Reasoning or Recall? Blind Evaluation of LLMs on Numerical Function Tasks

Ahmad A. Rushdi [1] **Kiana Jafari Meimandi** [1] **Mykel J. Kochenderfer** [1]

Abstract

For numerical tasks, existing LLM evaluations typically measure either the model's ability to generate code or to recall information. We examine a complementary capability: direct numerical inference from data, without tool use. To isolate it, we apply a uniform protocol across three tasks: multifidelity surrogate modeling, numerical integration, and global optimization. The input for each task is a table of point evaluations from a function whose formula is unknown — a setting typical of uncertainty quantification problems. To focus on reasoning, rather than recall or code execution, we ask LLMs not to guess the underlying function or run code. We evaluated 4 LLMs and standard test functions across growing sample sizes. Models were prompted to select a method (from a list) for each task, then predict the function at unseen query points, estimate the integral, and locate the global minimum (each within a per-task tolerance). LLMs often converged to the same method for the same task, regardless of the training points, while prediction quality and confidence clearly scaled up with sample size. A fraction of answers matched the ground truth to several decimal places, well beyond what sparse-sample interpolation can yield, a pattern consistent with recall rather than reasoning. We probed this with a control function we composed, where recall is not possible: no more matches occurred.

1. Introduction

Surrogate modeling underpins much of computational science and engineering (Forrester et al., 2008; Queipo et al., 2005), yet still demands expert judgment. Three core tasks share one input: a table of N point evaluations $\{\mathbf{x}, f\}$ from an *expensive* function on a domain $\Omega \subseteq \mathbb{R}^d$. *Multifidelity* (MF) surrogate modeling trains on paired low-fidelity (f_L) and high-fidelity (f_H) samples to recover $\widehat{f} \approx f_H$ on Ω. *Numerical integration* (INT) estimates $\widehat{I} \approx \int_\Omega f\, d\mathbf{x}$. *Global optimization* (OPT) estimates $\widehat{\mathbf{x}}^\star \approx \arg\min_\Omega f$. We ask whether large language models (LLMs) can produce these answers from data alone, no other access to f (Figure 1).[1]

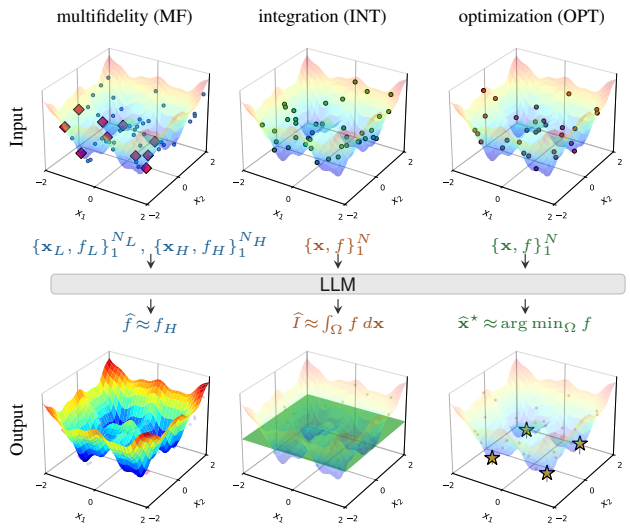

Figure 1. An illustration of our blind protocol over three tasks. *Inputs* are point evaluations sampled from an anonymous function (faded). MF input mixes many low-fidelity (blue) and a few high-fidelity (red, larger) samples. The LLM sees no formula or code. *Outputs*: an MF surrogate, an integral plate, and predicted minima.

Existing LLM-and-surrogate work (Lyu et al., 2025; Xie et al., 2025; Liu & Wang, 2026) measures a composite of two abilities: software engineering (executing code, recognizing named textbook functions) and reasoning from data. Code-generation benchmarks (Jimenez et al., 2024; Zhuo et al., 2024) cover the first well; the second is harder to isolate. We strip both confounds with a *blind protocol*: anonymous function name, unit-cube normalization, no code execution, and closed-form recall suppressed via the system prompt. Each response then falls into one of four categories (Figure 2): recall (from memory, e.g., actual formula in training data), code execution (a powerful but separate capability), reasoning (our focus), or no capability.

[1]Stanford University, Stanford, CA, USA. Correspondence to: Ahmad A. Rushdi <rushdi@stanford.edu>.

The 3rd *AI for Math Workshop* at the 43rd International Conference on Machine Learning (ICML), Seoul, South Korea, 2026.
Copyright 2026 by the author(s).

[1]Code, prompts, and per-session logs: https://github.com/aarushdi/reasoning-or-recall.

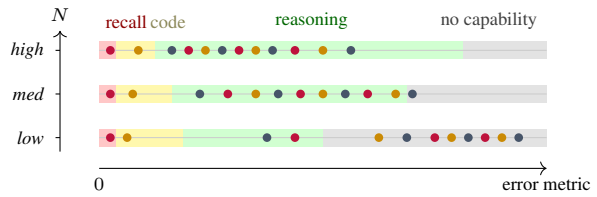

Figure 2. Illustration of response categories: *recall* (recalled formula evaluation), run *code*, sample-based *reasoning*, and *no capability*. As N grows, errors cluster tighter and leftward, the reasoning band expands, and the no-capability band shrinks.

Across the three tasks and four LLMs (Claude, GPT, Gemini, Muse), we observe: (i) When LLMs pick a method from a fixed list, all four converge to the same default per task, regardless of the data. (ii) Prediction quality scales with sample size. (iii) A *near-exact regime* (the recall band in Figure 2) in which a fraction of responses match the analytical ground truth to a precision well beyond what sparse interpolation can plausibly achieve, concentrated on canonical benchmarks, persistent under renaming (alias probe; Section 4.3), and *vanishing* on a hand-composed function, consistent with recall rather than inference from samples.

Our contributions are: (i) A blind cross-task evaluation protocol that strips code execution and function-name recognition simultaneously, applied identically across MF, INT, and OPT (Section 3) — to our knowledge the first such cross-task LLM numerical evaluation under matched elicitation. (ii) A four-regime error map (Figure 2) backed by a hand-composed novel function on which the near-exact rate falls to $0/12$ across all four LLMs (Section 4.3) — a falsifiable recall test prior benchmarks could not run. (iii) Cross-task empirical evidence on 4 frontier LLMs and 14 functions that isolates the recall channel from honest reasoning (Sections 4 and 4.3). We discuss the results in Sections 5 and 6.

2. Related Work

Recent work places LLMs in one of several roles inside a numerical pipeline. They have been used as direct surrogate models that predict outputs from inputs without an explicit numerical scheme (Rushdi et al., 2017; Zhang et al., 2025); as code-executing oracles that simulate program output (Lyu et al., 2025); as meta-controllers that pick from a library of concrete surrogates, often with explicit code execution tools (Xie et al., 2025; Liu & Wang, 2026); and as auxiliary signals that augment a PDE surrogate with LLM-derived embeddings (Lorsung & Barati Farimani, 2024). Each of these combines the reasoning we want to isolate with code execution or recall of known data.

Closest to our concern, Gupta et al. (2025) isolate reasoning in single-fidelity Bayesian optimization and find that LLMs do not productively use observed feedback; whether the same holds for coupled-fidelity prediction, quadrature, or blind-data optimization is open. We differ on two axes at once: a blind protocol that strips both code execution and function-recognition confounds (anonymous function name, unit-cube domain, no code execution, closed-form recall suppressed), and the *same* reasoning question posed across all three tasks (MF, INT, OPT). Each task has a mature numerical literature; we sketch the landscape below and draw a subset as reference baselines (specifics in Section 3). Test functions we use for all three tasks come from the SFU library (Surjanovic & Bingham, 2013).

For multifidelity surrogate modeling, additive and multiplicative corrections (Forrester et al., 2008) adjust a low-fidelity model by a learned residual or scale; co-kriging (Kennedy & O'Hagan, 2000; Forrester et al., 2007) generalizes these to a joint Gaussian process over both fidelities. Recursive (Le Gratiet & Garnier, 2014) and nonlinear NARGP (Perdikaris et al., 2017) variants address scalability and nonlinear fidelity coupling. Deep multifidelity Gaussian processes (Cutajar et al., 2019), composite neural networks (Meng & Karniadakis, 2020), physics-informed constructions (Penwarden et al., 2023; Raissi et al., 2019; Karniadakis et al., 2021; Howard et al., 2023), and Fourier neural operators (Li et al., 2021; Ravi et al., 2024) generalize the approach to deeper architectures; Peherstorfer et al. (2018) reviews the broader landscape. Multifidelity Bayesian optimization (Kandasamy et al., 2017; Poloczek et al., 2017; Wu et al., 2019; Sabanza-Gil et al., 2025) adds budget-aware query selection.

For integration, the Genz integrand families (Genz, 1984) provide standard test problems; classical quadrature rules (Gauss–Legendre, Simpson), Monte Carlo, quasi-Monte Carlo with Sobol sequences (Bratley et al., 1992), and Bayesian quadrature (Briol et al., 2019) cover the algorithmic spread. Welch et al. (1992) offers a high-dimensional scaling anchor. For global optimization (Kochenderfer & Wheeler (2019)), we benchmark against Bayesian optimization (model-based sequential search), CMA-ES (Hansen, 2016) (gradient-free evolution with adaptive covariance), L-BFGS-B (Byrd et al., 1995) (quasi-Newton with box constraints), random search, and Nelder–Mead.

These methods underpin practical workflows in fluid dynamics, where multifidelity surrogates accelerate expensive flow simulations (Carlberg et al., 2019; Morton et al., 2019; Partin et al., 2023); and in uncertainty quantification, where the same tasks drive propagation, sensitivity analysis, and design-under-uncertainty (Smith, 2024; Adams et al., 2023).

None of the above literature poses the same anonymous-function reasoning question across tasks under a single elicitation protocol. Existing cross-task work permits code execution or function access, so neither an observed failure nor success can be cleanly attributed to reasoning.

3. Experimental Design

Four ingredients turn the blind-protocol idea into a reproducible one: *functions* (Section 3.1), *LLMs and prompts* (Section 3.2), the *sweep grid* (Section 3.3), and *grading* (Section 3.4). Together they specify a $3\times4\times3\times4$ design (task $\times$ function $\times$ sample size $\times$ LLM); each cell yields a method selection (from a closed list) and a *requested quantity* — $\widehat{\mathbf{y}}$ for MF, $\widehat{I}$ for INT, or $(\widehat{\mathbf{x}}^\star, \widehat{f}^\star)$ for OPT. We use "$\star$" to denote gold and hats to denote LLM estimates.

3.1. Test functions

The benchmark draws from the SFU library (Surjanovic & Bingham, 2013), partitioned by task into three suites of four functions each (see Figure 3). All ground truths are analytical (LF/HF surfaces, closed-form integrals, exact global minima), so a human judge is not needed. Each suite spans an intrinsic-difficulty axis (smooth, multimodal, discontinuous, pathological).

MF uses Currin (d=2), Park 1, Park 2 (d=4 each), and Borehole (d=8) — canonical SFU pairs with native LF/HF couplings exercising $d \in \{2, 4, 8\}$. INT uses four integrands on $[0, 1]^2$: G1 oscillatory (smooth, high-frequency), G2 product-peak (smooth, corner-peaked), G4 Gaussian (smooth peaked), and G6 discontinuous (hard-edged plateau); each family's a, u parameters are drawn from a single fixed seed so every LLM, at every sample size, sees the same integrand. OPT uses Branin (three tied minima at $f^\star = 0.398$), Ackley (multimodal), Goldstein–Price (polynomial), and Rosenbrock (narrow curving valley) — landscapes that defeat local, gradient-based, and undirected search respectively. We also include two stretch cells beyond the main grid — G1 oscillatory at d=4 and Hartmann at d=3 — to probe dimensional scaling (Section 4.3, Appendix B). In our prompt design, we normalize to $[0, 1]^d$, hiding the native domain. We observed LLMs use the native domain (e.g., $[-5, 10] \times [0, 15]$) to guess the underlying function; normalization removes that channel.

3.2. LLM panel and prompting protocol

We evaluate four LLMs in their most capable reasoning modes: Claude Opus 4.7 (Anthropic, 2026) (hereafter **Claude**; *Adaptive Thinking*), GPT-5.5 (OpenAI, 2026) (hereafter **GPT**; *Thinking*), Gemini 3.1 Pro (Google, 2026) (hereafter **Gemini**; *Extended*), and Muse Spark (Meta, 2026) (hereafter **Muse**; *Thinking*).

All cells share the same elicitation protocol (Algorithm 1). The system message forbids tool use and closed-form recall; each cell returns a strict-JSON object with the method selection (from a closed list), the requested quantity, and a self-reported confidence in $[0, 100]$. The no-tool-use rule is enforced at the prompt level only; response patterns show no evidence of tool invocation — no code blocks, no execution-trace artifacts in the returned JSON, and no tool-call markers. Elicitation details (interface, settings, access window) are in Appendix D. Forbidding code execution is a design choice, not a limitation: code execution and named-function recall would re-introduce the very channels the protocol is designed to suppress, conflating reasoning under data scarcity with already-benchmarked code generation skill. The full per-task prompt templates are in Appendix A.

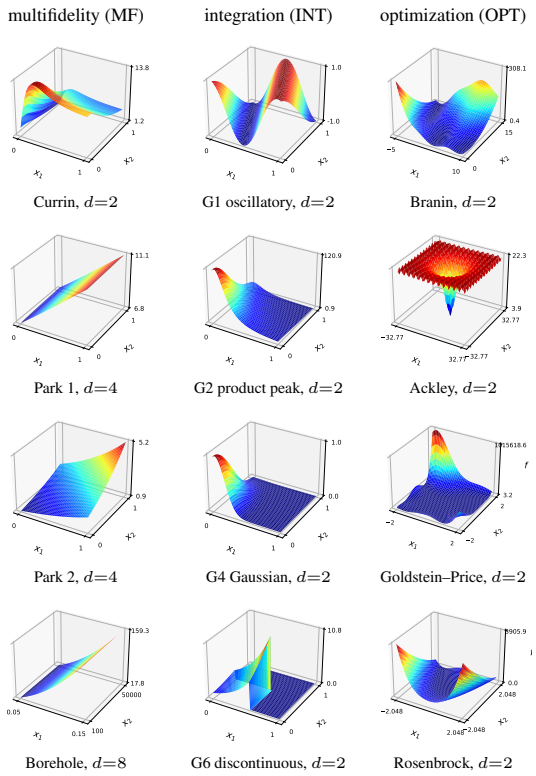

Figure 3. Test functions. Columns: MF (HF surface; 2d slice if d>2), INT (integrand), OPT (landscape). 3d surfaces shown on each function's native domain; but LLMs see it normalized to $[0, 1]^d$. Two d>2 stretch cells (G1 at d=4, Hartmann at d=3) and a hand-composed control (Section 4.3) are described in the text.

Algorithm 1 Blind elicitation cell.

Require: f, task T (MF, INT, OPT), axis $a \in \{\text{select}, \text{predict}\}$, LLM L, size N

1: $X \sim \text{LHS}([0, 1]^d, N)$, $Y \leftarrow f(X)$
2: Build (T, a) prompt P with an opaque hashed ID for f
3: $r \leftarrow L(P)$; tail-parse a whitelisted choice / number
4: Score r vs. gold (closed form / best baseline)

3.3. Sample sizes and factors

Sample size is swept across three levels per task: $N \in \{10, 100, 1000\}$ for INT, $N \in \{10, 30, 100\}$ for OPT (capped at 100 because N=1000 on a 2d Branin trivial-

izes argmin to a table lookup), and for MF a paired budget $N_H \in \{3, 10, 30\}$ with $N_L = 4N_H$ (fixed cost $c_H/c_L \approx 4$).

Two further design factors — sampling design (LHS, MC, Sobol, uniform, ...) and derivative information (values, or values+gradients) — are held fixed at (LHS, values only). A sweep over them is outside the scope of this work.

3.4. Ground truth and quality metrics

Every prompt yields: a method *selection* from a closed list and a numerical *prediction* graded by relevant metrics. Each prompt is scored correct/wrong within a per-task tolerance for: a selection outcome (against the candidate method that achieves the lowest error on the cell's data) and a prediction outcome (against closed-form gold).

For selection, Table 1 summarizes the selection list and gold criteria. We offer 6 MF architectures (cokriging-AR1 [`cokrig`], hierarchical recursive [`hier`], additive correction [`add`], multiplicative correction [`mult`], NARGP [`NARGP`], deep MF-GP [`deepMF`]), 5 quadrature schemes for INT (Gauss–Legendre [`G-L`], composite Simpson [`Simp`], Monte Carlo [`MC`], quasi-Monte Carlo [`qMC`], Bayesian quadrature [`BayQ`]), and five optimizers for OPT (random search [`rand`], CMA-ES [`CMA`], Bayesian optimization [`BOpt`], Nelder–Mead [`NM`], L-BFGS-B [`LBFGS`]). Selection gold fits each candidate on the cell's training samples and keeps the one with lowest error. Selection ties within 5% of the best held-out RMSE (MF) or best baseline error (INT, OPT) are accepted; otherwise selection is binary correct/wrong. Selection is framed as a *recommendation* ("which method would best solve the underlying task?"); baselines are scored under their native evaluation budgets, not constrained to the N samples.

Table 1. Selection axis: per-task closed list and gold criterion.

Task	Closed list	Gold
MF	6 architectures	within 5% of best HF RMSE
INT	5 quadrature schemes	min $\lvert\hat{I}-I^\star\rvert/\lvert I^\star\rvert$
OPT	5 optimizers	min $\lvert\hat{f}-f^\star\rvert$

For prediction, answers are scored independently against *analytical* or *empirical* gold. For MF, the held-out HF set is a fixed-seed (seed 42), 32-point Halton sequence on $[0,1]^d$; GP hyperparameters use the Emukit defaults (Matérn-5/2, maximum-likelihood length scales) without per-cell tuning. For INT, each scheme is run once with the same N. For OPT, the stochastic baselines (random search, CMA-ES, BO with EI) are repeated over 5 seeds and we take the median best-found $\widehat{f}^\star$ with a 50-evaluation budget; deterministic baselines (Nelder–Mead, L-BFGS-B) use 10 random starts. Implementation details are in Appendix D. Prediction gold is analytical throughout: closed-form HF surfaces $\mathbf{y}^\star$ for MF, Genz integrals $I^\star$ for INT, global minima $(\mathbf{x}^\star, f^\star)$ for OPT. The three prediction metrics we use are normalized RMSE for MF, relative error e for INT, and joint location/value error $(d, \Delta f)$ for OPT, given by:

$$\mathrm{nRMSE}(\widehat{\mathbf{y}}, \mathbf{y}^\star) = \frac{1}{\sigma(\mathbf{y}^\star)} \sqrt{\frac{1}{Q}\sum_{i=1}^{Q}(\hat{y}_i - y_i^\star)^2}, \tag{1}$$

$$\mathrm{e}(\hat{I}, I^\star) = \frac{|\hat{I} - I^\star|}{\max(|I^\star|, \varepsilon)}, \tag{2}$$

$$d(\widehat{\mathbf{x}}^\star, \mathcal{M}^\star) = \min_{\mathbf{m}\in\mathcal{M}^\star} \frac{\|\widehat{\mathbf{x}}^\star - \mathbf{m}\|_2}{\mathrm{diag}(\Omega)}, \quad \Delta f = \frac{|\widehat{f}^\star - f^\star|}{\sigma_\Omega(f)}, \tag{3}$$

with $\mathrm{e}_i = |\hat{y}_i - y_i^\star|/|y_i^\star|$; Q test points and τ ($\geq 75\%\, Q$). For this study, a small Q is deliberate. Beyond ~ 15 list entries, LLMs introduce silent transcription errors (e.g., rounding, dropped values). We keep the MF response well below that while supporting the $\mathrm{nRMSE} \leq 0.30$ verdict; $\varepsilon = 0.05$ stabilizes Equation (2) when $|I^\star| \to 0$; $\mathcal{M}^\star$ the set of valid global minima ($|\mathcal{M}^\star| = 3$ for Branin); $\sigma_\Omega(f)$ via 10^4-point quasi-Monte-Carlo. For OPT we additionally flag the trivial sample-copy solution (argmin coincides with a visible sample point), increasingly available as N grows. Across tasks, we report the *pass rate*: the fraction of cells whose prediction meets the pass thresholds of Table 2.

Table 2. Prediction axis. Per-task metric (with equation number), pass threshold, and near-exact threshold.

Task	Metric	Pass	Near-exact
MF	nRMSE (1)	≤ 0.30	$\#\{\mathrm{e}_i < 10^{-3}\} \geq \tau$
INT	e (2)	≤ 0.10	$\lvert\hat{I} - I^\star\rvert/\lvert I^\star\rvert < 10^{-3}$
OPT	d, Δf (3)	≤ 0.10 each	both $< 10^{-3}$

The 10^{-3} near-exact threshold is well beyond what sparse-sample interpolation can plausibly achieve: MF predictions all within 10^{-3} of analytical HF from ≤ 30 HF points on a $d \geq 2$ smooth function would require either a near-perfect posterior or a fortuitous sample alignment; the INT and OPT analogues follow the same argument. We therefore read near-exact matches as evidence of plausible recall, table lookup, or memorized constants, without ascribing intent or mechanism. The novel-function probe (Section 4.3) is the empirical sanity check: on a function absent from any benchmark, the rate of near-exact match drops to zero.

Self-reported confidence in $[0, 100]$ is reported per cell and compared qualitatively against pass outcomes (Figure 5). Headline rates aggregate per-cell binaries with 95% bootstrap intervals over 1000 resamples; per-task and per-N rates marginalize the other factors. Selection and prediction are scored independently. All prompts and ground-truth modules regenerate byte-for-byte from fixed seeds.

Table 3. Per-LLM method picks, by function and sample size. Each cell is the LLM's modal choice across the 3 independent sessions at that specific (function, N); each LLM contributes 3 cells (low / mid / high N). Methods are color-coded; the 4 cells at a given (function, N) are shaded green when all four LLMs agree on the same method. Abbreviations are listed in Section 3.4.

(a) MF: architecture

Function	Claude			GPT			Gemini			Muse		
(N_H)	3	10	30	3	10	30	3	10	30	3	10	30
Currin	cokrig	cokrig	cokrig	NARGP	NARGP	add	add	NARGP	cokrig	NARGP	NARGP	add
Park 1	cokrig	cokrig	cokrig	add	add	add	NARGP	cokrig	NARGP	NARGP	NARGP	cokrig
Park 2	add	cokrig	cokrig	add	add	NARGP	cokrig	cokrig	cokrig	cokrig	NARGP	add
Borehole	cokrig	cokrig	cokrig	add	NARGP	NARGP	cokrig	cokrig	cokrig	cokrig	cokrig	NARGP

(b) INT: quadrature scheme

Function	Claude			GPT			Gemini			Muse		
(N)	10	100	1000	10	100	1000	10	100	1000	10	100	1000
G1 oscillatory	BayQ	BayQ	G-L	BayQ	qMC	qMC	BayQ	BayQ	qMC	BayQ	qMC	qMC
G2 product-peak	BayQ	qMC	G-L	BayQ	qMC	qMC	BayQ	BayQ	qMC	BayQ	BayQ	MC
G4 Gaussian	BayQ	G-L	qMC	BayQ	BayQ	qMC	BayQ	BayQ	qMC	BayQ	BayQ	G-L
G6 discontinuous	qMC	MC	qMC	MC	MC	MC	BayQ	BayQ	qMC	BayQ	MC	MC

(c) OPT: optimizer

Function	Claude			GPT			Gemini			Muse		
(N)	10	30	100	10	30	100	10	30	100	10	30	100
Branin	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt
Ackley	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt
Goldstein–Price	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt
Rosenbrock	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt	BOpt

4. Results

We report results from 4 LLMs under the blind protocol of Section 3: each LLM completes 138 main-scoring sessions (108 on the main grid +21 on dimension/sparsity stretches +9 on alias probes) plus 5 side-probe sessions (Table 5 in Appendix B). The order mirrors Section 3.4: Section 4.1 on *selection*, Section 4.2 on *prediction quality and confidence*, Section 4.3 on the near-exact regime and the novel-function control, and Section 4.4 on the cross-task synthesis.

4.1. Method selection: what do LLMs pick?

Table 3 reports each LLM picks, with rows shaded green where the four LLMs agree unanimously. OPT is fully collapsed: every LLM picks `bayes-opt` on every function regardless of N, producing four unanimously green rows. We stress that Bayesian optimization is a *reasonable* default for low-dimensional black-box optimization from point evaluations. The issue is not that the pick is wrong but that it is nearly *invariant* across functions and sample sizes, so the LLMs default to one "safe" optimizer without conditioning on the cell's data. For MF, Claude anchors on `cokrig` across all 4 functions, Gemini mostly on `cokrig` too, while GPT prefers `additive` and Muse spreads more. INT is the only domain with within-LLM variation: Gemini anchors on `bayes-quad`, Claude/GPT/Muse switch between `qMC` and `bayes-quad` depending on integrand family, with G4 Gaussian the one unanimously `bayes-quad` row. Selection therefore shows a sharp ranking of *adaptivity to data*: OPT $\ll$ MF $\ll$ INT.

4.2. Prediction quality and self-reported confidence

Figure 4 reports per-LLM *quality* (top row, lower is better: nRMSE for MF, e for INT, d for OPT, mean over cells) and *pass rate* (bottom row, higher is better), each with 95% bootstrap CIs over cells. MF has the widest cross-LLM dispersion: Claude and GPT pass roughly half their cells (51%, 49%), Gemini lands at 31%, and Muse trails at 26% — a 25-point spread — on MF the choice of LLM matters more than on INT or OPT. INT clusters tightly around e ≈ 0.18 and $\sim$40% pass across all four LLMs: integration difficulty is dictated by the integrand class, not by the LLM. OPT is the easiest axis, with three of four LLMs above 0.70 pass and the lowest d medians.

Pass rate alone does not say whether the LLMs are calibrated. Figure 5 traces both pass rate (top row) and self-reported confidence (bottom row) against sample size per task, with 95% bootstrap bands per LLM. The pass-rate panels confirm a uniform scaling pattern — MF climbs from below 10% at N_H=3 to 50–90% at N_H=30, INT from 0% at N=10 to $\sim$ 75% at N=1000, OPT to $\sim$ 80% by N=30. Confidence also rises with N in all three tasks, but from a high floor and with a shallower slope than pass rate: Gemini reports 75–95% in every column at every N, including the N_H=3 column where its MF pass rate is below 10%. The headline observation is that confidence does not discriminate at fixed N — it does not separate Gemini's near-exact cells from its honest cells (Section 4.3), and cannot be used as a per-cell reliability signal.

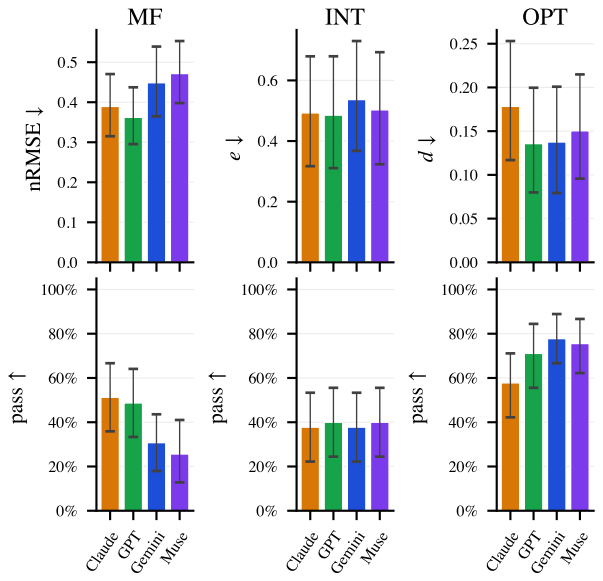

Figure 4. Per-LLM quality and pass rate, by task, pooled over the main grid (all N levels and all 4 functions per task; 12 cells per LLM/task, 3 sessions each, = 36 sessions/LLM/task). *Top row* (lower is better): the prediction metric, nRMSE for MF, e for INT, d for OPT, as the mean. *Bottom row* (higher is better): pass rate. All bars are means with 95% bootstrap CIs over cells.

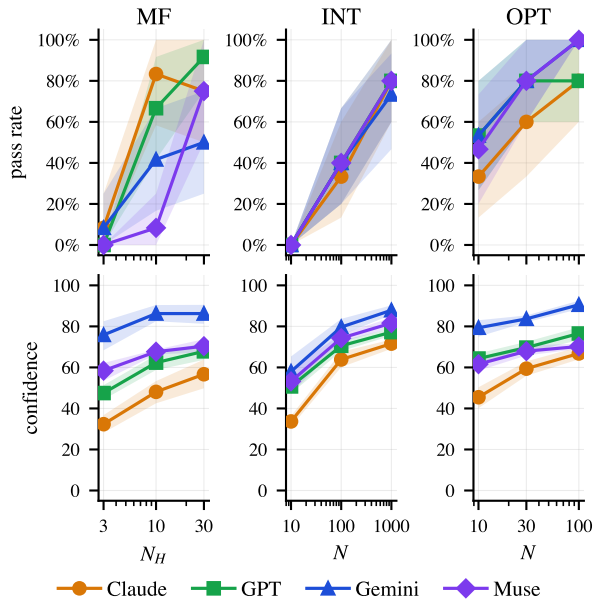

Figure 5. Pass rate and confidence vs. sample size, per task. *Top row*: pass rate. *Bottom row*: self-reported confidence in $[0, 100]$. Shaded bands are 95% bootstrap CIs over (cell, session). Pass rate rises monotonically with N in all three tasks; confidence also rises but never collapses with poor quality and never separates recall from honest reasoning.

4.3. Near-exact regime and protocol robustness

A small but systematic subset of LLM responses match analytical ground truth to within $\sim 10^{-3}$ relative precision on $\geq 6/8$ MF query points, to within 10^{-3} relative INT error, or to within 10^{-3} on *both* $\widehat{\mathbf{x}}^\star$ and $\widehat{f}^\star$ (OPT). Such precision is well beyond what honest sparse-sample interpolation would normally achieve, since the gold values are functions of input coordinates the LLM has not seen at training-data resolution. We report the rate without ascribing intent — the underlying mechanism (recall of a memorized closed form, lookup of a tabulated minimum, etc.) is not directly observable from the response.

The pattern is concentrated along three axes. *By task*: INT shows zero near-exact matches across all 4 LLMs and all 5 integrand families, totaling 180 sessions; closed-form Genz integrals are evidently not retrievable in the same low-error regime as MF surfaces and SFU minima, even when the integrand family is recognizable from sample patterns.

By function: MF matches concentrate on Currin (Gemini hits $5/12$, 42%, Wilson CI 19–68%; Park 1 $1/9$; Park 2 $0/9$; Borehole zero), the canonical 2d MF benchmark in nearly every multifidelity tutorial, while Borehole at $d{=}8$ is hard to recognize from 30 HF points alone. A baseline null on the same 4 Currin cells (Figure 6) achieves $0/4$ near-exact verdicts, ruling out sparse-sample interpolation as the explanation. OPT matches cluster on the most-cited 2d benchmarks (Branin, Ackley, Goldstein–Price), where the minimum is a well-known constant, and vanish on Rosenbrock and Hartmann 3.

By LLM: Muse records zero matches across all 138 sessions (a clean within-panel control); Claude 4 (one Park 2, two Ackley, one Goldstein); GPT 11 (concentrated in the OPT Branin / Ackley / Goldstein cluster); Gemini 16 (split across MF Currin and the OPT canonical-benchmark cluster). An anonymization-alias probe independently confirms the leak channel is value-pattern, not ID-string: the same Branin sample points under three different opaque IDs (`F-OPT-brani0100`, `F-foo-77`, `F-mystery-A`) elicit the same 44% near-exact rate for both GPT ($4/9$) and Gemini ($4/9$), statistically indistinguishable from the original Branin cells — renaming the function does not break the recognition channel.

Self-reported confidence on Gemini's near-exact cells (mean 95–100) is essentially indistinguishable from its confidence on honest cells (mean ∼90) — a direct echo of the at-fixed-N non-discrimination noted in Section 4.2. The structural lesson is that surface compliance with the blindness instruction (no function name spoken, no closed form recited) does not preclude leakage at the numerical output. We treat the near-exact rate as a measurable property of the protocol, not a claim about model intent. Figure 6 concretizes the recall-vs-reasoning contrast on the MF/Currin/$N_H{=}10$ cell:

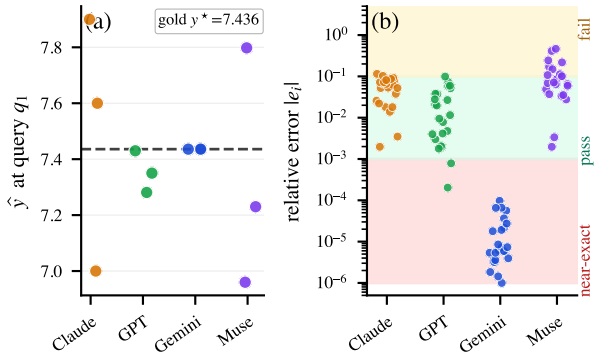

Figure 6. Currin / N_H=10 spotlight. (a) Predictions $\widehat{y}$ at a single query point q_i: gold value (dashed line) with all (LLM, session) markers — Gemini's markers sit on the gold line; Claude, GPT, Muse scatter. (b) Per-LLM relative error $|e_i|$ across all 8 query points and sessions on log scale, with shaded tolerance bands: *near-exact* ($|e|<10^{-3}$, suspicious), *pass* ($10^{-3}<|e|<10^{-1}$), and *fail* ($|e|>10^{-1}$). Gemini's points cluster in the near-exact band; the other three LLMs span pass and fail. **Empirical null**: across all 4 Currin cells (HF $\in\{1,3,10,30\}$), the best of four MF baselines (AR1 co-kriging, NARGP, additive, multiplicative) achieves $0/4$ near-exact verdicts (best per-cell $\mathrm{nRMSE}=0.08$ at HF $=30$; best per-point near-exact $=1/8$, far below the $\geq 6/8$ threshold) — quantitatively ruling out honest sparse-sample interpolation as the explanation for Gemini's residual collapse.

Claude, GPT, and Muse deviate from the analytical gold by amounts consistent with honest interpolation from sparse samples (residuals roughly in $[-2,+3]$, Muse the noisiest), while Gemini's predictions sit on the analytical curve to four-decimal precision across every session, with residuals indistinguishable from zero.

To test whether the near-exact regime tracks corpus exposure rather than the protocol itself, we ran the same blind protocol with the same suppression clause on a hand-composed 2d function not in any benchmark suite (full form in Appendix C). Three cells (MF, INT, OPT) were collected for each LLM, 12 sessions in total. The near-exact-match rate was $\mathbf{0/12}$ across all LLMs and all three axes: INT estimates converged on $\widehat{I}=0.5753$ (gold 0.5875), the honest sample-mean estimator; MF predictions showed honest nRMSE values (0.15–0.42); OPT predictions found a boundary minimum near $(0.75,1.0)$ (gold $(0.745,1.0)$) with $\widehat{f}^\star$ scattered between -0.42 and -0.35 around the gold -0.34.

Aggregating across the main grid, the four LLMs produce near-exact rates of Gemini $16/138$ (12%, 95% CI 7–18%), GPT $11/138$ (8%, 5–14%), Claude $4/138$ (3%, 1–7%), and Muse $0/138$ (0%, 0–3%); on the novel control: $0/12$ for every LLM. Two scaling probes are consistent with this reading: Branin and Ackley at N=200 elicit Gemini near-exact matches on both ($1/1$ each), and the G4 Gaussian integrand re-evaluated at d=4 reproduces the INT $0/4$ pattern. We frame these novel-control sessions as *evidence* that the near-exact regime is corpus-indexed rather than a property of the protocol.

Table 4. Cross-task failure modes. Each row is a falsifiable pattern that reproduces across the three tasks.

Failure mode	MF	INT	OPT
Default-answer bias	Claude / Gemini anchor on `cokrig`	Gemini anchors on `BayQ` ($\sim$2/3)	all 4 LLMs anchor on `BOpt`
Selection < prediction	S lags P across panel	S lags P in all 4 LLMs	smallest gap; OPT.P highest
Near-exact matches	Gemini 42% on Currin; 0% on Park 2, Borehole	0% across all 4 LLMs and 5 Genz families	GPT/Gemini $\geq$ 30% on Branin / GP; 0% on Rosenbrock, Hartmann 3

4.4. Cross-task failure modes

Table 4 compresses the four-LLM signal into three operational failure-mode rows that transfer across tasks: the default-answer bias documented in Table 3, the selection-versus-prediction gap visible by comparing Table 3 with Figure 4, and the near-exact matches of Section 4.3. The three rows partition cleanly: default-answer bias is a *selection-axis* failure (the LLM doesn't reason about which method fits the data); the S-vs-P gap is a *calibration-axis* failure (the LLM is more reliable at producing a number than at choosing a method); the near-exact-match row is a *protocol-robustness* failure (the blindness instruction is observed in text but not in output). Section 5 takes each in turn.

5. Discussion

5.1. Reading the three failure modes

The three failure modes of Table 4 sit at distinct levels of the protocol. *Default-answer bias* (selection axis): all four LLMs converge on small default sets — `cokrig` or `add` for MF, `BayQ` for INT (Gemini), `BOpt` for OPT panel-wide; the OPT default survives anonymized IDs and unit-cube normalization, ruling out a recognition artifact. One plausible explanation: BOpt is the dominant method discussed alongside black-box optimization in the training corpus and so becomes the safe default when cues are withheld.

Selection-versus-prediction asymmetry (calibration): LLMs are more reliable producing a number than picking from a list. *Near-exact regime*: surface compliance with the blindness instruction does not preclude near-exact agreement with analytical gold at the output; the novel-function control ($0/12$ matches) is evidence consistent with corpus-indexed recall rather than sample-based inference. We do not read recall and reasoning as binary: a model may recognize a function family from the sample pattern and then adapt from the points, so the near-exact band likely marks where recall dominates rather than a clean partition of responses.

5.2. Why blindness matters

These findings depend critically on blindness. Recent LLM-surrogate frameworks (Xie et al., 2025; Liu & Wang, 2026; Lyu et al., 2025) permit code execution or named-function access; those framings measure a composite of mathematical reasoning and code-synthesis fluency. We close both channels via three layers: an anonymous hashed identifier replaces the function name; unit-cube normalization hides distinctive native bounds (Branin's $[-5, 10]\times[0, 15]$, Borehole's heterogeneous physical ranges); and the MF selection axis uses synthetic LF discrepancies so the $(f, \mathrm{lf_archetype})$ pair has no exact precedent. In a pilot without normalization, two of four LLMs spelled "Branin" verbatim in their reasoning trace; with normalization, none did.

The cross-task design also preempts the single-task critique: a pattern in one domain could be a domain artifact, whereas a three-task sweep either reproduces it (a generalizable limitation) or maps where it transfers. Our data lands on the former — all three failure modes recur across MF, INT, and OPT.

5.3. Limitations

The blind protocol is artificially clean — real practitioners use formulas, plots, and code — so our claims are about reasoning under blindness, not deployment ergonomics. A second caveat is enforcement: code and tool use are suppressed at the *prompt level* only, not technically disabled or logged, so we cannot rule out silent tool use on a per-response basis. The novel-function control is our safeguard against this: genuine code execution would have solved the unseen control too, yet the near-exact rate falls to $0/12$ there (Sections C and 4.3) — so the near-exact matches on canonical functions are better explained by recall than by either reasoning or covert execution. OPT optimizer-gold labels are stochastic; we average over seeds. The four-LLM panel is a mid-2026 snapshot, and near-exact rates are conditioned on the publicly-indexed status of the SFU functions: the novel-function control (Appendix C) is one data point, and broader sweeps are the natural extension.

5.4. Practitioner recommendations

Six guidelines follow from the patterns above. (1) Treat method selection as the weak axis: ask the LLM what to do but verify by running the picked method (Table 4). (2) Cross-check selection across LLMs and majority-vote — per-LLM defaults differ (Table 3), so a panel vote covers more of the decision space. (3) Do not use self-reported confidence as a per-cell reliability signal — confidence rises with N alongside pass rate but at fixed N does not separate correct from incorrect cells (Section 4.2); use external sanity checks (held-out validation, physics-residuals on simulators). (4) Watch for default-answer bias under uncertainty: the OPT `BOpt` anchor persists through anonymized IDs and unit-cube normalization, so read it as "I am uncertain" rather than "this is the right tool." (5) Pair the LLM with a small empirical baseline sweep — fitting the candidate set and reporting held-out deltas catches selection failures even when the prediction passes. (6) Treat near-exact agreement with analytical gold as a flag, not a success: the pattern concentrates on canonical benchmarks (Currin, Branin, Goldstein–Price); the empirical test is whether the same model behaves the same way on a genuinely novel function.

5.5. Implications for benchmark design

These regularities directly impact benchmark design. First, the near-exact regime quantifies a recall-vs-reasoning ambiguity that any benchmark over canonical functions inherits: up to $\sim 12\%$ of a headline rate may reflect retrieval, so reported numbers on Currin, Branin, Goldstein–Price should carry a recall caveat. Second, the selection–prediction gap argues that pass-rate-only benchmarks overstate competence by masking the weaker axis; the closed-list selection metric (Table 1) is cheap to add and pulls a distinct signal on the same data. Third, the confidence non-discrimination rules out asking the model how sure it is as a reliability shortcut, *external sanity checks* are the practical fall-back. Taken together: blind protocols, paired selection-and-prediction axes, and novel-function controls as benchmark design defaults for LLM numerical reasoning.

6. Conclusion

Three canonical numerical tasks — multifidelity prediction, numerical integration, and global optimization — share one atomic input: a point-evaluation table from which a method choice and a numerical estimate must be produced. We proposed a unified protocol and applied it across four LLMs with $550+$ prompts. The results identify three reproducible failure modes: selection underperforms prediction at matched sample size despite being categorically simpler; self-reported confidence rises with N but at fixed N does not separate correct from incorrect cells; and a small fraction of responses match analytical ground truth to a precision well beyond what sparse-sample interpolation can plausibly achieve, while the novel-function control drives that rate to $0/12$ — evidence consistent with recall.

Three natural extensions complete the picture: a code-execution arm to separate reasoning from code-synthesis fluency; a $\sim$100-function novel-control sweep to bound the recall attribution; and the same protocol on neighboring tasks (interpolation, sensitivity analysis, ODE/PDE simulation) to test generalization. Together, the blind protocol and its novel-function recall test (Figure 2) offer a contamination-aware lens for numerical LLM benchmarks.

References

Adams, B. M., Bohnhoff, W. J., Dalbey, K. R., Ebeida, M. S., Eddy, J. P., Eldred, M. S., Hooper, R. W., Hough, P. D., Hu, K. T., Jakeman, J. D., Khalil, M., Maupin, K. A., Monschke, J. A., Ridgway, E. M., Rushdi, A. A., Seidl, D. T., Stephens, J. A., Swiler, L. P., and Winokur, J. G. *Dakota: A Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, and Sensitivity Analysis*. Sandia National Laboratories, 2023. Version 6.19, https://dakota.sandia.gov.

Anthropic. Claude (Opus 4.7 adaptive thinking version). Large language model, 2026. URL https://claude.ai. Accessed 2026-05-23.

Bratley, P., Fox, B. L., and Niederreiter, H. Implementation and tests of low-discrepancy sequences. *ACM Transactions on Modeling and Computer Simulation*, 2 (3):195–213, 1992.

Briol, F.-X., Oates, C. J., Girolami, M., Osborne, M. A., and Sejdinovic, D. Probabilistic integration: A role in statistical computation? *Statistical Science*, 34(1):1–22, 2019.

Byrd, R. H., Lu, P., Nocedal, J., and Zhu, C. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

Carlberg, K. T., Jameson, A., Kochenderfer, M. J., Morton, J., Peng, L., and Witherden, F. D. Recovering missing CFD data for high-order discretizations using deep neural networks and dynamics learning. *Journal of Computational Physics*, 395:105–124, 2019.

Cutajar, K., Pullin, M., Damianou, A., Lawrence, N., and González, J. Deep Gaussian processes for multi-fidelity modeling. *arXiv preprint arXiv:1903.07320*, 2019.

Forrester, A., Sóbester, A., and Keane, A. *Engineering Design via Surrogate Modelling: A Practical Guide*. Wiley, 2008.

Forrester, A. I. J., Sóbester, A., and Keane, A. J. Multi-fidelity optimization via surrogate modelling. *Proceedings of the Royal Society A*, 463:3251–3269, 2007.

Genz, A. Testing multidimensional integration routines. In *Tools, Methods, and Languages for Scientific and Engineering Computation*, pp. 81–94. North-Holland, 1984.

Google. Gemini (3.1 Pro extended version). Large language model, 2026. URL https://gemini.google.com. Accessed 2026-05-23.

Gupta, R., Hartford, J., and Liu, B. LLMs for Bayesian optimization in scientific domains: Are we there yet? *arXiv preprint arXiv:2509.21403*, 2025.

Hansen, N. The CMA evolution strategy: A tutorial. *arXiv preprint arXiv:1604.00772*, 2016.

Howard, A. A., Perego, M., Karniadakis, G. E., and Stinis, P. Multifidelity deep operator networks for data-driven and physics-informed problems. *Journal of Computational Physics*, 2023.

Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., and Narasimhan, K. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations (ICLR)*, 2024.

Kandasamy, K., Dasarathy, G., Schneider, J., and Póczos, B. Multi-fidelity Bayesian optimisation with continuous approximations. In *International Conference on Machine Learning (ICML)*, 2017.

Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., and Yang, L. Physics-informed machine learning. *Nature Reviews Physics*, 3:422–440, 2021.

Kennedy, M. C. and O'Hagan, A. Predicting the output from a complex computer code when fast approximations are available. *Biometrika*, 87(1):1–13, 2000.

Kochenderfer, M. J. and Wheeler, T. A. *Algorithms for Optimization*. MIT Press, 2019.

Le Gratiet, L. and Garnier, J. Recursive co-kriging model for design of computer experiments with multiple levels of fidelity. *International Journal for Uncertainty Quantification*, 4(5), 2014.

Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations (ICLR)*, 2021.

Liu, J. and Wang, N. AutoSurrogate: An LLM-driven multi-agent framework for autonomous construction of deep learning surrogate models in subsurface flow. *arXiv preprint arXiv:2604.11945*, 2026.

Lorsung, C. and Barati Farimani, A. Explain like I'm five: Using LLMs to improve PDE surrogate models with text. *arXiv preprint arXiv:2410.01137*, 2024.

Lyu, B., Huang, S., Liang, Z., Sun, Q., and Zhang, J. SURGE: On the potential of large language models as general-purpose surrogate code executors. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 3268–3308, 2025.

Meng, X. and Karniadakis, G. E. A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse PDE problems. *Journal of Computational Physics*, 401, 2020.

Meta. Meta AI (Muse Spark version). Large language model, 2026. URL https://meta.ai. Accessed 2026-05-23.

Morton, J., Witherden, F. D., and Kochenderfer, M. J. Parameter-conditioned sequential generative modeling of fluid flows. In *AIAA Scitech 2019 Forum*, pp. 1–15. American Institute of Aeronautics and Astronautics, 2019.

OpenAI. ChatGPT (GPT-5.5 thinking version). Large language model, 2026. URL https://chatgpt.com. Accessed 2026-05-23.

Partin, L., Geraci, G., Rushdi, A. A., Eldred, M. S., and Schiavazzi, D. E. Multifidelity data fusion in convolutional encoder/decoder networks. *Journal of Computational Physics*, 472:111666, 2023.

Peherstorfer, B., Willcox, K., and Gunzburger, M. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. *SIAM Review*, 60(3):550–591, 2018.

Penwarden, M., Zhe, S., Narayan, A., and Kirby, R. M. A metalearning approach for physics-informed neural networks (PINNs): Application to parameterized PDEs. *Journal of Computational Physics*, 477:111912, 2023.

Perdikaris, P., Raissi, M., Damianou, A., Lawrence, N. D., and Karniadakis, G. E. Nonlinear information fusion algorithms for data-efficient multi-fidelity modelling. *Proceedings of the Royal Society A*, 473(2198), 2017.

Poloczek, M., Wang, J., and Frazier, P. Multi-information source optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Queipo, N. V., Haftka, R. T., Shyy, W., Goel, T., Vaidyanathan, R., and Tucker, P. K. Surrogate-based analysis and optimization. *Progress in Aerospace Sciences*, 41(1):1–28, 2005.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

Ravi, K., Fediukov, V., Dietrich, F., Neckel, T., Buse, F., Bergmann, M., and Bungartz, H.-J. Multi-fidelity Gaussian process surrogate modeling of pediatric tissue expansion. *arXiv preprint arXiv:2404.11965*, 2024.

Rushdi, A. A., Swiler, L. P., Phipps, E. T., D'Elia, M., and Ebeida, M. S. VPS: Voronoi piecewise surrogate models for high-dimensional data fitting. *International Journal for Uncertainty Quantification*, 7(1), 2017.

Sabanza-Gil, V., Barbano, R., Pacheco Gutiérrez, D., Luterbacher, J. S., Hernández-Lobato, J. M., Schwaller, P., and Roch, L. Best practices for multi-fidelity Bayesian optimization in materials and molecular research. *Nature Computational Science*, 5(7):572–581, 2025.

Smith, R. C. *Uncertainty Quantification: Theory, Implementation, and Applications*. SIAM, 2nd edition, 2024.

Surjanovic, S. and Bingham, D. Virtual library of simulation experiments: Test functions and datasets. https://www.sfu.ca/~ssurjano/, 2013. Accessed: 2026-05-23.

Welch, W. J., Buck, R. J., Sacks, J., Wynn, H. P., Mitchell, T. J., and Morris, M. D. Screening, predicting, and computer experiments. *Technometrics*, 34(1):15–25, 1992.

Wu, J., Toscano-Palmerin, S., Frazier, P. I., and Wilson, A. G. Practical multi-fidelity Bayesian optimization for hyperparameter tuning. In *Uncertainty in Artificial Intelligence (UAI)*, 2019.

Xie, L., Li, G., Wang, Z., Chung, E., and Gong, M. Large language model-driven surrogate-assisted evolutionary algorithm for expensive optimization. *arXiv preprint arXiv:2507.02892*, 2025.

Zhang, X.-R., Gong, Y.-J., Zhong, Y.-T., Huang, T., and Zhang, J. Large language model as meta-surrogate for data-driven many-task optimization: a proof-of-principle study. *arXiv preprint arXiv:2503.08301*, 2025.

Zhuo, T. Y., Vu, M. C., Chim, J., Hu, H., Yu, W., Widyasari, R., Yusuf, I. N. B., Zhan, H., He, J., Paul, I., Brunner, S., Gong, C., Hoang, T., Zebaze, A. R., Hong, X., Li, W.-D., Kaddour, J., Xu, M., Zhang, Z., Yadav, P., Jain, N., Gu, A., Cheng, Z., Liu, J., Liu, Q., Wang, Z., Lo, D., Hui, B., Muennighoff, N., Fried, D., Du, X., de Vries, H., and Von Werra, L. BigCodeBench: Benchmarking code generation with diverse function calls and complex instructions. *arXiv preprint arXiv:2406.15877*, 2024.

A. Prompts and Sample Responses

The full elicitation protocol is summarized in Algorithm 1 in the main text. Figure 7 assembles the full prompt stack: every cell composes two pieces — a fixed shared frame applied to every (task, axis) pair, and a per-task template that specializes data, selection list, and reply format — followed by one representative cell per task and the four LLMs' responses side by side. Reasoning text is truncated; the structured JSON fields shown are the ones the scorer consumes.

B. Session accounting

Table 5 reconciles every session in the paper. "Cells" = task $\times$ function $\times$ N level. Sessions = cells $\times$ sessions/cell. Each LLM completes 138 main-scoring sessions, so the panel of 4 LLMs runs $4 \times 138 = 552$ main-scoring sessions; counting the 5 side-probe sessions per LLM, the grand total is $4 \times 143 = 572$.

Table 5. Session accounting per LLM.

Probe	Cells	Sess./cell	Sessions	Used in
Main grid ($3 \times 4 \times 3$)	36	3	$36 \times 3 = 108$	§4.1, 4.2
Stretch (1 MF + 3 INT + 3 OPT)	7	3	$7 \times 3 = 21$	§4.3
Alias probe (3 Branin IDs)	3	3	$3 \times 3 = 9$	§4.3
N=200 scaling (Branin, Ackley)	2	1	$2 \times 1 = 2$	§4.3
Novel control (MF / INT / OPT)	3	1	$3 \times 1 = 3$	App. C
Total per LLM	51	—	143	

C. Novel-function control

The hand-composed function used in the novel-function probe of Section 4.3 is

$$\begin{aligned} f(x_1, x_2) = {} & 0.60 \sin(3.5x_1 - 1.2) \cos(2.7x_2 + 0.4) \\ & + \; 1.10 \exp\!\big(-3(x_1 - 0.65)^2 - 4(x_2 - 0.35)^2\big) \\ & - \; 0.30\, x_1 x_2 + 0.50\, (x_1 + x_2 - 1)^2, \end{aligned}$$

on $[0, 1]^2$. The closed form is not in any benchmark suite or tutorial we are aware of; we verified by name- and substring-search on the SFU library and the Genz family. The probe samples MF, INT, and OPT cells under the same blind protocol as the main grid (12 sessions total: 4 LLMs $\times$ 3 cells). Gold values are computed analytically from the closed form above.

D. Implementation

Test-function callables, closed-form integrals (Genz), and SFU optimization minima are implemented in $\sim$600 lines of Python: `functions.py` (MF, via `mf2` + custom), `integration.py` (Genz family + Bratley), `optimization.py` (Branin / Rosenbrock / Ackley / Levy / Goldstein–Price / Hartmann 3). Architecture-fitter baselines use Emukit (AR1, NARGP, deep MF-GP) and scikit-learn (HF-only Matérn-5/2). Quadrature baselines use `scipy.integrate` and `scipy.stats` Sobol; the Bayesian-quadrature baseline uses Emukit's quadrature module. Optimization baselines use `scipy.optimize` (L-BFGS-B, Nelder–Mead), `cma` (CMA-ES), and a thin expected-improvement BO loop. LLM calls go through a thin `LLMBackend` protocol with Anthropic, Gemini, and OpenRouter backends; all responses are content-hash cached for reproducibility.

Elicitation and reproducibility. All reported results use each vendor's web-chat interface (one fresh chat per session, default vendor temperature, *Thinking* / *Extended* modes per Section 3.2); model versions and access dates are pinned in (Anthropic, 2026; OpenAI, 2026; Google, 2026; Meta, 2026). During pipeline development we verified that the same prompts replicate on the Anthropic, Gemini, and OpenRouter APIs at temperature 0.7; we do not pool web-chat and API runs in any reported number. Tool use is suppressed at the prompt level via the system message; we do not have API tool-call logs that formally rule out tool invocation, though response patterns (intermediate reasoning text, no code blocks, no tool-call markers) show no evidence of it.

The blind-protocol implementation, baseline fitters, scoring scripts, and per-cell prompt files will be publicly released with the final version of the paper.

MF / Currin, N_H=10

$N_L = 40$, $N_H = 10$, 8 queries on $[0, 1]^2$.
Gold $\mathbf{y}^\star$: 7.44, 4.93, 13.75, 6.94, 5.07, 9.38, 5.44, 8.38.

INT / G2 product-peak, N=100

100 random samples on $[0, 1]^2$.
Gold $I^\star = 20.194$.

OPT / Branin, N=100

100 samples; 3 tied minima.
Gold $f^\star = 0.398$ at $(0.124, 0.818)$, $(0.543, 0.152)$, $(0.962, 0.165)$.

Shared system frame (every cell, every task, every axis)

[SYSTEM] You are a careful numerical reasoner. You will be shown evaluations of an anonymous function at known inputs. You do NOT have access to the function's closed form, its name in the literature, or any code execution. Reason only from the point evaluations you see.

[ID] An opaque hashed identifier (*e.g.* `F-c679d`) replaces the function name. **[DOMAIN]** All inputs are normalized to $[0, 1]^d$ before the LLM sees them.

[REPLY] Strict JSON with fields `method` (one item from the closed list provided), `prediction` (numerical value(s) requested by the axis), and `confidence` $\in [0, 100]$. Reasoning text may precede the JSON but is parsed by tail-extracting the last whitelisted choice / numerical token.

MF prompt template

[DATA] N_L `LOW-FIDELITY` rows, N_H `HIGH-FIDELITY` rows, 8 `query_i` coords.
[SELECT] {`cokriging_AR1`, `hier_recursive`, `additive`, `multiplicative`, `NARGP`, `deep_MF_GP`}.
[PREDICT] $\widehat{\mathbf{y}}$ at the 8 queries.

INT prompt template

[DATA] N rows of $(\mathbf{x}, f(\mathbf{x}))$ on $[0, 1]^d$.
[SELECT] {`gauss_legendre`, `simpson`, `monte_carlo`, `quasi_monte_carlo`, `bayesian_quadrature`}.
[PREDICT] A single estimate $\widehat{I}$.

OPT prompt template

[DATA] N rows of $(\mathbf{x}, f(\mathbf{x}))$ on $[0, 1]^d$.
[SELECT] {`random_search`, `CMA_ES`, `bayesian_opt`, `nelder_mead`, `L_BFGS_B`}.
[PREDICT] $\widehat{\mathbf{x}}^\star \in [0, 1]^d$, $\widehat{f}^\star$.

Claude

arch: `cokriging_AR1` **conf:** 55
pred: 7.0, 4.6, 13.5, 7.3, 4.8, 8.3, 5.0, 8.5
Reason: "Nearby LF/HF pairs differ by small smooth offsets — approximately additive coupling…"

Claude

scheme: `quasi_monte_carlo` **conf:** 58 $\widehat{I} = 20.7$
Reason: "Scattered (non-grid) samples rule out Gauss–Legendre and Simpson; corner-peaked, MC slow…"

Claude

opt: `bayesian_opt` **conf:** 72
$\widehat{\mathbf{x}}^\star = (0.124, 0.818)$, $\widehat{f}^\star = 0.4$
Reason: "Competing basins of near-equal depth ⇒ gradient / simplex methods lock onto init…"

GPT

arch: `nonlinear_NARGP` **conf:** 72
pred: 7.28, 4.97, 13.24, 7.12, 5.06, 8.85, 5.84, 8.30
Reason: "LF/HF strongly correlated …"

GPT

scheme: `quasi_monte_carlo` **conf:** 72 $\widehat{I} = 20.71$
Reason: "Smooth 2d integrand with localized corner peak; low-discrepancy …"

GPT

opt: `bayesian_opt` **conf:** 82
$\widehat{\mathbf{x}}^\star = (0.124, 0.818)$, $\widehat{f}^\star = 0.398$
Reason: "Smooth multimodal landscape; GP-EI best use of a 50-eval budget…"

Gemini

arch: `nonlinear_NARGP` **conf:** 95
pred: 7.44, 4.93, 13.75, 6.94, 5.07, 9.38, 5.44, 8.38
Reason: "Spatially-varying LF–HF gap; ∼0 at [0.83, 0.92], substantial near [0.57, 0.77]…" *[All match gold to 3 decimals.]*

Gemini

scheme: `bayesian_quadrature` **conf:** 75 $\widehat{I} = 20.71$
Reason: "Limited samples with concentrated values ⇒ Bayesian quadrature most effective…"

Gemini

opt: `bayesian_opt` **conf:** 92
$\widehat{\mathbf{x}}^\star = (\mathbf{0.124}, \mathbf{0.818})$, $\widehat{f}^\star = \mathbf{0.398}$
Reason: "True minimum slightly deeper than observed best, around 0.398 near $(0.124, 0.818)$…"
[Analytical Branin minimum to 3 decimals.]

Muse

arch: `nonlinear_NARGP` **conf:** 68
pred: 7.80, 5.27, 10.75, 7.52, 4.88, 8.57, 6.37, 9.24
Reason: "LF reproduces smooth x_2-trend; HF–LF gap ∼0 at center…"

Muse

scheme: `bayesian_quadrature` **conf:** 72 $\widehat{I} = 20.71$
Reason: "Single smooth mode at the origin, falling by two orders of magnitude across the unit square…"

Muse

opt: `bayesian_opt` **conf:** 72
$\widehat{\mathbf{x}}^\star = (0.122, 0.83)$, $\widehat{f}^\star = 0.25$
Reason: "Lowest samples cluster at $\sim(0.12, 0.83)$, $y = 0.42$; second basin at $(0.55, 0.06)$ higher…"

Figure 7. Sample prompts and LLM responses, by task.