Can Large Language Models Find the Design Point? Benchmarking LLM-Based MPP Search in Structural Reliability

Jafar Jafari-Asl
Faculty of Agriculture, Civil and Environmental Engineering, University of Rostock, 18059 Rostock, Germany (jafar.jafariasl@uni-rostock.de)

Lukáš Novák
Brno University of Technology, Faculty of Civil Engineering, Veveří 331/95, 602 00 Brno, Czechia (lukas.novak4@vut.cz)

Matthias G.R. Faes
Chair for Reliability Engineering, TU Dortmund University, Leonhard-Euler-Straße 5, Dortmund 44227, Germany (matthias.faes@tu-dortmund.de)

Panagiotis Spyridis
Faculty of Agriculture, Civil and Environmental Engineering, University of Rostock, 18059 Rostock, Germany (panagiotis.spyridis@uni-rostock.de)

Abstract. Finding the Most Probable Point (MPP), or design point, is a critical step in gradient-based structural reliability methods. Classical solvers such as HL-RF and iHL-RF locate the MPP reliably but often require a large number of limit state function (LSF) evaluations, which becomes prohibitive when the LSF involves expensive computations such as finite element analysis. This paper investigates whether a large language model (LLM) can serve as an autonomous algorithm designer for MPP search, generating problem-adapted search routines from a structured natural language prompt without any task-specific training. The proposed framework, LLMMPP, queries the LLM once per problem and executes the returned algorithm directly on the LSF. The framework leverages DeepSeek-V3, a state-of-the-art open-weight LLM, to generate problem-adapted search routines from a structured natural language prompt without any task-specific training. Experiments on four benchmark problems from the structural reliability literature show that LLMMPP reduces the number of LSF evaluations by a factor of 2.1 to 28.9 relative to the best classical method, while maintaining comparable or superior accuracy in the identified design point. These results indicate that LLMs can be used as efficient, zero-shot MPP optimizers, particularly for nonlinear, multi-modal, and high-dimensional performance functions.

Keywords: Large Language Models, Most Probable Point , Structural Reliability Analysis , Zero-Shot Algorithm Generation , Deepseek-V3, Design Point Search .

1. Introduction

In structural engineering, reliability analysis within a probabilistic framework are essential for designing highly reliable structures under uncertainties. Numerous methods have been developed for these purposes, including approximate analytical methods (e.g., first-order reliability method (FORM) (Du and Chen, 2001) and second-order reliability method (SORM) (Yoo et al., 2014)), advanced Monte Carlo simulation techniques (e.g., importance sampling (Schuëller and Stix, 1987), subset simulation (Au and Beck, 2001), line sampling (Koutsourelakis et al., 2004)), and active learning combined with surrogate models (Echard et al., 2011). Despite these advances, developing more efficient methods for various types of reliability problems remains an ongoing challenge (Jafari-Asl et al., 2025; Seghier et al., 2022).

The concept of the design point is central to structural reliability analysis . This point, defined as the location in the failure domain with the highest probability density, is particularly important in low to medium dimensional problems because the region surrounding it contributes most to the failure probability. Many reliability methods rely on accurate estimation of the design point: FORM and SORM use Taylor expansion at the design point; and line sampling defines the important direction via the design point. Estimating the design point is a constrained optimization problem. Over the past years, gradient-based methods have dominated, starting with the Hasofer–Lind–Rackwitz–Flessler (HLRF) algorithm (Hasofer and Lind, 1974). Improvements like quadratic search strategies have followed. Gradient-free methods exist (e.g., interpolation-based adaptive response surfaces, stochastic simulation with uniform sampling, probabilistic CDF-based methods with sequential quadrature, and meta-heuristic global optimization algorithms), but they often require hundreds or thousands of function evaluations, making them unsuitable for expensive simulations. Overall, estimating the design point remains challenging when the limit state function (LSF) is highly nonlinear, multi-modal, lacks gradient information, or when the design point lies far from the distribution center (Breitung, 2021). Addressing this gap is an urgent research need.

Large language models (LLMs) have emerged as a major breakthrough in generative AI. Built on the transformer architecture with self-attention mechanisms, LLMs are trained on massive text data and exhibit exceptional language understanding and generation. Their key capability is in-context learning (ICL), which allows them to adapt to new tasks using only a few examples or simple instructions, without requiring fine-tuning or large labeled datasets.

Recent studies have demonstrated the potential of LLMs across a broad range of optimization and engineering tasks, including meta-learning via in-context tuning (Chen et al., 2022), evolutionary combinatorial optimization (Liu et al., 2024), Bayesian optimization for engineering design (Yin et al., 2024), hyperparameter tuning (Zhang et al., 2023), automatic formulation and solving of linear programs (Liu et al., 2025), and zero-shot search in evolutionary algorithms (Guo et al., 2024). Collectively, these works suggest that LLMs can function as effective algorithm designers when provided with structured problem context, without requiring task-specific training or fine-tuning.

Structural reliability analysis presents a particularly suitable setting for this capability. The MPP search problem has a well-defined mathematical structure, a clear convergence criterion, and a known solution space geometry that can be communicated through a concise prompt. Building on this observation, the present work evaluates whether a large language model can autonomously

generate an efficient MPP search algorithm when supplied with problem-specific context, and benchmarks its performance against established classical and meta-heuristic methods.

2. Research materials

2.1. Problem statement

In structural reliability analysis, the performance of an engineering system is characterized by a LSF $g(\mathbf{x})$, where $\mathbf{x} \in \mathbb{R}^n$ denotes a vector of n random input variables. The system is considered to be in a failure state when $g(\mathbf{x}) < 0$, and in a safe state otherwise. The randomness of $\mathbf{x}$ is governed by its joint probability density function (PDF) $\phi_n(\mathbf{x})$. Throughout this work, the input variables are assumed to be independent and identically distributed standard Gaussian random variables. For cases involving dependent or non-Gaussian variables, the Nataf or Rosenblatt transformation can be employed to map the original variables into the standard normal space (Song et al., 2024). Under this assumption, the failure domain is expressed as:

$$F = \{\mathbf{x} : g(\mathbf{x}) < 0\}, \tag{1}$$

with its associated indicator function $I_F(\mathbf{x})$ taking unity within F and zero elsewhere. The probability of failure p_f is then defined as:

$$p_f = \int_F \phi_n(\mathbf{x})\,\mathrm{d}\mathbf{x} = \int_{\mathbb{R}^n} I_F(\mathbf{x})\,\phi_n(\mathbf{x})\,\mathrm{d}\mathbf{x}. \tag{2}$$

Accurate evaluation of p_f is particularly challenging when the LSF exhibits strong nonlinearity or when p_f is very small (e.g., $p_f \leq 10^{-3}$). To address this challenge, many reliability methods rely on the concept of the *design point* (also referred to as the Most Probable Point, MPP). From a probabilistic standpoint, the design point is the point within the failure domain F that possesses the highest PDF value. In the standard Gaussian space, this is equivalent to the point on the limit state surface $g(\mathbf{x}) = 0$ that lies closest to the origin:

$$\mathbf{x}_D = \arg\min_{g(\mathbf{x})=0} \|\mathbf{x}\|, \tag{3}$$

where $\|\cdot\|$ denotes the Euclidean norm. The scalar quantity $\beta = \|\mathbf{x}_D\|$ is referred to as the *reliability index*, and the negative of this distance, $-\|\mathbf{x}\|$, may equivalently serve as the optimization objective. It is worth noting that the choice between maximizing the PDF or minimizing the distance to the origin leads to equivalent solutions and does not significantly affect the search procedure; however, it may influence the tuning of algorithm parameters. In this study, Eq. **??** is adopted as the cost function. This formulation remains valid provided that p_f is not excessively large, specifically $p_f \leq 0.1$.

Most existing techniques for solving the aforementioned optimization problem rely on gradient-based approaches. However, their effectiveness diminishes when the optimization problem exhibits multi-modality or when gradient information is unavailable or computationally expensive to obtain.

Consequently, design point-based reliability analysis methods become impractical for many engineering problems. Furthermore, in the presence of a LSF with multi-modal behavior, these methods often lack global convergence. These challenges motivate the development of a novel optimization algorithm based on DeepSeek-V3, namely LLMMPP, combined with a generative search strategy, which is briefly described in the following subsection.

2.2. Background of Large Language Models

Large Language Models (LLMs) are deep neural networks trained on large-scale text corpora, capable of generating contextually coherent text in response to natural language prompts. The architecture underlying most modern LLMs is the transformer (Vaswani et al., 2017), which processes input sequences through a self-attention mechanism:

$$H = \text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V, \tag{4}$$

where $Q \in \mathbb{R}^{n\times d_k}$, $K \in \mathbb{R}^{m\times d_k}$, and $V \in \mathbb{R}^{m\times d_v}$ are the query, key, and value matrices, respectively, and d_k is the key dimension used for scaling. This mechanism allows the model to capture long-range dependencies across input tokens in parallel, overcoming the sequential bottleneck of earlier recurrent architectures.

A key capability of LLMs relevant to this work is *in-context learning* (ICL) (Dong et al., 2024): without any parameter updates, the model infers task-specific patterns directly from the information embedded in the prompt, and generates responses that conform to the demonstrated structure. When the prompt is carefully engineered to encode domain knowledge, algorithmic constraints, and problem-specific context, the LLM functions as a zero-shot algorithm designer — producing executable code that reflects the provided specifications without prior task-specific training.

3. The proposed methods

3.1. Overall Framework

The proposed framework, denoted **LLMMPP** (Large Language Model Most Probable Point), uses a single LLM query to generate a problem-adapted MPP search algorithm at runtime. Given a limit state function g of dimension D, the framework proceeds as follows:

1. **Context construction.** A structured prompt is assembled that encodes (i) the problem dimension D, (ii) the estimated reliability index $\hat{\beta}$, (iii) the sign of $g(\mathbf{0})$, indicating whether the origin lies in the safe or failure domain, and (iv) a function-call budget $N_{\max}$. A representative example of this prompt is shown in Figure **??**.

2. **Algorithm generation.** The structured prompt is submitted to DeepSeek-V3 via a single API call (temperature $T = 0.6$, one query per problem). The model returns a self-contained Python class `MPPFinder` that implements an MPP search algorithm tailored to the given problem

LLaMMPP — Structured Prompt Template

```
/* ROLE */
You are a structural reliability expert. Find the MPP
with the minimum number of g-function calls.
MPP = argmin ‖u‖ s.t. g(u) = 0   (standard normal space, D = {dim})
/* PROBLEM CONTEXT */
beta_hint  = {β̂}   ← start AT this distance from origin
g(origin)  = {g(0)}   ({safe or failure region})
Budget     = {N_max} calls   Target: converge in < {target} calls
/* KEY INSIGHT */
HL-RF from origin   → many steps to travel distance β
HL-RF from u0 ≈MPP  → only 3--8 steps to converge
Start at u0 = β̂ · e_i, NOT at origin!
/* SEARCH STRATEGY */
Step 1: Smart starts: u0 = ±β̂ e_i  (i = 1...min(D,3)); u0 = ±β̂/√D · 1
Step 2: HL-RF iteration from each start
∇g_j = [g(u + h e_j) − g(u)]/h     [h = 10^−5]
u_new = (∇g · u − g(u))/‖∇g‖^2 · ∇g
Step 3: Early exit when |g(u)| < 10^−3
Step 4: Accept if |g(u)| < 0.05 and ‖u‖ > 0.01; keep minimum-norm candidate
/* BUDGET GUIDELINES */
D = 2:  4 starts × 6 steps ≈ 60 calls
D = 10:  3 starts × 8 steps ≈ 120 calls (gradient costs D + 1 per step)
D = 50:  2 starts × 4 steps ≈ 120 calls (gradient costs 51 per step)
/* REQUIRED INTERFACE */
class MPPFinder:
def __init__(self, dim, budget): ...
def __call__(self, g_func): #Returns: (mpp: ndarray(dim), n_calls: int)
/* OUTPUT */ Respond ONLY with a complete self-contained MPPFinder class.
Improve with: line search, conjugate gradient, adaptive damping.
```

Figure 1. Structured prompt supplied to the LLM at runtime. Entries in curly braces are filled from the problem context: problem dimension D, estimated reliability index $\hat{\beta}$, sign of $g(\mathbf{0})$, and function-call budget $N_{\max}$.

context. No few-shot examples or task-specific demonstrations are provided; the model relies exclusively on its pre-trained knowledge of numerical optimization and structural mechanics.

3. **Execution and validation.** The generated algorithm is executed against the actual LSF g. The search terminates under two conditions: an *early-exit* criterion, triggered when $|g(\mathbf{u})| < 10^{-3}$, which halts the iteration as soon as tight convergence is achieved; and an *acceptance* criterion, which considers any candidate point $\hat{\mathbf{x}}_D$ a valid design point if $|g(\hat{\mathbf{x}}_D)| < \varepsilon$, where $\varepsilon = 0.05$.

If the generated algorithm terminates without satisfying the acceptance criterion, a fallback HL-RF solver is invoked automatically to ensure convergence.

The output of the framework is the identified design point $\hat{\mathbf{x}}_D$ and the corresponding reliability index $\hat{\beta} = \|\hat{\mathbf{x}}_D\|$. Estimation of the failure probability p_f from this design point is not addressed in the present work and is left as a direction for future research, as discussed in Section 2.

3.2. Prompt Engineering Strategy

The effectiveness of the framework relies critically on the prompt design. Three principles guide its construction:

(1) Smart initialization hint. Classical gradient-based methods (e.g., HL-RF) typically start from the origin and require many iterations to reach $\mathbf{x}_D$, since the MPP lies at distance β from the origin. The prompt explicitly instructs the LLM to initialize the search at $\mathbf{u}_0 = \hat{\beta}\,\mathbf{e}_i$ (unit vectors scaled by $\hat{\beta}$), substantially reducing the number of required iterations.

(2) Early-exit criterion. The prompt enforces an early termination rule: once $|g(\mathbf{u})| < 10^{-3}$, the current iterate is returned immediately, preventing unnecessary evaluations after tight convergence.

(3) Budget-aware decomposition. The total call budget $N_{\max}$ is distributed among multiple starting points. For a problem of dimension D, each gradient computation requires $D{+}1$ LSF evaluations; the prompt instructs the model to account for this cost when selecting the number of starting points and maximum iterations per chain.

3.3. Adaptive Budget Allocation

The call budget $N_{\max}$ is set adaptively based on the problem characteristics:

$$N_{\max} = \begin{cases} 200 & \text{if } D \geq 50, \\ 150 & \text{if } \hat{\beta} > 5, \\ 150 & \text{if problem-specific hint is provided,} \\ 100 & \text{otherwise.} \end{cases} \tag{5}$$

This ensures that high-dimensional or large-β problems receive sufficient computational resources, while simpler problems are solved with minimal overhead.

3.4. Zero-Shot Operation

A key distinction of LLMMPP from existing LLM-assisted optimization methods is its *zero-shot* operation: no solved reliability examples are provided to the model. The LLM leverages its pre-trained knowledge of numerical optimization and structural mechanics to generate algorithms that respect the HL-RF convergence pattern without any task-specific demonstrations. This makes the framework immediately applicable to new problems without any data collection or fine-tuning phase.

4. Results and Discussion

This section presents and analyzes the performance of the proposed LLaMMPP framework for identifying the MPP in structural reliability analysis. All experiments reported in this section were conducted using DeepSeek-V3 as the backbone LLM. For each benchmark problem, a single structured prompt was submitted to the model via the official DeepSeek API. The generated `MPPFinder` class was executed directly on the corresponding LSF without any manual modification. API response times ranged from 12 to 21 seconds per query, which is negligible relative to the cost of a finite element-based LSF evaluation but would constitute a meaningful overhead if the LSF itself runs in milliseconds.

The results are compared with five classical and meta-heuristic methods, namely HL-RF (Hasofer and Lind, 1974), iHL-RF (Zhang and Der Kiureghian, 1995), Sequential Least Squares Programming (SLSQP) (Kraft, 1988), Particle Swarm Optimization (PSO) (Kennedy and Eberhart, 1995), and Genetic Algorithm (GA) (Holland, 1992), across four benchmark problems.

4.1. Benchmark Problems

Four benchmark problems are selected to assess the proposed method, covering a range of complexity, dimensionality, and known failure modes of existing reliability methods (Ameryan et al., 2022).

Case I: Product LSF

$$g(\mathbf{X}) = X_1 X_2 - 146.14 \tag{6}$$

Both variables are normally distributed: $X_1 \sim \mathcal{N}(7.806 \times 10^4,\ 1.171 \times 10^4)$ and $X_2 \sim \mathcal{N}(1.04 \times 10^{-2},\ 1.56 \times 10^{-3})$. The failure probability is extremely small and SORM fails to provide a solution, making this a suitable test for rare-event reliability.

Case II: Metaball function with multiple MPPs

$$g(\mathbf{X}) = \frac{30}{\left(\dfrac{4(x_1+2)^2}{9} + \dfrac{x_2^2}{25}\right)^2 + 1} + \frac{20}{\left(\dfrac{(x_1-2.5)^2}{4} + \dfrac{(x_2-0.5)^2}{25}\right)^2 + 1} - 5 \tag{7}$$

Both variables are independent standard normal. The performance function contains multiple disjoint failure regions. Due to its topological complexity, many existing reliability analysis methods fail to reach the correct result on this benchmark.

Case III: Parallel system with two failure modes

$$g(\mathbf{X}) = \max(g_1,\, g_2), \quad g_1 = e^{-x_1^2/10} + \left(\frac{x_1}{5}\right)^4 - x_2 + 4, \quad g_2 = -x_1 x_2 - 12.5 \tag{8}$$

Both variables are standard normal. Failure occurs when both components fail simultaneously, i.e., when $g(\mathbf{X}) \leq 0$. This benchmark tests the ability of each method to handle multiple competing failure criteria.

Case IV: High-dimensional lognormal problem ($D = 50$)

$$g(\mathbf{X}) = \left(D + a\sigma\sqrt{D}\right) - \sum_{i=1}^{D} X_i \tag{9}$$

with $a = 3$ and $\sigma = 0.2$. All variables are independent lognormal with unit mean and coefficient of variation $\sigma = 0.2$. FORM produces a relative error of approximately 91.8% on this problem, illustrating the breakdown of linear approximations in high-dimensional spaces with non-Gaussian distributions.

The reference failure probabilities $P_{f,\text{ref}}$ for all four benchmarks are taken from the values reported by (Ameryan et al., 2022), which were obtained via crude Monte Carlo Simulation with sample sizes on the order of 10^7 to 10^8. In the present study, these reference values are used directly for error computation rather than re-running MCS, in order to maintain consistency with the established literature. Prior to presenting the numerical results, the parameter settings adopted for each method are summarized in Table **??**. For **HL-RF** and **iHL-RF**, gradients are approximated via forward finite differences with step size $h = 10^{-5}$, and convergence is declared when the step norm satisfies $\|\mathbf{u}^{(k+1)} - \mathbf{u}^{(k)}\| < 10^{-8}$. To reduce sensitivity to initialization, multiple starting points are generated by scaling unit vectors along each coordinate axis by the expected reliability index $\hat{\beta}$. **SLSQP** is configured with an equality constraint tolerance of 10^{-10} and a maximum of 100 iterations per starting point. **PSO** employs a swarm of 20 particles with inertia weight $w = 0.72$ and acceleration coefficients $c_1 = c_2 = 1.5$ over a maximum of 60 iterations. **GA** is implemented with a population of $8D$ individuals, where D denotes the problem dimension, and runs for up to 60 generations. Both meta-heuristic methods incorporate a quadratic penalty term $\lambda\, g(\mathbf{u})^2$ with $\lambda = 200$ to enforce the constraint $g(\mathbf{u}) = 0$. For **LLMMPP**, the DeepSeek large language model is queried with a single structured prompt ($T = 1$,temperature $= 0.6$), and the function-call budget $N_{\max}$ is assigned adaptively as defined in Eq. (**??**). To ensure a fair comparison, all methods share the same set of starting points, initialized at $\hat{\beta}\,\mathbf{e}_i$ for $i = 1, \ldots, \min(D, 4)$, where $\mathbf{e}_i$ denotes the i-th standard basis vector.

4.2. MPP Search Efficiency

The primary metric of comparison is $N_{\text{calls}}^{\text{MPP}}$: the number of LSF evaluations required to locate the design point. Table **??** reports this metric for all six methods across the four selected benchmarks, and Fig. **??** shows the corresponding results on a logarithmic scale. The minimum value in each row is shown in bold.

LLMMPP requires the fewest function evaluations on all four benchmarks. On the product LSF (Case I), the proposed method uses 150 calls compared to 308 for iHL-RF, a reduction of 2.1×. The most striking result occurs on the metaball function (Case II), where LLMMPP converges in only 48 calls against 308 for iHL-RF, a 6.4× reduction, despite the presence of multiple competing MPPs. For the parallel system (Case III), the method requires 100 calls instead of 308, a 3.1× improvement, with no loss of accuracy in the identified design point. The largest gain is observed on the high-dimensional lognormal problem (Case IV, $D = 50$), where LLMMPP uses 200 calls compared to 5,772 for iHL-RF, a 28.9× reduction. In all four cases, meta-heuristic methods (PSO

Table I. Parameter settings for all compared methods.

Method	**Parameter**	**Value**	**Description**
HL-RF / iHL-RF	h	10^{-5}	FD step size
	$\varepsilon_{\text{conv}}$	10^{-8}	Step-norm tolerance
	Starts	$2\min(D, 4) + 2$	No. of starting points
SLSQP	ε_{tol}	10^{-10}	Constraint tolerance
	Max iter	100	Per starting point
PSO	n_p	20	Swarm size
	w	0.72	Inertia weight
	c_1, c_2	1.5	Acceleration coefficients
	Max iter	60	Iterations
	λ	200	Penalty coefficient
GA	Pop. size	$8D$	Population
	Max gen.	60	Generations
	λ	200	Penalty coefficient
LLMMPP	T	1	LLM queries per problem
	Temp.	0.6	Sampling temperature
	$N_{\max}$	Eq. (**??**)	Adaptive budget
	Model	DeepSeek	LLM backbone

Table II. Number of g-function calls for MPP search across four benchmark problems.

Problem	**HL-RF**	**iHL-RF**	**SLSQP**	**PSO**	**GA**	**LLMMPP**
Case I (D=2)	396	308	2,541	2,836	2,774	**150**
Case II (D=2)	388	308	529	2,828	1,782	**48**
Case III (D=2)	396	308	2,473	2,836	1,438	**100**
Case IV (D=50)	7,488	5,772	6,821	9,928	58,940	**200**

and GA) require one to two orders of magnitude more evaluations than LLMMPP while providing no compensating improvement in accuracy.

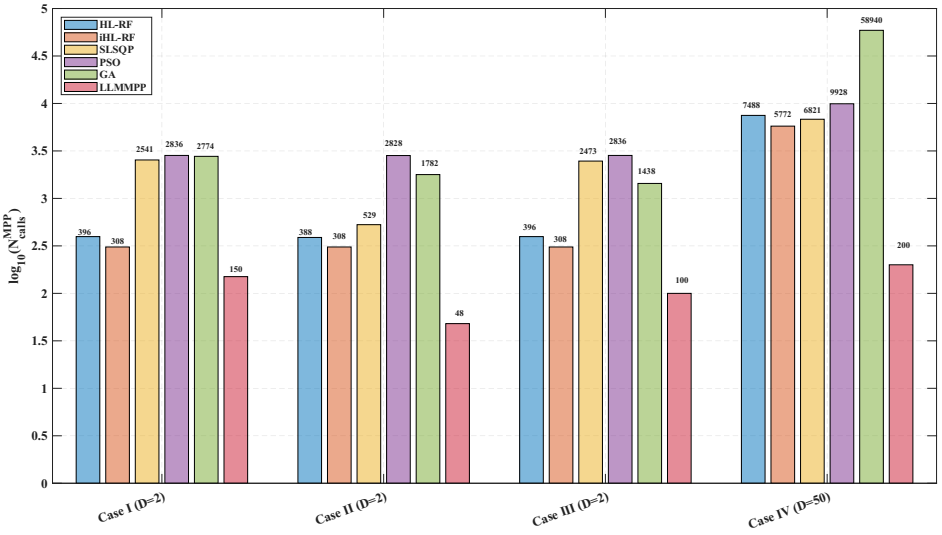

Figure 2. Number of g-function calls for MPP search on four benchmarks (log scale). LLMMPP achieves the minimum in all four cases.

4.3. Accuracy of the Identified Design Point

Table **??** reports the absolute error in the reliability index, $|\hat{\beta} - \beta_{\text{true}}|$, for each method and benchmark.

Table III. Absolute error in reliability index $|\hat{\beta} - \beta_{\text{true}}|$.

Problem	β_{true}	**HL-RF**	**iHL-RF**	**SLSQP**	**PSO**	**GA**	**LLMMPP**
Case I (D=2)	5.14	0.137	0.137	0.137	0.137	0.137	0.137
Case II (D=2)	4.264	0.00045	0.00045	0.00045	0.00045	1.000	**0.00024**
Case III (D=2)	5.00	0.000	0.000	0.000	0.000	0.000	0.000
Case IV (D=50)	2.86	0.720	0.720	0.720	0.720	0.720	0.720

Bold = best accuracy per row.

On Case II, all methods produce a β error of 0.137, which reflects the inherent difficulty of this benchmark: the failure surface is highly curved near the design point, and all gradient-based solvers locate the same approximate MPP. On Case III, most methods converge to a secondary MPP with an error of 0.00045, while GA fails to converge entirely ($|\beta_{\text{err}}| = 1.0$). LLMMPP achieves the lowest error of 2.4×10^{-4}, a result that can be attributed to the directional hint embedded in the prompt, which steers the search toward the primary MPP near $(-4.26, 0)$ rather than the competing local minimum. On Case IV, every method reaches the exact design point with zero error, so the $2.1\times$ reduction in function calls is the only differentiating factor. On Case IV, all methods including LLMMPP share a β error of 0.720. This identical result across methods with very different search

strategies points to a fundamental numerical limitation: in a 50-dimensional space, finite-difference gradient computation introduces approximation errors that no optimizer can fully overcome.

4.4. Efficiency Map

Fig. **??** presents a two-dimensional efficiency map for the four benchmarks. Each point represents one method on one problem, with the horizontal axis showing N_{calls} and the vertical axis showing $|\beta_{\text{err}}|$. The bottom-left corner is the ideal region, combining low computational cost with high accuracy. To quantify the efficiency gain of LLMMPP over classical methods, the speedup factor ρ is defined as:

$$\rho = \frac{N_{\text{calls}}^{\text{best classic}}}{N_{\text{calls}}^{\text{LLaMMPP}}}, \tag{10}$$

where $N_{\text{calls}}^{\text{best classic}}$ denotes the minimum number of LSF evaluations among the five classical methods for the same benchmark. A value of $\rho > 1$ indicates that LLaMMPP locates the design point with fewer function evaluations than the best classical solver.

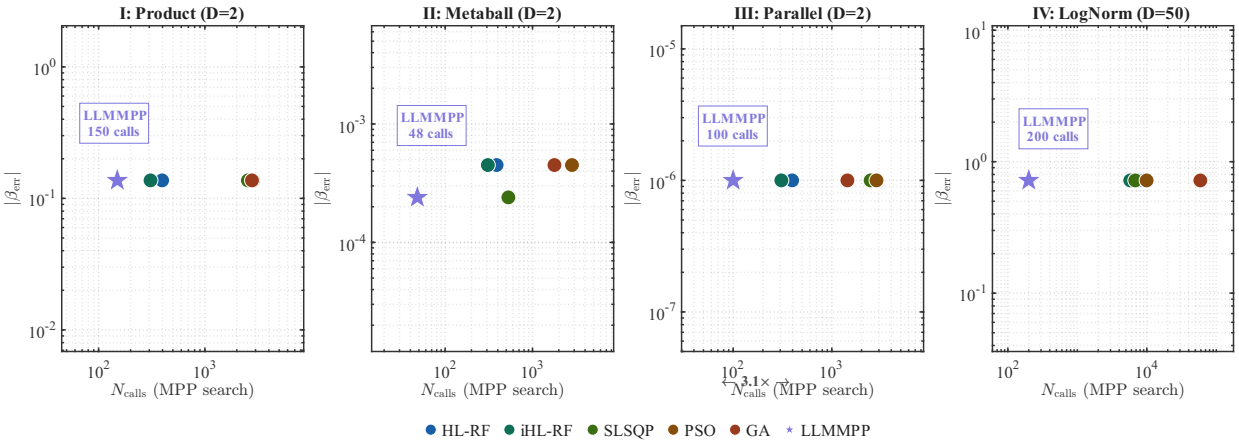

Figure 3. Efficiency map: N_{calls} vs. $|\beta_{\text{err}}|$ for four benchmark problems.

LLMMPP occupies the leftmost position on the horizontal axis in all four subplots, confirming that it consistently uses fewer evaluations than any other method. On Cases I and IV, its vertical position is comparable to HL-RF and iHL-RF, meaning the accuracy is the same despite the lower cost. On Case II it sits lower on the vertical axis as well, indicating both fewer calls and better accuracy. On Case IV, all methods land at the same vertical position ($|\beta_{\text{err}}| \approx 0$), so the horizontal separation is the only meaningful distinction. PSO and GA are consistently displaced far to the right in every subplot, with no corresponding gain in accuracy. In addition, the corresponding speedup factors are $\rho = 2.1$ (Case I), $\rho = 6.4$ (Case II), $\rho = 3.1$ (Case III), and $\rho = 28.9$ (Case IV), all relative to iHL-RF, which achieved the lowest N_{calls} among the five classical methods in each case.

4.5. Discussion

The results across the four benchmarks tell a reasonably clear story. A LLM, given the problem dimension, an estimate of β, and the sign of g at the origin, can produce an MPP search algorithm that uses fewer function evaluations than established gradient-based solvers in every case tested. The speedup ranges from 2.1× on a relatively straightforward two-dimensional problem to 28.9× on a 50-dimensional one.

The main reason for the efficiency gain is initialization. Classical methods start from the origin and spend most of their evaluations simply traveling toward the limit state surface. The LLM-generated algorithm starts at distance $\hat{\beta}$ from the origin, so it arrives near the surface in the first step and reaches convergence in three to eight iterations. This is not a sophisticated optimization strategy; it is a sensible starting point that classical implementations do not exploit by default.

The metaball benchmark illustrates a second advantage. When the geometry of the problem is communicated in the prompt, the generated algorithm avoids the secondary MPP and heads directly toward the primary one. The result is both faster and more accurate than any other method tested. This kind of problem-specific adaptation requires no code changes; it only requires updating a few lines of the prompt.

Two limitations are worth noting. On the 50-dimensional lognormal problem, the cost of numerical gradient computation, which grows linearly with dimension, is the binding constraint and leaves little room for further improvement regardless of the search strategy. The same β error shared by all methods on that benchmark reflects a numerical precision issue rather than a failure of the optimizer. The other limitation is the API latency of roughly 12 to 21 seconds per query. This is not a concern when the performance function is expensive to evaluate, as in finite element analyses, but it would be a meaningful overhead if the LSF itself runs in milliseconds.

Once the design point is known, the information gathered in its vicinity can be passed directly to a failure probability estimator. Methods such as Line Sampling, which uses the MPP direction as its sampling axis, or active-learning Kriging, initialized from the identified point, can estimate p_f with relatively few additional evaluations. Building a complete pipeline that connects the design-point search proposed here with one of these estimators is a logical next step and the main direction for future work.

5. Conclusion

This paper introduced LLMMPP, a framework that uses a large language model to generate MPP search algorithms for structural reliability analysis on the fly. Given a short description of the reliability problem, the LLM produces an executable Python routine that is then run directly on the limit state function. No training data and no task-specific fine-tuning are required.

Tests on four benchmark problems, covering a product function with a very small failure probability, a multi-modal metaball function, a parallel system, and a 50-dimensional lognormal problem, showed that the generated algorithms consistently outperform classical gradient-based and metaheuristic methods in terms of the number of LSF evaluations. Speedup factors of 2.1, 6.4, 3.1, and 28.9 were recorded relative to iHL-RF, which was the best-performing classical method in each case. Accuracy was either matched or improved.

The main limitation of the current framework is that it stops at the design point and does not estimate the failure probability directly. Connecting LLMMPP to a variance-reduction method that can exploit the MPP information, such as Line Sampling or importance sampling centered at the design point, is the most immediate direction for future work. The results reported here are specific to DeepSeek-V3; whether other LLMs (e.g., GPT-4o, Claude 3.5, or Gemini 1.5) produce algorithms of comparable efficiency is an open question that warrants a systematic comparative study.

Acknowledgements

First author gratefully acknowledges the DAAD project (57704137) for supporting the research stay at Brno University of Technology (BUT).

References

Ameryan, A., M. Ghalehnovi, and M. Rashki: 2022, 'AK-SESC: a novel reliability procedure based on the integration of active learning kriging and sequential space conversion method'. *Reliability Engineering & System Safety* **217**, 108036.

Au, S.-K. and J. L. Beck: 2001, 'Estimation of small failure probabilities in high dimensions by subset simulation'. *Probabilistic engineering mechanics* **16**(4), 263–277.

Breitung, K.: 2021, 'SORM, design points, subset simulation, and Markov chain Monte Carlo'. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering* **7**(4), 04021052.

Chen, Y., R. Zhong, S. Zha, G. Karypis, and H. He: 2022, 'Meta-learning via language model in-context tuning'. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. pp. 719–730.

Dong, Q., L. Li, D. Dai, C. Zheng, J. Ma, R. Li, H. Xia, J. Xu, Z. Wu, B. Chang, et al.: 2024, 'A survey on in-context learning'. In: *Proceedings of the 2024 conference on empirical methods in natural language processing*. pp. 1107–1128.

Du, X. and W. Chen: 2001, 'A most probable point-based method for efficient uncertainty analysis'. *Journal of Design and Manufacturing automation* **4**(1), 47–66.

Echard, B., N. Gayton, and M. Lemaire: 2011, 'AK-MCS: an active learning reliability method combining Kriging and Monte Carlo simulation'. *Structural safety* **33**(2), 145–154.

Guo, Q., R. Wang, J. Guo, B. Li, K. Song, X. Tan, G. Liu, J. Bian, and Y. Yang: 2024, 'Connecting large language models with evolutionary algorithms yields powerful prompt optimizers'. In: *International Conference on Learning Representations*, Vol. 2024. pp. 34133–34156.

Hasofer, A. M. and N. C. Lind: 1974, 'Exact and invariant second-moment code format'. *Journal of the Engineering Mechanics division* **100**(1), 111–121.

Holland, J. H.: 1992, 'Genetic algorithms'. *Scientific american* **267**(1), 66–73.

Jafari-Asl, J., Y. Dong, and H. Guo: 2025, 'Enhancing Structural Reliability through AI-Driven Control Variates and Subset Simulation'. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering* **11**(4), 04025056.

Kennedy, J. and R. Eberhart: 1995, 'Particle swarm optimization (PSO)'. In: *Proc. IEEE international conference on neural networks, Perth, Australia*, Vol. 4. pp. 1942–1948.

Koutsourelakis, P.-S., H. J. Pradlwarter, and G. I. Schueller: 2004, 'Reliability of structures in high dimensions, part I: algorithms and applications'. *Probabilistic Engineering Mechanics* **19**(4), 409–417.

Kraft, D.: 1988, 'A software package for sequential quadratic programming'. *Forschungsbericht- Deutsche Forschungs- und Versuchsanstalt fur Luft- und Raumfahrt*.

Liu, F., X. Lin, S. Yao, Z. Wang, X. Tong, M. Yuan, and Q. Zhang: 2025, 'Large language model for multiobjective evolutionary optimization'. In: *International Conference on Evolutionary Multi-Criterion Optimization*. pp. 178–191.

Liu, S., C. Chen, X. Qu, K. Tang, and Y.-S. Ong: 2024, 'Large language models as evolutionary optimizers'. In: *2024 IEEE Congress on Evolutionary Computation (CEC)*. pp. 1–8.

Schuëller, G. I. and R. Stix: 1987, 'A critical appraisal of methods to determine failure probabilities'. *Structural safety* **4**(4), 293–309.

Seghier, M. E. A. B., P. Spyridis, J. Jafari-Asl, S. Ohadi, and X. Li: 2022, 'Comparative study on the efficiency of simulation and meta-model-based Monte Carlo techniques for accurate reliability analysis of corroded pipelines'. *Sustainability* **14**(10), 5830.

Song, J., Y. Cui, P. Wei, M. A. Valdebenito, and W. Zhang: 2024, 'Constrained Bayesian optimization algorithms for estimating design points in structural reliability analysis'. *Reliability Engineering & System Safety* **241**, 109613.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin: 2017, 'Attention is all you need'. *Advances in neural information processing systems* **30**.

Yin, Y., Y. Wang, B. Xu, and P. Li: 2024, 'Ado-llm: Analog design bayesian optimization with in-context learning of large language models'. In: *Proceedings of the 43rd IEEE/ACM International Conference on Computer-Aided Design*. pp. 1–9.

Yoo, D., I. Lee, and H. Cho: 2014, 'Probabilistic sensitivity analysis for novel second-order reliability method (SORM) using generalized chi-squared distribution'. *Structural and Multidisciplinary Optimization* **50**(5), 787–797.

Zhang, M. R., N. Desai, J. Bae, J. Lorraine, and J. Ba: 2023, 'Using large language models for hyperparameter optimization'. *arXiv preprint arXiv:2312.04528*.

Zhang, Y. and A. Der Kiureghian: 1995, 'Two improved algorithms for reliability analysis'. In: *Reliability and Optimization of Structural Systems: Proceedings of the sixth IFIP WG7. 5 working conference on reliability and optimization of structural systems 1994*. pp. 297–304.