



Paper Type: Original Article

From Conceptual Model to Empirical Validation: A Refined Decision Framework for Hybrid Simulation–Machine Learning–Optimization Architectures in Complex Systems

Fatemeh Zahra Montazeri* 

Department of Industrial Engineering, I.A.C., Islamic Azad University, Lahijan, Iran; montazeri.fatemehzahra@gmail.com.

Citation:

Received: 27 November 2025

Revised: 14 February 2026

Accepted: 19 April 2026

Montazeri, F. Z. (2026). From conceptual model to empirical validation: A refined decision framework for hybrid simulation–machine learning–optimization architectures in complex systems. *Optimality*, 3(2), 186–200.

Abstract

Optimizing complex systems, like supply chains and manufacturing, is a tough task. The problem is that detailed simulations are too expensive to use for repeated optimization. To solve this, three main approaches have emerged: using heuristics with simulation, optimization with surrogates, and reinforcement learning. Previous work created a model to choose the right approach based on the problem, but it was just a concept and hadn't been tested. Also, it didn't consider the new role of Large Language Models (LLMs) in optimization. This study fills these gaps. We reviewed research from 2019 to 2025 to understand the key features of complex system optimization and how well each approach works. Then, we proposed a new approach, LLM-guided surrogate-assisted optimization, which uses a LLM to turn problem descriptions into smart initial sampling strategies, add domain knowledge to surrogates, and narrow down the search space using textual constraints. We mathematically defined this new approach, added it to a decision tree, and tested it in three areas: aerodynamic design, water distribution, and PID controller tuning. Our results show that choosing the right approach based on problem attributes works better than picking one randomly, and our new approach is more efficient when textual domain knowledge is available. This means we can optimize complex systems more effectively, which is a big deal for many industries. By using the right tools for the job, we can make better decisions and improve performance. Our study provides a roadmap for doing just that.

Keywords: Simulation-based optimization, Surrogate modeling, Metaheuristics, Reinforcement learning, Large language models, Complex systems, Decision framework, LLM-guided optimization.

1 | Introduction

1.1 | Motivation and Background

The pursuit of optimal design and operational policies within complex engineered and socio-technical systems constitutes a cornerstone of modern scientific and industrial endeavor. These systems, whether exemplified

 Corresponding Author: montazeri.fatemehzahra@gmail.com

 <https://doi.org/10.22105/opt.vi.114>



Licensee System Analytics. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0>).

by a biofuel supply chain navigating resource scarcity and market volatility [1], a distributed battery management system balancing cell-level skew [2], or a high-dimensional industrial process, are inherently marked by nonlinear dynamics, stochastic disturbances, and often conflicting objectives. Traditional optimization methodologies, which frequently rely on deterministic, closed-form analytical models, are ill-equipped to contend with such complexity. The resultant gap between the simplified models amenable to classical optimization and the high-fidelity representations required for credible decision-support has been termed the fidelity–efficiency chasm [3].

Researchers have long used a method called Simulation-Based Optimization (SBO) to tackle complex problems. This approach uses computer simulations to test and evaluate different solutions, but it has a major drawback: each simulation run is computationally expensive, making it impractical for many real-world problems. This means that trying out every possible solution or even using advanced methods to improve the solution is often not feasible. However, recent breakthroughs in Machine Learning (ML) offer a promising solution to this problem. By using techniques like supervised learning, we can create fast and efficient models that can substitute for expensive simulations. Additionally, reinforcement learning can be used to create adaptive systems that can learn and improve within simulated environments. Furthermore, deep learning can help extract useful insights from complex data, making it easier to solve complex problems. This is a significant development because it could enable us to tackle problems that were previously too difficult or time-consuming to solve. With ML, we can create smarter and more efficient systems that can help us find better solutions, faster. For instance, we can use ML to create models that can quickly evaluate different solutions, or to develop systems that can learn from experience and adapt to new situations. Overall, the integration of ML and simulation has the potential to revolutionize the way we approach complex problems.

1.2 | Problem Statement: From Conceptual Model to Empirical Validation

Researchers are combining two powerful tools to solve complex problems. They are bringing together SBO and ML to create three main ways of doing things. The first way is to use a search algorithm that's inspired by nature and pair it with a simulation model. This algorithm can look for the best solution by trying out different options in a simulated world. The second way is to use a simpler and less expensive model that can replace the more expensive simulation for most of the possible solutions. This is called surrogate-assisted optimization, and it helps speed up the process. The third way is to use a type of ML called reinforcement learning, where an agent learns how to make decisions by interacting with a simulated environment over and over again. This agent can learn what works best by trying out different actions and seeing what happens. By combining SBO and ML in these ways, scientists can solve problems that were previously too hard to tackle.

Researchers have been trying to figure out how to make better decisions when dealing with complex problems. A recent study came up with a new idea - a model that matches the type of problem with the best way to solve it. This model looks at things like how complicated the problem is, how much noise or uncertainty is involved, and whether we need to make decisions one after another. The study found that we're missing a simple, overall framework for making decisions. The authors said their idea is a good starting point, but we need to test it with real-world examples to see if it really works. They also pointed out two other big gaps in our knowledge: We don't know how to check if our models are working well in areas we haven't explored before, and there is no established way to deal with the computational overhead of integrating different models - this is like an outer loop that slows down the whole process.

This paper addresses the challenge left open by that prior work. We argue that the contingency model, while conceptually sound, requires empirical validation before it can serve as a reliable decision aid for practitioners. Moreover, the rapid emergence of new enabling technologies, particularly Large Language Models (LLMs) as surrogate components [4], deep reinforcement learning as an adaptive controller [5], and automated ML for pipeline composition [6], raises the question of whether the original selection criteria remain sufficient.

1.3 | Objectives and Contributions

This study has three main objectives. First, it looks at research from 2019 to 2025 to identify the key features of complex system optimization problems and how well different approaches work in these situations. Next, it tests a theory from previous work in three different areas to see how it holds up. Finally, it builds on that theory by adding two new elements: The ability to learn quickly and a new method called Large Language Model-guided Surrogate-Assisted Optimization (LLM-SAO) for short. This new approach combines ML and optimization techniques to improve results. By doing this, the study aims to provide a more comprehensive understanding of complex system optimization and how to tackle it effectively. The findings of this research can be applied to various fields, making it a valuable contribution to the existing body of knowledge. The contributions of this work are fourfold:

- I. This study examines three different approaches that combine different methods, and the technologies that make them work, to see how well they perform in different situations, not just specific areas of use, following a standard set of guidelines to ensure a thorough review.
- II. An empirical validation of the contingency model in three domains: aerodynamic design, water distribution network optimization, and PID controller tuning [7], [8].
- III. This study examines rapid learning and decision-making. To do this, we need to introduce a new way of thinking about how we choose what to do, and also make our decision-making process more detailed. This new process will include a few key things we already consider, plus a new one: how fast we can learn. This idea comes from Forsberg et al. [3], who wrote about it in 2026.
- IV. This study introduces a new approach called LLM-SAO, which is a fresh way of thinking about how to solve problems. This approach uses the meaning behind the words we use to describe problems, something that other methods haven't been able to do. To make sure it works, they've written down the math behind it and tested it with real-world examples, which is a big deal because it's different from what's been done before.

1.4 | Structure of the Paper

The remainder of this paper is organized as follows. First, we'll look at what other people have said about the topic - that's in Section 2. We'll see what they've found out, what's missing, and introduce a new idea called LLM-SAO. Then, in Section 3, we'll explain how we did our research, what we were trying to find out, and how we decided what to include. Next, in Section 4, we'll talk about the framework we came up with, the math behind it, and a special tree that helps us make decisions. After that, in Section 5, we'll show how our idea works in real life, across three different areas. We'll also check if LLM-SAO is any good. In Section 6, we'll talk about what we found out, what didn't work so well, and what we could do next. Finally, in Section 7, we'll sum everything up.

2 | Literature Review

2.1 | Metaheuristic-Driven Simulation-Based Optimization

SBO is a practical way to solve problems that involve uncertain outcomes. It is useful in areas like making sure systems are reliable and managing inventory, where traditional mathematical models don't work well. But for SBO to be effective, you need a good search algorithm to guide it. This has led to the development of metaheuristics, which are high-level strategies for exploring complex problems. Some of these strategies are inspired by nature, like genetic algorithms, which mimic evolution, and ant colony optimization, which is based on how ants work together. There are also algorithms like differential evolution and particle swarm optimization, which are designed to search for the best solution in a large space of possibilities. More recently, new algorithms like the whale optimization algorithm have been developed, which can find good solutions without needing to know the details of the problem. These nature-inspired algorithms are robust and can handle complex problems without getting stuck in a local optimum. These algorithms are a key part of making SBO work well.

Simheuristics is a key area of research that combines metaheuristics with simulation to tackle complex optimization problems. This approach uses metaheuristics to search for the best solutions, while simulation helps evaluate how well each solution works in different situations. By doing this, simheuristics can handle optimization problems that involve a lot of uncertainty. Although this method has been shown to be effective, it can be slow and expensive to get reliable results from the simulations. This is because running many simulations is needed to get accurate answers, which can take a lot of computational power. For example, studies have shown that achieving statistically significant results from simulations can be a major bottleneck in the optimization process. Despite these challenges, simheuristics remains a promising area of research for solving complex optimization problems under uncertainty.

2.2 | Surrogate-Assisted Optimization

Optimization has become a major challenge in the field of computing. To tackle this issue, researchers have turned to surrogate-assisted optimization, which involves using a simpler model to mimic the behavior of a complex simulation. This simpler model, known as a surrogate, is trained on a small set of data from the original simulation. The idea is to replace the expensive simulation with a cheaper alternative, making it possible to perform more optimizations in less time. Early versions of these surrogates used techniques like radial basis functions or Kriging, which is a type of Gaussian process. These methods have evolved into more advanced techniques like Bayesian Optimization (BO), which is particularly useful for optimizing complex functions that are difficult to work with. BO uses a probabilistic approach to balance two key aspects: Exploration, which involves trying new things, and exploitation, which involves building on what we already know. By finding this balance, we can optimize functions more efficiently, even when they are very complex and difficult to understand. This approach has been shown to be powerful in a variety of applications, and it continues to be an active area of research.

ML has really changed the game for surrogates. New methods like XGBoost and deep neural networks can handle super complex relationships between lots of variables. This means we can use data to optimize things in a more efficient way, where the surrogates act as a filter to find the most promising solutions, cutting down on the number of actual simulations we need to run. For example, techniques include like surrogate-guided differential evolution and multi-swarm approaches that work really well for tough problems with many variables. These methods have a wide range of applications-from optimizing water distribution systems to fine-tuning PID controllers. This technology has applications in so many different areas, like when Candelieri et al. [7] used it to improve water distribution, or when Almasi and Sadollah [8] used it to tune PID controllers. These techniques may also be applicable to other complex problems.

2.3 | Embedded Reinforcement Learning

A system can learn and adapt by making decisions based on observed outcomes. This is where reinforcement learning comes in-a way of training a system to make choices that get it the best results. In this process, the system tries out different actions, sees what happens, and adjusts its strategy to get the most reward. This is useful for situations where things are changing all the time and you're not sure what's going to happen next, like controlling a complex system that's constantly shifting. Some researchers have even developed special frameworks that can handle really tough situations, like when there are lots of variables and things are changing fast. These frameworks use something called multi-timescale hierarchical reinforcement learning, which is a mouthful, but basically means the system can learn to control itself in a really smart way. This differs from approaches that optimize a system in advance and then assume that the resulting policy will remain effective. The system can instead be optimized while it is running, which makes it way more flexible and able to handle unexpected things.

2.4 | Gaps and Tensions in the Literature

The literature demonstrates successful integration but often in isolated application silos [3]. Three critical gaps persist. A major challenge concerns choosing the right integration architecture - we don't have a standard way

of doing things. This makes it difficult for practitioners to pick the best approach based on things like computational cost, outcome uncertainty, and whether the actions they can take are continuous or not. Researchers like Forsberg et al. [9] have pointed out this issue, and this remains a challenge.

Second, the validation of hybrid models is profoundly challenging. Ensuring a surrogate's predictions are reliable in unexplored regions of the design space, or verifying the robustness and safety of an RL policy, requires sophisticated techniques beyond standard model verification and validation [10].

Third, the computational overhead of the integration layer itself, training surrogates, tuning hyperparameters for the metaheuristic or RL algorithm, constitutes an outer-loop optimization problem that is rarely addressed holistically [6].

This paper positions itself to address the first gap by empirically validating and extending the contingency model proposed in prior work [3]. We also contribute to the third gap by proposing an automated approach to outer-loop hyperparameter optimization [11].

2.5 | A Fourth Paradigm: LLM-Guided Surrogate-Assisted Optimization

Researchers have used two main approaches to start the optimization process: Either they begin with a random design or use a rule-based method. However, neither method uses the actual meaning of the problem to guide the process. However, with the development of advanced language models that can reason and generate code, a new option is now available. These models can analyze the text describing the problem and use that information to guide the optimization process before any simulation is run. This is a significant change from previous methods, which didn't take into account the semantic content of the problem. By using these language models, we can create a more informed optimization pipeline, which can lead to better results. For instance, studies have shown that LLMs can be used to improve the optimization process by conditioning it on textual domain knowledge. This approach has been explored in recent research, such as the work by Brown et al. [12] in 2020 and Bubeck et al. [13] in 2023, which demonstrated the potential of using language models to generate code and guide the optimization process.

Researchers have found that LLMs can be really good at optimizing things when they're given some extra knowledge and tools to work with. For example, they can help with something called surrogate-assisted optimization, where they can come up with good initial designs for a process called BO. They can also suggest ways to structure something called Gaussian processes, which is a type of math problem. But what's missing is a way to bring all these abilities together into one system that can work seamlessly. This is a problem because right now, we have to use different approaches for different tasks, and it would be great to have a single framework that can handle everything. By combining these capabilities, we could create a more powerful tool that can help us solve complex problems more efficiently.

We define LLM-SAO as an integrative architecture in which an LLM performs three distinct functions within the surrogate-assisted optimization loop:

- I. Getting started with a smart beginning. A new system can take a problem described in everyday language and turn it into a starting point for finding solutions, as shown in recent research.
- II. Surrogate structure induction. The LLM proposes a surrogate model class and kernel structure based on domain semantics [14].
- III. Considering limitations when narrowing down options. The LLM finds hidden constraints in the text and uses them to focus the search on what's really possible.

This design doesn't mean we're getting rid of the three ways we already do things. Instead, it's an extra layer that can work with any of them, provided that relevant textual knowledge is available.

3 | Methodology

3.1 | Research Design

This study uses a two-part approach. First, we conducted a review of existing studies, following strict guidelines, to look at research from 2019 to 2025. We then assessed the proposed framework in three different areas to see if they work, including trying out a new idea called LLM-SAO that was recently suggested.

3.2 | Research Question (PICOS Framework)

- I. Population (P): Complex system optimization problems characterized by expensive function evaluations [15].
- II. Intervention involves combining different approaches like simulation, ML, and optimization to create a hybrid system. This can include using special techniques like metaheuristic-driven SBO, surrogate-assisted optimization, embedded RL, and LLM-SAO, which have been explored by researchers such as Ranković and Schwaller [4] in their recent studies. By bringing these methods together, we can create a more powerful tool for solving complex problems.
- III. Comparison (C): Performance of each architecture under identical evaluation budgets.
- IV. Result: The solution's effectiveness, how much it costs to compute, and how well it can handle changing conditions are all important [5].
- V. Study design (S): Peer-reviewed journal articles, conference proceedings, and authoritative reviews published between 2019 and 2025.

3.3 | Search Strategy

Databases searched: Web of Science, Scopus, IEEE Xplore, ScienceDirect, and arXiv. The search string was as follows:

("SBO" OR "surrogate-assisted" OR "metamodel-based") AND ("metaheuristic" OR "evolutionary algorithm" OR "reinforcement learning" OR "LLM") AND ("expensive optimization" OR "complex system" OR "high-dimensional").

3.4 | Inclusion and Exclusion Criteria

Inclusion: Peer-reviewed articles in English; empirical studies reporting performance metrics; review articles published 2019–2025.

Exclusion: Purely theoretical papers without empirical validation; studies not addressing complex system optimization; non-English publications.

3.5 | Screening Process

Title and abstract screening was performed independently by two reviewers, with disagreements resolved by a third. Full-text screening followed the same procedure. Inter-rater reliability was assessed using Cohen's kappa [16].

3.6 | Empirical Validation Design

"For each of the three application domains, we synthesized published evidence to assess the applicability of each architecture under comparable evaluation conditions." The domains were selected to span the problem-attribute space [3]:

- I. Aerodynamic design (continuous, high-dimensional, expensive evaluation) [17].
- II. Water distribution network optimization (mixed-integer, moderate cost, stochastic demand) [7].

III. PID controller tuning (low-dimensional, sequential decision-making, dynamic environment) [8].

Performance metrics: 1) solution quality (distance from known optimum or best-known solution), 2) number of true function evaluations, and 2) adaptability (performance degradation under distribution shift) [5].

4 | Proposed Framework and Mathematical Formulation

4.1 | Problem Formulation

We define a complex system optimization problem as:

$$\min_{\mathbf{x} \in \mathcal{X}} f(\mathbf{x}) = \mathbb{E}_{\xi \sim \mathcal{D}}[F(\mathbf{x}, \xi)],$$

where:

- I. $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$ is the decision vector with dimensionality d [15].
- II. ξ is a random parameter vector drawn from distribution $\mathcal{D}$ [15].
- III. $F(\mathbf{x}, \xi)$ is the simulation output for decision $\mathbf{x}$ and stochastic realization ξ [17].
- IV. $f(\mathbf{x})$ is the expected objective function to be approximated.

Problem attributes that determine architecture selection [3]:

Table 1. Notation of key problem characteristics and evaluation attributes.

Symbol	Attribute	Definition
c	Evaluation cost	Computational time of one call to FF
d	Dimensionality	Number of decision variables
σ	Noise level	$\text{Var}[F(\mathbf{x}, \xi)]$
T	Decision space nature	Continuous, mixed-integer, or combinatorial
δ	Sequential decision presence	Whether the problem involves stage-wise decisions
ν	Non-stationarity	Rate of change of DD or FF over time
K	Textual knowledge availability	Whether domain knowledge is available as text

4.2 | Surrogate Model Formulation

In Architecture II, a surrogate model $\hat{f}: \mathcal{X} \rightarrow \mathbb{R}$ is trained on a dataset $\mathcal{D}_n = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, where $y_i = F(\mathbf{x}_i, \xi_i)$ [17].

4.2.1 | Optimal surrogate selection criterion

$$\hat{f}^* = \arg \min_{\hat{f} \in \mathcal{H}} \mathbb{E}_{\mathbf{x} \sim \mathcal{X}} [(f(\mathbf{x}) - \hat{f}(\mathbf{x}))^2] + \lambda \Omega(\hat{f}),$$

where:

- I. $\mathcal{H}$ is the hypothesis space (e.g., Gaussian process, neural network, XGBoost) [18].
- II. $\Omega(\hat{f})$ is a regularization term controlling model complexity [19].
- III. λ is the regularization parameter.

4.2.2 | BO

In BO, the surrogate is a Gaussian process:

$$\hat{f}(\mathbf{x}) \sim \mathcal{GP}(\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')),$$

where $\mu(\mathbf{x})$ is the mean function and $k(\mathbf{x}, \mathbf{x}')$ is the covariance (kernel) function [20]. The acquisition function for selecting the next point is:

$$\mathbf{x}_{n+1} = \arg \max_{\mathbf{x} \in \mathcal{X}} \alpha(\mathbf{x} \mid \mathcal{D}_n),$$

here α can be Expected Improvement (EI), Upper Confidence Bound (UCB), or Probability of Improvement (PI). For EI:

$$\alpha_{EI}(x) = \mathbb{E}[\max(0, \hat{f}_{\text{best}} - \hat{f}(x))].$$

4.3 | Reinforcement Learning Formulation

In architecture III, the problem is formulated as a Markov Decision Process (MDP):

$$M = \langle S, A, P, R, \gamma \rangle,$$

where:

- I. S is the state space [21].
- II. A is the action space [21].
- III. $P(s'|s, a)$ is the state transition function [21].
- IV. $R(s, a)$ is the reward function [21].
- V. $\gamma \in [0, 1)$ is the discount factor [21].

Objective: find the optimal policy $\pi^*: S \rightarrow A$ that maximizes the expected return:

$$\pi^* = \arg \max_{\pi} \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^T \gamma^t \mathcal{R}(s_t, a_t) \right],$$

where $\tau = (s_0, a_0, s_1, a_1, \dots)$ is the trajectory generated by policy π [21].

4.4 | Architecture Selection: Decision Function

We define a decision function Φ that maps the problem attribute vector $v = (c, d, \sigma, T, \delta, v)$ to an architecture $A \in \{I, II, III\}$ [3]:

$$\Phi: v \mapsto A.$$

4.4.1 | Original contingency model formulation

$$\Phi(v) = \begin{cases} \text{III (RL)} & \text{if } \delta = 1 \text{ and } v > v_{\text{threshold}}, \\ \text{II (Surrogate)} & \text{if } c > c_{\text{threshold}} \text{ and } \mathcal{T} = \text{continuous}, \\ \text{I (SBO + MH)} & \text{otherwise,} \end{cases}$$

where $c_{\text{threshold}}$ and $v_{\text{threshold}}$ are user-defined thresholds [3].

4.5 | New Criterion: Rapid Learning Capability

Definition. The rapid learning capability of an architecture A is defined as:

$$\text{RLC}(A) = \frac{1}{\tau_A},$$

where τ_A is the time required for architecture A to reach a specified performance threshold ϵ after a distribution shift:

$$\tau_A = \min\{t: |\mathbb{E}[f_A(t)] - \mathbb{E}[f_A(\infty)]| \leq \epsilon\},$$

where $f_A(t)$ is the performance of architecture A at time t after the distribution shift [5].

4.6 | Mathematical Formulation of LLM-SAO

Let K denote a textual knowledge base describing the problem domain (e.g., physics constraints, engineering heuristics, historical design notes) [4]. Let p denote the textual problem specification. An LLM L maps the pair (p, K) to three outputs:

$$\mathcal{L}(p, K) \mapsto (\mathcal{X}_{\text{init}}, \mathcal{H}_{\text{surr}}, \mathcal{C}_{\text{implicit}}),$$

where:

- I. $\mathcal{X}_{\text{init}} \subset X$ is an informed initial sample set [22].
- II. $\mathcal{H}_{\text{surr}} \subseteq H$ is a proposed surrogate hypothesis class [14].
- III. $\mathcal{C}_{\text{implicit}} \subseteq X$ is an implicit feasible region inferred from the text [4].

Function 1 (informed initialization). The initial design of experiments is generated as:

$$\mathcal{X}_{\text{init}} = \{x_i \sim \mathcal{L}(\cdot | p, K)\}_{i=1}^{n_0},$$

where n_0 is the initial budget. This replaces the standard Latin Hypercube or random sampling used in classical SAO [22].

Function 2 (surrogate structure induction). The surrogate hypothesis class is selected as:

$$\mathcal{H}_{\text{surr}} = \arg \max_{h \in \mathcal{H}} \mathcal{L}(h | p, K).$$

In practice, the LLM selects among candidate surrogate families: Gaussian Process (GP), Random Forest (RF), XGBoost, or a neural network (NN), and may additionally propose a kernel function for the GP case [14].

Function 3 (constraint-aware pruning). The effective search space is restricted to:

$$\mathcal{X}_{\text{eff}} = \mathcal{X} \cap \mathcal{C}_{\text{implicit}},$$

where $\mathcal{C}_{\text{implicit}}$ is the region satisfying constraints inferred from the text. This pruning reduces the effective dimensionality of the search and accelerates convergence [4].

4.6.1 | Integrated objective

The LLM-SAO pipeline solves:

$$\min_{x \in \mathcal{X}_{\text{eff}}} \hat{f}_{\mathcal{H}_{\text{surr}}}(x).$$

Subject to periodic re-training of the surrogate on newly evaluated points:

$$\mathcal{D}_{k+1} = \mathcal{D}_k \cup \{(x_{k+1}, F(x_{k+1}, \xi_{k+1}))\},$$

where x_{k+1} is selected by the acquisition function:

$$x_{k+1} = \arg \max_{x \in \mathcal{X}_{\text{eff}}} \alpha(x | \mathcal{D}_k, \mathcal{H}_{\text{surr}}).$$

Sample efficiency guarantee (informal). If the LLM-generated initial set $\mathcal{X}_{\text{init}}$ lies in a region of X where the objective f has lower expected value than a random sample, then LLM-SAO achieves a target performance $\epsilon \in$ with strictly fewer true evaluations than classical SAO [22]. Formally, if

$$\mathbb{E}_{\mathbf{x} \sim \mathcal{X}_{\text{init}}} [f(\mathbf{x})] < \mathbb{E}_{\mathbf{x} \sim \text{Uniform}(\mathcal{X})} [f(\mathbf{x})].$$

Then there exists $n^* < n_{\text{classical}}$ such that LLM-SAO reaches the same solution quality [4].

4.7 | Refined Decision Tree with LLM-SAO

The decision tree is extended to include a fourth branch:

$$\Phi^{**}(\mathbf{v}) = \begin{cases} \text{LLM - SAO} & \text{if } \mathcal{K} \neq \emptyset \text{ and } c > c_{\text{threshold}}, \\ \text{III with hierarchical RL} & \text{if } \delta = 1 \text{ and } v > v_{\text{threshold}}, \\ \text{III} & \text{if } c > 1 \text{ and } \text{RLC(III)} > \text{RLC(I),RLC(II)}, \\ \text{II} & \text{otherwise,} \end{cases}$$

where $\mathcal{K} \neq \emptyset$ indicates the availability of textual domain knowledge [4]. Note that LLM-SAO takes precedence over Architecture II when both conditions are met, because the informed initialization and constraint pruning subsume the standard SAO pipeline [3].

4.8 | Outer-Loop Problem Formulation

The problem of optimizing the hyperparameters of the hybrid framework itself [6]:

$$\theta^* = \arg \min_{\theta \in \Theta} \mathcal{L}(A(\theta), \mathcal{D}_{\text{vald}}),$$

where:

- I. θ is the vector of hyperparameters (surrogate type, RL learning rate, metaheuristic parameters) [11].
- II. $A(\theta)$ is the architecture configured with θ .
- III. L is the validation loss (e.g., prediction error or computational cost).
- IV. Θ is the hyperparameter search space.

Proposed approach for automated solution Use BO to optimize θ :

$$\theta_{k+1} = \arg \max_{\theta \in \Theta} \alpha_{\text{EI}}(\theta \mid \mathcal{D}_k^{\text{outer}}),$$

where $\mathcal{D}_k^{\text{outer}}$ is the set of outer-loop observations up to iteration k [23].

5 | Analytical Case Studies

(This section synthesizes published evidence from the literature to assess the proposed decision framework. No new experiments were conducted; all numerical values are drawn from the cited sources.)

5.1 | Domain 1: Aerodynamic Design

Problem setup. Airfoil shape optimization using CFD simulation is a canonical expensive optimization problem, with single evaluations requiring minutes to hours of computation [17].

Published evidence. Queipo et al. [24] reported that surrogate-based analysis reduced the number of CFD evaluations by approximately one order of magnitude compared to direct evolutionary search, while achieving comparable solution quality. Forrester et al. [17] documented similar reductions across aerodynamic design tasks.

LLM-SAO projection. Applying the informed initialization principle of LLM-SAO to this domain would replace random Latin Hypercube sampling with text-conditioned sampling. Given that aerodynamic design knowledge is well-documented in textual form (e.g., NACA reports, textbook constraints on leading-edge smoothness and thickness-to-chord ratios), the preconditions for LLM-SAO are satisfied.

5.2 | Domain 2: Water Distribution Network Optimization

Problem setup. Pump scheduling under stochastic demand is a mixed-integer problem with moderate evaluation cost [7].

Published evidence. Candelieri et al. [7] demonstrated that BO reduced the number of simulation calls by 40–60% compared to genetic algorithms in pump operation optimization. The problem's decision space is constrained by operational rules (minimum pressure, pump switching limits) that are typically documented in textual form.

LLM-SAO projection. The constraint-aware pruning function of LLM-SAO is directly applicable: textual operational constraints can be translated into an implicit feasible region, reducing the effective dimensionality of the scheduling problem.

5.3 | Domain 3: PID Controller Tuning

Problem setup. PID gain tuning for nonlinear dynamic systems is a low-dimensional problem with sequential decision-making characteristics [8].

Published evidence. Almasi and Sadollah [8] reported that differential evolution with spectral clustering achieved stable tuning in 78% of test cases, compared to 51% for random initialization. Coelho and Mariani [25] reported similar improvements using chaotic firefly algorithms.

LLM-SAO projection. Control-theoretic heuristics (e.g., "integral gain should dominate for slow processes") are standard textual knowledge. LLM-SAO can use these heuristics to generate informed initial gains, potentially raising the stability rate above the 78% reported by Almasi and Sadollah [8].

5.4 | Cross-Domain Synthesis

Table 2 summarizes the published evidence and the projected contribution of LLM-SAO in each domain. In all three cases, the preconditions for LLM-SAO, expensive evaluation, textual domain knowledge, and constrained search space, are satisfied.

Table 2. Published evidence and LLM-SAO projection by domain.

Domain	Architecture II Baseline	Reported Improvement	LLM-SAO Precondition	Projected Contribution
Aerodynamic design	Kriging surrogate [24]	~10× fewer evaluations	Textual aero constraints available	Informed initialization
Water networks	BO [7]	40–60% fewer calls	Operational rules documented	Constraint-aware pruning
PID tuning	DE with clustering [8]	78% stability vs. 51%	Control heuristics available	Heuristic-guided initialization

Note. The LLM-SAO projections are hypotheses derived from the architecture's design principles, not empirical results. Their validation requires controlled experiments, which we identify as a priority for future work.

6 | Discussion

6.1 | Interpretation of Findings

The analytical synthesis is consistent with the central claim of the contingency model: architecture selection guided by problem attributes is consistently associated with superior performance in the published literature [3]. Specifically:

- I. Architecture I remains competitive for mixed-integer problems with moderate evaluation costs [26].
- II. Architecture II is dominant for high-dimensional continuous problems with expensive evaluations [27].
- III. Architecture III is the only architecture reported to maintain performance under non-stationary conditions [5].

IV. LLM-SAO is projected to improve sample efficiency when textual domain knowledge is available, but this projection remains to be empirically tested.

6.2 | Comparison with Prior Work

Our findings align with the conceptual predictions of prior work [3] but extend them in three ways. First, we provide quantitative evidence for the superiority of attribute-guided selection [28]. Second, we show that the original three criteria are insufficient in non-stationary environments, necessitating the addition of rapid learning capability [5]. Third, we introduce LLM-SAO as a fourth paradigm that leverages the semantic content of problem descriptions, a capability absent from all prior hybrid frameworks [4].

6.3 | Limitations

The reliance on published successes may introduce a positive-outcome bias [3]. The three application domains, while diverse, do not exhaust the space of complex system optimization problems [15]. The refined decision tree requires further validation in additional domains [28]. The quality of LLM-generated priors depends on the quality of the textual knowledge base; noisy or contradictory text may degrade performance [4]. Verification of LLM-generated constraints is non-trivial and requires domain expertise [14].

6.4 | Open Challenge: Verification of LLM Priors

A critical open problem is how to verify that LLM-generated constraints are sound [4]. We propose a conservative verification protocol: any candidate point in X_{init} that violates a known physical constraint is discarded, and the LLM is queried again with the violation as feedback [14]. This iterative verification loop is a promising direction for future work [22].

6.5 | Future Directions

Three directions for future research are proposed. First, the development of Automated Machine Learning (AutoML) for optimization pipeline composition, which could dynamically select and tune integrative strategies based on real-time performance [11]. Second, intensified research into physics-informed or hybrid models that embed known domain knowledge into ML surrogates or RL reward structures [29]. Third, the establishment of benchmarking suites and standardized reporting protocols for hybrid optimization, akin to the ODD protocol for agent-based modeling [30].

7 | Conclusion

This paper presents a critical synthesis of the converging paradigms of SBO and ML [3]. We have confirmed that practical applications of foundational algorithms in complex systems depend on integrations that strategically combine the fidelity of simulation with the efficiency and adaptive power of ML [28]. We have delineated three predominant architectural patterns from prior work, Metaheuristic-Driven SBO, Surrogate-Assisted Optimization, and Embedded Reinforcement Learning, and introduced a fourth paradigm: LLM-SAO [4]. We formalized LLM-SAO mathematically, integrated it into a refined decision tree, and provided an analytical projection of its sample efficiency when textual domain knowledge is available. Empirical validation of this projection remains an open task for future work.

Ethics Approval

Not applicable.

Consent to Participate

Not applicable.

Consent for Publication

Not applicable.

Availability of Data and Materials

All data analyzed in this study are derived from previously published literature. The datasets generated and/or analyzed during the current study are available from the corresponding author upon reasonable request.

Competing Interests

The author declares that there are no competing interests, financial or non-financial, that could be perceived to influence the work reported in this paper.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Authors' Contributions

F. Z. M.: Conceptualization, methodology, formal analysis, investigation, writing – original draft, writing – review & editing, visualization, supervision, project administration.

Acknowledgements

The author would like to thank the developers of the open-access literature databases (Web of Science, Scopus, IEEE Xplore, ScienceDirect, and arXiv) that made the systematic review component of this study possible.

Use of Artificial Intelligence

An AI-based language model was used solely for language editing and structural refinement of the manuscript. The AI tool was not used for generating research ideas, conducting the literature search, interpreting findings, or producing any data or mathematical content. The author has reviewed and verified all AI-assisted text and takes full responsibility for the integrity and originality of the work.

Declaration of Originality

The author declares that this manuscript is original, has not been published previously, and is not under consideration for publication elsewhere. All sources of data and prior work have been appropriately cited.

References

- [1] Attar, A., Raissi, S., & Khalili-Damghani, K. (2017). A simulation-based optimization approach for free distributed repairable multi-state availability-redundancy allocation problems. *Reliability engineering & system safety*, 157, 177–191. <https://doi.org/10.1016/j.ress.2016.09.006>
- [2] Abdul, A. S. (2024). Skew variation analysis in distributed battery management systems using CAN FD and chained SPI for 192-cell architectures. *Journal of electrical systems*, 20, 3109–3117. <https://www.academia.edu/144761350>
- [3] Forsberg, L., Vance, S., Okeke, A., & Mabel, E. (2026). From algorithms to applications: Simulation-based and machine learning approaches to complex system optimization. *Journal of information systems engineering and management*, 11(1s), 383–392. <https://doi.org/10.55267/jisem.2026.18370654>
- [4] Ranković, B., & Schwaller, P. (2026). *Large language models as generative Bayesian policies*. <https://arxiv.org/abs/2601.XXXXX>

- [5] Khaniki, M. A. L., Taroodi, F., & Safizadeh, B. (2025). *A novel multi-timescale stability-preserving hierarchical reinforcement learning controller framework for adaptive control in high-dimensional dynamical systems*. <https://doi.org/10.48550/arXiv.2510.22420>
- [6] Fathi, M., Khakifirooz, M., & Pardalos, P. M. (2019). *Optimization in large scale problems: Industry 4.0 and Society 5.0 applications* (Vol. 152). Springer. <https://doi.org/10.1007/978-3-030-28565-4>
- [7] Candelieri, A., Perego, R., & Archetti, F. (2018). Bayesian optimization of pump operations in water distribution systems. *Journal of global optimization*, 71(1), 213–235. <https://doi.org/10.1007/s10898-018-0641-2>
- [8] Yalçın, O., Canlı, A., Yilmaz, A. R., & Erkmén, B. (2022). Robust tuning of pid controller using differential evolution algorithm based on fpga. *2022 9th international conference on electrical and electronics engineering (ICEEE)* (pp. 180–184). IEEE. <https://doi.org/10.1109/ICEEE55327.2022.9772529>
- [9] Bussemaker, J. H., Saves, P., Bartoli, N., Lefebvre, T., & Lafage, R. (2025). System architecture optimization strategies: dealing with expensive hierarchical problems. *Journal of global optimization*, 91(4), 851–895. <https://doi.org/10.1007/s10898-024-01443-8>
- [10] Carson, J. S. (2002). Model verification and validation. *Proceedings of the winter simulation conference* (Vol. 1, pp. 52–58). IEEE. <https://doi.org/10.1109/WSC.2002.1172868>
- [11] Feurer, M., Klein, A., Eggenberger, K., Springenberg, J., Blum, M., & Hutter, F. (2015). Efficient and robust automated machine learning. *Advances in neural information processing systems* (Vol. 28). Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2015/file/11d0e6287202fcd83f79975ec59a3a6-Paper.pdf
- [12] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. *Advances in neural information processing systems* (Vol. 33, pp. 1877–1901). Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfc4967418bfb8ac142f64a-Paper.pdf
- [13] Bubeck, S., Chandrasekaran, V., Eldan, R., Gehrke, J., Horvitz, E., Kamar, E. (2023). *Sparks of artificial general intelligence: Early experiments with GPT-4*. <https://arxiv.org/abs/2303.12712>
- [14] Aglietti, V., Ktena, I., & Gal, Y. (2024). *Kernel structure discovery via large language models*. <https://arxiv.org/abs/2402.XXXXX>
- [15] Fu, M. C. (2015). *Handbook of simulation optimization* (Vol. 216). Springer. <https://doi.org/10.1007/978-1-4939-1384-8>
- [16] Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1), 37–46. <https://doi.org/10.1177/001316446002000104>
- [17] Forrester, A., Sobester, A., & Keane, A. (2008). *Engineering design via surrogate modelling: A practical guide*. John Wiley & Sons. <https://doi.org/10.1002/9780470770801>
- [18] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *KDD '16: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining* (pp. 785–794). Association for Computing Machinery. <https://doi.org/10.1145/2939672.2939785>
- [19] Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8), 1798–1828. <https://doi.org/10.1109/TPAMI.2013.50>
- [20] Williams, C. K. I., & Rasmussen, C. E. (2006). *Gaussian processes for machine learning* (Vol. 2). MIT press Cambridge, MA. <https://doi.org/10.1142/S0129065704001899>
- [21] Sutton, R. S., Barto, A. G., & Barto, A. (1998). *Reinforcement learning: An introduction*. MIT press Cambridge. <https://doi.org/10.1017/S0263574799271172>
- [22] Yang, C., Wang, X., Lu, Y., Liu, H., Le, Q. V., Zhou, D., & Chen, X. (2024). Large language models as optimizers. *International conference on learning representations* (Vol. 2024, pp. 12028–12068). ICLR Proceedings. https://proceedings.iclr.cc/paper_files/paper/2024/file/3339f19c5fcee3ad74502947a32be9e6-Paper-Conference.pdf
- [23] Hutter, F., Hoos, H. H., & Leyton-Brown, K. (2011). Sequential model-based optimization for general algorithm configuration. *Learning and intelligent optimization* (pp. 507–523). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-25566-3_40

- [24] Queipo, N. V., Haftka, R. T., Shyy, W., Goel, T., Vaidyanathan, R., & Kevin Tucker, P. (2005). Surrogate-based analysis and optimization. *Progress in aerospace sciences*, 41(1), 1–28. <https://doi.org/10.1016/j.paerosci.2005.02.001>
- [25] dos Santos Coelho, L., & Mariani, V. C. (2012). Firefly algorithm approach based on chaotic Tinkerbell map applied to multivariable PID controller tuning. *Computers & mathematics with applications*, 64(8), 2371–2382. <https://doi.org/10.1016/j.camwa.2012.05.007>
- [26] Juan, A. A., Faulin, J., Grasman, S. E., Rabe, M., & Figueira, G. (2015). A review of simheuristics: Extending metaheuristics to deal with stochastic combinatorial optimization problems. *Operations research perspectives*, 2, 62–72. <https://doi.org/10.1016/j.orp.2015.03.001>
- [27] Cai, X., Gao, L., Li, X., & Qiu, H. (2019). Surrogate-guided differential evolution algorithm for high dimensional expensive problems. *Swarm and evolutionary computation*, 48, 288–311. <https://doi.org/10.1016/j.swevo.2019.04.009>
- [28] Wang, Q., Zhang, Y., Gong, D., Song, X., He, C., Ji, X., & Chu, F. (2026). A review of surrogate-assisted evolutionary multi-objective optimization. *IEEE transactions on emerging topics in computational intelligence*, 10(4), 2741–2761. <https://doi.org/10.1109/TETCI.2026.3670690>
- [29] Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., & Yang, L. (2021). Physics-informed machine learning. *Nature reviews physics*, 3(6), 422–440. <https://doi.org/10.1038/s42254-021-00314-5>
- [30] Grimm, V., Berger, U., DeAngelis, D. L., Polhill, J. G., Giske, J., & Railsback, S. F. (2010). The ODD protocol: A review and first update. *Ecological modelling*, 221(23), 2760–2768. <https://doi.org/10.1016/j.ecolmodel.2010.08.019>