JS 26
INTELLIGENT TOOLS AND METHODS FOR RADIO SCIENCE

Impact des évolutions matérielles et algorithmiques sur l'optimisation électromagnétique par modèles de substitution

Impact of Hardware and Algorithm Improvement on Surrogate-Based Electromagnetic Optimization

***Dawei ZHAO*[1], *Christophe ROBLIN*[1]**

[1]*Telecom Paris, LTCI, Comelec, RFM², {dawei.zhao,christophe.roblin}@telecom-paris.fr*

Keywords: Antenna Optimization; Surrogate Model; Bayesian Optimization; Surrogate-Assisted Evolutionary Algorithms; Heterogeneous Computing; Benchmarking;

Abstract

This paper investigates how the computational balance between surrogate-based optimization and high-fidelity electromagnetic (EM) simulation has evolved over the past decade. A decoupled benchmarking methodology is proposed to separately evaluate three major components of the optimization workflow: EM simulation, surrogate model training, and surrogate inference. Multiple hardware platforms, spanning legacy CPU-only systems and modern heterogeneous CPU–GPU workstations, are benchmarked under representative software environments. The results show that both EM simulation and surrogate-based optimization have benefited significantly from hardware and software evolution, while the optimization-side overhead has improved even more rapidly. Consequently, the relative time consumed by surrogate-based optimization remains consistently low, even when the "heavy engine" EM solver is substantially accelerated. In particular, surrogate inference for large candidate batches remains at the millisecond scale, confirming its role as an ultra-fast "light engine" within the optimization loop. These benchmarking also suggest that the traditional description of surrogate-based optimization as "computationally expensive" should be interpreted mainly in terms of algorithmic structure and internal loop complexity, rather than practical wall-clock overhead in modern EM design workflows.

1 Introduction

Over the past three decades, full-wave EM simulation algorithms based on numerical solvers—such as the Finite Element Method (FEM), Method of Moments (MoM), Finite Difference Time Domain (FDTD), and Finite Integration Technique (FIT)—alongside commercial software packages (e.g., HFSS®, CST Microwave Studio®, and Feko®), have become indispensable. They are now crucial steps in the design of antennas, RF/microwave circuits, millimetre-wave systems, optoelectronics, and Very-Large-Scale Integration (VLSI). These tools empower designers to accurately predict device performance and identify physical bottlenecks prior to fabrication.

Historically, high-fidelity EM simulations have been computationally expensive, with a single evaluation often requiring hours or even days. To reduce this burden, various surrogate-assisted and model-based optimization strategies have been introduced for RF and EM design. An early influential example is John W. Bandler's Space Mapping (SM) framework [1], which exploited computationally cheap coarse models to accelerate expensive optimization. In its original setting, SM was especially suitable for RF and microwave circuit problems, where equivalent circuit models could often serve as physically meaningful coarse models. This philosophy later helped motivate the application of more general surrogate modelling and optimization techniques from the optimization community [2][3] to antenna and RF design, as systematically developed by Koziel and co-workers [4]. More recently, Mesh Space Mapping (MSM), developed by Cheng, Feng, Bandler, and collaborators [5], further extended the SM idea to a much broader range of electromagnetic structures. In parallel, Bayesian optimization (BO), represented by the Efficient Global Optimization (EGO) algorithm [6] and its multi-objective extensions like ParEGO [7], EHVI [8] and MOEA/D-EGO [9], became another important framework for expensive black-box optimization in EM design [10].

Although originally introduced to bypass costly heavy engine evaluations, the internal training overhead of these model-and-inference-based optimization methods is traditionally considered a computational bottleneck in its own right [11]., and we seek to achieve a clear understanding of the relationship between the time consumed by optimization algorithms and the duration of EM simulations. Specifically, we investigate how the proportion of model training overhead in the total

optimization time evolves throughout the EGO process, and how this trend is influenced by hardware platform performance and the calculation frameworks that utilize them. For example, beyond general hardware evolution, CST's 2012 presentation at the NVIDIA GTC conference marked the maturity of the GPU-accelerated T-Solver [12]. In 2019, CST officially extended support for T-Solver acceleration to consumer-grade GPUs via CUDA 9.2 [13], significantly lowering the barrier to high-performance EM simulation on affordable desktop systems. On the other hand, PyTorch-based frameworks such as GPyTorch (2018) [14] and BoTorch (2020) [15] have also substantially improved the fitting speed of Gaussian processes and BO workflows.

To systematically investigate this issue, we selected two distinct groups of hardware platforms. The first group utilizes an underclocked i7-7700K processor to simulate the typical performance of 4-core/8-thread hardware prevalent from 2012 to 2017. The second group comprises three existing workstations sourced from our research group, which naturally feature consumer-grade CPUs and GPUs across different generations and performance tiers. The detailed configurations of these evaluated platforms are summarized in Table 1. Across these platforms, we conducted benchmark tests encompassing two primary categories: baseline EM simulations and surrogate model training.

Table 1. Hardware Configurations of Evaluated Platforms

	Platform ID	**CPU (Target / Actual)**	**GPU**
Group 1 *(Legacy 4C8T)*	1.1	i7-2700K (underclocked i7-7700K)	N/A
	1.2	i7-4790K (underclocked i7-7700K)	N/A
	1.3	i7-6700K (underclocked i7-7700K)	N/A
	1.4	i7-7700K (Stock)	N/A
Group 2 *(Modern Workstations)*	2.1	i7-9700K	NVIDIA RTX 2060
	2.2	i7-12700K	NVIDIA RTX 4060 Ti
	2.3	Core Ultra 7 265K	NVIDIA RTX 5070 Ti

2 Methodology and Benchmarking Strategy

To systematically evaluate the impact of hardware evolution on surrogate-based or surrogate-assisted optimization, we first deconstruct the algorithmic workflows. Figure 1(a) and Figure 1(b) illustrate the standard flowcharts for Generation-Based Surrogate-Assisted Evolutionary Algorithms (SAEA) and EGO, which is 2 of the most widely applied surrogate-based optimization frameworks, respectively. Despite their distinct mathematical foundations, both frameworks share a highly comparable architecture. They inherently operate on a four-step epistemological loop: knowledge modeling (surrogate training or UQ modelling), decision-making, validation via high-fidelity EM simulation, and knowledge updating.

The fundamental difference between them lies actually in the selection criteria, operational scale and execution ratio among these steps. In the classic EGO loop, the relationship is tightly coupled: a complete model training and optimization cycle typically serves only a single expensive EM evaluation, and UQ information is applied to searching the next candidate. Conversely, in generation-based SAEA, a single model update often sustains an extensive evolutionary search spanning multiple generations, which is subsequently followed by a batch of EM validations, and UQ information is not adequately used, even you applied a surrogate model with UQ capability like kriging interpolation.

These structural characteristics also imply that the benchmarking strategy should differ for the two frameworks. For generation-based SAEA, the computational overhead can be meaningfully decomposed into three components: high-fidelity EM simulation, surrogate model training, and surrogate model inference. This decoupled evaluation directly reveals how hardware evolution affects each component separately, and provides practical time references for both the modelling stage and the "light-engine" surrogate evaluation stage. In contrast, for EGO, the model construction, acquisition-function optimization, and expensive evaluation are tightly coupled at each iteration. Therefore, instead of decomposing its internal steps, we benchmark the complete optimization loop directly on standard test functions, so as to capture the practical overhead of the full sequential BO process under different hardware and software environments. This decomposition method clearly shows how hardware evolution and algorithm improvements based on these hardware impacts each component and allows readers to use our benchmark data as a practical time reference for modelling and surrogate evaluation overheads in their own future optimization tasks.

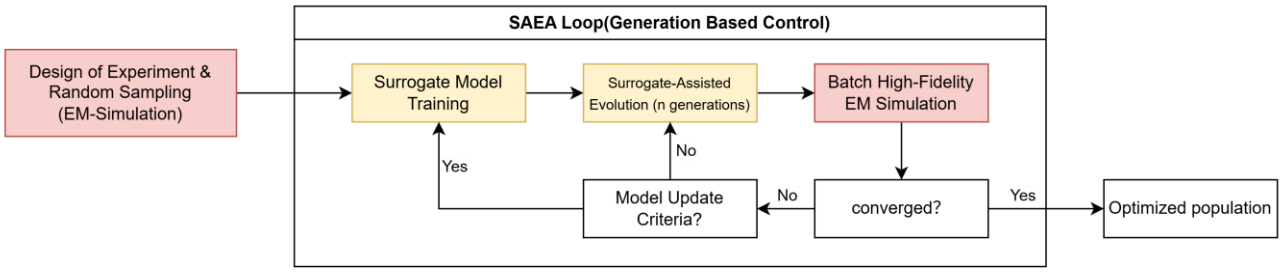

Figure 1(a) schematic of SAEA

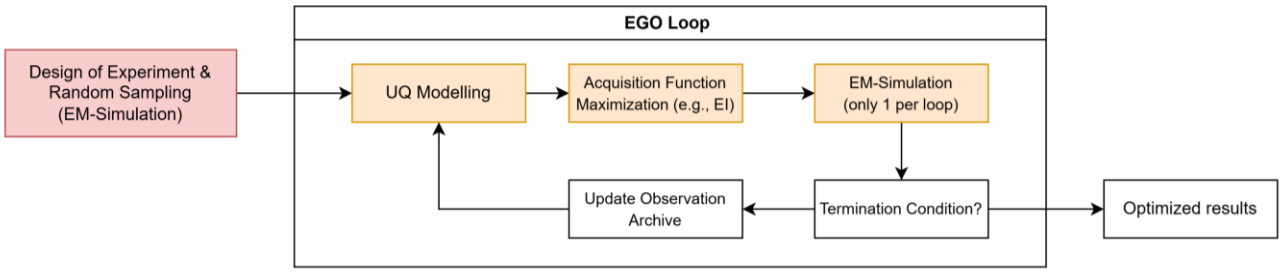

Figure 1(b) schematic of EGO

2.1 High-Fidelity EM Simulation Benchmark

To establish a realistic computational baseline for the high-fidelity EM evaluation which often referred to as the "heavy engine" in the optimization loop, we selected a Multislot Antenna with a Screening Backplane (MSA-BP) [16] as our primary test object. This antenna is originally designed for Ultra-Wideband (UWB) Wireless Body Area Network (WBAN) applications, and incorporates a backplane to effectively shield the human body and reduce the Specific Absorption Rate (SAR).

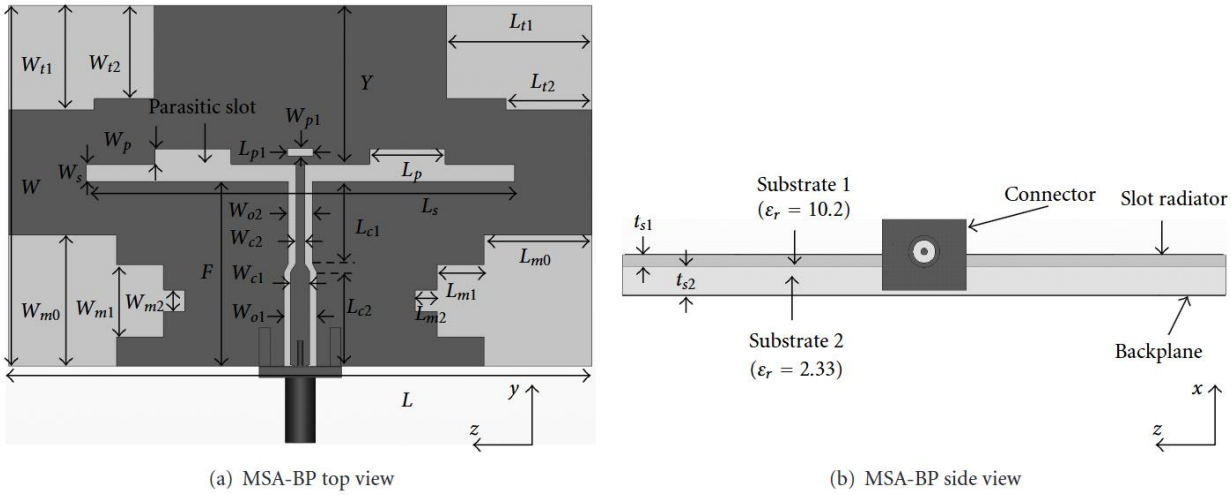

Figure 2 Geometry of MSA-BP

There are two reasons for selecting the MSA-BP. First, it exhibits a moderate, and representative computational complexity. Configured with a mesh density of 20 cells per wavelength (20 cells/λ), the discretization of the structure yields approximately 1M hexahedral mesh cells. Second, as illustrated by its intricate geometry, the MSA-BP possesses a rich set of geometric parameters, making it an ideal exemplar for benchmarking medium-scale optimization problems, which typically spanning 10 to 30 dimensions.

Furthermore, in a practical optimization scenario, the duration of a single EM evaluation is not static; it fluctuates continuously because parametric variations alter the physical structure, inherently triggering localized or global mesh reallocation. To accurately capture this dynamic computational overhead rather than relying on a single idealized run, we first conducted a preliminary sensitivity analysis to isolate 20 dominant design parameters (see table 2). Subsequently, we applied Latin Hypercube Sampling (LHS) to generate 16 distinct design variations, perturbing these selected parameters within a ±20% range around their nominal values. This LHS-based perturbation setup ensures that our recorded benchmark times reflect the true variance in simulation costs encountered during an active SBO process.

Table 2 design parameters of MSA-BP

	Random Sampled Parameters									
Variable Name	L	W	F	Y	Ls	W_s	L_p	W_p	L_{p1}	W_{p1}
Initial Value(mm)	68.1	42.1	21.6	18.5	50	2	8.84	1.8	3	0.9
Variable Name	L_{t1}	W_{t1}	L_{t2}	W_{t2}	L_{m0}	W_{m0}	L_{m1}	W_{m1}	L_{m2}	W_{m2}
Initial Value(mm)	17	12.2	10	10.9	12.6	15.3	5.5	8.4	2.5	2.5
	Fixed or non-independent parameters									
Variable Name	L_{c1}		L_{c2}	W_{c1}	W_{c2}	W_{o1}	W_{o2}	t_{s1}	t_{s2}	
Initial Value(mm)	F-L_{c2}-1		11	2.25	1	3.8	2.8	1.27	3.18	

2.2 Benchmark Setting for Bayesian Optimization/EGO

In this study, the benchmark for BO is constructed to simulate the computational stress of high-dimensional, long-budget engineering optimizations. This section details the parameterization of the EGO framework and the selection process for the benchmark functions.

2.2.1 Experimental Configuration and ParEGO Framework

To ensure the GP model encounters sufficient complexity across different hardware tiers, we configured a multi-scale test matrix. The dimensionality is set as $n_{dim} \in \{5,10,15,20\}$. Correspondingly, the initial Design of Experiments (DoE) size follows the theory of $n_{init} \approx 10 \times n_{dim}$, specifically $n_{init} \in \{64,128,192,256\}$. To evaluate the hardware's endurance under sustained high-load modeling (where the $O(N^3)$ complexity of GP becomes dominant), the total optimization budget is fixed at $n_{total} = 4 \times n_{init}$., which in practical engineering application can be $2 \times n_{init}$ to $10 \times n_{init}$.

For multi-objective scenarios, we employ the ParEGO [7] framework. ParEGO simplifies the multi-objective problem into a single-objective surrogate modeling task by applying an Augmented Tchebycheff Scalarization at each iteration with a randomly sampled weight vector $\mathbf{\Lambda}$. The scalarized objective function $s\big(f(x)\big)$ is defined as:

$$s(f(x), \mathbf{\Lambda}) = \max_{i=1,\dots,m} [\Lambda_i (f_i(x) - z_i^*)] + \rho \sum_{i=1}^{m} \Lambda_i (f_i(x) - z_i^*) \quad (1)$$

where m is the number of objectives, z_i^* is the estimated ideal point (the minimum value for objective i seen so far), and ρ is a small positive augmentation coefficient (typically set to 0.05) to ensure the strict Pareto optimality of the solution. The ParEGO, with this augmented Tchebycheff scalarization, can ensure the coverage of non-convex Pareto front in optimization, which linear scalarization can't.

2.2.2 Benchmark Selection via Pre-testing

To select an appropriate test function and ensure it maintains meaningful exploration value throughout the full $n_{total} = 4 \times n_{init}$ BO budget—rather than prematurely converging to an optimum and rendering subsequent iterative sampling computationally trivial—we conducted a small-scale preliminary test. In this pre-test, the BO loop was executed with a limited budget of $n_{total} = 2 \times n_{init}$ (i.e., the number of sequential BO iterations will be equal to the initial DoE size). We utilized the Integrated Mean Squared Error (IMSE), a widely adopted metric in BO, as the evaluation criterion to monitor the global modelling status.

dim	5		10		15		20	
$n_{samples}$	64	128	128	256	192	384	256	512
functions	IMSE							
zdt1	1.54E-03	3.22E-04	9.71E-04	4.93E-04	9.86E-04	9.72E-04	9.33E-04	5.80E-04
zdt2	3.05E-04	1.92E-04	1.27E-04	4.73E-04	1.91E-04	4.43E-04	1.91E-04	1.24E-04
zdt3	4.83E-02	2.22E-04	5.32E-02	2.08E-03	5.14E-02	9.50E-03	1.45E-01	1.44E-02
dtlz1	1.93E+02	5.73E+01	1.56E+03	1.22E+02	6.56E+02	2.18E+02	2.12E+03	3.56E+02
dtlz2	1.58E-03	4.52E-04	9.20E-03	9.72E-04	1.28E-02	1.87E-03	1.72E-02	7.06E-03
dtlz3	8.80E+02	3.38E+02	2.79E+04	9.19E+02	4.21E+03	9.68E+02	5.67E+03	1.41E+03

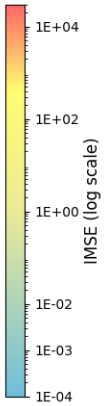

Figure 3: Kriging model IMSE results for different benchmark functions after an Incremental budget of n_{init}.

The pre-test results across various classical test functions [18][19] and dimensions are summarized in Figure 3. As the data indicates, smoother functions experience a rapid collapse in IMSE, leading to early exploitation that fails to stress the algorithm's computational limits. Based on these results, DTLZ1[1] was explicitly selected as the designated benchmark function for the BO loop, as its complexity successfully prevents premature convergence and guarantees sustained algorithmic load across the entire optimization budget.

2.3 Benchmark Setting for SAEA

To benchmark the computational overhead associated with Generation-Based SAEA, it is necessary to independently profile both the model training phase and the batch inference (evaluation) phase. In this section, we evaluate three classic global surrogate modeling techniques widely adopted in EM optimization: Artificial Neural Networks (ANN), Polynomial Chaos Expansion (PCE), and Kriging. The UQLab [20] toolbox by ETH Zurich is applied for easier implementation of PCE and Kriging Surrogate.

2.3.1 Configuration Matrix of Surrogate Models

To ensure a fair and statistically valid comparison across different hardware platforms, the complexity of the surrogate models must scale proportionally with the dimensionality of the problem ($d \in \{5, 10, 15, 20, 25, 30\}$). We constructed a configuration matrix strictly adhering to established empirical rules in surrogate modeling, as detailed in Table 4.

Table3: Surrogate Model Initial Sampling Numbers and Correspondence Model Complexity

dim	**Samples**	**ANN (FcNN)**			**PCE (N=3)**	
		Size	Params	Samples / Params	Coeffs	Samples / Coeffs
10	512	(6, 6)	115	~4.45×	286	~1.79×
15	1024	(8, 8)	209	~4.90×	816	~1.25×
20	2048	(12, 12)	421	~4.86×	1771	~1.16×
25	4096	(18, 18)	829	~4.94×	3276	~1.25×
30	8192	(28, 28)	1709	~4.79×	5456	~1.50×

For the PCE models, the polynomial order was truncated at $N = 3$. The corresponding number of sampling points ($N_{samples}$) was specifically allocated to maintain a slightly overdetermined system (i.e., $N_{samples}$ is 1.16 to 1.79 times the number of PCE coefficients), thereby preserving its advantage in uncertainty quantification and preventing rank deficiency. For the ANN models, double-hidden-layer architectures were employed. The size of the hidden layers was carefully tuned such that the number of training samples strictly remains approximately 5 times the total number of network parameters, effectively mitigating the risk of overfitting while ensuring sufficient model capacity. Kriging, lacking analogous structural

[1] whose Pareto-optimal front is a hyperplane.

hyperparameters, are usually considered more input-efficient, so was evaluated under the 1/8 of $N_{samples}$ to reach a similar accuracy level

2.3.2 Benchmark Selection via NRMSE Pre-testing

It should be noted that, unlike EGO, which is characterized by a high degree of predictability and convergence, SAEA always requires a high degree of model accuracy before applying EA or other metaheuristic algorithms; in this case, the choice of test function for the SAEA simulation follows a different theory than that of EGO.

To identify an appropriate benchmark, we conducted a deterministic pre-test by fitting the $N = 3$ PCE model across six standard test functions. The Normalized Root Mean Square Error (NRMSE) on an independent test set (10% of total samples) was recorded, as visualized in Table 5. The results reveal that highly multimodal functions (e.g., DTLZ1, DTLZ3) provoke catastrophic underfitting (NRMSE consistently exceeding 15%), rendering them unsuitable for simulating valid global model training. Conversely, excessively smooth functions (e.g., ZDT2) yield near-zero errors, failing to provide sufficient "fitting resistance".

dim	5		10		15		20		25		30	
Object	1	2	1	2	1	2	1	2	1	2	1	2
zdt1	1.33E-07	1.10E-02	5.27E-07	6.05E-03	4.80E-07	2.20E-02	7.13E-07	3.80E-02	3.06E-07	2.09E-02	2.36E-07	1.29E-02
zdt2	1.32E-07	7.64E-04	8.37E-08	3.84E-04	4.05E-07	3.76E-04	6.38E-07	4.30E-04	4.28E-07	2.35E-04	1.57E-07	1.17E-04
zdt3	1.35E-07	7.57E-02	1.35E-07	1.78E-01	8.01E-07	2.88E-01	2.45E-07	4.93E-01	1.63E-07	3.26E-01	2.92E-07	1.56E-01
dtlz1	1.52E-01	1.61E-01	1.52E-01	1.77E-01	2.21E-01	3.85E-01	4.67E-01	4.53E-01	1.98E-01	2.10E-01	1.21E-01	1.16E-01
dtlz2	8.45E-03	8.32E-03	1.07E-02	1.09E-02	2.03E-02	2.05E-02	3.82E-02	4.11E-02	2.05E-02	1.83E-02	1.30E-02	1.20E-02
dtlz3	1.76E-01	1.44E-01	1.91E-01	2.17E-01	4.49E-01	3.51E-01	5.89E-01	6.62E-01	2.64E-01	2.52E-01	1.38E-01	1.38E-01

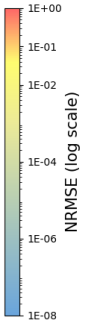

Figure 4: NRMSE of PCE (N=3) Across Different Benchmark Functions

Ultimately, DTLZ2 was selected as the designated benchmark function for the SAEA evaluation. Across all tested dimensions, DTLZ2 consistently falls into the accuracy most similar to realistic engineering application with an NRMSE between 1% and 5% (equivalent to $R^2 \approx 0.95$). This level of accuracy perfectly mimics the typical fitting behavior observed when modeling continuous but highly non-linear EM responses (e.g., broadband S-parameters or antenna far-field characteristics), ensuring our benchmarking results are meaningful of real-world SBO applications.

3 Results and Discussion

3.1 Impact of Hardware on High-Fidelity EM Simulation

The baseline EM simulation constitutes the most time-consuming phase of any traditional optimization loop. Figure 5 illustrates the execution time of a single MSA-BP evaluation across the tested hardware platforms.

In the legacy CPU era (Platforms 1.1 to 1.4), the simulation time gradually decreases from 480 s to 334 s. While sequential hardware upgrades within this paradigm do provide continuous performance improvements, the gains exhibit clear diminishing marginal returns. This trend highlights the inherent bottleneck of relying solely on CPU clock frequencies and moderate core counts to accelerate dense numerical solvers like the FIT-based T-Solver. Even on the modern CPU of Platform 2.3, the pure-CPU simulation time plateaus at 266 s.

However, the computational paradigm shifts dramatically with the introduction of GPU-accelerated T-Solvers in the modern heterogeneous era (Platforms 2.1 to 2.3). By offloading highly parallelizable matrix operations to the GPU, the computational overhead is decisively curtailed. Notably, on the highest-tier Platform 2.3, the simulation time is compressed to an unprecedented 91s—achieving a remarkable 5.2x speedup compared to the 2012 baseline. This profound acceleration of the "heavy engine" fundamentally alters the absolute time-cost distribution within the SBO process. It establishes a much faster baseline for physical validation, which makes metaheuristic algorithms based on massive simulation ever more accessible, thereby forcing a critical re-evaluation of whether the surrogate modeling algorithms can maintain their "negligible cost" status under such extreme simulation speeds.

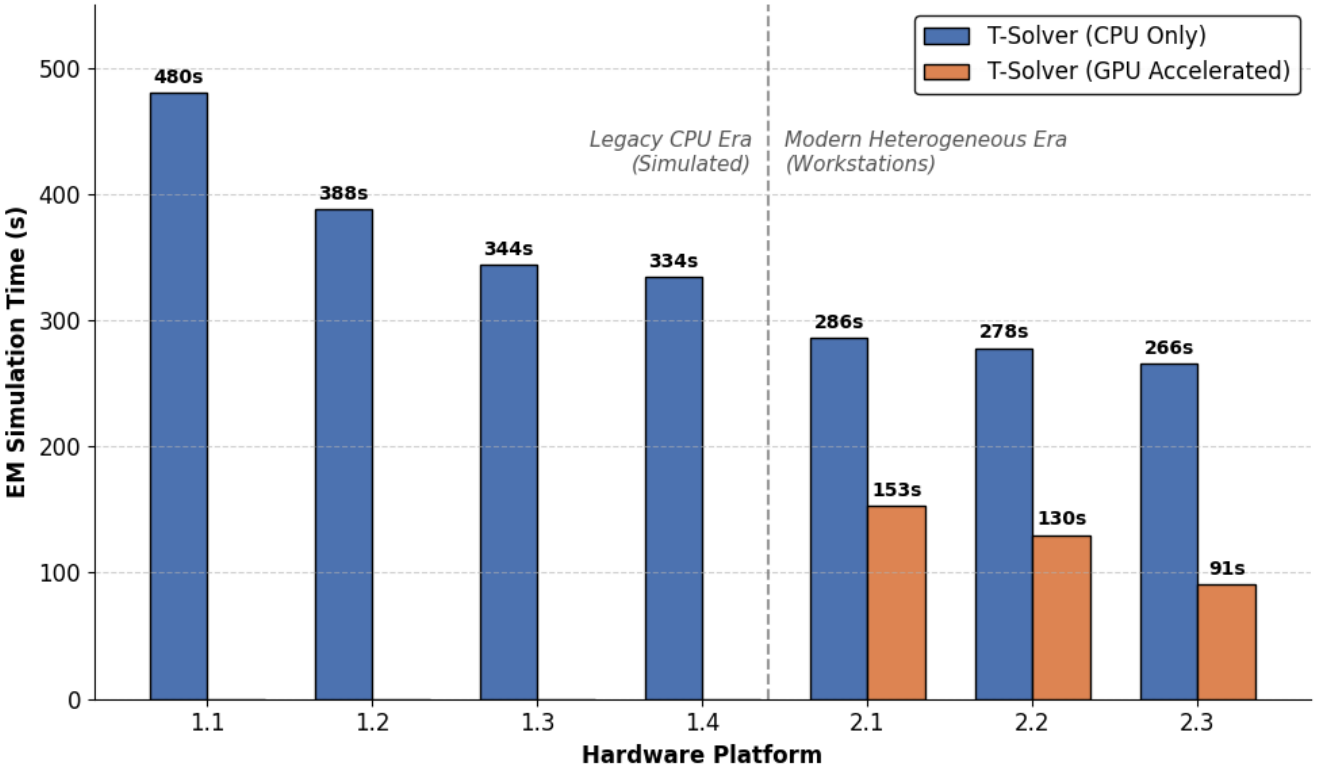

Figure 5: T-Solver Benchmark Results

3.2 Improvements on Surrogate Modelling and Bayesian Optimization

Historically, the internal algorithmic overhead of EGO—specifically the Maximum Likelihood Estimation (MLE) and the covariance matrix inversion with $\mathcal{O}(N^3)$ complexity required by Gaussian Process models—has often been considered as computationally prohibitive for high-dimensional or large-budget tasks. However, empirical benchmarking challenges this persistent dogma. As evaluated on our baseline Platform 1.1 (representing a legacy 2012 architecture), executing a 20-dimensional EGO process scaling from 256 initial samples to a 1024 evaluation budget using a classic DACE [21] implementation required a cumulative algorithmic time of merely 120 minutes. Even under such old hardware and software conditions, the algorithmic overhead averaged only about 2% of the total EM simulation cost per iteration. This cost fraction has also been continuously diluted by the evolution of mathematical software and hardware architecture. The introduction of optimized Gaussian Process regression tools (e.g., MATLAB's Statistics and Machine Learning Toolbox, since 2015 [22]) fundamentally accelerated the matrix operations, compressing the overhead by a factor of about 4. Subsequent paradigm shifts toward modern tensor-based computation frameworks (e.g., GPyTorch), even running plainly on CPU, have further marginalized this cost. On the highest-tier Platform 2.3 (2025), the optimization-to-simulation cost ratio has plummeted to a negligible 0.2%. Figure 6 explicitly illustrates this asynchronous evolutionary trend. Despite the absolute EM simulation time (grey bars) decreasing drastically over the past decade, the algorithmic cost of EGO (blue lines) has dropped at an even steeper logarithmic rate across all tested dimensionalities, cementing its status as an essentially "free" operational layer in contemporary SBO workflows.

Notably, this baseline represents the computational cost of a single scalarized problem with 768 iteration and 20 input dimensions. For higher-dimensional objective spaces, decomposition-based BO frameworks such as MOEA/D-EGO provide a more scalable alternative. Under the Das and Dennis systematic weight-generation scheme [16], the total overhead may be extrapolated from the ParEGO baseline by multiplying it by the number of scalarization subproblems m, with

$$m = \binom{H + M - 1}{M - 1} \qquad (2)$$

where $M$is the number of objectives and $H$is the number of divisions along each coordinate of the simplex.

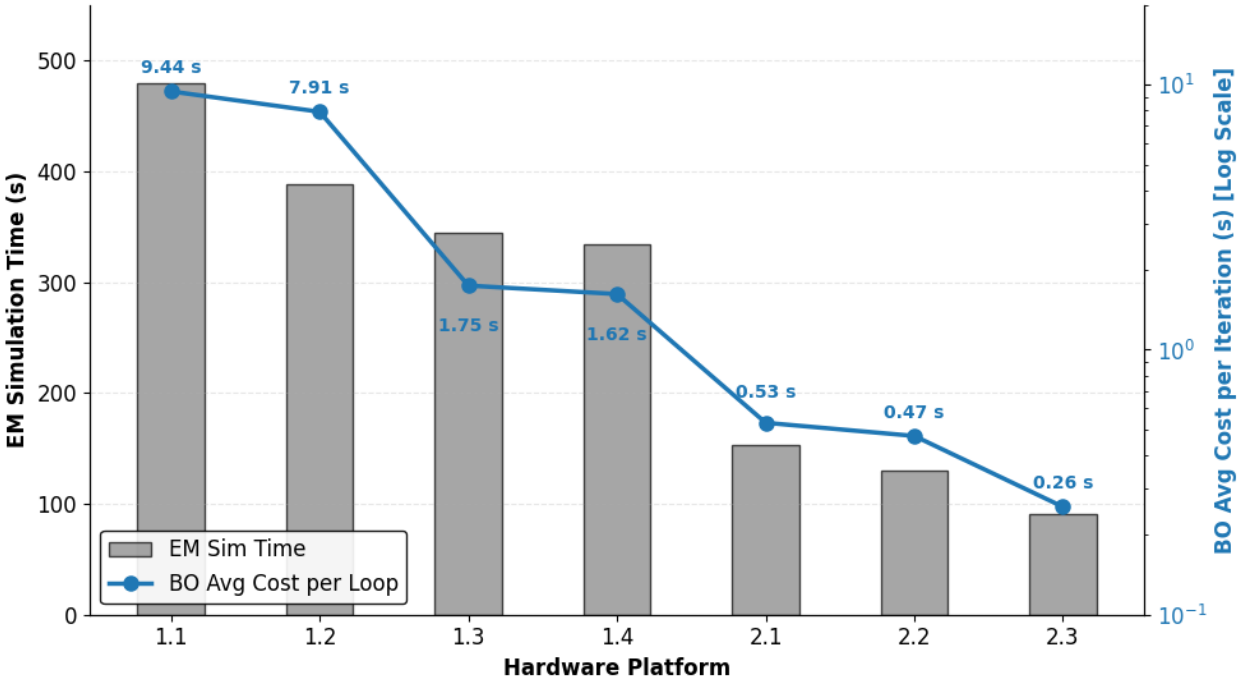

Figure 6: EM simulation time and BO Algorithm Overhead with best available algorithm at that time

This observation extends naturally to the generation-based SAEA paradigm. As illustrated in Fig. 7, the training overhead of global surrogate models across varying problem scales and hardware configurations remains negligible. Even for computationally demanding models such as large-scale Kriging and ANNs, the training cost is insignificant when compared to the dominant expense of full-batch high-fidelity EM evaluations, which typically involve tens of parallel simulations per generation. It should also be noted that, although the surrogate kriging method is the most time-consuming, it also offers the best sampling efficiency, as it achieves a similar R^2 with only 1/8 of the samples.

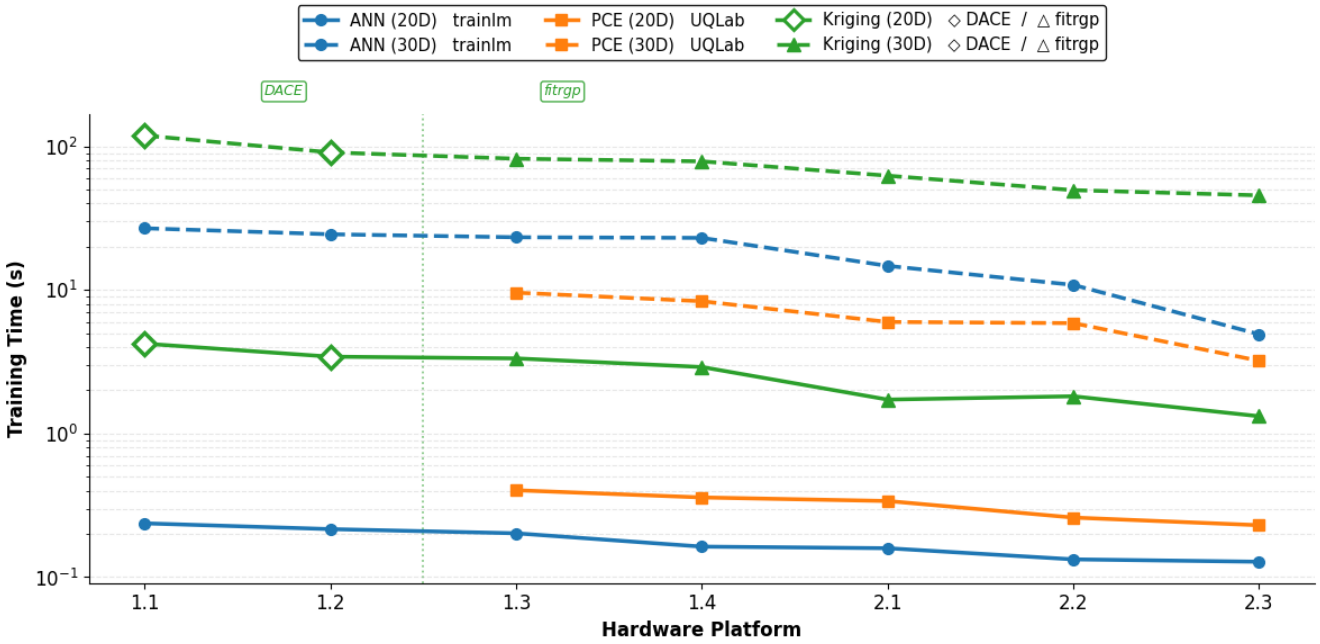

Figure 7: Training time of medium and large-scale surrogate model across platforms

Furthermore, the strength of SAEA lies in its ability to predict large candidate pools at extremely low cost. Fig. 8 reports the inference overhead for evaluating 8,192 surrogate predictions on Platform 2.1, which ranges from only a few to a few hundred milliseconds. This stark contrast in computational scale demonstrates that the surrogate model consistently functions as an ultra-fast "light engine", efficiently handling large-scale exploration, comparing with the high-fidelity "heavy-engine" EM simulation.

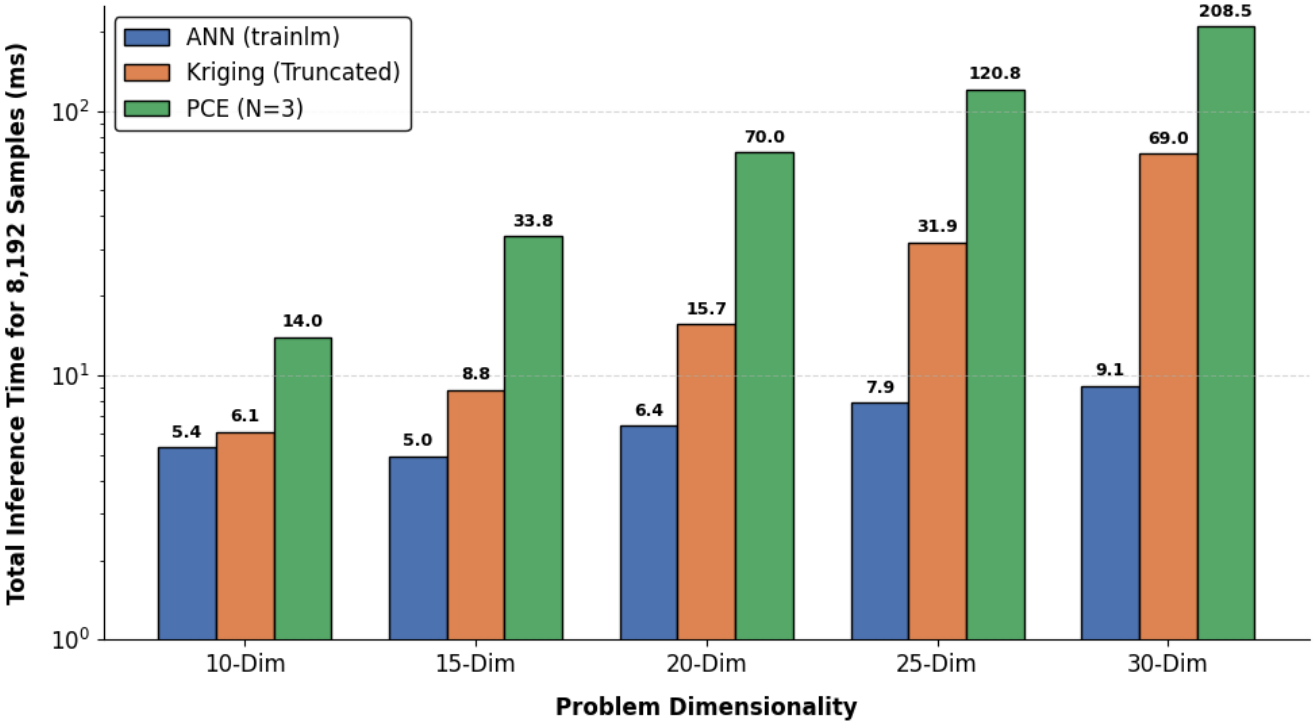

Figure 8: surrogate batch evaluation cost on platform 2.1

3.3 Discussion on Evolution Trends of Surrogate Based EM Optimization

By synthesizing the decoupled benchmarking results presented above, a clear evolution trend in surrogate-based EM optimization can be identified. A key observation is that the widely held assumption that Bayesian optimization or large-scale surrogate modelling is "computationally expensive" does not hold in practice.

When compared to the execution time of high-fidelity EM simulations—even for moderately complex structures such as the MSA-BP—the overhead of the EGO algorithm is consistently negligible. As demonstrated in our benchmark, even as early as Platform 1.1 (representative of circa 2012 performance and algorithm levels), the BO computational cost are only about 1.5% of the total EGO optimization time. This suggests that the commonly used term "computationally expensive" for BO often reflects theoretical complexity (e.g., the $\mathcal{O}(N^3)$ scaling of Gaussian Processes or the complex loop structure) rather than practical computation cost. In real-world settings, such complexity is largely mitigated by modern hardware.

In addition, the transition from Platform 1.1 to 2.1 reveals a clear pattern of asymmetric acceleration. While both EM simulation and surrogate modelling benefit from hardware improvements, the latter exhibits a significantly greater performance gain. This asymmetry is closely tied to differences in software ecosystems. Surrogate modelling and machine learning are mainstream fields and have benefited from extensive global improvement efforts, leading to highly efficient tensor-based frameworks such as PyTorch and GPyTorch. In contrast, full-wave EM solvers remain tightly coupled to specific field of demands and physics-based discretization schemes, which inherently limits their ability to fully leverage generalized hardware acceleration.

4 Conclusions and Future Perspectives

This paper proposed a decoupled benchmarking methodology to systematically investigate the impact of hardware and also algorithm evolution on the computational structure of surrogate-based optimization in electromagnetics. By separating EM simulation, surrogate training, and inference stages, the results consistently show that the algorithmic overhead of both EGO and SAEA constitutes only a negligible fraction of the total optimization cost in modern heterogeneous computing environments.

These findings suggest a fundamental shift: the practical bottleneck of surrogate-based optimization is no longer dominated by model construction or acquisition optimization, but increasingly governed by the interaction between surrogate models and search strategies. As a result, future research should place greater emphasis on control mechanisms, robustness, and stability of optimization dynamics, rather than solely on improving nominal algorithmic efficiency.

In parallel, the accessibility of advanced optimization techniques is rapidly improving. Recent developments in large language models (LLMs) with strong code-generation capabilities are lowering the implementation barrier of complex SBO pipelines. This enables EM researchers and antenna designers to more readily adopt and adapt advanced optimization frameworks, even without extensive backgrounds in machine learning or numerical optimization.

Despite the insights provided, several important directions remain for future work:

(1) Expansion to Diverse EM Solvers. This study focused on time-domain solvers. Future investigations should extend to frequency-domain solvers, which are typically CPU-bound and dominated by sparse matrix operations, as well as asymptotic ray-based solvers, where performance is closely tied to double-precision GPU capabilities.

(2) Expansion to Modern Optimization Paradigms. While this work employed ParEGO as a representative BO framework, more recent advances such as EHVI and its noise relief and batch variants (e.g., qNEHVI) [23] offer improved convergence behaviour and scalability, and also demand higher computational cost. In addition, hybrid frameworks that tightly integrate uncertainty quantification with evolutionary search, such as K-RVEA [24] and SAPEO [25], should also be considered in further benchmarking.

References

[1] J. W. Bandler, R. M. Biernacki, S. H. Chen, P. A. Grobelny, and R. H. Hemmers, "Space mapping technique for electromagnetic optimization," IEEE Trans. Microw. Theory Tech., vol. 42, no. 12, pp. 2536–2544, Dec. 1994.

[2] Jin, Y.C., "A comprehensive survey of fitness approximation in evolutionary computation," *Soft Computing* **9**, 3–12 (2005). https://doi.org/10.1007/s00500-003-0328-5

[3] A. I. J. Forrester, A. Sóbester, and A. J. Keane, "Engineering Design via Surrogate Modelling: A Practical Guide. Chichester," U.K.: Wiley, 2008.

[4] S. Koziel and S. Ogurtsov, "Rapid design optimization of antennas using space mapping and response surface approximation models," Int. J. RF Microw. Comput.-Aided Eng., vol. 21, no. 5, pp. 611–620, Sep. 2011.

[5] F. Feng, J. Zhang, W. Zhang, Z. Zhao, J. Jin, and Q. J. Zhang, "Coarse- and fine-mesh space mapping for EM optimization incorporating mesh deformation," IEEE Microw. Wireless Compon. Lett., vol. 29, no. 7, pp. 510–512, Jul. 2019.

[6] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," J. Global Optim., vol. 13, no. 4, pp. 455–492, Dec. 1998.

[7] J. Knowles, "ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems," IEEE Trans. Evol. Comput., vol. 10, no. 1, pp. 50–66, Feb. 2006.

[8] W. Ponweiser, T. Wagner, D. Biermann, and M. Vincze, "Multiobjective optimization on a limited budget of evaluations using model-assisted S-metric selection," in Proc. Parallel Problem Solving from Nature (PPSN X), 2008, pp. 784–794.

[9] Q. Zhang, W. Liu, E. Tsang, and B. Virginas, "Expensive multiobjective optimization by MOEA/D with Gaussian process model," IEEE Trans. Evol. Comput., vol. 14, no. 3, pp. 456–474, Jun. 2010.

[10] H. L. Southall and T. H. O'Donnell, "Antenna design using the efficient global optimization (EGO) algorithm," Air Force Res. Lab., Wright-Patterson AFB, OH, USA, Tech. Rep. ADA544460, May 2011.

[11] A. Tan Wei Min, Y. S. Ong, A. Gupta, and C. K. Goh, "Multi-Problem Surrogates: Transfer Evolutionary Multiobjective Optimization of Computationally Expensive Problems," IEEE Trans. Evol. Comput., vol. 23, no. 4, pp. 559–573, Aug. 2019, doi: 10.1109/TEVC.2018.2871003.

[12] A. Buhr, A. Langwost, and F. Zanella, "GPU computing advances in 3D electromagnetic simulation," presented at the NVIDIA GPU Technology Conf. (GTC), San Jose, CA, USA, May 2012. [Online]. Available: https://developer.download.nvidia.com/GTC/PDF/GTC2012/PresentationPDF/S0069-GTC2012-Advances-3D-Electromagnetic.pdf

[13] Dassault Systèmes, CST Studio Suite GPU Computing Guide, 2019. [Online]. Available: https://updates.cst.com/downloads/GPU_Computing_Guide_2019.pdf

[14] J. R. Gardner, G. Pleiss, D. Bindel, K. Q. Weinberger, and A. G. Wilson, "GPyTorch: Blackbox matrix-matrix Gaussian process inference with GPU acceleration," in Advances in Neural Information Processing Systems 31 (NeurIPS 2018), 2018.

[15] M. Balandat et *al*., "BoTorch: A framework for efficient Monte-Carlo Bayesian optimization," in Advances in Neural Information Processing Systems 33 (NeurIPS 2020), 2020.

[16] Y.F. Wei and C. Roblin, "Multislot Antenna with a Screening Backplane for UWB WBAN Applications," IJAP, Oct. 2012.

[17] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems," SIAM J. Optim., vol. 8, no. 3, pp. 631–657, Aug. 1998.

[18] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," Evol. Comput., vol. 8, no. 2, pp. 173–195, Jun. 2000.

[19] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," in Evolutionary Multiobjective Optimization, A. Abraham, L. Jain, and R. Goldberg, Eds. London, U.K.: Springer, 2005, pp. 105–145.

[20] S. Marelli and B. Sudret, "UQLab: A framework for uncertainty quantification in MATLAB," in *Proc. 2nd Int. Conf. Vulnerability Risk Anal. Manage. (ICVRAM)*, Liverpool, U.K., 2014, pp. 1–10.

[21] S. N. Lophaven, H. B. Nielsen, and J. Søndergaard, "DACE: A MATLAB Kriging Toolbox, Version 2.0," Tech. Univ. Denmark, Lyngby, Denmark, Tech. Rep. IMM-TR-2002-12, 2002.

[22] The MathWorks, Inc., “Gaussian process regression (fitrgp),” MATLAB Documentation. [Online]. Available: https://www.mathworks.com/help/stats/gaussian-process-regression.html

[23] S. Daulton, M. Balandat, and E. Bakshy, “Parallel Bayesian optimization of multiple noisy objectives with expected hypervolume improvement,” in Proc. 35th Conf. Neural Inf. Process. Syst. (NeurIPS 2021), 2021, pp. 2187–2200.

[24] T. Chugh, Y. Jin, K. Miettinen, J. Hakanen, and K. Sindhya, “A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization,” IEEE Trans. Evol. Comput., vol. 22, no. 1, pp. 129–142, Feb. 2018.

[25] V. Volz, G. Rudolph, and B. Naujoks, “Surrogate-assisted partial order-based evolutionary optimisation,” in Parallel Problem Solving from Nature – PPSN XIV, ser. Lecture Notes in Computer Science, vol. 9921, Springer, 2016, pp. 562–572.