Universiteit
Leiden
The Netherlands

Data Science and Artificial Intelligence

Multiobjective Bayesian Hyperparameter Optimization for a Traveling Salesman Problem

Leon Altfeld

Supervisors:
Furong Ye & Elena Raponi

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

10/07/2025

Abstract

We evaluate Bayesian hyperparameter optimization (HPO) for the Min–Max Ant System (MMAS) applied to the Traveling Salesman Problem (TSP), comparing different acquisition functions for single-objective optimization against the multi-objective approaches ParEGO and Expected Hypervolume Improvement (EHVI). Hyperparameters were tuned for the objectives of Anytime Fitness (AF) and the Best-Found Solution (BFS). All Bayesian Optimization (BO) frameworks are implemented using BoTorch. Our findings demonstrate that optimized hyperparameter configurations consistently outperform the default settings in terms of final solution quality. Furthermore, AF is shown to be a more reliable performance metric than solely optimizing the final best solution. For optimizing the objectives AF and BFS, we recommend HPO-UCB, tuned for AF, or ParEGO-UCB, due to their better convergence properties in the tuning process. However, EHVI is also a viable alternative.

Contents

1 Introduction

Meta-heuristic algorithms, such as Ant Colony Optimization (ACO), have demonstrated notable performance on combinatorial optimization problems, including the Traveling Salesman Problem (TSP) (MM15). Among these, the Min–Max Ant System (MMAS) (SH00) is widely used due to its strong performance. However, it heavily relies on hyperparameters to balance exploration and exploitation (PFM06). Selecting appropriate hyperparameter configurations is crucial for achieving high-quality solutions, but is often time-consuming and problem-dependent.

Hyperparameter optimization (HPO) aims to automate this search, making HPO accessible to inexperienced practitioners and often results in better performance (KPM+23). While Traditional HPO approaches typically focus on a single-objective (YDWB21), real-world optimization problems require balancing multiple potentially conflicting objectives.

Multi-objective hyperparameter optimization (MOHPO) methods have been proposed to address multiple objectives simultaneously. Techniques, including grid search, random search, scalarization-based methods, and evolutionary algorithms (EAs), have been extended to tackle the multi-objective setting (KPM+23). However, model-based methods such as Multi-Objective Bayesian Optimization (MOBO) methods offer a promising alternative due to their sample efficiency in black-box optimization tasks (RC25). MOBO has demonstrated success in various domains, including aerodynamic design (ZPS19) and MOHPO (CPA22). Despite this, its application for tuning the hyperparameters of meta-heuristic solvers for combinatorial optimization problems remains relatively under-explored.

Previous work on hyperparameter tuning of meta-heuristic algorithms has focused on single-objective HPO. Opez-Ibáñez et al. propose automated tuning for a MMAS using Anytime Fitness (AF) as an objective (OIS12), while Yin and Wijk (YW21) apply Bayesian Optimization (BO) to ACO, optimizing for the Best-Found Solution (BFS). Ye et al. (YDWB21) compare different HPO methods for tuning a Genetic Algorithm (GA) for Expected Running Time (ERT) and AF respectively, and propose a multi-objective approach to HPO for meta-heuristics.

Building on these foundations, this thesis investigates the application of MOBO to tune the hyperparameters of an MMAS for the TSP using two objectives, AF and BFS. Our research evaluates the performance of MMAS across multiple TSPLIB benchmark instances (Rei91), comparing MOBO to random search and single-objective methods.

This research aims to address the following research question:

How can Multi-Objective Bayesian Optimization be used to optimize hyperparameters for TSP solvers?

To investigate this, we consider the following sub-questions:

- Can multi-objective hyperparameter optimization enhance the performance of ACO-based TSP solvers?

- How do MOBO methods compare to methods like random search and traditional hyperparameter optimization when applied to a multi-objective hyperparameter optimization task?
- What objectives can be considered for the multi-objective hyperparameter optimization of ACO-based TSP solvers?

Addressing these questions, this research makes the following contributions:

- We apply Multi-Objective Bayesian Optimization to tune the hyperparameters of a Min–Max Ant System for solving TSP instances from the TSPLIB library.
- We compare Multi-Objective Bayesian Optimization with traditional hyperparameter optimization methods, including random search and single-objective BO.
- We evaluate solver performance based on Anytime Fitness and the Best-Found Solution.

The code repository of this thesis can be accessed via the following link: https://github.com/LeonAQA/BachelorThesis_MOHPO. This thesis is organized as follows. This chapter contains the introduction; Section 2 discusses related work; Section 3 describes the methodology. Section 4 describes the experiments; Section 5 demonstrates the results; Section 6 concludes this thesis, and suggests potential directions for further research.
This bachelor's thesis was conducted at the Leiden Institute of Advanced Computer Science (LIACS), Leiden University, under the supervision of Dr. Furong Ye and Dr. Elena Raponi.

2 Related Work

In this section, we discuss prior research conducted on HPO, BO, MOHPO, and meta-heuristic solvers for the TSP. We focus on methods relevant to our application of MOBO to tuning meta-heuristic algorithms.

2.1 Hyperparameter Optimization

Grid and random search are commonly used baseline methods for HPO. Grid search is an exhaustive search method that systematically evaluates all combinations from a user-specified hyperparameter grid. This process is computationally expensive and can miss high-performing configurations that lie in between grid points (YZ20).

Random search samples configurations from an underlying distribution and is often preferred due to better empirical performance (BB12). However, both methods treat evaluations independently and do not use prior evaluations to influence future searches.

In contrast, evolutionary algorithms (EAs) guide the search following evolutionary-inspired principles such as generations, parents, offspring, recombination, and selection (YDWB21). While this form of guided search outperforms the naive baselines (BBL+21), it lacks an approximation of the underlying objective function. BO uses probabilistic surrogates to model the objective function and an acquisition function to select promising configurations (SSW+16).

2.2 Bayesian Optimization

BO is a surrogate-based framework known for its sample efficiency in solving expensive black-box optimization problems (HLGR+16). BO constructs and updates a probabilistic model of the objective function and selects new evaluation points by maximizing an acquisition function that balances exploration and exploitation (SSW+16).

There are several possible acquisition functions, including Probability Improvement (PI), Expected Improvement (EI), and Upper Confidence Bounds (UCB). De Ath et al. find UCB and EI to comparatively balance the exploration-exploitation tradeoff effectively, highlighting the importance of a mostly greedy search approach (DAERF21).

Gaussian processes are widely used as surrogate models (KPM+23). However, they can struggle in higher-dimensional spaces, leading to the search for alternative surrogates such as random forests (HB16). Neural network-based surrogates have been proposed for improved scalability with an increasing number of observations (SRS+15).

BO has been successfully applied to reinforcement learning (BCdF10), combinatorial optimization (HHLB11), and hyperparameter tuning (WCZ+19). Although traditional BO focuses on single-objective optimization, recent work extends it for multi-objective optimization (MOO), providing a broader perspective of objective interactions by constructing a Pareto front of non-dominated solutions (MJ11).

2.3 Multi-Objective Hyperparameter Optimization

Single-objective random search, grid search, evolutionary algorithms, and model-based methods can be extended to MOHPO (KPM+23).

Multi-objective evolutionary algorithms (MOEAs) are a well-established family of MOHPO methods (KPM+23). Popular MOEAs include Pareto-based methods, such as NSGA-II (DPAM02), decomposition-based methods, like MOEA/D (ZL07), and indicator-based methods, with SMS-EMOA (HNE07). While MOEAs demonstrate competitive performance (TIO17) on a wide range of tasks and generate new candidate solutions more rapidly than MOBO (LTRE20), they are often sample-inefficient, due to the reliance on many objective-evaluation calls to find competitive solutions (RC25). This limits their suitability for black-box optimization problems, motivating the application of MOBO.

2.4 Multi-Objective Bayesian Optimization

MOBO acquisition strategies are categorized into scalarization-based approaches, combining multiple objectives into a singular one, and Pareto-based approaches, using one surrogate per objective (KPM+23).

A common scalarization-based method is ParEGO (Kno06), which uses a set of weight vectors to explore the search space. It uses a variation of the Tchebycheff function to combine multiple objectives into a single scalarized objective (Kno06).

In contrast, Expected Hypervolume Improvement (EHVI) utilizes a separate surrogate for each objective, optimizing the expected increase in the Hypervolume (HV) dominated by the Pareto front. This strategy quantifies the gain of volume relative to a reference point to predict new candidate solutions (EGN06a).

Zuhal et al. (ZPS19) compare ParEGO and EHVI across multiple material design optimization problems. Their work demonstrates that EHVI offers a more diverse Pareto front, while ParEGO is more computationally efficient.

Despite its strengths, EHVI's computational complexity suffers in high-dimensional problem settings, as it scales poorly with the number of objectives (BKJ+20). Variations such as qEHVI (DBB20a) try to mitigate this by extending EHVI to enable parallel evaluations leveraging differentiation.

Recent strategies such as PESMO (HLHLSA16) and USeMO (BDJD22) offer improved decision-making under uncertainty and in higher-dimensional scenarios. However, PESMO requires nested Monte Carlo entropy estimates, and USeMO relies on multiple surrogate optimizations per objective, making them computationally expensive in resource-constrained settings.

2.5 Combinatorial Optimization and TSP-solvers

The TSP is a classical combinatorial NP-hard problem. Slightly modified, it has wide-ranging applications in fields such as logistics, network design, and DNA sequencing (GTSS22). The solution space of a TSP scales factorially with the number of cities, making exhaustive search methods infeasible for large instances (KNRS24).

To tackle such instances, meta-heuristic methods such as Simulated Annealing (SA) (And90), GAs (GP03), and ACO (DMC96) have been proposed to approximate near-optimal solutions.

Among these, the ACO, particularly the MMAS variant, shows the best performance (SH00), finding better solutions compared to GA and SA under the same budget (TGHD22).

Recent Works have explored HPO methods for combinatorial solvers (OIS12; YW21; PSB10). Pellegrini et al. compare online and offline tuning methods on an MMAS, finding that offline approaches yield superior results (PSB10). Opez-Ibáñez et al. apply automated tuning methods to MMAS for optimizing anytime performance (OIS12), while Yin and Wijk apply BO for an ACO TSP-solver focusing on BFS (YW21).

However, these works limit their focus to a single objective. Our work addresses this gap by applying MOBO to tune the hyperparameters of an MMAS for solving a TSP. We optimize for AF and BFS across multiple TSPLIB benchmark instances (Rei91).

3 Methodology

This work focuses on MOBO for tuning a variation of the MMAS applied to symmetric TSPs. We compare the performance of single- and multi-objective BO, with both approaches evaluated on the objectives, AF, and the BFS.

For single-objective BO, we evaluate UCB and Log Expected Improvement (LogEI) as acquisition functions. For multi-objective BO, we consider ParEGO applied with LogEI and UCB as well as EHVI, a Pareto-based acquisition strategy. The following sections provide an overview of each component.

3.1 Traveling-Salesman-Problem

The TSP is a classical NP-hard problem. Given a weighted graph V and a distance function D, the goal is to find a tour π visiting each city exactly once and returning to the starting point, minimizing the total travel cost.

Variants of the TSP include symmetric and asymmetric versions. In the symmetric TSP, distances are direction-independent: $D(v_1, v_2) = D(v_2, v_1)$. In contrast, the asymmetric TSP allows direction-dependent distances: $D(v_1, v_2) \neq D(v_2, v_1)$.

We focus on the symmetric TSP with Euclidean distances. Meta-heuristic algorithms such as ACO achieve more reliable convergence and better solution quality (OKN+17) on such instances. These properties make the symmetric TSP well-suited for benchmarking.

3.2 Meta-heuristic Algorithms

Meta-heuristic Algorithms include a wide range of algorithms, such as GAs, SA, and ACO. These algorithms do not guarantee optimality but can approximate a near-optimal solution. This makes them suitable for large search spaces where exact methods become infeasible.

Meta-heuristics rely on various hyperparameters to strike a balance between exploration and exploitation. These parameters must be carefully tuned to ensure robust convergence (OIS12).

3.2.1 Ant Colony Optimization

ACO was first proposed by Dorigo et al. (DMC96) and is inspired by the foraging behavior of ants. Tosoni et al. (TGHD22) found that ACO outperforms other meta-heuristic algorithms, such as Particle Swarm Optimization, SA, Tabu search, and GAs, on TSP benchmarks.

When applied to the TSP, artificial ants construct solutions by probabilistically selecting the next node based on pheromone levels. Pheromone levels are updated each iteration, with better solutions receiving higher pheromone levels. As such, frequented paths receive higher reinforcement, making them more likely to be traversed in subsequent iterations.

The probabilistic selection p for ant $k \in N_{ants}$ for moving from node i to node j at iteration t is based on the current pheromone level τ_{ij} raised to the power of α and the heuristic term η_{ij} raised to a power of β. For the TSP, η_{ij} is defined as the inverse of the distance between i and j (SH00). With $\mathcal{N}_i$ denoting the set of cities not yet visited p_{ij} is given by:

$$p_{ij}^k(t) = \frac{[\tau_{ij}(t)]^\alpha \cdot [\eta_{ij}]^\beta}{\sum_{l \in \mathcal{N}_i^k} [\tau_{il}(t)]^\alpha \cdot [\eta_{il}]^\beta} \quad \text{if } j \in \mathcal{N}_i^k \tag{1}$$

Multiple variants of ACO have been proposed, including the Rank-based Ant System (BHS99), and the MMAS (SH00). MMAS extends the original ACO by aiming to exploit the BFS more strongly and avoid premature convergence (SH00), making the MMAS particularly effective on large TSP instances.

The MMAS uses a modified pheromone update that only reinforces the global best solution (SH00). Here τ_{ij} denotes the pheromone level on edge (i, j) and ρ is the evaporation.

$$\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \sum_{k=1}^{N_{ants}} \Delta\tau_{ij}^{best}(t) \tag{2}$$

$$\Delta\tau_{ij}^{\text{best}}(t) = \begin{cases} \frac{1}{L_{\text{best}}}, & \text{if edge } (i,j) \text{ is in the best tour} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

The pheromone levels τ are limited to the range between τ_{min} and τ_{max}, to avoid premature convergence (SH00).

$$\tau_{min} \leq \tau \leq \tau_{max} \tag{4}$$

τ_{min} and τ_{max} are dynamically updated based on the BFS. With N as the number of cities, we follow Zhang et al.'s definition of τ_{min} (ZCZ14) and with L_{best} as the optimal tour, τ_{max} (DS04) is calculated by:

$$\tau_{min} = \frac{\tau_{max}}{2N} \tag{5}$$

$$\tau_{max} = \frac{1}{\rho \cdot L_{best}} \tag{6}$$

During initialization, the pheromone levels are set to τ_{max} to encourage exploration. The optimal tour length, L_{best}, is initially estimated using the nearest neighbor greedy solution.

Additional enhancements include 3-opt local search, restart strategies, and candidate lists.

Our implementation is based on the core MMAS framework, but it uses a simplified version of certain components. We replace 3-opt with 2-opt for local search, as well as omit the candidate list and restart mechanisms. These modifications were chosen to balance our limited computational resources while maintaining the core functionality of MMAS

All hyperparameters involved in the tuning process are listed in Table 1.

Parameter	Interpretation
α	Relative importance of pheromone information
β	Relative importance of heuristic trails
ρ	Pheromone evaporation rate
N_{ANTS}	Number of ants

Table 1: Hyperparameters used in MMAS.

3.3 Objective Functions for tuning ACO

The objectives used to evaluate the MMAS implementation for optimization are AF (YLC24) and the BFS.

Given a TSP instance, the best found tour is denoted by π. We compute the average π_{avg} over independent runs R.

$$\pi_{\text{avg}} = \frac{1}{R} \sum_{r=1}^{R} \pi^{(r)} \tag{7}$$

We express best-found as the deviation from the optimal solution π^* as a percentage clipped between 0 and 1, π_{dif}.

$$\pi_{dif} = \frac{\pi_{avg} - \pi^*}{\pi^*} \tag{8}$$

Because our optimization framework assumes maximization by default, we optimize for the negated deviation $-\pi_{dif}$ to ensure higher values correspond to closer proximity to the optimal solution.

Our AF implementation follows the method proposed by Ye et. al (YLC24). AF calculates the Empirical Cumulative Distribution Function (ECDF) for a given set of log-spaced targets $\{\tau_1, ..., \tau_K\}$. For each target τ_j and run r, we define $h_{j,r} \in [0, 1]$ as the fraction of total evaluations needed to reach τ_j. With $t \in [0, 1]$ as normalized evaluation time at which the target is reached, the ECDF is computed as:

$$\text{ECDF}(t) = \frac{1}{KR} \sum_{j=1}^{K} \sum_{r=1}^{R} \mathbf{1} \{h_{j,r} \leq t\}, \quad t \in [0, 1] \tag{9}$$

The final AF is the Area Under the Curve (AUC) of the ECDF.

$$\text{Anytime Fitness } = \int_0^1 \text{ECDF}(t) \mathrm{d}t \tag{10}$$

We focus on the two objectives, AF and BFS. Combining both enables a more detailed evaluation of the convergence behavior and the quality of the final solution.

For each configuration, both objectives are averaged over $R = 5$ runs per instance and aggregated across all benchmark instances.

3.4 Optimization Setup

This section introduces the optimization strategies employed, which include single-objective BO, two MOBO methods — ParEGO and EHVI — as well as a random search baseline.

3.4.1 Bayesian Optimization

BO is a sample-efficient framework for global optimization of black-box problems. BO is based on a probabilistic surrogate model and an acquisition function. The surrogate model quantifies the uncertainty, and the acquisition function quantifies the importance of new samples.

This work focuses on the commonly applied Gaussian Process (GP) as a surrogate model. Rasmussen (Ras04) defines the GP f as a distribution over functions, where for any set of inputs $\{x_1, x_2, ...x_n\}$ the corresponding outputs $f(x_1), f(x_2), f(x_n)$ follow a multivariate normal distribution. The GP consists of a mean and a kernel function.

The mean function $m(x)$ represents the expected value of the function for an input x.

$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})] \tag{11}$$

The kernel function $k(x, x')$ encodes both the uncertainty of the function and the similarity between inputs.

$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}\left[(f(\mathbf{x}) - m(\mathbf{x}))\left(f(\mathbf{x}') - m(\mathbf{x}')\right)\right] \tag{12}$$

The GP is defined as:

$$f(\mathbf{x}) \sim GP\left(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\right) \tag{13}$$

Using the mean and kernel function, the GP provides both a prediction and the uncertainty estimate for new inputs, which are critical for guiding the acquisition function.

The acquisition function, using the GP's mean and variance, is optimized to strike a balance between exploring uncertain regions and exploiting high-performing regions. For single-objective optimization, we evaluate UCB, a standard acquisition function, and LogEI, a more robust variation of Expected Improvement (ADE+25).

UCB has separate terms for exploration through predicted uncertainty (σ) and exploitation through the predicted mean (μ). These can be balanced by adjusting the parameter β, scaling the importance of sampling uncertain regions. De Ath et al. recommend prioritizing exploitation while still encouraging occasional exploration (DAERF21). Aligning with this, we set the exploration scalar in UCB to $\beta_{ucb} = 0.2$.

$$\text{UCB}(x) = \mu(x) + \beta_{ucb}\sigma(x) \tag{14}$$

EI quantifies the expected gain of the GP-prediction $f(x)$ over the current best-observed value f_{best}. LogEI modifies this by applying a logarithmic transformation, improving robustness, particularly in flat regions of the objective space (ADE+25).

$$\text{LogEI}(x) \;=\; \log\Big(\mathbb{E}\Big[\max\big(f(x) - f_{\text{best}},\, 0\big)\Big]\Big) \tag{15}$$

Initial sampling provides the training data to fit the GP, with more samples leading to a better initial approximation. However, in a fixed-budget experiment, this comes at the cost of fewer Bayesian evaluation steps. Balancing this tradeoff is crucial for initializing a Bayesian framework (DOW+22).

To minimize the discrepancy of random initial samples, we use Sobol sampling. Sobol sampling is a quasi-random, low-discrepancy sequence offering better space-filling capabilities than uniform random sampling. Tarantola et al. recommend Sobol for globally sensitive methods (TBZ12). Our setup utilizes 5 Sobol-initialized samples, which are shared across all Bayesian approaches to reduce computation and ensure fairness.

3.4.2 Multi-Objective Bayesian Optimization

The MOBO methods we consider are ParEGO (Kno06) and EHVI (EGN06b). These are popular methods in MOBO and follow different approaches to multiple objectives, making a comparison relevant. Both are available in BoTorch, allowing for a direct comparison using a single framework.

More recent methods, such as PESMO (HLHLSA16) and UseMO (BDJD22), are not considered due to their computational and implementation complexity, making them less suitable for our experimental setting.

ParEGO is a scalarization-based MOBO method introduced by Knowles (Kno06). It converts multiple objectives into a single objective by sampling weight vectors λ to express scalarization preferences on our k objectives $f_1, ..., f_k$ at each iteration. The weight vectors and our objectives are combined into a single objective through the augmented Tchebycheff function (Kno06).

$$f_\lambda(x) = \max_{j=1}^{k} (\lambda_j . f_j(x)) + \rho_{pgo} \sum_{j=1}^{k} \lambda_j . f_j(x) \tag{16}$$

The max term ensures we can attain non-supported Pareto-optimal points while the ρ_{pgo}-scaled linear term penalizes solutions weakly dominated by the Pareto-front. We set $\rho_{pgo} = 0.05$, following the original implementation by Knowles (Kno06).

We compare ParEGO using LogEI and UCB. Over iterations, this process provides a diverse set of solutions, each optimized using a different weighting of the objectives. The collected solutions approximate a Pareto front by covering different trade-offs in the objective space. To reduce randomness and ensure the search space is fairly represented, we use Sobol sampling to generate weight vectors.

Because ParEGO uses scaling to combine multiple objectives into a singular one, objectives must be normalized, posing difficulties for the BFS when optimal solutions are unknown. A common workaround is to estimate the BFS using long runs. In our case, we use benchmark TSP instances

with known optima, avoiding the need for heuristic estimation, which allows us to focus on evaluating the feasibility and comparative performance of different MOBO methods under controlled conditions.

Expected Hypervolume Improvement, proposed by Emmerich et al. (EGN06b), is a Pareto-based acquisition function that optimizes for the expected gain in HV. EHVI fits an independent surrogate model for each objective and computes the EI in dominated volume relative to the current Pareto set $\mathcal{P}$ and a reference point r for a candidate x.

HV of a Pareto front $\mathcal{P} = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$ for a reference point r is defined as the volume of the region in the d-dimensional objective space dominated by $\mathcal{P}$ and bounded by r (YEDB19). With λ_d as the Lebesgue measure, HV is defined as:

$$\mathrm{HV}(\mathcal{P}) = \lambda_d \left(\cup_{\mathbf{y} \in \mathcal{P}} [\mathbf{r}, \mathbf{y}] \right) \tag{17}$$

Given a predicted objective $f(x)$ for a candidate x the hypervolume improvement (HVI) (YEDB19) is:

$$\mathrm{HVI}(f(x), \mathcal{P}) = \mathrm{HV}(\mathcal{P} \cup \{f(x)\}) - \mathrm{HV}(\mathcal{P}) \tag{18}$$

EHVI uses the expectation of this improvement under the GP (DBB20b):

$$\mathrm{EHVI}(x) = \mathbb{E}[\mathrm{HVI}(\mathcal{P} \cup \{f(x)\})] \tag{19}$$

We use a static reference point set to $[-0.05, -1.05]$, chosen to lie below all feasible objective values. This ensures that even marginal improvements in the worst-case regions contribute to the HV gain. The use of a static reference point ensures consistency and aligns with the approach proposed by Yang et al. (YEDB19).

EHVI is well suited for problems with conflicting objectives, as it explicitly targets improvements to the Pareto front and encourages solutions that are diverse and well-balanced (ZPS19). While EHVI becomes computationally expensive in many-objective settings due to the complexity of HV computations, its cost remains manageable in our bi-objective setup (BKJ+20).

Zuhal et al. (ZPS19) evaluate ParEGO and EHVI on different bi-objective material design problems. They find that EHVI consistently outperforms ParEGO in terms of HV and the quality of the Pareto front. However, their results show that ParEGO remains a viable alternative in computationally constrained settings, offering faster acquisition steps and competitive performance. These findings support the decision to compare these methods in an HPO setting, optimizing for AF and BFS.

3.4.3 Random Search

Random Search provides a simple baseline optimization strategy that samples configurations uniformly from a predefined search space. It has been shown to outperform grid search methods for HPO in high-dimensional spaces (BB12). In this thesis, we use random search as a baseline for both HPO and MOHPO methods. Random search uniformly samples a hyperparameter tuple $(\alpha, \beta, \rho, N_{ANTS})$ from our predefined bounds. This configuration is then evaluated on the test set

based on AF and BFS. We use the same budgets and hyperparameter bounds as for our Bayesian methods to ensure a fair comparison.

3.5 Experimental Setup

All tuning methods were given 50 iterations. Each MMAS was evaluated under a budget of 500 and 2000 function evaluations.

3.5.1 TSP-Instances

We evaluated each optimization framework on 14 TSP instances of varying sizes. To reduce the inherent variance of meta-heuristic algorithms, each configuration was run 5 times per instance, using unique deterministic seeds to ensure the reproducibility of the experiment and independence for separate runs.

To assess generalization, the performance was tested on five separate test instances not used during optimization. Each configuration was run 10 times on the test instances to ensure the reliability of the findings.

The training instances were selected to span a range of varying numbers of cities and distances. The number of cities and optimal tour lengths are reported in Table 2.

Instance	**Cities**	**Optimum**
st70	70	675
kroA100	100	21282
eil101	101	629
lin105	105	14379
bier127	127	118282
ch130	130	6110
u159	159	42080
d198	198	15780
tsp225	225	3916
gil262	262	2378
a280	280	2579
rd400	400	15281
fl417	417	11861
p654	654	34643

Table 2: Selected TSPLIB instances for training sorted by increasing number of cities, showing the known optimum.

Test instances (Table 3) were chosen to reflect the diversity of the training set, while avoiding overlap with training instance families.

Instance	Cities	Optimum
berlin52	52	7542
pr152	152	73682
ts225	225	126643
pcb442	442	14379
d657	657	50778

Table 3: Selected TSPLIB instances for testing sorted by increasing number of cities, showing the known optimum.

3.5.2 Hyperparameter Ranges

The Hyperparameter ranges are summarized in Table 4. These are derived from Pellegrini et al.'s study on the effect of parameters for MMAS (PFM06), López-Ibáñez and Stützle's HPO method (OIS12), and the default parameter settings for the MMAS (OIS12).

The bounds for α combine the default parameters mentioned by López-Ibáñez and Stützle (OIS12) and the values proposed by Pelligrini et al. (PFM06), expanded to explore larger and finer-grained parameter interactions. For ρ, β and N_{ants} we adopt the ranges of López-Ibáñez and Stützle (OIS12) excluding extreme values ($\rho = 1$, $\beta = 0$, $\alpha = 0$) to prevent complete pheromone loss or neglect of either heuristic or pheromone information.

Parameter	Our Bounds	Effect of Parameters	HPO	Default Settings
α	[0.5, 5.0]	{1, 2, 3}	Not specified	1.0
β	[1.0, 20.0]	{2, 3, 4, 5}	[0, 20]	2.0
ρ	[0.01, 0.99]	{0.02, 0.04, 0.06, 0.08}	[0.01, 1.0]	0.2
N_{ANTS}	[1, 100]	{0, 100, 200, 300}	[1, 100]	25

Table 4: Hyperparameter ranges for MMAS with literature settings (OIS12; PFM06).

3.5.3 Tools

Initial sampling was performed using Sobol sampling provided by PyTorch. For the optimization framework, we use BoTorch (BKJ+20), an open-source library built on PyTorch for BO.

For our single-objective and ParEGO methods, we use *SingleTaskGP* provided by BoTorch (BKJ+20) as the Gaussian Process. *SingleTaskGP* uses the zero-mean function and a 5/2 maternal kernel by default. For EHVI, we use *ModelListGP* and two separate *SingleTaskGPs*, one for each objective. Furthermore, the acquisition functions logEI, UCB, and EHVI are provided by BoTorch. These are optimized using *optimize_acqf* (BKJ+20).

The TSP instances were sourced from TSPLIB (Rei91), a widely used benchmark library for combinatorial optimization with known optimal solutions.

4 Results

This section evaluates the performance of our optimization methods. We begin by comparing the convergence behavior of our single-objective methods, UCB and LogEI, optimizing either the BFS or AF. We evaluate our multi-objective methods by comparing ParEGO using UCB and LogEI with EHVI in terms of convergence, HV, and the final Pareto front. Finally, we compare the best hyperparameter configurations of each approach on the test set. From here on out, EI will be used synonymously with LogEI. Furthermore, we introduce the notation *method-acquisition function-metric*, where e.g., HPO-EI-AF stands for single-objective BO using the Expected Improvement acquisition function focusing on Anytime Fitness, or EHVI-AF as Expected Hypervolume Improvement focusing on Anytime Fitness.

4.1 Convergence Comparison

To visualize the difference in convergence behavior between objectives, we use separate convergence plots for AF and the BFS. For the multi-objective methods, we also examine the HV increase to understand how the different objectives impact the HV space.

4.1.1 Single-Objective

Figure 1 shows the average convergence curves for UCB and EI optimizing either the BFS or AF under two different budgets, 500 (below) and 2000 (above) function evaluations. Each line represents the best-observed objective value across iterations, averaged over five runs.

For 2000 function evaluations on the left, when observing AF, UCB-AF achieves the highest final AF ($\sim$*0.79*), followed by UCB-BF ($\sim$*0.78*). Interestingly, EI-BF, while converging more slowly, converges to a similar AF as EI-AF ($\sim$*0.77*), indicating that when using EI, optimizing for BFS can yield strong AF.

On the right, BFS, UCB-AF, and UCB-BF converge to a similar, low error value ($\sim$*0.6*), with EI-BF ($\sim$ *0.75*) achieving a slightly worse score. EI-AF converges more quickly and ends with a similarly high error.

Random Search performs poorly on both metrics with minimal improvement over iterations.

The 500 function evaluation plots mirror the interactions observed for 2000 function evaluations, in finding that UCB-based methods outperform their EI-based counterparts. However, terminating at lower AF values and higher error rates. Interestingly, EI is more competitive in the 500-function evaluation setting, outperforming the UCB-BF method on the AF objective.

Overall, UCB-based methods consistently outperform EI on both objectives, with UCB-AF achieving the highest AF and the lowest error.

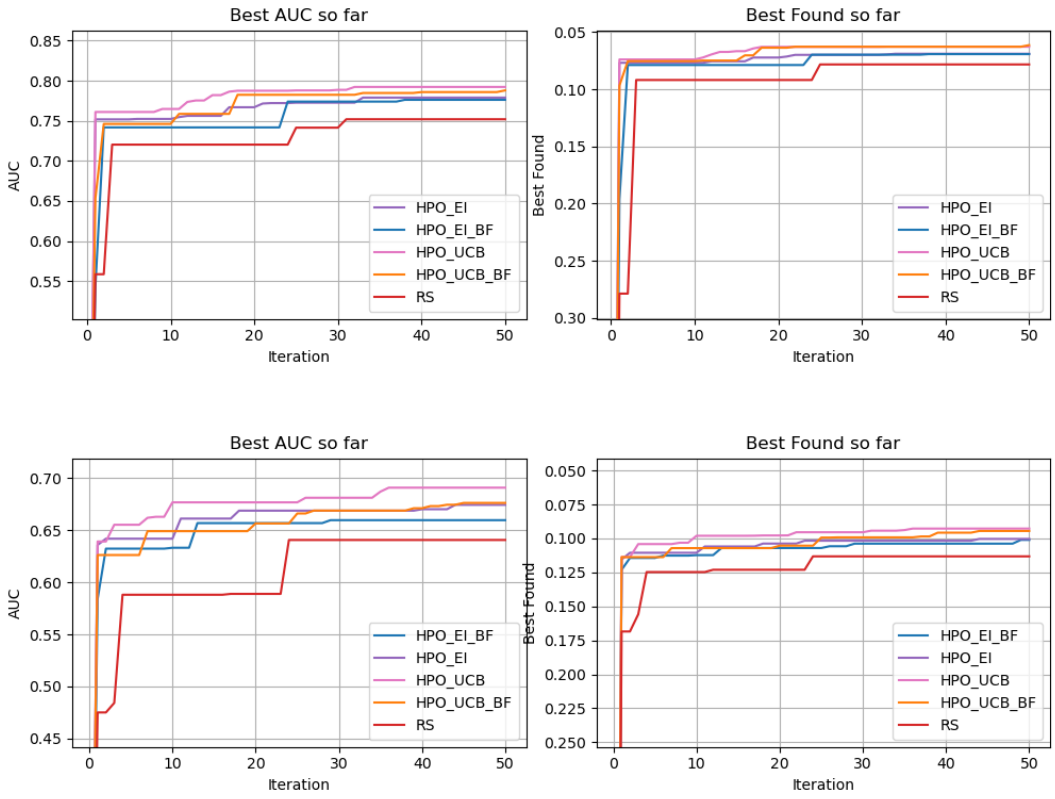

Figure 1: Comparison of the average Anytime Fitness (left) and Best Found Solution (right) during the tuning process over 50 iterations for Single-Objective Bayesian Optimization methods (EI, UCB) with a random search baseline. The results for tuning a Min-Max Ant-system with 500 function evaluations are shown below, and for 2000 function evaluations are shown above.

4.1.2 Multi-Objective

Convergence across objectives Figure 2 shows the best-so-far values for AF and BFS over 50 iterations for ParEGO using EI and UCB, EHVI and Random Search.

For 2000 function evaluations (above) on the left, ParEGO-UCB achieves the highest final AF ($\sim$*0.78*), followed by ParEGO-EI ($\sim$*0.775*), with EHVI slightly below ($\sim$*0.77*).

On the right, ParEGO-UCB again outperforms the other methods, achieving the lowest error value ($\sim$*0.65*), which indicates strong final solution quality. EHVI is competitive, achieving ($\sim$*0.67*). Interestingly, ParEGO-EI shows a similar performance to Random Search, both of which terminate at around *0.75*. These results indicate that ParEGO-UCB offers the strongest performance across both objectives and EHVI is a competitive alternative.

Random Search exhibits poor anytime performance but is comparable to ParEGO-EI in terms of final solution quality.

The plots for 500 function evaluations also show ParEGO-UCB outperforming the other methods on both metrics. EHVI exhibits competitive performance early on but stagnates after iteration 15. ParEGO-EI converges more slowly early on but outperforms EHVI in later iterations. Random Search is less competitive for 500 function evaluations than for 2000 function evaluations.

Overall, ParEGO-UCB performs best among the multi-objective methods.

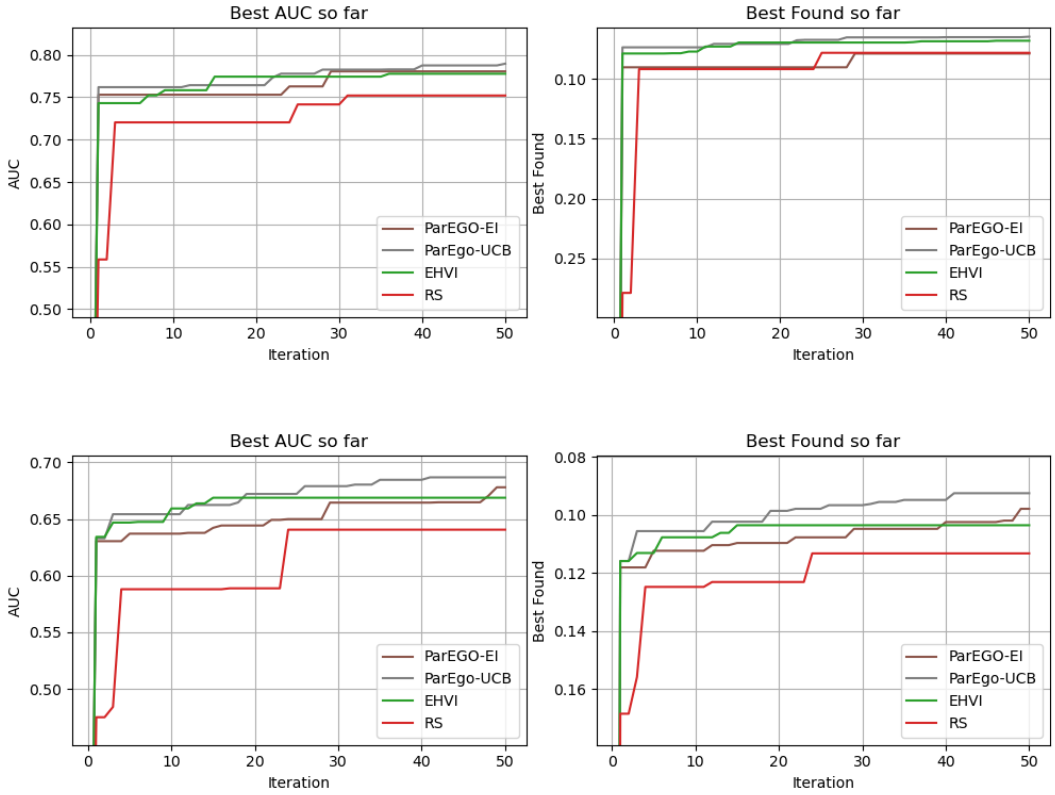

Figure 2: Comparison of the average Anytime Fitness (left) and Best Found Solution (right) during the tuning process over 50 iterations for Multi-Objective Bayesian Optimization methods (ParEGO-EI, ParEGO-UCB, and EHVI) with a random search baseline. The results for tuning a Min-Max Ant-system with 500 function evaluations are shown below, and for 2000 function evaluations are shown above.

Hypervolume Figure 3 shows the HV increase over time, averaged across five runs. For 2000 function evaluations, ParEGO-EI initially achieves the steepest increase in HV, outperforming the other methods up to iteration 11. After the 11th iteration, EHVI overtakes the other methods. However, both ParEGO methods overtake EHVI in later iterations, with ParEGO achieving the highest HV final hypervolume ($\sim$*0.838*).

For 500 function evaluations, the plot shows that ParEGO-UCB achieves the highest HV and the quickest increase. EHVI increases consistently but remains mostly behind ParEGO-UCB. ParEGO-EI shows a weak early HV increase but overtakes EHVI in later iterations.

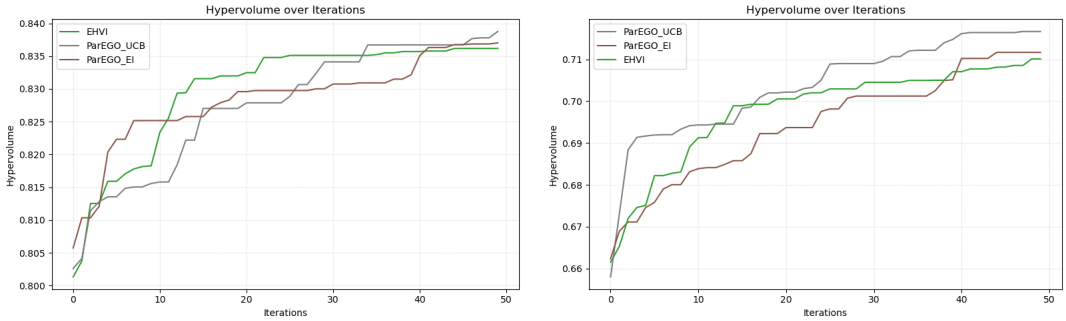

Figure 3: Comparison of the average hypervolume during the tuning process over 50 iterations for Multi-Objective Bayesian Optimization methods (ParEGO-EI, ParEGO-UCB, and EHVI). The results for tuning a Min-Max Ant-system with 500 function evaluations are shown below, and for 2000 function evaluations are shown above.

Pareto Front Figure 4 shows the non-dominated solutions from EHVI, ParEGO-UCB, and Random Search, as well as the non-dominated solutions obtained by single-objective HPO methods. For 2000 function evaluations, EHVI provides a Pareto front of three configurations while ParEGO provides five non-dominated points, offering minor trade-offs between AF and error.

The single-objective methods find multiple configurations dominating those found by the multi-objective methods. HPO-UCB-AF finds the highest overall AF values ($\sim$*0.814*), while HPO-UCB-BF finds the lowest BFS value ($\sim$*0.0565*).

Random Search contributes two non-dominated points that are dominated by almost all other methods, indicating poor overall performance.

For 500 function evaluations, ParEGO-UCB finds two non-dominated configurations, with one being practically dominant compared to the other, achieving an error of approximately *0.084*. HPO-UCB-AF again finds the configuration with the highest AF value. EHVI provides four non-dominated configurations offering trade-offs between AF and the BFS. However, these are all dominated by the configurations found by ParEGO-UCB. Again, random search is non-competitive.

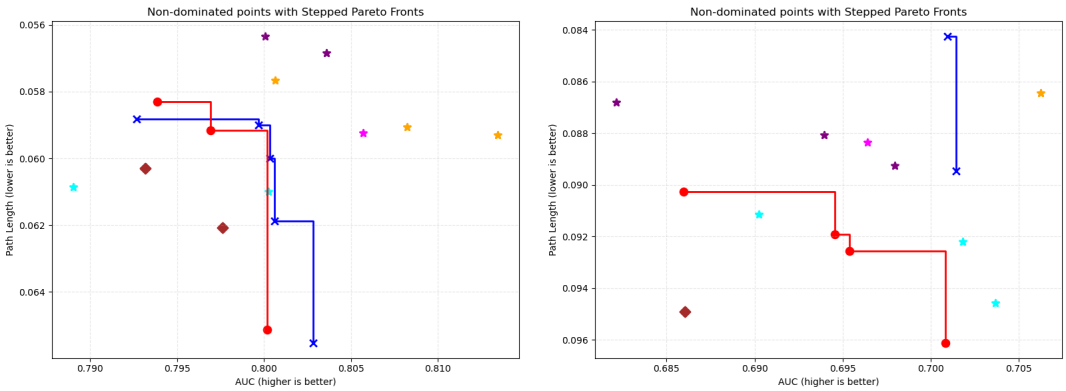

Figure 4: Non-dominated solutions and Pareto fronts identified by each method. The top plot displays results for 2000 function evaluations, and the bottom plot shows results for 500 calls. Orange stars: HPO-UCB optimizing Anytime Fitness (AF). Purple stars: HPO-UCB optimizing Best-Found Solution (BFS). Cyan stars: HPO-EI-AF. Magenta stars: HPO-EI-BFS. Blue line and crosses: Pareto front and solutions from ParEGO-UCB. Red line and circles: Pareto front and solutions from EHVI. Brown diamonds: Random Search.

Full Comparison Figure 5 presents the convergence of the best-performing single-objective method (*UCB-AF*), the best scalarization-based MOBO method (*ParEGO-UCB*), the Pareto-based method (*EHVI*), and Random Search across AF and BFS.

For 2000 function evaluations, on the left, HPO-UCB-AF achieves the highest final AF, followed closely by ParEGO-UCB. EHVI stabilizes at a lower AF.

On the right, HPO-UCB-AF again performs best, achieving the lowest BFS. ParEGO-UCB follows closely with a slightly higher error, while EHVI trails slightly behind both.

Random Search exhibits the weakest performance across objectives, consistently lagging behind other methods.

The plots for 500 function evaluations echo these findings.

Overall, HPO-UCB tuned for AF consistently outperforms the other methods on both objectives. ParEGO-UCB offers competitive performance, trailing slightly. EHVI converges the slowest among the guided methods, and random search is uncompetitive, unable to match the quality of the guided methods.

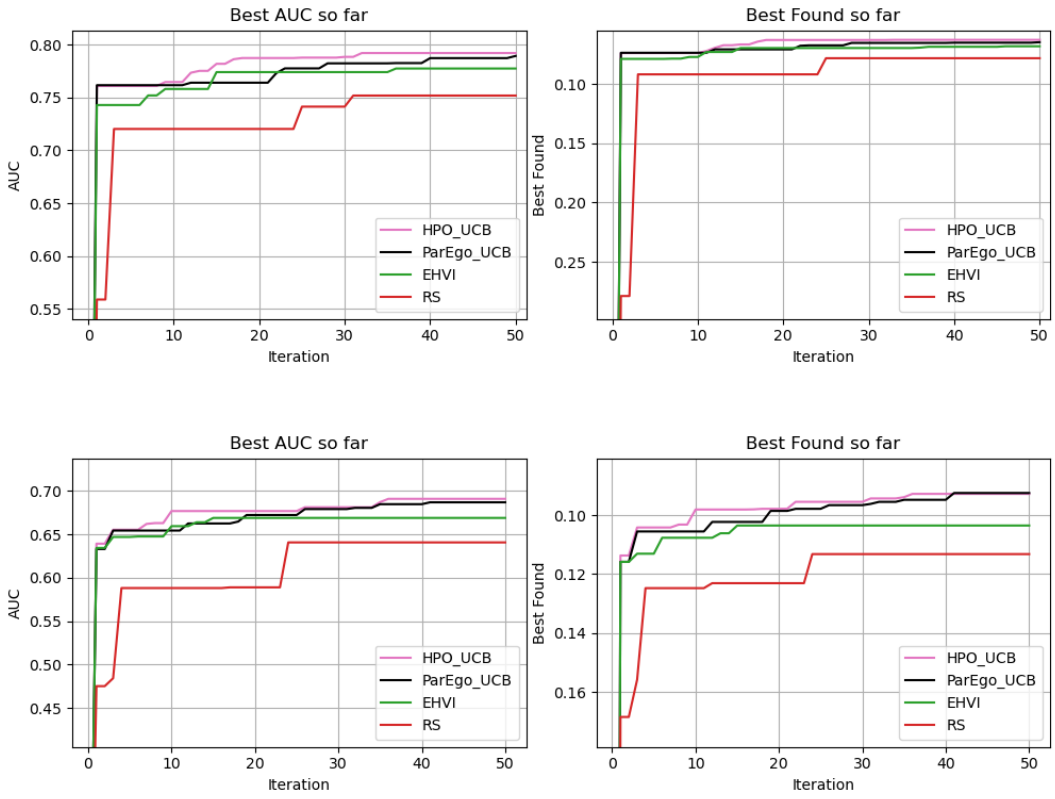

Figure 5: Comparison of the average Anytime Fitness (left) and Best Found Solution (right) during the tuning process over 50 iterations for the best performing Bayesian Optimization methods (HPO-UCB-AF, ParEGO-UCB, and EHVI) with a random search baseline. The results for tuning a Min-Max Ant-system with 500 function evaluations are shown below, and for 2000 function evaluations are shown above.

4.2 Tuned-ACO Comparison

To better understand the impact of hyperparameter tuning on an MMAS, we compare the configurations with the highest AF and the lowest BFS found by each method against the default settings mentioned by Stützle et al.(OIS12). Each configuration is evaluated on all test instances. The average performance over evaluations is shown in Figure 7, and the best average tour lengths are summarized in Table 5.

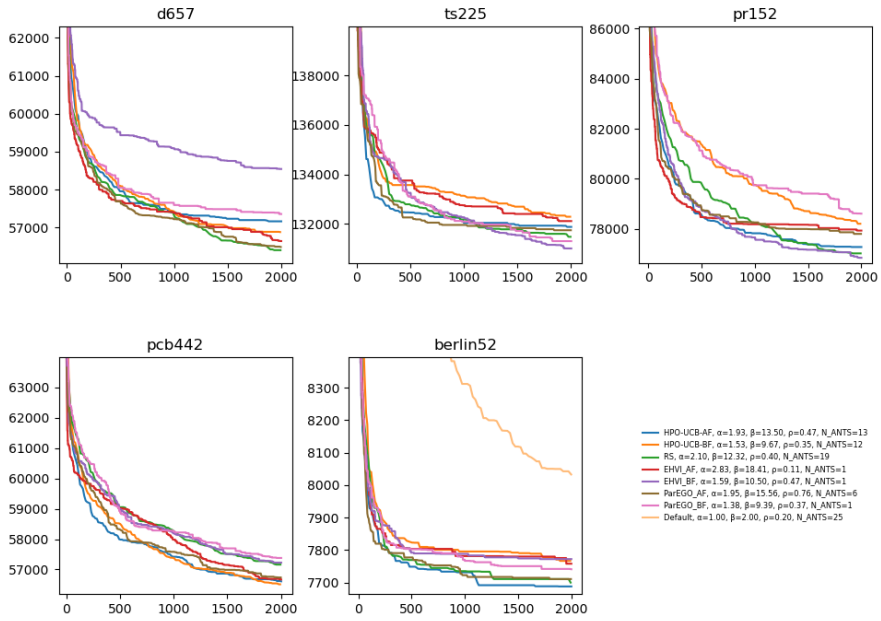

Figure 6: Comparison of the test set performance of the configurations with the best respective AF and BFS values obtained during the tuning process with a random search baseline for tuning a Min-Max Ant-system with 2000 function evaluations.

Method	**berlin52**	**pr152**	**ts225**	**pcb442**	**d657**
Optimal	7 542.00	73 682.00	126 643.00	50 778.00	48 912.00
Default	8 032.80	86 754.80	152 190.90	78 784.70	82 321.50
HPO-UCB-AF	**7 687.90**	77 281.30	131 877.90	56 610.00	57 165.60
HPO-UCB-BF	7 766.60	78 209.80	132 288.90	**56 512.60**	56 882.60
RS	7 699.70	77 026.90	131 477.40	57 161.40	**56 405.60**
EHVI_AF	7 757.90	77 936.40	132 113.10	56 670.10	56 645.30
EHVI_BF	7 772.10	**76 851.50**	**131 005.30**	57 227.10	58 544.60
ParEGO_AF	7 710.60	77 805.20	131 742.70	56 736.90	56 489.90
ParEGO_BF	7 740.80	78 613.10	131 301.80	57 379.50	57 358.40

Table 5: Average best-found tour lengths on the test set per instance over 2000 function evaluations by hyperparameter-tuning method and TSPLIB optima.

For 2000 function evaluations, no one configuration performs best across all instances. Most perform well in some instances and poorly in others. Interestingly, the best random search configuration is quite competitive, even showing the best performance on d657. Across all instances the default settings are not competitive.
Configurations with higher AF values outperform configurations found using BFS values on more instances. ParEGO-AF has a lower average on d657, pcb442, berlin52, and pr152 compared to

its BFS counterpart. A similar pattern holds for HPO-UCB-AF and HPO-UCB-BF, as well as EHVI-AF and EHVI-BF, with the AF configurations performing better on three of the five test instances.

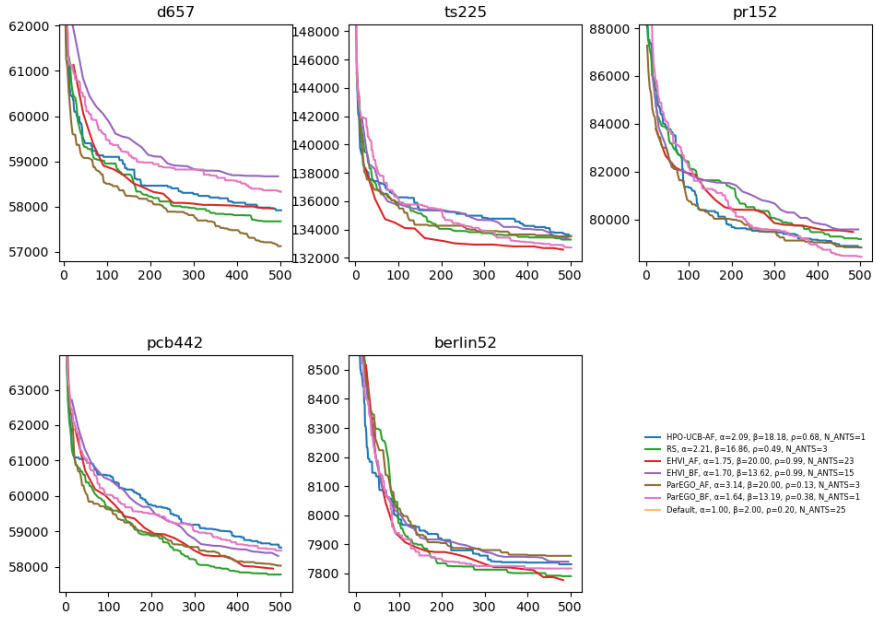

Figure 7: Comparison of the test set performance of the configurations with the best respective AF and BFS values obtained during the tuning process with a random search baseline for tuning a Min-Max Ant-system with 500 function evaluations.

Method	**berlin52**	**pr152**	**ts225**	**pcb442**	**d657**
Optimal	7 542.00	73 682.00	126 643.00	50 778.00	48 912.00
Default	9 058.00	101 932.40	180 355.30	91 050.80	94 672.10
HPO-UCB-AF	7 831.10	78 830.60	133 530.80	58 543.70	57 915.60
RS	7 790.10	79 185.90	133 283.10	**57 782.30**	57 669.80
EHVI_AF	**7 776.30**	79 476.80	**132 576.00**	57 947.80	57 960.10
EHVI_BF	7 840.20	79 584.00	133 365.30	58 306.10	58 666.70
ParEGO_AF	7 860.00	78 841.40	133 509.20	58 034.10	**57 120.30**
ParEGO_BF	7 816.40	**78 441.20**	132 726.90	58 458.90	58 326.50

Table 6: Average best-found tour lengths on the test set per instance over 500 function evaluations by hyperparameter-tuning method and TSPLIB optima.

For 500 function evaluations, similar interactions arise. Again, random search is competitive, achieving low averages across the test set and the lowest average on pcb442. For EHVI, it holds that the AF configuration outperforms the BF configuration, yielding lower averages across instances. For

ParEGO, this is no longer the case, as the Bf configuration achieves lower averages than its AF counterpart on four instances.

Table 7 shows the results of the Wilcoxon signed-rank test comparing all tuned configurations to the default for 2000 function evaluations. The results show all tested methods significantly outperform ($p < 0.05$) the default configuration, reinforcing the importance of hyperparameter tuning.

Comparison	**d657**	**ts225**	**pr152**	**pcb442**	**berlin52**
HPO-UCB-AF vs Default	**0.0020**	**0.0020**	**0.0020**	**0.0020**	**0.0020**
HPO-UCB-BF vs Default	**0.0020**	**0.0020**	**0.0020**	**0.0020**	**0.0059**
RS vs Default	**0.0020**	**0.0020**	**0.0020**	**0.0020**	**0.0039**
EHVI_AF vs Default	**0.0020**	**0.0020**	**0.0020**	**0.0020**	**0.0098**
EHVI_BF vs Default	**0.0020**	**0.0020**	**0.0020**	**0.0020**	**0.0371**
ParEGO_AF vs Default	**0.0020**	**0.0020**	**0.0020**	**0.0020**	**0.0137**
ParEGO_BF vs Default	**0.0020**	**0.0020**	**0.0020**	**0.0020**	**0.0059**

Table 7: Wilcoxon signed-rank test p-values of final tour lengths for 2000 function evaluations: Default vs. Other Methods (significant $p < 0.05$ in **bold**)

Table 8 shows that for 500 function evaluations, all methods also significantly outperform the default configuration.

Method	**d657**	**ts225**	**pr152**	**pcb442**	**berlin52**
HPO-UCB-AF	**0.0020**	**0.0020**	**0.0020**	**0.0020**	**0.0020**
RS	**0.0020**	**0.0020**	**0.0020**	**0.0020**	**0.0020**
EHVI_AF	**0.0020**	**0.0020**	**0.0020**	**0.0020**	**0.0020**
EHVI_BF	**0.0020**	**0.0020**	**0.0020**	**0.0020**	**0.0020**
ParEGO_AF	**0.0020**	**0.0020**	**0.0020**	**0.0020**	**0.0020**
ParEGO_BF	**0.0020**	**0.0020**	**0.0020**	**0.0020**	**0.0020**

Table 8: Wilcoxon signed-rank test p-values of final tour lengths for 500 function evaluations: Default vs. Other Methods (significant $p < 0.05$ in **bold**)

Furthermore, we compare the configurations with the highest AF with those with the lowest BF produced by the guided methods.

The results for 2000 function evaluations are reported in Table 9. These show only one significant interaction between the EHVI methods, with the AF configuration outperforming EHVI-BF on d657.

Comparison	d657	ts225	pr152	pcb442	berlin52
HPO-UCB-AF vs HPO-UCB-BF	0.5566	0.6953	0.6250	0.8457	0.2031
EHVI_AF vs EHVI_BF	**0.0039**	0.1055	0.1602	0.3750	0.6953
ParEGO_AF vs ParEGO_BF	0.0645	0.2754	0.3750	0.2754	0.6523

Table 9: Wilcoxon signed-rank test p-values of final tour lengths for Comparisons between the highest AF and the lowest BFS between method families for 2000 function evaluations (significant $p < 0.05$ in **bold**)

Table 10 shows the intra-family results of the Wilcoxon Signed-Rank Test for the multi-objective configurations for 500 evaluations. The single-objective methods produced a single non-dominated configuration, making a comparison between AF and BF versions obsolete. The results show that ParEGO, with a higher AF value, significantly outperforms its counterpart on d567. For EHVI, no configuration significantly outperforms the others.

Comparison	d657	ts225	pr152	pcb442	berlin52
EHVI_AF vs EHVI_BF	0.4316	0.1309	0.8457	0.7695	0.3105
ParEGO_AF vs ParEGO_BF	**0.0273**	0.3750	0.6953	0.6953	0.4766

Table 10: Wilcoxon signed-rank test p-values of final tour lengths for Comparisons between the highest AF and the lowest BFS between method families for 500 function evaluations (significant $p < 0.05$ in **bold**)

Table 11 reports the results of the Wilcoxon signed-rank test comparing the single- and multi-objective BO approaches.

The results show that for 2000 function evaluations, HPO-UCB-AF's final tour length significantly differs from EHVI-BF and ParEGO-AF on d657. HPO-UCB-AF outperforms EHVI-BF and is outperformed by ParEGO-AF on this instance. HPO-UCB-BF achieves a lower final tour length on d657, showing a significant difference from EHVI-BF.

Comparison	d657	ts225	pr152	pcb442	berlin52
HPO-UCB-AF vs RS	0.2754	0.4316	0.8457	0.1602	0.7695
HPO-UCB-AF vs EHVI_AF	0.1309	0.9219	0.2754	0.3750	**0.0449**
HPO-UCB-AF vs EHVI_BF	**0.0137**	0.1055	0.5566	0.2754	0.4316
HPO-UCB-AF vs ParEGO_AF	**0.0371**	0.6250	0.4922	0.7695	0.7344
HPO-UCB-AF vs ParEGO_BF	0.5566	0.4316	0.1055	0.1602	0.3223
HPO-UCB-BF vs RS	0.4316	0.4316	0.1602	0.5566	0.4316
HPO-UCB-BF vs EHVI_AF	0.7695	1.0000	0.9219	0.7695	0.9219
HPO-UCB-BF vs EHVI_BF	**0.0098**	0.1602	0.1602	0.1934	0.7695
HPO-UCB-BF vs ParEGO_AF	0.6250	1.0000	0.3750	0.4922	0.6250
HPO-UCB-BF vs ParEGO_BF	0.4316	0.2754	0.6953	0.1602	0.7695

Table 11: Wilcoxon signed-rank test p-values of final tour lengths for Comparisons between single- and multi-objective methods for 2000 function evaluations (significant $p < 0.05$ in **bold**)

Table 12 shows the results of the Wilcoxon signed-rank test for single- and multi-objective methods for 500 function evaluations. The results show no significant differences between methods.

Comparison	**d657**	**ts225**	**pr152**	**pcb442**	**berlin52**
HPO-UCB-AF vs RS	0.6953	0.6250	0.9219	0.1602	0.6953
HPO-UCB-AF vs EHVI_AF	1.0000	0.5566	0.6250	0.2754	0.3223
HPO-UCB-AF vs EHVI_BF	0.3223	0.5566	0.5566	0.6953	0.9219
HPO-UCB-AF vs ParEGO_AF	0.1055	0.6953	0.8457	0.3223	0.7695
HPO-UCB-AF vs ParEGO_BF	0.4922	0.6953	0.9219	0.8457	0.6523

Table 12: Wilcoxon signed-rank test p-values of final tour lengths for Comparisons between single- and multi-objective methods for 500 function evaluations(significant $p < 0.05$ in **bold**)

Figure 8 shows violin plots for the top-10 hyperparameter values across methods, separately for AF and BFS, as well as for 500 and 2000 function evaluations.

The results indicate that high-performing configurations for both objectives tend to arise from low α ($\sim$1-3) and high β ($\sim$13-17) values, with β being higher for 500 function evaluations. The number of ants ranges between 1 and 15 for 2000 function evaluations. However, most top configurations use $N_{ants} = 1$. The evaporation rate ρ spans the entire range from 0.01 and 0.99 in both objectives.

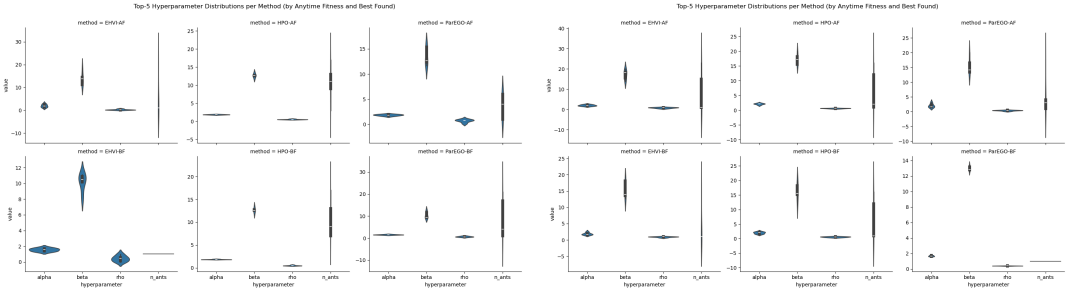

Figure 8: Violin Plots showing the top-5 hyperparameter values per best performing method for both objectives separately for 500 (right) and 2000 function evaluations (left)

5 Discussion

Across all benchmark instances, the default settings were non-competitive, resulting in substantially worse solutions than those of any of the tuned methods. This confirms that HPO can significantly enhance the performance of a ACO-based TSP solver. These findings align with Yin and van Wijk's results on applying BO to optimize the hyperparameters of an ACO (YW21) and Opez-Ibá nez and Stützle's work on automating the hyperparameter search for an MMAS (OIS12). Both find a substantial improvement using tuned values over default settings.

The general trends for high-performing MMAS configurations indicate a preference for low α values and high β values, consistent with Pellegrini et al.'s study on the MMAS hyperparameters (PFM06). Our MMAS implementation was evaluated under 500 and 2000 evaluations, prioritizing

early convergence. Pellegrini et al. similarly observed that configurations with low α and high β promote faster exploitation for shorter runtimes. In line with their findings, ρ does not exhibit a strong interaction with performance. However, in our experiments, high-performing configurations were found across a broad range of values, including ranges not explored in Pellegrini et al.'s study.

The use of different acquisition functions for single-objective and ParEGO-based optimization shows that UCB yields better results overall, often finding superior solutions than methods tuned using EI. This suggests that UCB offers a more effective balance between exploration and exploitation in the context of MMAS tuning.

While Random Search offers a competitive baseline, finding high-performing configurations that outperform the default settings and the guided methods on some instances, its tuning convergence is inconsistent and inferior to that of guided search methods.

Comparing the use of different objective functions in Figure 1 shows that single-objective tuned for AF leads to better convergence, with BO finding configurations with higher final AF as well as a lower final BFS earlier. Furthermore, configurations with a higher AF achieve a lower average on the test set on most instances. This suggests that AF offers a more comprehensive optimization target, resulting in more robust and high-quality solutions.

The success of the single-objective UCB tuned for AF in improving both objectives, as well as the limited configurations on the Pareto front, indicate that AF and BFS show little conflict, allowing scalarized and single-objective methods to perform well. In contrast, the Pareto-based EHVI method, which attempts to balance both objectives simultaneously, appears to be less suited in this setting.

5.1 Implications

Our findings demonstrate that both single-objective and multi-objective BO outperform random search when tuning an MMAS for non-conflicting objectives. Furthermore, all tuned methods significantly outperform the out-of-the-box methods on the test set. While none of the tuned methods consistently achieves the best performance across all instances, we observe that single-objective BO using UCB and tuning for AF yields the most reliable convergence towards high-performing configurations for both objectives.

Additionally, we find that AF is a more stable and informative optimization target than final solution quality. Tuning an MMAS for AF results in better tuning convergence and shorter final tours on most instances. We argue that researchers and practitioners should adopt holistic performance metrics, such as AF, when tuning meta-heuristic algorithms. While this conclusion is drawn from MMAS for a TSP, it likely extends to other metaheuristics where convergence behavior matters more than isolated final solutions.

Finally, our results suggest that the default parameters for the MMAS are insufficient for achieving effective search performance. Relying on such out-of-the-box settings in practical settings introduces

a risk of suboptimal solutions. Even a moderate investment in BO yields more robust and efficient convergence, highlighting the importance of HPO.

5.2 Limitations

This study has multiple limitations that must be acknowledged. Firstly, all experiments were conducted on symmetric TSP instances with Euclidean distance. As such, our findings may not generalize to non-Euclidean TSP variations, asymmetric TSPs, or other combinatorial optimization problems.

Second, the scope of hyperparameter tuning was limited to 4 parameters of an MMAS ($\alpha, \beta, \rho, n_{ants}$). While these are among the most influential parameters, MMAS variants with additional or alternative parameters were not considered. Moreover, by restricting the range of the search space, potentially relevant interactions between a larger set of parameters may have been overlooked.

Furthermore, our study focuses exclusively on a bi-objective setting with minimally conflicting objectives. While single-objective and scalarization-based methods are more effective, this may not be reflected in many-objective or more conflicting multi-objective settings, where trade-offs could be more pronounced. In such cases, the performance of Pareto-based or scalarization-based methods may differ significantly. Our results should not be generalized to optimization problems with more conflicting objectives.

Additionally, all our optimization methods were given a budget of 55 iterations, 5 Sobol samples, and 50 further evaluations. We cannot claim that the performance hierarchy of our methods remains the same when budgets are higher. Similarly, the preferred MMAS configurations identified for 500 and 2000 function evaluations may not generalize to scenarios with larger budgets or those using fixed iteration limits.

Moreover, we employed Gaussian processes in BoTorch with LogEI, UCB, ParEGO, and EHVI. Alternative surrogates, such as random forests, or acquisition functions, such as PES or NEHVI, were not explored. These alternatives could recommend different configurations or exhibit improved performance on certain instances.

Another limitation arises from the stochastic nature of MMAS. Due to limited computational resources, each hyperparameter configuration was evaluated five times. This limited number of replications may not fully capture the variance in performance across independent runs. As a result, observed differences may be affected by randomness rather than algorithmic superiority.

Finally, our implementation focused on BO frameworks, due to their sample efficiency. However, other optimization methods, such as NSGA-II or PesMO, were not included. These methods may provide competitive performance or outperform the methods considered in certain scenarios.

6 Conclusions and Further Research

This study demonstrated that hyperparameter optimization (HPO) and multi-objective hyperparameter optimization (MOHPO) find hyperparameter configurations for an Min–Max Ant System (MMAS) that consistently outperform the default settings. While Expected Hypervolume Improvement (EHVI) and ParEGO provide a Pareto Front of trade-offs between objectives, the configurations with higher Anytime Fitness (AF) values show better performance on more instances. No single method outperforms all others across all instances, highlighting that all approaches are viable. For the objectives AF and Best-Found Solution (BFS), we recommend HPO-UCB, tuned for AF, or ParEGO-UCB, due to their better convergence properties in the tuning process.

Further research could explore higher-dimensional tuning, incorporating additional algorithmic components, or expanding the search space. This would further test the capabilities of Multi-Objective Bayesian Optimization (MOBO) and could yield novel hyperparameter interactions.

In addition, applying different surrogate and acquisition functions, particularly methods more robust to noise, such as Random Forests or NEHVI, could provide valuable insights.

Finally, the use of different objectives should be explored. In particular, the use of more conflicting objectives could produce differing results, highlighting the importance of a more diverse Pareto front.

By addressing these directions, future research could further validate and extend the use of MOHPO for meta-heuristics.

References

[ADE+25] Sebastian Ament, Samuel Daulton, David Eriksson, Maximilian Balandat, and Eytan Bakshy. Unexpected improvements to expected improvement for bayesian optimization, 2025.

[And90] Bjarne Andresen. *The Use of Simulated Annealing to Solve Extremely Large and Complex Problems*, pages 239–241. Springer US, Boston, MA, 1990.

[BB12] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(10):281–305, 2012.

[BBL+21] Bernd Bischl, Martin Binder, Michel Lang, Tobias Pielok, Jakob Richter, Stefan Coors, Janek Thomas, Theresa Ullmann, Marc Becker, Anne-Laure Boulesteix, Difan Deng, and Marius Lindauer. Hyperparameter optimization: Foundations, algorithms, best practices and open challenges, 2021.

[BCdF10] Eric Brochu, Vlad M. Cora, and Nando de Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning, 2010.

[BDJD22] Syrine Belakaria, Aryan Deshwal, Nitthilan Kannappan Jayakodi, and Janardhan Rao Doppa. Uncertainty-aware search framework for multi-objective bayesian optimization, 2022.

[BHS99] Bernd Bullnheimer, Richard Hartl, and Christine Strauss. A new rank based version of the ant system - a computational study. *Central European Journal of Operations Research*, 7:25–38, 01 1999.

[BKJ+20] Maximilian Balandat, Brian Karrer, Daniel R. Jiang, Samuel Daulton, Benjamin Letham, Andrew Gordon Wilson, and Eytan Bakshy. Botorch: A framework for efficient monte-carlo bayesian optimization, 2020.

[CPA22] Antonio Candelieri, Andrea Ponti, and Francesco Archetti. Fair and green hyperparameter optimization via multi-objective and multiple information source bayesian optimization, 2022.

[DAERF21] George De Ath, Richard M. Everson, Alma A. M. Rahat, and Jonathan E. Fieldsend. Greed is good: Exploration and exploitation trade-offs in bayesian optimisation. *ACM Transactions on Evolutionary Learning and Optimization*, 1(1):1–22, April 2021.

[DBB20a] Samuel Daulton, Maximilian Balandat, and Eytan Bakshy. Differentiable expected hypervolume improvement for parallel multi-objective bayesian optimization, 2020.

[DBB20b] Samuel Daulton, Maximilian Balandat, and Eytan Bakshy. Differentiable expected hypervolume improvement for parallel multi-objective bayesian optimization, 2020.

[DMC96] M. Dorigo, V. Maniezzo, and A. Colorni. Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 26(1):29–41, 1996.

[DOW+22] Mike Diessner, Joseph O'Connor, Andrew Wynn, Sylvain Laizet, Yu Guan, Kevin Wilson, and Richard D. Whalley. Investigating bayesian optimization for expensive-to-evaluate black box functions: Application in fluid dynamics. *Frontiers in Applied Mathematics and Statistics*, 8, December 2022.

[DPAM02] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[DS04] Marco Dorigo and Thomas Stützle. *Ant Colony Optimization*. The MIT Press, 06 2004.

[EGN06a] M. Emmerich, Kyriakos C. Giannakoglou, and Boris Naujoks. Single- and multiobjective evolutionary optimization assisted by gaussian random field metamodels. *IEEE Transactions on Evolutionary Computation*, 10:421–439, 2006.

[EGN06b] M.T.M. Emmerich, K.C. Giannakoglou, and B. Naujoks. Single- and multiobjective evolutionary optimization assisted by gaussian random field metamodels. *IEEE Transactions on Evolutionary Computation*, 10(4):421–439, 2006.

[GP03] Nilesh Gambhava and G. H. Patel. Traveling salesman problem using genetic algorithm. 2003.

[GTSS22] Amey Gohil, Manan Tayal, Tezan Sahu, and Vyankatesh Sawalpurkar. Travelling salesman problem: Parallel implementations & analysis. *ArXiv*, abs/2205.14352, 2022.

[HB16] Daniel Horn and Bernd Bischl. Multi-objective parameter configuration of machine learning algorithms using model-based optimization. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8, 2016.

[HHLB11] Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *Proceedings of the 5th International Conference on Learning and Intelligent Optimization*, LION'05, page 507–523, Berlin, Heidelberg, 2011. Springer-Verlag.

[HLGR+16] José Hernández-Lobato, Michael Gelbart, Brandon Reagan, Robert Adolf, Daniel Hernandez-Lobato, Paul Whatmough, David Brooks, Gu-Yeon Wei, and Ryan Adams. Designing neural network hardware accelerators with decoupled objective evaluations. 12 2016.

[HLHLSA16] Daniel Hernandez-Lobato, Jose Hernandez-Lobato, Amar Shah, and Ryan Adams. Predictive entropy search for multi-objective bayesian optimization. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1492–1501, New York, New York, USA, 20–22 Jun 2016. PMLR.

[HNE07] Nicola Hochstrate, Boris Naujoks, and Michael Emmerich. Sms-emoa: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181:1653–1669, 02 2007.

[Kno06] J. Knowles. Parego: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10(1):50–66, 2006.

[KNRS24] Aryan Kothari, V. R. N. S. Nikhil, Namana Rohit, and Apurvanand Sahay. Harnessing meta-heuristic algorithms to optimize travelling salesman problem. In *2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, pages 1–9, 2024.

[KPM+23] Florian Karl, Tobias Pielok, Julia Moosbauer, Florian Pfisterer, Stefan Coors, Martin Binder, Lennart Schneider, Janek Thomas, Jakob Richter, Michel Lang, Eduardo C. Garrido-Merchán, Juergen Branke, and Bernd Bischl. Multi-objective hyperparameter optimization in machine learning—an overview. *ACM Trans. Evol. Learn. Optim.*, 3(4), December 2023.

[LTRE20] Gongjin Lan, Jakub M. Tomczak, Diederik M. Roijers, and A. E. Eiben. Time efficiency in optimization with a bayesian-evolutionary algorithm, 2020.

[MJ11] Avijit Maji and Manoj Jha. Comparison of single and multi objective highway alignment optimization algorithms. *Advances in Transportation Studies*, pages 5–16, 01 2011.

[MM15] Hosam Mukhairez and Ashraf Maghari. Performance comparison of simulated annealing, ga and aco applied to tsp. *International Journal of Intelligent Computing Research (IJICR)*, Volume 6:647 – 654, 12 2015.

[OIS12] Manuel L Opez-Ibáñez and Thomas Stützle. Automatically improving the anytime behaviour of optimisation algorithms. 2012.

[OKN+17] Julius Beneoluchi Odili, Mohd Nizam Mohmad Kahar, A. Noraziah, M. Zarina, and Riaz Ul Haq and. Performance analyses of nature-inspired algorithms on the traveling salesman's problems for strategic management. *Intelligent Automation & Soft Computing*, 0(0):1–11, 2017.

[PFM06] Paola Pellegrini, Daniela Favaretto, and Elena Moretti. On max - min ant system's parameters. In *ANTS Workshop*, 2006.

[PSB10] Paola Pellegrini, Thomas Stützle, and Mauro Birattari. Tuning max – min ant system with off-line and on-line methods. 2010.

[Ras04] Carl Edward Rasmussen. *Gaussian Processes in Machine Learning*, page 13. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[RC25] Ashwin Renganathan and Kade E. Carlson. qpots: Efficient batch multiobjective bayesian optimization via pareto optimal thompson sampling, 2025.

[Rei91] Gerhard Reinelt. Tsplib—a traveling salesman problem library. *ORSA Journal on Computing*, 3(4):376–384, 1991.

[SH00] Thomas Stützle and Holger H. Hoos. Max-min ant system. *Future Gener. Comput. Syst.*, 16:889–914, 2000.

[SRS+15] Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Md. Mostofa Ali Patwary, Prabhat, and Ryan P. Adams. Scalable bayesian optimization using deep neural networks, 2015.

[SSW+16] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando de Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016.

[TBZ12] Stefano Tarantola, William E. Becker, and Dirk Zeitz. A comparison of two sampling methods for global sensitivity analysis. *Comput. Phys. Commun.*, 183:1061–1072, 2012.

[TGHD22] Deniz Tosoni, Christopher Galli, Thomas Hanne, and Rolf Dornberger. Benchmarking metaheuristic optimization algorithms on travelling salesman problems. In *Proceedings of the 8th International Conference on E-Society, e-Learning and e-Technologies*, ICSLT '22, page 20–25, New York, NY, USA, 2022. Association for Computing Machinery.

[TIO17] Ryoji Tanabe, Hisao Ishibuchi, and Akira Oyama. Benchmarking multi- and many-objective evolutionary algorithms under two optimization scenarios. *IEEE Access*, 5:19597–19619, 2017.

[WCZ+19] Jia Wu, Xiuyun Chen, H. Zhang, Li-Dong Xiong, Hang Lei, and Sihao Deng. Hyper-parameter optimization for machine learning models based on bayesian optimization. *Journal of Electronic Science and Technology*, 17:26–40, 2019.

[YDWB21] Furong Ye, Carola Doerr, Hao Wang, and Thomas Bäck. Automated configuration of genetic algorithms by tuning for anytime performance. *IEEE Transactions on Evolutionary Computation*, 26:1526–1538, 2021.

[YEDB19] Kaifeng Yang, Michael Emmerich, André Deutz, and Thomas Bäck. Multi-objective bayesian global optimization using expected hypervolume improvement gradient. *Swarm and Evolutionary Computation*, 44:945–956, 2019.

[YLC24] Furong Ye, Chuan Luo, and Shaowei Cai. Better Understandings and Configurations in MaxSAT Local Search Solvers via Anytime Performance Analysis. *arXiv e-prints*, page arXiv:2403.06568, March 2024.

[YW21] Emmy Yin and Klas Wijk. Bayesian parameter tuning of the ant colony optimization algorithm : Applied to the asymmetric traveling salesman problem, 2021.

[YZ20] Tong Yu and Hong Zhu. Hyper-parameter optimization: A review of algorithms and applications, 2020.

[ZCZ14] Tao Zhang, Wanpracha Chaovalitwongse, and Yuejie Zhang. Integrated ant colony and tabu search approach for time dependent vehicle routing problems with simultaneous pickup and delivery. *Journal of Combinatorial Optimization*, 28, 07 2014.

[ZL07] Qingfu Zhang and Hui Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007.

[ZPS19] Lavi Rizki Zuhal, Pramudita Satria Palar, and Koji Shimoyama. A comparative study of multi-objective expected improvement for aerodynamic design. *Aerospace Science and Technology*, 91:548–560, 2019.