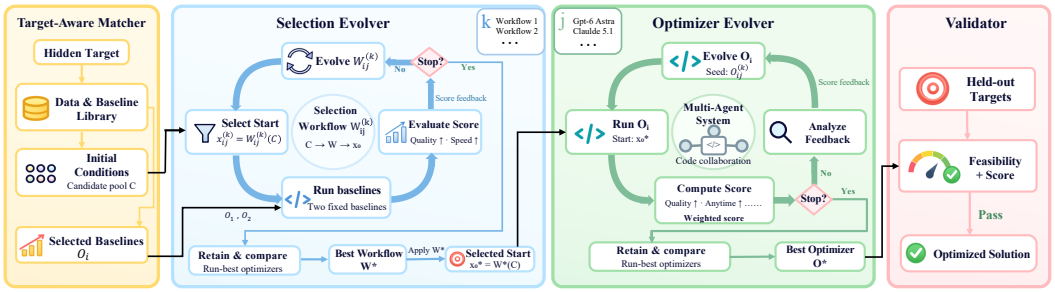


Figure 1: **Overview of ADAPT for Science.** A target-aware matcher retrieves historical designs and reference optimizers to form a target-specific search prior. The selection evolver determines *where to search*, while the optimizer evolver determines *how to search* from execution feedback. Together, they define a target-specific search policy that is evaluated on held-out targets under the original scientific objective and physical constraints.

how to search. By jointly adapting initialization, resource allocation, and optimization dynamics, ADAPT enables efficient scientific optimization under physical constraints and limited budgets.

3 ADAPT FOR SCIENCE

Figure 1 summarizes ADAPT. Rather than optimizing only scientific variables, ADAPT optimizes the complete search policy through two coupled decisions, where to search and how to search. Historical experience provides a target-specific prior, online execution feedback adapts the optimization procedure, and an independent validator evaluates the resulting policy on unseen targets.

3.1 Problem Formulation

For a scientific target $\tau \in \mathcal{T}$, let $x \in \mathcal{X}_\tau$ denote the design variables, $L_\tau(x)$ the objective, and $\mathcal{F}_\tau \subseteq \mathcal{X}_\tau$ the feasible region. Under a finite evaluation budget, performance depends on both the initialization and the optimization procedure. We therefore define the search policy as $\pi = (W, O)$, where W selects the initial design x_0 and O generates the trajectory $\mathcal{T}_\pi = \{x_0, \ldots, x_T\}$. For budget B, we define $J_\tau(\pi; B) = \min_{x_t \in \mathcal{T}_\pi \cap \mathcal{F}_\tau,\, c_t \le B} L_\tau(x_t)$, where c_t is the cumulative evaluation cost.

Conventional optimization fixes W and O in advance, whereas ADAPT adapts both to each scientific target,

$$(W_\tau^*, O_\tau^*) = \arg \min_{W \in \mathcal{W},\, O \in \mathcal{O}} J_\tau((W, O); B), \tag{1}$$

where $\mathcal{W}$ and $\mathcal{O}$ denote the spaces of selection policies and optimization procedures. W_τ^* determines where to search, while O_τ^* determines how to search.

3.2 Historical Scientific Experience

We construct a reusable experience library from historical gravitational-wave detector designs and optimization records. After cleaning, representation standardization, and transfer compatibility filtering, the design library contains 15,409 valid topology–design records, each pairing one topology with a compatible parameterized design and its evaluated performance. A separate baseline library contains ten established optimization procedures and their historical behavior. The former supplies candidate initializations, while the latter provides reference search behavior for policy evolution. Held-out targets and their outcomes are excluded from library construction and all subsequent strategy development.

3.3 Target-Aware Retrieval and Initialization

For each target topology, ADAPT retrieves $K = 64$ distinct historical sources using a fixed structure-aware distance. A size-3 topology contains nine interior symbols $\mathbf{c}$ and twelve boundary symbols

b. For target T and source S,

$$d(T,S) = \frac{1}{9}\sum_{i=1}^{9}\delta(c_i^T, c_i^S) + \frac{1}{12}\sum_{j=1}^{12}\mathbf{1}\left[b_j^T \neq b_j^S\right], \tag{2}$$

where $\delta(a,b) = 0$ for identical symbols, 0.25 for different orientations within the same component family, and 1 for different families. Symbols A–D denote beamsplitters and E–H directional beamsplitters, with four orientations per family. Symbols are compared at fixed positions without rotational or reflection alignment.

Sources with the same complete topology as the target are excluded. The remainder are ranked by increasing $d(T,S)$, with lexicographic ordering used for ties. Incompatible records are skipped and the first 64 eligible sources are retained, without dataset-specific quotas.

Parameters are transferred by exact semantic matching of component names, types, parameter properties, and complete coupling groups. Matched bounded values are copied directly. Unmatched parameters use frozen training-set medians in the order component-type and property, property only, and global prior, followed by mapping to target bounds. For parameter $x_i^{(k)}$ with bounds ℓ_i and h_i, $u_i^{(k)} = \mathrm{clip}((x_i^{(k)} - \ell_i)/(h_i - \ell_i), \epsilon, 1-\epsilon)$ and $z_i^{(k)} = \log(u_i^{(k)}/(1-u_i^{(k)}))$, with $\epsilon = 10^{-9}$. The resulting $C_\tau = \{\mathbf{z}^{(k)}\}_{k=1}^{64}$ forms the candidate initialization pool.

3.4 Selection Evolution

Given C_τ, the Selection Evolver searches for a policy W_τ that returns one initialization under a fixed scientific evaluation budget. Evolution starts from a direct 64-to-1 baseline that evaluates all 64 candidates and selects the best single initialization. Intermediate candidate counts, screening stages, pruning rules, and budget-allocation strategies are not predefined and are left to evolution.

A candidate policy may use historical attributes, transferred parameters, target evaluations, feasibility, constraint violations, and observed optimization progress to determine which candidates are evaluated, retained, or discarded and how budget is allocated among them. ADAPT evolves these policies with GEPA, OpenEvolve, ShinkaEvolve, and ASI using GPT-6 Astra and Claude-fable-5.1. Each workflow–model pair is compared with two reference baselines, yielding 16 configurations.

At iteration k, a policy $W^{(k)}$ is executed under the fixed budget, its trace is converted into feedback $F_W^{(k)}$, and the feedback is used to propose $W^{(k+1)}$. Strong previous policies are retained for direct comparison. Selection is based on measured scientific performance, and the final policy produces $x_{0,\tau}^* = W_\tau^*(C_\tau)$.

3.5 Optimizer Evolution

Starting from $x_{0,\tau}^*$, the Optimizer Evolver searches over executable optimization programs evaluated directly by the differentiable physics simulator under the same budget and numerical constraints. The search space includes update rules, gradient and state handling, learning-rate and momentum schedules, trajectory management, numerical safeguards, and allocation of the remaining evaluation budget.

The same workflow–model pool proposes and revises optimizers. At iteration k, $O^{(k)}$ is executed, its trace is converted into feedback $F_O^{(k)}$, and the feedback is used to generate $O^{(k+1)}$. Strong previous optimizers remain available for direct comparison. Candidate procedures are ranked by

$$S = w_q S_{\text{quality}} + w_a S_{\text{anytime}} + w_f S_{\text{feasibility}} + w_i S_{\text{improvement}}, \tag{3}$$

where the four terms measure final feasible quality, performance throughout execution, physical validity, and improvement over the initialization. This score is used only for strategy evolution. Reported results use the native scientific objective and feasibility criteria. The selected optimizer O_τ^* and selection policy W_τ^* define $\pi_\tau^* = (W_\tau^*, O_\tau^*)$.

3.6 Independent Validation

The final policy π_τ^* is evaluated on detector topologies absent from the experience library and excluded from retrieval, baseline selection, policy evolution, and model selection. Their outcomes are

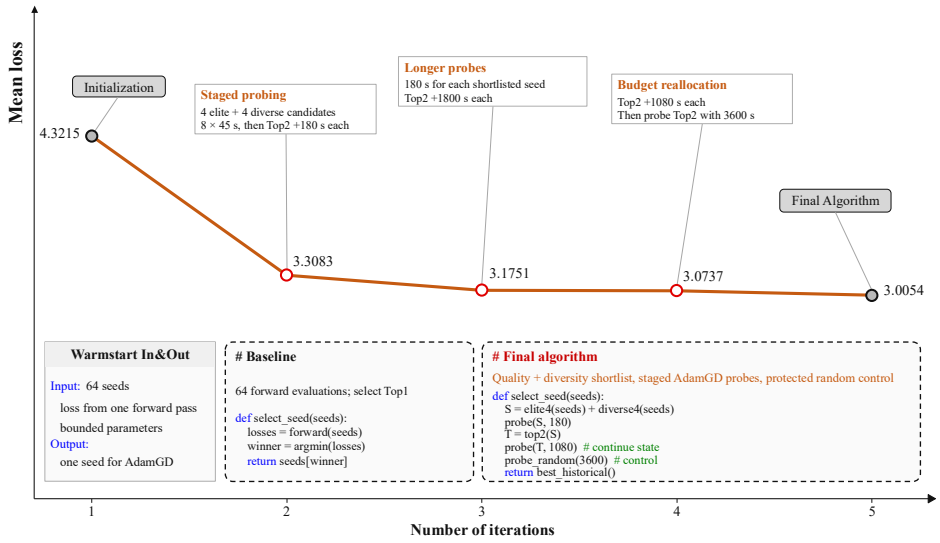


Figure 2: **Evolution of the selection policy.** Mean loss across successive selection-policy iterations. Starting from direct selection after 64 forward evaluations, evolution introduces quality–diversity shortlisting, staged probing, and progressive budget reallocation. The final policy reaches a mean loss of 3.0054.

never returned to either evolver. Each held-out target is evaluated under the same scientific budget as the comparison methods, and all reported results use the native objective, physical feasibility, and anytime optimization performance.

4 Results

We first examine the policies discovered by the two evolution stages and then evaluate their combined effect on eight held-out gravitational-wave detector topologies. All comparisons use matched random seeds, hardware, numerical precision, and the same total wall-clock budget. For ADAPT, this budget covers both warmup (candidate screening and staged probing) and subsequent optimization; no additional time is allocated for warmup.

4.1 Selection evolution discovers progressive search

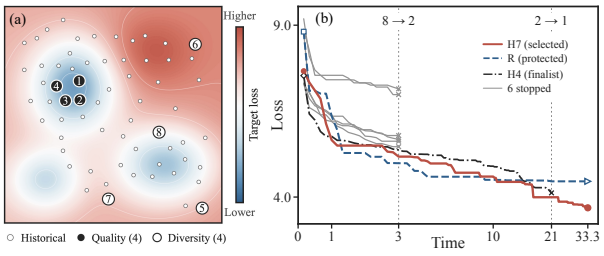


Figure 3: **Selection evolution discovers progressive search.** Starting from a 64-to-1 objective, the evolved policy selects four high-quality feasible candidates and four diversity-preserving candidates, then progressively concentrates evaluation as $8 \to 2 \to 1$ while retaining protected exploration.

The Selection Evolver is given 64 transferred initializations and a single objective, to return one start under a fixed evaluation budget. No intermediate candidate count, pruning schedule, or budget-allocation rule is specified. Figure 2 shows how the policy develops from direct Top-1 selection into a staged search procedure. Successive iterations introduce quality–diversity shortlisting, longer target-specific probes, and progressive budget reallocation, reducing the mean loss to 3.0054. The resulting policy converges to the $64 \to 8 \to 2 \to 1$ workflow shown in Figure 3.

The first stage evaluates all 64 starts once on the target without optimizer updates and records loss, feasibility, and constraint violation. Among feasible candidates, the four lowest-loss starts form an elite set. Four additional starts are then selected for coverage in transferred parameter space. For

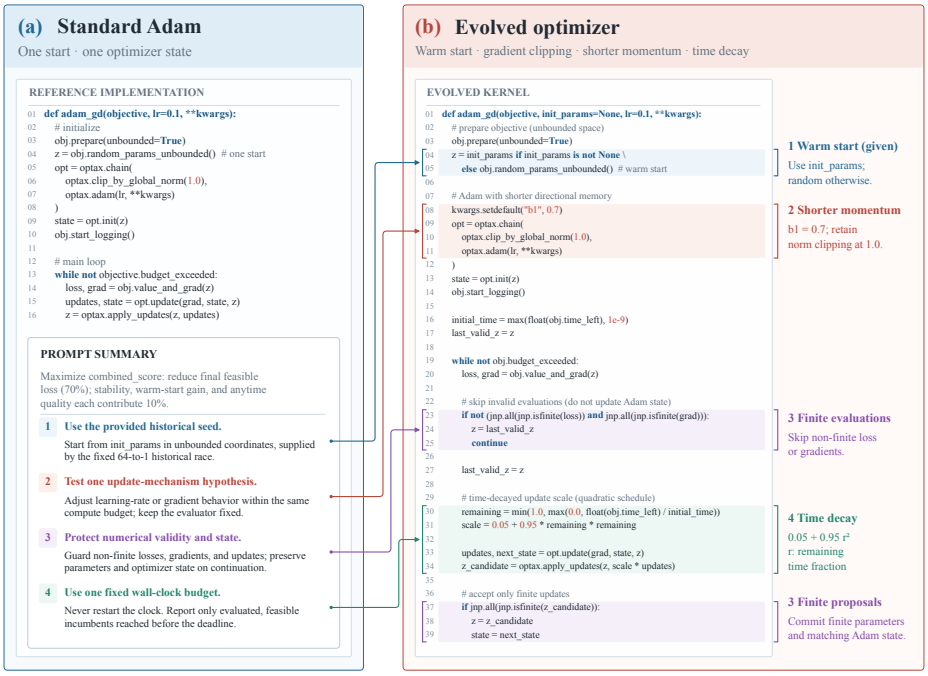


Figure 4: **Optimizer evolution discovers budget-aware search dynamics.** The evolved procedure retains the Adam structure while using the selected initialization, shorter first-moment memory, time-dependent updates, and explicit rejection of invalid evaluations and proposals under a fixed scientific budget.

candidate k, each bounded parameter is normalized as $q_{k,i} = (x_{k,i} - \ell_i)/(h_i - \ell_i)$, and pairwise distance is $D(k,j) = \sqrt{d^{-1}\sum_i (q_{k,i} - q_{j,i})^2}$. Starting from the four elites, each additional candidate is chosen by $k^\star = \arg\max_{k \in \mathcal{F}\setminus S} \min_{j \in S} D(k,j)$ until eight starts are retained. Ties are resolved by candidate index. If fewer than eight feasible starts exist, all feasible starts are retained and numerically valid infeasible candidates are added in order of constraint violation, loss, and candidate index. Fewer than eight numerically valid candidates terminates the selection run.

The eight retained starts then enter a target-specific race. Each receives a 3-minute optimization run, after which two leading trajectories are retained and continued to 21 minutes. A protected exploratory branch is maintained during pruning, and the remaining budget is concentrated on the final start. The discovered policy therefore moves from historical quality and parameter coverage to target-specific execution evidence before committing the remaining budget to a single direction.

4.2 Optimizer evolution discovers budget-aware search dynamics

The preceding selection stage supplies the initialization, while the Optimizer Evolver determines how the subsequent local search proceeds. Figure 5 traces this process from the reference AdamGD implementation to the final optimization procedure. Successive iterations test finite-state protection, time-dependent update scaling, stopping behavior, and shorter optimizer memory. These changes are retained or revised according to measured execution performance, ultimately yielding the optimizer shown in Figure 4.

Figure 4 shows the resulting optimizer selected by ADAPT. Rather than replacing Adam, evolution retains its basic structure and modifies its behavior under the finite evaluation horizon.

The discovered procedure shortens the first-moment memory to $\beta_1 = 0.7$ while retaining global gradient clipping. It further scales each update by $s(r) = 0.05 + 0.95r^2$, where r is the remaining fraction of the evaluation budget, producing larger updates early and increasingly conservative updates near the deadline. Non-finite losses or gradients are skipped, and non-finite parameter proposals are rejected together with the corresponding optimizer-state update.

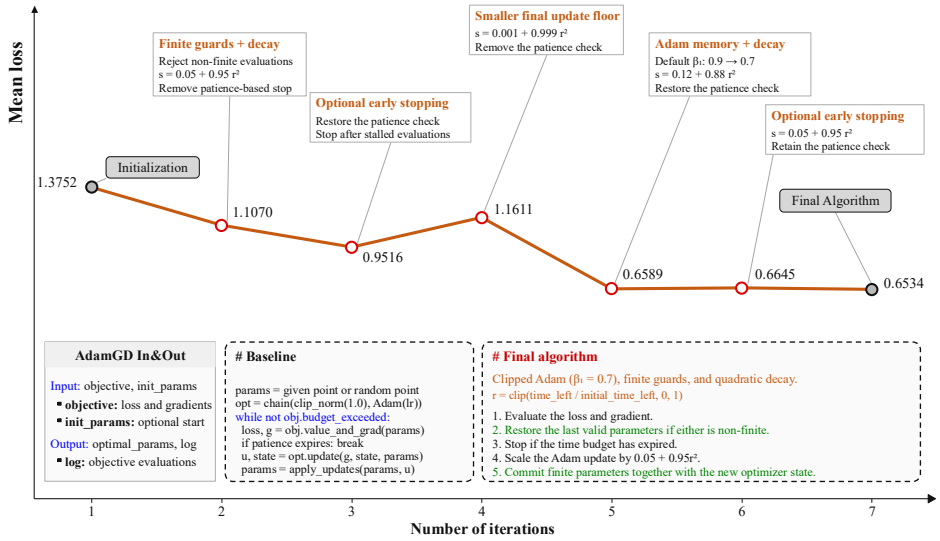


Figure 5: **Evolution of the optimization procedure.** Mean loss across successive optimizer-evolution iterations, with annotations summarizing the changes tested at each stage. Starting from the reference AdamGD implementation, evolution tests numerical safeguards, stopping behavior, update scaling, and optimizer memory. The final procedure reaches a mean loss of 0.6534 during strategy evolution.

These changes couple the local search to both the selected historical start and the remaining evaluation horizon. The resulting optimizer remains close to Adam computationally, but adapts its directional memory, update magnitude, and numerical state handling to the observed scientific search process.

4.3 Overall Performance on Held-Out Topologies

We compare the complete ADAPT policy with fixed classical optimizers and general-purpose agentic evolution methods on eight held-out detector topologies. Lower loss indicates better performance.

Among fixed optimizers, NAAdam gives the lowest mean loss at 0.452 ± 0.150, whereas ADAPT reaches $\mathbf{0.204 \pm 0.101}$, a **54.9% reduction**. ADAPT also achieves the lowest loss on all eight held-out topologies among the evaluated baselines, showing that the mean improvement is not driven by a small subset of targets.

The same advantage persists against agentic evolution baselines. GEPA gives the strongest mean result among these methods at 0.540 ± 0.263, compared with 0.204 ± 0.101 for ADAPT, corresponding to a **62.2% reduction**. Thus, program evolution alone does not match the full system under the present protocol. Combining target-specific historical initialization with evolved selection and optimization policies yields the stronger held-out performance.

4.4 Representative Detector Optimization

Figure 6 shows how the numerical gain appears in a representative held-out detector. Standard Adam largely preserves the initial structure while adjusting continuous parameters. ADAPT instead suppresses three optical branches whose optimized contributions approach zero, yielding a simpler effective configuration while preserving the remaining optical structure. Under the same evaluation budget, Adam reaches a loss of 0.666 and ADAPT reaches 0.155, a **76.7% reduction**. The performance gain therefore coincides with a distinct, physically interpretable change in the optimized detector rather than a small perturbation of the baseline solution. Additional held-out examples are provided in Appendix E.

Table 1: **Optimization performance of classical optimizers across eight held-out gravitational-wave detector targets.** Lower loss is better. Red percentages indicate the relative increase in loss compared with Ours. The final column reports mean ± standard deviation.

Method	T1	T2	T3	T4	T5	T6	T7	T8	Mean ± Std.
Adam	0.423 +36.0%	0.887 +117.4%	1.833 +2344.0%	0.332 +45.6%	0.379 +152.7%	0.692 +397.8%	0.993 +509.2%	0.666 +329.7%	0.776 ± 0.458 +280.4%
NAAdam	0.423 +36.0%	0.696 +70.6%	0.423 +464.0%	0.334 +46.5%	0.415 +176.7%	0.375 +169.8%	0.695 +326.4%	0.255 +64.5%	0.452 ± 0.150 +121.6%
Lion	4.358 +1301.3%	3.841 +841.4%	2.028 +2604.0%	3.028 +1228.1%	2.004 +1236.0%	2.641 +1800.0%	2.663 +1533.7%	2.325 +1400.0%	2.861 ± 0.793 +1302.5%
SGDM	4.151 +1234.7%	0.445 +9.1%	0.468 +524.0%	0.473 +107.5%	0.581 +287.3%	0.448 +222.3%	0.196 +20.2%	0.415 +167.7%	0.897 ± 1.234 +339.7%
Sophia	0.506 +62.7%	0.412 +1.0%	0.627 +736.0%	1.749 +667.1%	0.653 +335.3%	0.574 +312.9%	0.430 +163.8%	1.659 +970.3%	0.826 ± 0.514 +304.9%
Adan	1.404 +351.4%	1.321 +223.8%	1.376 +1734.7%	2.783 +1120.6%	2.212 +1374.7%	0.673 +384.2%	1.355 +731.3%	1.387 +794.8%	1.564 ± 0.602 +666.7%
Ours	**0.311**	**0.408**	**0.075**	**0.228**	**0.150**	**0.139**	**0.163**	**0.155**	**0.204 ± 0.101**

Table 2: **Comparison with agentic evolution baselines across eight held-out gravitational-wave detector targets.** Lower loss is better. Red percentages indicate the relative increase in loss compared with Ours. The final column reports mean ± standard deviation.

Method	T1	T2	T3	T4	T5	T6	T7	T8	Mean ± Std.
GEPA	0.557 +79.1%	0.410 +0.5%	0.480 +540.0%	1.218 +434.2%	0.379 +152.7%	0.435 +212.9%	0.475 +191.4%	0.365 +135.5%	0.540 ± 0.263 +164.7%
OpenE.	0.707 +127.3%	0.492 +20.6%	0.449 +498.7%	0.261 +14.5%	0.665 +343.3%	0.426 +206.5%	0.439 +169.3%	0.954 +515.5%	0.549 ± 0.202 +169.1%
ShinkaE.	0.561 +80.4%	0.440 +7.8%	0.674 +798.7%	2.632 +1054.4%	0.366 +144.0%	0.210 +51.1%	0.643 +294.5%	0.402 +159.4%	0.741 ± 0.729 +263.2%
ASI	0.725 +133.1%	0.483 +18.4%	0.624 +732.0%	0.653 +186.4%	1.309 +772.7%	0.450 +223.7%	1.681 +931.3%	1.067 +588.4%	0.874 ± 0.410 +328.4%
Ours	**0.311**	**0.408**	**0.075**	**0.228**	**0.150**	**0.139**	**0.163**	**0.155**	**0.204 ± 0.101**

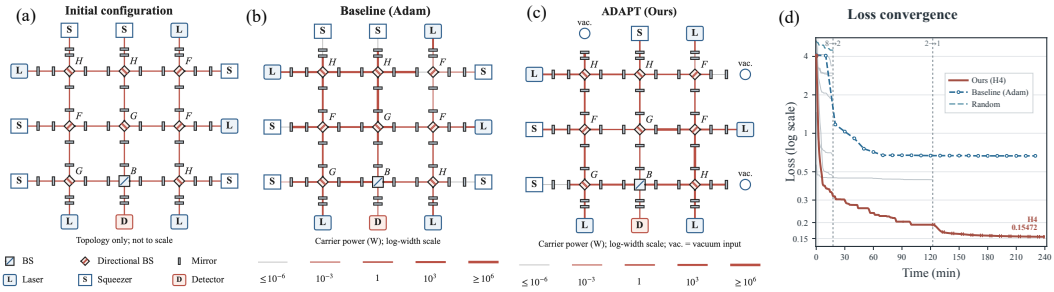


Figure 6: **Representative held-out detector optimization.** (a) Initial detector configuration. (b) Design obtained with standard Adam and its optimized carrier-power distribution. (c) Final ADAPT design. ADAPT suppresses three optical branches with near-zero optimized contributions and replaces them with vacuum inputs while preserving the remaining detector structure. (d) Loss convergence under the same scientific evaluation budget. ADAPT reaches a final reported loss of 0.155.

4.5 Ablation of Where and How to Search

Table 3: **Ablation of selection and optimizer evolution.** Lower is better. Mean ± standard deviation is computed across the eight held-out topologies.

Target	Direct+GEPA	Fixed+Adam	No-Prompt	Ours
T1	0.416 +33.8%	0.320 +2.9%	0.424 +36.3%	**0.311**
T2	0.986 +141.7%	0.371 -9.1%	0.443 +8.6%	**0.408**
T3	0.373 +397.3%	0.180 +140.0%	0.145 +93.3%	**0.075**
T4	0.476 +108.8%	0.246 +7.9%	0.287 +25.9%	**0.228**
T5	0.164 +9.3%	0.200 +33.3%	0.264 +76.0%	**0.150**
T6	0.138 -0.7%	0.143 +2.9%	0.148 +6.5%	**0.139**
T7	0.413 +153.4%	0.415 +154.6%	0.499 +206.1%	**0.163**
T8	0.421 +171.6%	0.172 +11.0%	0.210 +35.5%	**0.155**
Mean	0.423± 0.243 +107.4%	0.256± 0.094 +25.5%	0.303± 0.128 +48.5%	**0.204± 0.101**

We isolate the contributions of adaptive initialization, optimizer evolution, and science-specific guidance. **Direct+GEPA** bypasses evolved start selection and applies GEPA from a single retrieved initialization. **Fixed+Adam** retains the selected initialization but replaces the evolved optimizer with fixed Adam. **No-Prompt** removes the science-specific guidance while retaining the corresponding agentic evolution pipeline.

Removing evolved start selection increases the mean loss to 0.423±0.243, and full ADAPT reduces this value by **51.8%**. ADAPT is lower on seven of eight targets, with T6 effectively unchanged at 0.138 versus 0.139.

Retaining the selected start but replacing the evolved optimizer with Adam gives 0.256 ± 0.094. ADAPT reduces this loss by **20.3%** and is lower on seven targets, with T2 as the exception. Removing science-specific guidance gives 0.303 ± 0.128, whereas full ADAPT is **32.7%** lower and improves all eight targets.

These ablations separate the two adaptive decisions in ADAPT. Start selection reduces sensitivity to initialization, optimizer evolution improves subsequent local search, and science-specific guidance strengthens the evolution process across held-out targets.

5 Limitations and Future Work

ADAPT is designed for scientific optimization with reusable historical experience, expensive evaluations, and explicit physical constraints. The present study focuses on gravitational-wave detector design, where a differentiable simulator provides controlled evaluation and a large historical design library. The results therefore establish the effectiveness of the framework in one challenging scientific domain, but do not yet demonstrate universality across scientific disciplines.

The method also depends on the quality and coverage of its experience library. Nearest-neighbor retrieval is effective when historical designs contain useful local structure, but its advantage may diminish for targets that lie far outside the observed design distribution. In addition, evolving both selection workflows and optimization programs introduces computational overhead relative to a single fixed optimizer. This overhead is justified when scientific evaluations are sufficiently costly, but should be reduced for cheaper optimization problems.

Several extensions follow naturally. First, ADAPT can be evaluated on scientific problems with different search spaces, simulators, and physical constraints, including non-differentiable and experimental settings. Second, the experience library can be updated online so that successful solutions and optimizers become reusable knowledge for future targets. Third, more efficient strategy evolution could reduce the number of workflow–model evaluations through learned transfer, uncertainty-aware allocation, or meta-optimization. Finally, extending the independent validator to real experimental measurements would test whether self-evolving optimization remains reliable beyond simulation.

6 Conclusion

We introduced **ADAPT for Science**, a framework that treats the complete scientific search policy as the object of optimization. ADAPT decomposes this policy into two coupled decisions: *where to search* and *how to search*. Historical scientific experience guides target-specific initialization and resource allocation, while online execution feedback adapts the optimization procedure itself.

On eight held-out gravitational-wave detector topologies, ADAPT achieves a mean loss of 0.204 ± 0.101, reducing mean loss by **54.9%** relative to the strongest classical optimizer and by **62.2%** relative to the strongest general-purpose agentic evolution baseline under matched scientific evaluation budgets. ADAPT also achieves the lowest loss on all eight held-out topologies among the evaluated baselines. Ablation results further show that adaptive initialization, optimizer evolution, and science-specific guidance each contribute to the full-system performance.

These results support a broader view of scientific automation in which the search procedure itself is adaptive. Rather than optimizing scientific parameters under a fixed workflow, ADAPT combines prior scientific experience with execution feedback to continually refine how the search is initialized and conducted.

AI use statement

Large language models were used as components of the agentic optimization workflows to propose and revise candidate search policies and optimization procedures based on execution feedback. All candidates were evaluated by the scientific simulator and objective functions.

Generative AI tools were used only for language editing and manuscript polishing. The authors reviewed all AI-assisted edits and take full responsibility for the final manuscript.

Reproducibility statement

The supplementary material provides the scientific objective, data construction, optimization procedures, evaluation protocol, implementation settings, and additional experimental details needed to reproduce the study.

References

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, et al. Gepa: Reflective prompt evolution can outperform reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 8479–8565, 2026.

Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. *Advances in neural information processing systems*, 29, 2016.

Daniil A Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. Autonomous chemical research with large language models. *Nature*, 624(7992):570–578, 2023.

Benjamin Burger, Phillip M Maffettone, Vladimir V Gusev, Catherine M Aitchison, Yang Bai, Xiaoyan Wang, Xiaobo Li, Ben M Alston, Buyi Li, Rob Clowes, et al. A mobile robotic chemist. *Nature*, 583(7815):237–241, 2020.

Xiangning Chen, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, Yifeng Lu, et al. Symbolic discovery of optimization algorithms. *Advances in neural information processing systems*, 36:49205–49233, 2023.

Zhantao Chen, Alexander N Petsch, Aidan J Israelski, Rajan Plumley, Lingjia Shen, Cong Wang, Cheng Peng, Yuan Ni, Arun Bansil, Sugata Chowdhury, et al. An agentic artificially intelligent x-ray scientist. *Nature Machine Intelligence*, pp. 1–12, 2026.

Ching-An Cheng, Allen Nie, and Adith Swaminathan. Trace is the next autodiff: Generative optimization with rich feedback, execution traces, and llms. *Advances in Neural Information Processing Systems*, 37:71596–71642, 2024.

Connor W Coley, Dale A Thomas III, Justin AM Lummiss, Jonathan N Jaworski, Christopher P Breen, Victor Schultz, Travis Hart, Joshua S Fishman, Luke Rogers, Hanyu Gao, et al. A robotic platform for flow synthesis of organic compounds informed by ai planning. *Science*, 365(6453): eaax1566, 2019.

David Eriksson, Michael Pearce, Jacob Gardner, Ryan D Turner, and Matthias Poloczek. Scalable global optimization via local bayesian optimization. *Advances in neural information processing systems*, 32, 2019.

Alhussein Fawzi, Matej Balog, Aja Huang, Thomas Hubert, Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Francisco J R. Ruiz, Julian Schrittwieser, Grzegorz Swirszcz, et al. Discovering faster matrix multiplication algorithms with reinforcement learning. *Nature*, 610(7930):47–53, 2022.

Zhaoran Feng, Jiangzhi Chen, Ce Wang, and Jie Ren. Complexity-powered machine intelligent classification of quantum many-body dynamics. *Physical Review E*, 113(6):065301, 2026.

Jacob R Gardner, Matt J Kusner, Zhixiang Eddie Xu, Kilian Q Weinberger, and John P Cunningham. Bayesian optimization with inequality constraints. In *Icml*, volume 2014, pp. 937–945, 2014.

Jarosław M Granda, Liva Donina, Vincenza Dragone, De-Liang Long, and Leroy Cronin. Controlling an organic synthesis robot with machine learning to search for new reactivity. *Nature*, 559 (7714):377–381, 2018.

Florian Hase, Loïc M Roch, Christoph Kreisbeck, and Alán Aspuru-Guzik. Phoenics: a bayesian optimizer for chemistry. *ACS central science*, 4(9):1134–1145, 2018.

Ziyao Huang, Weiwei Wu, Kui Wu, Wei-Bin Lee, and Jianping Wang. Calm: Co-evolution of algorithms and language model for automatic heuristic design. In *International Conference on Learning Representations*, volume 2026, pp. 72468–72510, 2026.

Donald R Jones, Matthias Schonlau, and William J Welch. Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492, 1998.

Robert Lange, Yuki Imajuku, and Edoardo Cetin. Shinkaevolve: Towards open-ended and sample-efficient program evolution. In *International Conference on Learning Representations*, volume 2026, pp. 74026–74078, 2026.

Fei Liu, Xialiang Tong, Mingxuan Yuan, Xi Lin, Fu Luo, Zhenkun Wang, Zhichao Lu, and Qingfu Zhang. Evolution of heuristics: Towards efficient automatic algorithm design using large language model. *arXiv preprint arXiv:2401.02051*, 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.

Benjamin P MacLeod, Fraser GL Parlane, Thomas D Morrissey, Florian Häse, Loïc M Roch, Kevan E Dettelbach, Raphaell Moreira, Lars PE Yunker, Michael B Rooney, Joseph R Deeth, et al. Self-driving laboratory for accelerated discovery of thin-film materials. *Science Advances*, 6(20): eaaz8867, 2020.

Daniel J Mankowitz, Andrea Michi, Anton Zhernov, Marco Gelmi, Marco Selvi, Cosmin Paduraru, Edouard Leurent, Shariq Iqbal, Jean-Baptiste Lespiau, Alex Ahern, et al. Faster sorting algorithms discovered using deep reinforcement learning. *Nature*, 618(7964):257–263, 2023.

Amil Merchant, Simon Batzner, Samuel S Schoenholz, Muratahan Aykol, Gowoon Cheon, and Ekin Dogus Cubuk. Scaling deep learning for materials discovery. *Nature*, 624(7990):80–85, 2023.

Zhichu Ren, Zekun Ren, Zhen Zhang, Tonio Buonassisi, and Ju Li. Autonomous experiments using active learning and ai. *Nature Reviews Materials*, 8(9):563–564, 2023.

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.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1): 148–175, 2015.

Asankhaya Sharma. Openevolve: an open-source evolutionary coding agent. *GitHub repository. Accessed*, pp. 08–10, 2025.

Yiding Shi, Jianan Zhou, Wen Song, Jieyi Bi, Yaoxin Wu, Zhiguang Cao, and Jie Zhang. Generalizable heuristic generation through llms with meta-optimization. In *International Conference on Learning Representations*, volume 2026, pp. 119314–119349, 2026.

Benjamin J Shields, Jason Stevens, Jun Li, Marvin Parasram, Farhan Damani, Jesus I Martinez Alvarado, Jacob M Janey, Ryan P Adams, and Abigail G Doyle. Bayesian reaction optimization as a tool for chemical synthesis. *Nature*, 590(7844):89–96, 2021.

Parshin Shojaee, Kazem Meidani, Shashank Gupta, Amir Barati Farimani, and Chandan Reddy. Llm-sr: Scientific equation discovery via programming with large language models. In *International Conference on Learning Representations*, volume 2025, pp. 16054–16085, 2025.

Jasper Snoek, Hugo Larochelle, and Ryan Adams. Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25, 2012.

Nathan J Szymanski, Bernardus Rendy, Yuxing Fei, Rishi E Kumar, Tanjin He, David Milsted, Matthew J McDermott, Max Gallant, Ekin Dogus Cubuk, Amil Merchant, et al. An autonomous laboratory for the accelerated synthesis of inorganic materials. *Nature*, 624(7990):86–91, 2023.

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

Haoran Ye, Jiarui Wang, Zhiguang Cao, Federico Berto, Chuanbo Hua, Haeyeon Kim, Jinkyoo Park, and Guojie Song. Reevo: Large language models as hyper-heuristics with reflective evolution. *Advances in neural information processing systems*, 37:43571–43608, 2024.

Jiaqi Zhang, Louis Cammarata, Chandler Squires, Themistoklis P Sapsis, and Caroline Uhler. Active learning for optimal intervention design in causal models. *Nature Machine Intelligence*, 5(10): 1066–1075, 2023.

Zhi Zheng, Zhuoliang Xie, Zhenkun Wang, and Bryan Hooi. Monte carlo tree search for comprehensive exploration in llm-based automatic heuristic design. *arXiv preprint arXiv:2501.08603*, 2025.

Yuhang Zhou, Yixin Cao, Yuchen Ni, Shihan Dou, Xutian Chen, Ge Zhang, Xiang Liu, and Guangnan Ye. Prism: Probabilistic reward model with inherent structural modeling. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12345–12362, 2026.