ALPHASAS: STRUCTURE-AWARE NEURAL-SYMBOLIC SEARCH FOR AUTOMATED ALPHA DISCOVERY

Anonymous authors
Paper under double-blind review

ABSTRACT

The automated mining of predictive signals, or alphas, is a central challenge in quantitative finance. While Reinforcement Learning (RL) and neural-symbolic methods have enabled automated alpha generation, existing frameworks face three key limitations. First, large-scale exploration over expression spaces introduces multiple hypothesis testing concerns, causing factors with strong historical performance to exhibit limited out-of-sample reliability. Second, existing approaches rely on sequential representations that inadequately capture the structural relationships governing mathematical expressions and factor interactions. Third, current methods do not effectively incorporate previous search outcomes, resulting in inefficient exploration of redundant or unstable alpha structures. To address these challenges, we introduce AlphaSAS (Structure-Aware Neural-Symbolic Search for Automated Alpha Discovery), a framework built upon three innovations: (1) a structure-aware neural-symbolic representation that captures mathematical expression properties and factor relationships; (2) an adaptive candidate evaluation mechanism that incorporates previous discovery outcomes during search; and (3) an exploration strategy that promotes complementary alpha generation while reducing redundant factor discovery. Experiments across multiple equity markets demonstrate that AlphaSAS discovers more robust, diverse, and persistent alpha portfolios, establishing a systematic framework for reliable automated alpha discovery.

1 INTRODUCTION

The primary objective of quantitative alpha research is to identify signals that contain stable information about future asset returns. Formulaic alphas are useful because each signal is written as a short mathematical program built from market variables, rolling operators, and cross-sectional transformations. These formulas are interpretable and can be combined under explicit portfolio constraints. The main difficulty is scale: the number of valid expressions grows rapidly with program length, operator choice, and lookback windows, so an automated method may evaluate many related formulas before retaining a small subset.

Traditional factor research starts from an economic or statistical hypothesis and then tests a small number of candidate signals. Automated alpha mining instead generates many expressions and uses earlier outcomes to decide which expressions should be explored next. Genetic programming provides an early tree-based search formulation (Koza, 1992). Deep symbolic regression learns a proposal policy over mathematical expressions (Petersen et al., 2021). In quantitative finance, AlphaGen uses reinforcement learning for formula-set discovery (Yu et al., 2023), AlphaQCM uses distributional reinforcement learning (Zhu and Zhu, 2025), and AlphaForge combines formula generation with dynamic factor weighting (Shi et al., 2025). The resulting selection is strongly data dependent. Repeated evaluation on one development sample can make the selected factor appear stronger than a prespecified factor. Sullivan et al. study this issue for technical trading rules (Sullivan et al., 1999). White formalizes a reality-check approach to data snooping (White, 2000). Romano and Wolf develop stepwise multiple-testing control (Romano and Wolf, 2005). Harvey et al. discuss the same concern in the factor literature (Harvey et al., 2016), while Bailey and López de Prado study selection bias in Sharpe-ratio evaluation (Bailey and López de Prado, 2014). McLean and Pontiff report weaker average factor returns after publication (McLean and Pontiff, 2016). Harvey and Liu study both false and missed discoveries in financial economics (Harvey and Liu, 2020). Together, these results motivate a clear separation between discovery and later testing.

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.

A second issue is representation. Mathematical expressions are not ordinary token sequences: operator type, argument order, nesting, window parameters, and algebraic equivalence determine their meaning. A typed abstract syntax tree (AST) represents these relations directly. Battaglia et al. describe graph networks as a general framework for relational structure (Battaglia et al., 2018). Schlichtkrull et al. introduce relation-specific graph convolution (Schlichtkrull et al., 2018), while Xu et al. analyze the expressive power of graph aggregation (Xu et al., 2019). These ideas motivate different edge roles for ordered operations such as subtraction and division. A canonical form is also useful because equivalent formulas should not consume separate search effort only because they use different valid syntax (Willsey et al., 2021).

A third issue is the limited use of search history. Many procedures retain only the current best score or current factor set, discarding information from failed, redundant, or unstable formulas. Ellis et al. show that learned program components can reduce later search effort (Ellis et al., 2018). Wong et al. use learned abstractions and search heuristics (Wong et al., 2021), while DreamCoder jointly learns reusable program structure and search guidance (Ellis et al., 2023). Upper-confidence methods in black-box optimization use a related idea by combining estimated quality with uncertainty when choosing the next evaluation (Srinivas et al., 2010). For alpha discovery, previous search outcomes can therefore be treated as an explicit source of information rather than as discarded intermediate results.

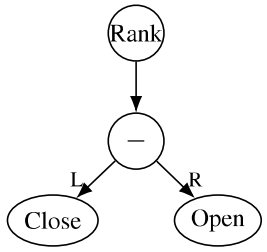

Figure 1: Typed abstract syntax tree (AST) for a candidate alpha. Operator nodes encode the formula structure, and leaf nodes represent its input variables.

To address these limitations, we introduce **AlphaSAS**, a structure-aware neural-symbolic search framework. Each candidate is represented by a typed expression graph and by its relation to the current selected factor set. A search history stores previous candidate outcomes and provides local estimates of expected quality, uncertainty, instability, and redundancy. These quantities form an adaptive evaluation score that guides the proposal model. Factor diversity is measured through the increase in a regularized determinant score when a candidate is added to the current factor set, so search favors candidates that add new information rather than repeat existing directions. Finally, a candidate selected during development is specified before later observations enter a separate sequential test process, preventing development outcomes from being reused as later evidence.

In summary, our contributions are as follows:

- We introduce a **structure-aware neural-symbolic representation** that combines relation-specific AST message passing, a safe canonical form, and context from the current selected factor set.
- We develop a **history-aware candidate evaluation mechanism** that uses previous discovery outcomes through local estimates of historical quality, uncertainty, instability, and program complexity.
- We propose a **diversity-aware exploration strategy** based on a regularized determinant gain, together with a separate post-selection testing rule for repeated adaptive search.

2 Background and Related Work

2.1 Alphas in quantitative trading

A formulaic alpha maps information available at date t to a cross-sectional score for each tradable asset. The score is commonly standardized and transformed into a portfolio signal subject to exposure, leverage, liquidity, and turnover constraints. Predictive quality is summarized by the information coefficient (IC), rank information coefficient (RIC), and their information ratios, while portfolio evaluation uses return, drawdown, turnover, and Sharpe ratio. Formulaic alpha collections provide a transparent representation of trading signals (Kakushadze, 2016). Novy-Marx and Velikov show that anomaly evaluation also depends materially on trading costs (Novy-Marx and Velikov, 2016). A production system therefore usually maintains a set of factors rather than a single factor because individual signals can decay, become regime dependent, or overlap with other predictors. A useful

new factor should therefore contain predictive information and add information beyond the factors already selected.

2.2 Alpha mining and adaptive selection

Automated alpha mining searches a large program space instead of testing a small set of fixed hypotheses. Genetic programming constructs formulas through mutation and crossover (Koza, 1992). Deep symbolic regression instead learns a token-level proposal policy (Petersen et al., 2021). Recent financial methods formulate alpha generation as sequential decision making. AlphaGen optimizes the contribution of a formula to a factor collection (Yu et al., 2023); AlphaQCM uses distributional reinforcement learning for a nonstationary search objective (Zhu and Zhu, 2025); and AlphaForge combines formula mining with dynamic factor reweighting (Shi et al., 2025). These methods improve search, but the final formula is still chosen after many evaluations. Sullivan et al. quantify data-snooping effects in technical trading rules (Sullivan et al., 1999). White develops a reality-check test for searched model collections (White, 2000). Romano and Wolf provide stepwise family-wise control (Romano and Wolf, 2005). Harvey et al. show why similar concerns arise in large factor collections (Harvey et al., 2016). The deflated Sharpe ratio adjusts for repeated strategy evaluation (Bailey and López de Prado, 2014). McLean and Pontiff document weaker average factor returns after publication (McLean and Pontiff, 2016), and Harvey and Liu study the balance between false and missed discoveries (Harvey and Liu, 2020). AlphaSAS therefore separates development search from later statistical testing.

2.3 Neural-symbolic structure representation

A symbolic expression has both program structure and numerical behavior. Ellis et al. combine explicit programs with learned search guidance (Ellis et al., 2018). Petersen et al. learn a policy over symbolic expressions (Petersen et al., 2021). DreamCoder learns reusable program structure together with neural search guidance (Ellis et al., 2023). For formulaic alphas, an AST directly represents operator nesting and argument roles. Graph networks provide a general representation for relational structure (Battaglia et al., 2018). Relation-specific graph convolution then allows different transformations for different edge roles (Schlichtkrull et al., 2018). This matters for ordered operators such as subtraction, division, and lagged covariance. A canonical form provides a separate symbolic step: safe algebraic rewrites can map equivalent expressions to one program identity and reduce duplicate evaluations (Willsey et al., 2021). AlphaSAS combines these symbolic and learned components and then adds information about the current selected factor set.

2.4 Search history, testing, and factor diversity

Adaptive search uses previous evaluations to decide which candidate should be tested next. Program induction can improve future synthesis by learning reusable program parts (Ellis et al., 2018). LAPS learns program abstractions and search heuristics (Wong et al., 2021), while DreamCoder learns reusable structure over repeated tasks (Ellis et al., 2023). Upper-confidence methods provide a related principle by combining a response estimate with uncertainty (Srinivas et al., 2010). AlphaSAS applies this principle through an explicit search history containing both successful and unsuccessful formula evaluations.

The statistical question is separate. A formula chosen after repeated search should not be treated as if it had been fixed before the data were examined. Benjamini and Hochberg introduce false-discovery-rate control for a fixed family (Benjamini and Hochberg, 1995). Foster and Stine introduce alpha-investing for sequential testing (Foster and Stine, 2008). Javanmard and Montanari develop online FDR rules (Javanmard and Montanari, 2018), while SAFFRON provides an adaptive online procedure (Ramdas et al., 2018). Howard et al. develop time-uniform confidence sequences (Howard et al., 2021). Ramdas et al. discuss anytime-valid sequential inference (Ramdas et al., 2023), and Wang and Ramdas develop FDR control with e-values (Wang and Ramdas, 2022). AlphaSAS uses a conservative family-wise rule in its main theory. Factor diversity is handled separately through the determinant of the factor similarity matrix. Determinantal point processes favor diverse subsets through determinant-based scores (Kulesza and Taskar, 2012). Krause et al. use related determinant and information criteria in subset selection (Krause et al., 2008). Classical greedy set-selection results formalize the resulting diminishing-gain property (Nemhauser et al., 1978).

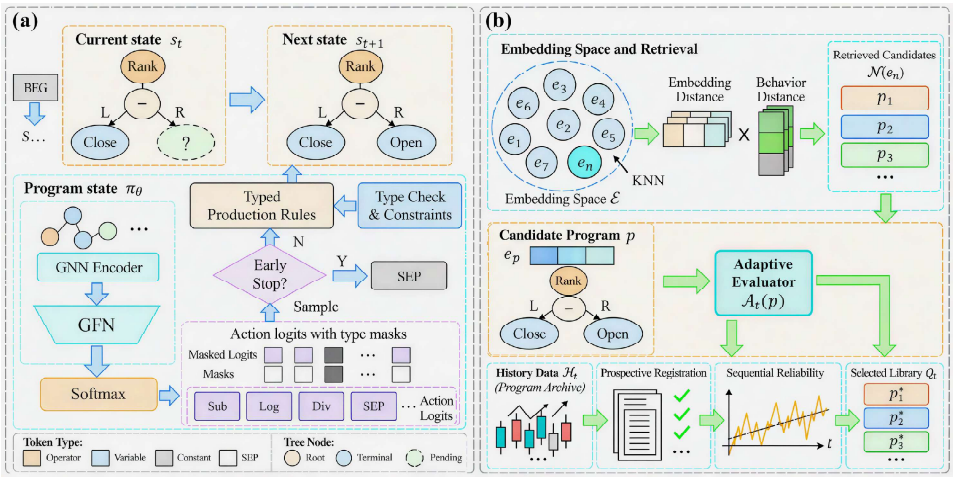

Figure 2: Overview of AlphaSAS. A typed symbolic generator creates valid ASTs. The structure encoder represents each candidate, while the search history provides outcomes of related earlier candidates. The adaptive evaluator combines development utility, history, uncertainty, failure risk, complexity, and factor diversity. Selected candidates are specified before later observations update sequential test processes. Retained factors are combined by a rolling linear model.

3 METHODOLOGY

3.1 FRAMEWORK OVERVIEW AND PROBLEM FORMULATION

Let $i \in \{1, \dots, N_t\}$ index assets available at decision date t. The information set $\mathcal{F}_t$ contains only records released and observable by t, previously emitted predictions, past realized outcomes, and search decisions already made. A valid alpha program $p \in \mathcal{P}$ maps the point-in-time feature tensor X_t to a cross-sectional score vector

$$z_t^p = p(X_t) \in \mathbb{R}^{N_t}. \tag{1}$$

Each program is represented by a typed AST $G_p = (V_p, E_p)$. Let $\mathcal{C}$ be a fixed rewrite system that contains only transformations valid under the declared missing-value and protected-arithmetic rules. We define

$$p \sim_{\mathcal{C}} q \quad \Longleftrightarrow \quad \mathrm{can}_{\mathcal{C}}(p) = \mathrm{can}_{\mathcal{C}}(q), \tag{2}$$

and assign one persistent identity $\mathrm{ID}(p)$ to each canonical class. The search process maintains a *search history* $\mathcal{H}_t$ containing outcomes of earlier candidates and a *selected factor set* $\mathcal{Q}_t$ containing factors retained by the method.

AlphaSAS separates candidate construction, history-aware evaluation, diversity-aware search, and post-selection testing. A structural encoder represents the mathematical expression. The search history summarizes how related candidates behaved in earlier search rounds. A diversity score measures whether a new candidate adds a direction not already represented by $\mathcal{Q}_t$. A candidate that passes development screening is then *specified* at time τ_p before later outcomes are used for statistical testing. Development data may determine which formula is selected, but those same outcomes are not reused in its later test process.

For a completed program, define the development utility

$$U_{\mathrm{dev}}(p) = \omega_1 \widehat{\mathrm{RICIR}}(p) + \omega_2 \widehat{\mathrm{NetSR}}(p) - \omega_3 \,\mathrm{Turnover}(p) - \omega_4 \,\mathrm{Instab}(p), \tag{3}$$

where all terms are computed on the current development prefix. The coefficients $\omega_j \geq 0$ are selected on training and validation data and are frozen before final testing. U_{dev} guides the search; it is not a statistical test. The complete flow is shown in figure 2.

3.2 STRUCTURE-AWARE NEURAL-SYMBOLIC REPRESENTATION

A formula is parsed into a typed AST. Nodes represent raw features, constants, unary operators, binary operators, and rolling operators. Edge labels encode roles such as unary operand, left operand, right operand, feature argument, and window argument. Let $h_v^{(0)}$ denote the initial embedding of node v. We use relation-specific message passing

$$h_v^{(\ell+1)} = \sigma\left(W_0^{(\ell)} h_v^{(\ell)} + \sum_{r\in\mathcal{R}} \sum_{u\in\mathcal{N}_r(v)} \frac{1}{c_{v,r}} W_r^{(\ell)} h_u^{(\ell)} \right), \tag{4}$$

where $\mathcal{R}$ is the relation set, $\mathcal{N}_r(v)$ is the r-type neighborhood of v, and $c_{v,r}$ is a normalization factor (Schlichtkrull et al., 2018). A graph representation is obtained by weighted pooling,

$$e_p = \sum_{v\in V_p} a_v h_v^{(L)}, \qquad a_v = \frac{\exp(q^\top h_v^{(L)})}{\sum_{u\in V_p} \exp(q^\top h_u^{(L)})}. \tag{5}$$

Unlike a flat token sequence, equation (4) can assign different parameters to left and right operands. This distinction matters for subtraction, division, lagged covariance, and other ordered operations.

Proposition 1 (Ordered Operand Sensitivity)**.** *Consider a non-commutative binary node v with left child l and right child r. Suppose $W_L \neq W_R$, $h_l \neq h_r$, and σ is injective on the two local inputs to the activation. If*

$$(W_L - W_R)(h_l - h_r) \neq 0, \tag{6}$$

then exchanging the left and right children changes the updated representation of v.

The proof is given in Appendix C. For a commutative operator, the two operand edges share one relation type, so exchanging the operands leaves the local sum unchanged.

Structural information is combined with factor interaction. Let $\widetilde{z}^p \in \mathbb{R}^T$ denote the standardized output of a completed candidate on admissible development observations. For $q \in \mathcal{Q}_t$, define

$$A_{pq} = \mathrm{Corr}(\widetilde{z}^p, \widetilde{z}^q), \qquad \rho_{pq} = |A_{pq}|. \tag{7}$$

A selected-set context vector is

$$g_p = \sigma\left(U_0 e_p + \sum_{q\in\mathcal{Q}_t} \frac{\kappa(\rho_{pq})}{\sum_{j\in\mathcal{Q}_t} \kappa(\rho_{pj}) + \varepsilon} U_1 e_q \right), \tag{8}$$

where κ is a nonnegative kernel. The final candidate representation is

$$\xi_p = \mathrm{LayerNorm}(W_e e_p + W_g g_p + W_s s_p), \tag{9}$$

where s_p contains program statistics such as depth, token count, maximum lookback, valid coverage, and numerical-warning rates. Thus ξ_p contains both expression structure and its observed relation to the selected factor set. A canonical form is assigned before ξ_p is stored, so algebraic duplicates share one identity and one history record (Willsey et al., 2021).

Boundary of the symbolic search space. AlphaSAS uses finite, acyclic ASTs. The DSL permits nested pointwise, cross-sectional, lag, and fixed-window rolling operators subject to fixed limits on program length, tree depth, and maximum lookback. It excludes recursion, temporal loops, data-dependent iteration counts, mutable program state, and external model calls. Every candidate therefore has a finite, deterministic point-in-time evaluation rule.

3.3 HISTORY-AWARE CANDIDATE EVALUATION

After $t-1$ completed evaluations, the search history is

$$\mathcal{H}_t = \{(\xi_j, U_j, R_j, F_j, L_j)\}_{j=1}^{t-1}, \tag{10}$$

where U_j is development utility, R_j is redundancy with the selected factor set, $F_j \in [0,1]$ is an instability or failure score, and L_j is description length. For a new candidate p, define

$$k_{tj}(p) = \exp\left[-\frac{\|M(\xi_p - \xi_j)\|_2^2}{2\tau_h^2} \right], \qquad D_t(p) = \kappa_0 + \sum_{j<t} k_{tj}(p), \tag{11}$$

and normalized weights

$$\omega_{t0}(p) = \frac{\kappa_0}{D_t(p)}, \qquad \omega_{tj}(p) = \frac{k_{tj}(p)}{D_t(p)}, \quad \omega_{t0} + \sum_{j<t} \omega_{tj} = 1. \tag{12}$$

Here M is a learned diagonal metric, $\tau_h > 0$ controls neighborhood width, and (m_0, v_0) is a fixed training-data anchor. The local history mean, variance, and failure estimate are

$$\widehat{m}_t(p) = \omega_{t0} m_0 + \sum_{j<t} \omega_{tj} U_j, \tag{13}$$

$$\widehat{v}_t(p) = \omega_{t0}\big[v_0 + (m_0 - \widehat{m}_t)^2\big] + \sum_{j<t} \omega_{tj}(U_j - \widehat{m}_t)^2, \tag{14}$$

$$\widehat{f}_t(p) = \sum_{j<t} \omega_{tj} F_j. \tag{15}$$

The anchor prevents a region with little search history from receiving an artificial zero variance. Failed and redundant candidates remain in $\mathcal{H}_t$ and continue to affect later search. The effective local history size is reported in Appendix B.

Scaling of proposal terms. The candidate-score components have different numerical ranges, so they are not added on their raw scales. For component $q_{t,k}(p)$, AlphaSAS uses a chronological robust standardization based only on candidates completed earlier,

$$\widetilde{q}_{t,k}(p) = \mathrm{clip}\left(\frac{q_{t,k}(p) - \mathrm{med}_{j<t}\, q_{j,k}}{\max\{1.4826\ \mathrm{MAD}_{j<t}(q_{j,k}), s_{\min}\}}, -c_{\mathrm{sc}}, c_{\mathrm{sc}}\right). \tag{16}$$

Training-history values are used at cold start, and all scaling rules are frozen before final testing. The current candidate never sets its own scale. This prevents one unusually large diversity or utility value from dominating the proposal update only because its numerical units are larger. The coefficients below therefore control intended relative importance rather than raw measurement scale.

The adaptive candidate score is

$$\begin{aligned}\mathcal{A}_t(p) = \widetilde{U}_{\mathrm{dev},t}(p) + \eta_m \widetilde{m}_t(p) + \beta_t \widetilde{\sigma}_t(p) + \eta_d \widetilde{\Delta}_{\mathrm{div}}(p \mid \mathcal{Q}_t) \\ - \eta_f \widetilde{f}_t(p) - \eta_L \widetilde{L}_t(p),\end{aligned} \tag{17}$$

where each tilde denotes equation (16), $\widetilde{\sigma}_t$ is the scaled form of $\sqrt{\widehat{v}_t}$, and Δ_{div} is defined in section 3.4. The uncertainty term gives additional evaluations to poorly resolved regions; the failure and complexity terms reduce repeated evaluation of unstable or unnecessarily long expressions.

The neural proposal model operates on partial ASTs. If s_k is a partial program, m_t is a pooled summary of nearby history records, and $\mathcal{V}(s_k)$ is the set of grammar-valid edits, then

$$\pi_\theta(a \mid s_k, m_t) = \frac{\exp\{f_\theta(h_{s_k}, m_t, a)/\tau_s\}}{\sum_{a' \in \mathcal{V}(s_k)} \exp\{f_\theta(h_{s_k}, m_t, a')/\tau_s\}}, \quad a \in \mathcal{V}(s_k). \tag{18}$$

For a minibatch $\mathcal{B}$ of completed trajectories, define

$$\widetilde{w}_j = \frac{\exp\{(\mathcal{A}_t(p_j) - b_t)/\tau_w\}}{\sum_{k \in \mathcal{B}} \exp\{(\mathcal{A}_t(p_k) - b_t)/\tau_w\}}, \tag{19}$$

and update the proposal model with

$$\mathcal{L}_{\mathrm{search}}(\theta) = -\sum_{j \in \mathcal{B}} \widetilde{w}_j \sum_{k=0}^{n_j - 1} \log \pi_\theta(a_{j,k} \mid s_{j,k}, m_t) - \beta_H \sum_{j,k} \mathcal{H}(\pi_\theta(\cdot \mid s_{j,k}, m_t)). \tag{20}$$

The first term shifts proposal mass toward trajectories with high history-adjusted scores; the entropy term reduces early concentration of the action distribution. The search history remains an explicit record even when the proposal network changes.

3.4 FACTOR DIVERSITY AND POST-SELECTION TESTING

Let $Z_{\mathcal{Q}} \in \mathbb{R}^{T\times K}$ contain standardized outputs of the current K selected factors and let $z_p \in \mathbb{R}^T$ be a standardized candidate. Define

$$C_{\mathcal{Q}} = \frac{1}{T} Z_{\mathcal{Q}}^\top Z_{\mathcal{Q}}, \qquad G_{\mathcal{Q}} = C_{\mathcal{Q}} + \varepsilon I_K, \qquad \varepsilon > 0. \tag{21}$$

We use the factor-diversity score

$$D_\varepsilon(\mathcal{Q}) = \log\det\left(I_K + \varepsilon^{-1} C_{\mathcal{Q}}\right). \tag{22}$$

The diversity gain of candidate p is

$$\Delta_{\text{div}}(p \mid \mathcal{Q}) = D_\varepsilon(\mathcal{Q} \cup \{p\}) - D_\varepsilon(\mathcal{Q}). \tag{23}$$

Determinant-based diversity scores measure joint set structure (Kulesza and Taskar, 2012). Related determinant criteria are also used in information-based subset selection (Krause et al., 2008). The score therefore measures joint redundancy rather than only the largest pairwise correlation.

Proposition 2 (Marginal Diversity Gain). *Assume $T^{-1} z_p^\top z_p = 1$ and let $c = T^{-1} Z_{\mathcal{Q}}^\top z_p$. Then*

$$\Delta_{\text{div}}(p \mid \mathcal{Q}) = \log\left(\frac{1 + \varepsilon - c^\top G_{\mathcal{Q}}^{-1} c}{\varepsilon}\right). \tag{24}$$

Hence the gain is smaller when the current factors explain more of the candidate output.

Theorem 1 (Diminishing Diversity Gain). *Let $K \succeq 0$ be the factor-similarity matrix of a finite candidate collection and define*

$$F(S) = \log\det\left(I + \varepsilon^{-1} K_S\right), \qquad S \subseteq \mathcal{P}_0. \tag{25}$$

Then $F(\varnothing) = 0$, adding a factor cannot decrease F, and for $A \subseteq B$ and $p \notin B$,

$$F(A \cup \{p\}) - F(A) \geq F(B \cup \{p\}) - F(B). \tag{26}$$

Thus a candidate's diversity gain cannot increase after more factors have already been selected.

The proofs are given in Appendix C. The theorem concerns the diversity score only; it does not claim that predictive utility follows the same property.

Adaptive search also creates a testing problem. Let τ_p be the candidate specification time. At τ_p, the program identity, sign, normalization, payoff definition, and testing rule are fixed before later outcomes are observed. Let w_t^p be the resulting position and let $m_{t+1|t}^b$ be a return forecast fitted using only $\mathcal{F}_t$. Define the raw cost-adjusted payoff

$$Z_{t+1}^p = (w_t^p)^\top \left(r_{t+1} - m_{t+1|t}^b\right) - C_t(w_t^p, w_{t-1}^p). \tag{27}$$

With a predictable scale $B_t > 0$, define the bounded test variable

$$X_{t+1}^p = \text{clip}\left(\frac{Z_{t+1}^p}{B_t}, -1, 1\right). \tag{28}$$

Because clipping changes the tested quantity, the null is stated directly on X_{t+1}^p:

$$H_{0,p}: \quad \mathbb{E}[X_{t+1}^p \mid \mathcal{F}_t] \leq 0 \quad \text{for all } t \geq \tau_p. \tag{29}$$

Choose a predictable $\lambda_t^p \in [0, \lambda_{\max})$, $\lambda_{\max} < 1$, and define the sequential test process

$$E_T^p = \prod_{t=\tau_p}^{T-1} \left(1 + \lambda_t^p X_{t+1}^p\right). \tag{30}$$

Theorem 2 (Sequential Family-Wise Error Control). *Suppose every candidate is specified at an $\{\mathcal{F}_t\}$-stopping time, the post-specification process in equation (30) is a nonnegative test process under $H_{0,p}$, and $\sum_p \pi(p) \leq 1$. Then*

$$\mathbb{P}\left(\exists p, \exists T: \ H_{0,p} \text{ holds and } E_T^p \geq [\alpha\pi(p)]^{-1}\right) \leq \alpha. \tag{31}$$

The result allows adaptive candidate creation and repeated monitoring, provided observations used before candidate specification are not inserted into the later test process.

The proof is given in Appendix C. The theorem controls false statistical selection; it does not claim that the adaptive candidate score predicts future performance without error.

Selection also checks whether a candidate adds value beyond the current selected factor set. Let $\widehat{r}^{\mathcal{Q}}_{t+1|t}$ be a return forecast from factors in $\mathcal{Q}_t$, fitted using only information available at time t. Define

$$\epsilon^{\mathcal{Q}}_{t+1} = r_{t+1} - \widehat{r}^{\mathcal{Q}}_{t+1|t}, \qquad Z^{p|\mathcal{Q}}_{t+1} = (w^p_t)^\top \epsilon^{\mathcal{Q}}_{t+1} - \Delta C_t(p \mid \mathcal{Q}). \tag{32}$$

A second sequential test process $E^{p|\mathcal{Q}}_T$ is formed from the bounded version of equation (32). With error allocations $\alpha_s + \alpha_i \le \alpha$, a generic selection rule is

$$\begin{aligned}\mathsf{Select}_T(p) = \mathbf{1}\Big[&E^{p,\mathrm{stand}}_T \ge (\alpha_s \pi_s(p))^{-1},\ E^{p|\mathcal{Q}}_T \ge (\alpha_i \pi_i(p))^{-1},\\ &\underline{\mu}^{p|\mathcal{Q}}_T > 0,\ \mathrm{Turnover}(p) \le \tau_{\max},\ \mathrm{Coverage}(p) \ge \rho_{\min},\ \mathrm{NumValid}(p) = 1\Big],\end{aligned} \tag{33}$$

where $\underline{\mu}^{p|\mathcal{Q}}_T$ is an anytime lower confidence bound for the bounded added-value mean. Appendix C gives the detailed construction.

3.5 Alpha combination

The final portfolio signal uses a rolling linear combiner. Let Z_u contain the standardized outputs of selected factors available at date u and let y_{u+1} be the subsequent return target. At date t, coefficients are fitted only on earlier observations:

$$\widehat{a}_t = \arg\min_a \left\{ \sum_{u<t} \rho^{t-1-u} \|y_{u+1} - Z_u a\|_2^2 + \lambda_a \|a\|_2^2 \right\}, \tag{34}$$

with $0 < \rho \le 1$ and $\lambda_a > 0$. The combined score is $s_t = Z_t \widehat{a}_t$ and is mapped to portfolio weights using the same cross-sectional normalization and trading constraints applied to all methods. The linear form keeps factor contributions visible and keeps the evaluation focused on alpha discovery rather than on a separate nonlinear allocation model.

4 Experiments and Evaluation Protocol

4.1 Experiment setting

Evaluation metrics. We evaluate both predictive quality and portfolio behavior. Correlation metrics are information coefficient (IC), IC information ratio (ICIR), rank information coefficient (RIC), and RIC information ratio (RICIR). Portfolio metrics are annualized return (AR), maximum drawdown (MDD), Sharpe ratio (SR), and turnover. Because AlphaSAS also targets reliable discovery, we report the number of evaluated canonical programs, duplicate-rejection rate, statistical-rejection rate, time from candidate specification to statistical selection, and the average diversity gain of selected factors. Exact definitions are given in Appendix D.

Datasets. The principal evaluation uses CSI300, CSI500, CSI1000, and S&P500 daily equity universes. The Chinese split is 2010–2020 for training, 2021 for validation, and 2022–2024 for test. The U.S. split is 2010–2016 for training, 2017 for validation, and 2018–2020 for test. Membership, price adjustments, and all features are handled point-in-time. The search history, normalization parameters, proposal model, and hyperparameters are fitted only on the permitted training and validation history. Test labels are read only by the final metric implementation.

Baselines. The comparison includes standard predictors (MLP, LightGBM, and XGBoost) and genetic programming. AlphaGen represents reinforcement-learning formula mining (Yu et al., 2023). AlphaQCM represents distributional reinforcement-learning search (Zhu and Zhu, 2025). AlphaForge represents neural formula generation with dynamic factor weighting (Shi et al., 2025). A relation-aware graph-based formula miner may also be included as a direct representation baseline. Every formula method uses the same raw feature set, operator grammar, maximum expression length,

Table 1: Performance comparison on CSI300, CSI500, and CSI800 using correlation-based evaluation metrics. Results are reported as mean ± standard deviation over four random seeds (0–3). Higher values are better for all metrics. Best and second-best performances are shown in **bold** and underlined, respectively; rankings are determined from the unrounded mean values.

Dataset	Method	IC	ICIR	RIC	RICIR
CSI300	MLP	0.030 ± 0.017	0.218 ± 0.119	0.035 ± 0.024	0.269 ± 0.186
	LightGBM	0.024 ± 0.003	0.242 ± 0.028	0.020 ± 0.001	0.191 ± 0.010
	Neural ODE	0.037 ± 0.004	0.260 ± 0.028	0.032 ± 0.006	0.214 ± 0.040
	Neural SDE	0.032 ± 0.005	0.239 ± 0.040	0.026 ± 0.005	0.188 ± 0.035
	AlphaGen	$\underline{0.041 \pm 0.011}$	$\mathbf{0.302 \pm 0.080}$	$\underline{0.046 \pm 0.021}$	$\mathbf{0.326 \pm 0.156}$
	AlphaQCM	0.039 ± 0.007	0.276 ± 0.051	0.043 ± 0.009	0.301 ± 0.073
	AlphaSAGE	0.033 ± 0.016	0.246 ± 0.080	0.037 ± 0.017	0.266 ± 0.088
	AlphaSAS	$\mathbf{0.041 \pm 0.004}$	$\underline{0.281 \pm 0.037}$	$\mathbf{0.049 \pm 0.005}$	$\underline{0.305 \pm 0.046}$
CSI500	MLP	0.031 ± 0.011	0.269 ± 0.088	0.034 ± 0.018	0.299 ± 0.162
	LightGBM	0.027 ± 0.003	0.311 ± 0.027	0.031 ± 0.007	0.364 ± 0.077
	Neural ODE	0.029 ± 0.002	0.239 ± 0.014	0.028 ± 0.002	0.218 ± 0.018
	Neural SDE	0.025 ± 0.005	0.215 ± 0.030	0.024 ± 0.007	0.186 ± 0.055
	AlphaGen	0.036 ± 0.005	0.273 ± 0.025	0.041 ± 0.007	0.339 ± 0.061
	AlphaQCM	0.035 ± 0.007	0.296 ± 0.053	0.042 ± 0.012	0.360 ± 0.085
	AlphaSAGE	$\mathbf{0.041 \pm 0.007}$	$\underline{0.336 \pm 0.039}$	$\underline{0.059 \pm 0.016}$	$\mathbf{0.497 \pm 0.061}$
	AlphaSAS	$\underline{0.039 \pm 0.003}$	$\mathbf{0.343 \pm 0.016}$	$\mathbf{0.059 \pm 0.003}$	$\underline{0.449 \pm 0.018}$
CSI800	MLP	0.015 ± 0.007	0.176 ± 0.082	0.018 ± 0.013	0.206 ± 0.152
	LightGBM	0.026 ± 0.003	$\underline{0.305 \pm 0.035}$	0.025 ± 0.004	0.286 ± 0.045
	Neural ODE	0.026 ± 0.002	0.237 ± 0.024	0.024 ± 0.003	0.197 ± 0.033
	Neural SDE	0.028 ± 0.003	0.264 ± 0.020	0.028 ± 0.003	0.239 ± 0.022
	AlphaGen	0.028 ± 0.004	0.258 ± 0.054	0.039 ± 0.007	0.344 ± 0.088
	AlphaQCM	$\underline{0.029 \pm 0.017}$	0.260 ± 0.147	0.041 ± 0.019	0.332 ± 0.150
	AlphaSAGE	0.025 ± 0.020	0.208 ± 0.159	$\underline{0.045 \pm 0.015}$	$\mathbf{0.396 \pm 0.082}$
	AlphaSAS	$\mathbf{0.043 \pm 0.006}$	$\mathbf{0.311 \pm 0.074}$	$\mathbf{0.055 \pm 0.006}$	$\underline{0.355 \pm 0.085}$

completed-program budget, portfolio rule, and transaction-cost convention. Search methods are run with paired random seeds.

4.2 Overall Performance

Across CSI300, CSI500, and CSI800, AlphaSAS shows a consistent advantage on the principal correlation-based measures, although the relative improvement varies across datasets and metrics. On CSI300, AlphaSAS achieves the highest IC and RIC and the second-highest ICIR and RICIR, indicating strong predictive correlation together with competitive temporal stability. On CSI500, it achieves the highest ICIR and RIC while ranking second on IC and RICIR, whereas on CSI800 it records the highest IC, ICIR, and RIC and remains second on RICIR. The relatively small standard deviations for several leading AlphaSAS results, particularly CSI300 IC and RIC and CSI500 ICIR and RIC, further indicate that the observed performance is stable across the four random seeds rather than being determined by an isolated run. The agreement between the dataset-level, average, and median comparisons suggests that the improvement is not concentrated in a single market universe and is particularly consistent for IC-based predictive strength and stability. At the same time, AlphaSAGE remains stronger on aggregate RICIR, showing that the relative advantage of AlphaSAS is more pronounced in average correlation strength and IC-based temporal consistency than in the stability of rank correlation. Complete cross-dataset average and median values for all competing methods and metrics are provided in Appendix E.

4.3 Sensitivity Analysis

The sensitivity analysis examines the two deployment parameters used in the final AlphaSAS calibration: signed score power $\gamma \in \{0.5, 0.75, 1, 1.5, 2\}$ and causal smoothing span $h \in \{1, 3, 5, 10\}$. Figure 3 reports their effect on IC, Rank IC, ICIR, and Rank ICIR across CSI300, CSI500, and CSI800, where the black open circles indicate the configuration selected using three chronological validation folds only. The remaining test curves are shown solely to characterize the local response around this frozen configuration and are not used for subsequent tuning. The resulting surfaces are generally smooth rather than being dominated by a single isolated setting, although the response

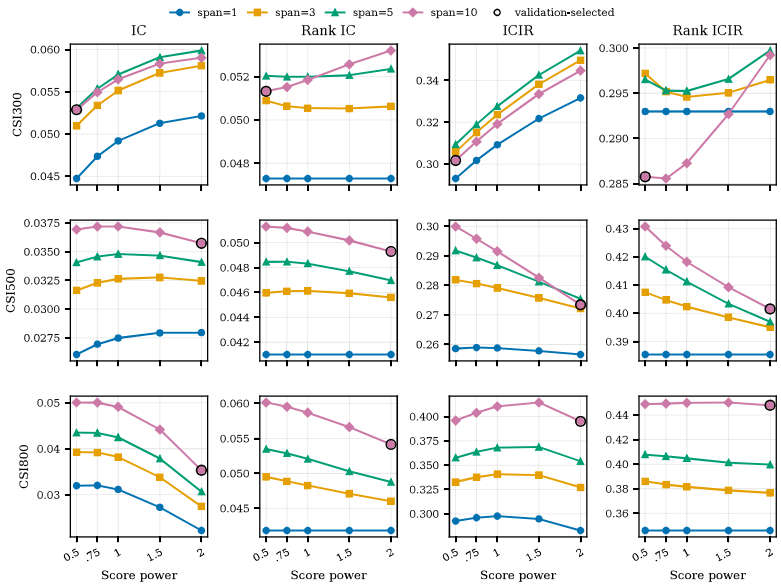

Figure 3: Sensitivity to signed score power and causal smoothing span on CSI300, CSI500, and CSI800. The columns report IC, Rank IC, ICIR, and Rank ICIR; the rows correspond to the three universes. Open circles denote the validation-selected setting. The test curves are reported only to show sensitivity around the frozen choice.

differs across market universes. CSI300 tends to benefit from a stronger score-power transformation for IC and ICIR when smoothing is applied, whereas several CSI500 and CSI800 metrics weaken as γ becomes large. The effect of the smoothing span is similarly dataset dependent, with longer spans improving selected CSI300 and CSI800 configurations but producing weaker ICIR behavior on CSI500. Importantly, the validation-selected settings do not always coincide with the highest values observed on the test curves; several CSI500 and CSI800 panels contain stronger test values away from the open-circle configuration. This separation supports the chronological calibration procedure and indicates that the displayed sensitivity results characterize robustness around a preselected setting rather than an additional search over the test data. Full numerical sensitivity results are reported in Appendix E.

5 Conclusion

We introduced AlphaSAS, a structure-aware neural-symbolic framework for automated alpha discovery that integrates mathematical expression structure, previous search outcomes, and factor-set information within a single stateful search process. Relation-specific graph encoding preserves typed and ordered operator relations, while canonicalization avoids repeated evaluation of equivalent expressions; the search history retains both successful and unsuccessful candidates so that subsequent proposals can use local estimates of historical quality, uncertainty, instability, and complexity; and the factor-diversity term evaluates whether a candidate contributes information beyond the factors already selected. These components are kept distinct from the subsequent statistical evaluation, where a candidate is fixed before later observations contribute to its sequential test process, thereby separating adaptive search from post-selection evidence. Across CSI300, CSI500, and CSI800, AlphaSAS achieves the highest cross-dataset average and median IC, ICIR, and RIC among the compared methods, while ranking second on RICIR, indicating that its gains are consistent across both linear and rank-based predictive measures rather than being concentrated in a single universe. The sensitivity analysis further shows that performance changes smoothly around the validation-selected score-power and smoothing configurations and that the frozen validation choices do not systematically coincide with the best values observed on the test curves. Together, these results support the central design of AlphaSAS: automated alpha discovery benefits from jointly modelling expression structure, search history, and factor-set context while maintaining a clear chronological separation between candidate discovery and subsequent statistical evaluation.

REFERENCES

David H. Bailey and Marcos López de Prado. The deflated sharpe ratio: Correcting for selection bias, backtest overfitting, and non-normality. *The Journal of Portfolio Management*, 40(5):94–107, 2014. doi: 10.3905/jpm.2014.40.5.094.

Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Caglar Gulcehre, Francis Song, Andrew Ballard, Justin Gilmer, George E. Dahl, Ashish Vaswani, Kelsey Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matthew Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018. URL https://arxiv.org/abs/1806.01261.

Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B*, 57(1):289–300, 1995. doi: 10.1111/j.2517-6161.1995.tb02031.x.

James S. Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems*, volume 24, pages 2546–2554, 2011.

Kevin Ellis, Lucas Morales, Mathias Sablé-Meyer, Armando Solar-Lezama, and Josh Tenenbaum. Learning libraries of subroutines for neurally-guided bayesian program induction. In *Advances in Neural Information Processing Systems*, volume 31, pages 7816–7826. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/hash/7aa685b3b1dc1d6780bf36f7340078c9-Abstract.html.

Kevin Ellis, Lionel Wong, Maxwell Nye, Mathias Sablé-Meyer, Luc Cary, Lore Anaya Pozo, Luke Hewitt, Armando Solar-Lezama, and Joshua B. Tenenbaum. Dreamcoder: Growing generalizable, interpretable knowledge with wake-sleep bayesian program learning. *Philosophical Transactions of the Royal Society A*, 381(2251):20220050, 2023. doi: 10.1098/rsta.2022.0050.

Stefan Falkner, Aaron Klein, and Frank Hutter. BOHB: Robust and efficient hyperparameter optimization at scale. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1437–1446. PMLR, 2018. URL https://proceedings.mlr.press/v80/falkner18a.html.

Dean P. Foster and Robert A. Stine. α-investing: A procedure for sequential control of expected false discoveries. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(2): 429–444, 2008. doi: 10.1111/j.1467-9868.2007.00643.x.

Peter Reinhard Hansen. A test for superior predictive ability. *Journal of Business & Economic Statistics*, 23(4):365–380, 2005. doi: 10.1198/073500105000000063.

Campbell R. Harvey and Yan Liu. False (and missed) discoveries in financial economics. *The Journal of Finance*, 75(5):2503–2553, 2020. doi: 10.1111/jofi.12951.

Campbell R. Harvey, Yan Liu, and Heqing Zhu. . . . and the cross-section of expected returns. *The Review of Financial Studies*, 29(1):5–68, 2016. doi: 10.1093/rfs/hhv059.

Steven R. Howard, Aaditya Ramdas, Jon McAuliffe, and Jasjeet Sekhon. Time-uniform, nonparametric, nonasymptotic confidence sequences. *The Annals of Statistics*, 49(2):1055–1080, 2021. doi: 10.1214/20-AOS1991.

Adel Javanmard and Andrea Montanari. Online rules for control of false discovery rate and false discovery exceedance. *The Annals of Statistics*, 46(2):526–554, 2018. doi: 10.1214/18-AOS1559.

Zura Kakushadze. 101 formulaic alphas. *Wilmott*, 2016(84):72–81, 2016. doi: 10.1002/wilm.10525.

John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, 1992.

Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9:235–284, 2008.

Alex Kulesza and Ben Taskar. Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning*, 5(2–3):123–286, 2012. doi: 10.1561/2200000044.

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*, 18(185):1–52, 2018. URL https://jmlr.org/papers/v18/16-558.html.

R. David McLean and Jeffrey Pontiff. Does academic research destroy stock return predictability? *The Journal of Finance*, 71(1):5–32, 2016. doi: 10.1111/jofi.12365.

George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming*, 14:265–294, 1978. doi: 10.1007/BF01588971.

Robert Novy-Marx and Mihail Velikov. A taxonomy of anomalies and their trading costs. *The Review of Financial Studies*, 29(1):104–147, 2016. doi: 10.1093/rfs/hhv063.

Brenden K. Petersen, Mikel Landajuela Larma, T. Nathan Mundhenk, Claudio P. Santiago, Soo K. Kim, and Joanne T. Kim. Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=m5Qsh0kBQG.

Dimitris N. Politis and Joseph P. Romano. The stationary bootstrap. *Journal of the American Statistical Association*, 89(428):1303–1313, 1994. doi: 10.1080/01621459.1994.10476870.

Aaditya Ramdas, Tijana Zrnic, Martin Wainwright, and Michael Jordan. SAFFRON: An adaptive algorithm for online control of the false discovery rate. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4286–4294. PMLR, 2018. URL https://proceedings.mlr.press/v80/ramdas18a.html.

Aaditya Ramdas, Peter Grünwald, Vladimir Vovk, and Glenn Shafer. Game-theoretic statistics and safe anytime-valid inference. *Statistical Science*, 38(4):576–601, 2023. doi: 10.1214/23-STS894.

Joseph P. Romano and Michael Wolf. Stepwise multiple testing as formalized data snooping. *Econometrica*, 73(4):1237–1282, 2005. doi: 10.1111/j.1468-0262.2005.00615.x.

Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *The Semantic Web – ESWC 2018*, volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer, 2018. doi: 10.1007/978-3-319-93417-4_38.

Hao Shi, Weili Song, Xinting Zhang, Jiahe Shi, Cuicui Luo, Xiang Ao, Hamid Arian, and Luis Angel Seco. AlphaForge: A framework to mine and dynamically combine formulaic alpha factors. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(12):12524–12532, 2025. doi: 10.1609/aaai.v39i12.33365. URL https://ojs.aaai.org/index.php/AAAI/article/view/33365.

Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, volume 25, pages 2951–2959, 2012.

Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias W. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proceedings of the 27th International Conference on Machine Learning*, pages 1015–1022, 2010.

Ryan Sullivan, Allan Timmermann, and Halbert White. Data-snooping, technical trading rule performance, and the bootstrap. *The Journal of Finance*, 54(5):1647–1691, 1999. doi: 10.1111/0022-1082.00163.

Ruodu Wang and Aaditya Ramdas. False discovery rate control with e-values. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 84(3):822–852, 2022. doi: 10.1111/rssb.12489.

Halbert White. A reality check for data snooping. *Econometrica*, 68(5):1097–1126, 2000. doi: 10.1111/1468-0262.00152.

Max Willsey, Chandrakana Nandi, Yisu Remy Wang, Oliver Flatt, Zachary Tatlock, and Pavel Panchekha. egg: Fast and extensible equality saturation. *Proceedings of the ACM on Programming Languages*, 5(POPL):1–29, 2021. doi: 10.1145/3434304. URL https://doi.org/10.1145/3434304.

Catherine Wong, Kevin M. Ellis, Joshua B. Tenenbaum, and Jacob Andreas. Leveraging language to learn program abstractions and search heuristics. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 11193–11204. PMLR, 2021. URL https://proceedings.mlr.press/v139/wong21a.html.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.

Shuo Yu, Hongyan Xue, Xiang Ao, Feiyang Pan, Jia He, Dandan Tu, and Qing He. Generating synergistic formulaic alpha collections via reinforcement learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5476–5486. ACM, 2023. doi: 10.1145/3580305.3599831. URL https://doi.org/10.1145/3580305.3599831.

Zhoufan Zhu and Ke Zhu. AlphaQCM: Alpha discovery in finance with distributional reinforcement learning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 80463–80479. PMLR, 2025. URL https://proceedings.mlr.press/v267/zhu25ag.html.