# Optimizing CMA-ES with CMA-ES

André Thomaser[1,2][a], Marc-Eric Vogt[1][b], Thomas Bäck[2][c] and Anna V. Kononova[2][d]
[1] *BMW Group, Knorrstraße 147, Munich, Germany*
[2] *LIACS, Leiden University, Niels Bohrweg 1, Leiden, The Netherlands*

Keywords: Parameter Tuning, CMA-ES, Benchmarking, Mixed-Integer Optimization, TPE, SMAC, BBOB.

Abstract: The performance of the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is significantly affected by the selection of the specific CMA-ES variant and the parameter values used. Furthermore, optimal CMA-ES parameter configurations vary across different problem landscapes, making the task of tuning CMA-ES to a specific optimization problem a challenging mixed-integer optimization problem. In recent years, several advanced algorithms have been developed to address this problem, including the Sequential Model-based Algorithm Configuration (SMAC) and the Tree-structured Parzen Estimator (TPE).
In this study, we propose a novel approach for tuning CMA-ES by leveraging CMA-ES itself. Therefore, we combine the modular CMA-ES implementation with the margin extension to handle mixed-integer optimization problems. We show that CMA-ES can not only compete with SMAC and TPE but also outperform them in terms of wall clock time.

## 1 INTRODUCTION

The Covariance Matrix Adaptation Evolution Strategy (CMA-ES) (Hansen and Ostermeier, 1996) is a popular algorithm used for solving complex black-box optimization problems. It has gained significant attention due to its ability to handle nonlinear and multimodal optimization problems. Over the years several different variants have been developed, each offering unique advantages (Bäck et al., 2013).

To achieve optimal performance with CMA-ES, it is crucial to tune the parameters of CMA-ES and to explore different CMA-ES variants (van Rijn et al., 2016). However, manual parameter tuning can be laborious and time-consuming. As an alternative approach, automatic parameter tuning has been proposed (Bäck, 1994; Grefenstette, 1986). This approach treats parameter tuning as an additional optimization problem besides the primary objective of solving the original problem.

Therefore, tuning CMA-ES parameters involves optimizing an optimization algorithm itself. The objective of such meta-optimization is to select the most suitable set of parameter values to enhance the performance of the optimizer on the original optimization problem. Figure 1 illustrates the relationship and distinction between solving the original optimization problem and tuning the parameters. While CMA-ES optimizes the quality of solutions found (goodness of solutions is referred to as *fitness*) for the original problem, a meta-algorithm is employed to optimize the quality of the CMA-ES parameters (goodness of performance is referred to as *utility*) (Eiben and Smit, 2011).

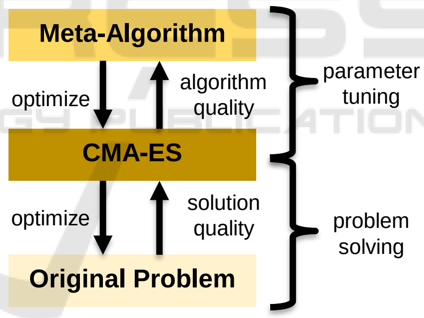


Figure 1: Solving an optimization problem with CMA-ES and parameter tuning with a meta-algorithm as two different optimization problems (Eiben and Smit, 2011).

CMA-ES parameter tuning can be formulated as a mixed-integer optimization problem where, in addition to continuous CMA-ES parameters, different

[a] https://orcid.org/0000-0002-6210-8784
[b] https://orcid.org/0000-0003-3476-9240
[c] https://orcid.org/0000-0001-6768-1478
[d] https://orcid.org/0000-0002-4138-7024



Thomaser, A., Vogt, M., Bäck, T. and Kononova, A.
Optimizing CMA-ES with CMA-ES.
DOI: 10.5220/0012179400003595
In *Proceedings of the 15th International Joint Conference on Computational Intelligence (IJCCI 2023)*, pages 214-221
ISBN: 978-989-758-674-3; ISSN: 2184-3236

combinations of discrete parameter values and CMA-ES variants can be selected to find the optimal configuration. Several meta-algorithms have been developed to address such a challenge.

A popular algorithm for parameter tuning is the Sequential Model-based Algorithm Configuration (SMAC) (Hutter et al., 2011). SMAC is a sequential model-based optimization (SMBO) approach that combines Bayesian optimization with random forest regression models (Breiman, 2001). SMAC has been successfully applied to various machine learning tasks, including algorithm configuration, feature selection, and deep neural architecture search (Feurer et al., 2015; Lindauer et al., 2022).

Another SMBO algorithm is the Tree-structured Parzen Estimator (TPE) (Bergstra et al., 2011). TPE utilizes a distinct approach based on tree-structured density estimation to efficiently search for optimal parameter settings. Tuning the parameters of CMA-ES with TPE has been shown to improve the performance of CMA-ES on a number of benchmark optimization problems (Zhao and Li, 2018).

Recently, an extension of CMA-ES called CMA-ES with margin (Hamano et al., 2022) has been introduced. This extension enhances the capabilities of CMA-ES to effectively handle discrete and mixed-integer optimization problems. As a result, CMA-ES with the margin extension can be used as a meta-algorithm for solving the mixed-integer optimization problem of tuning CMA-ES for specific optimization problems.

The goal of this study is to explore the potential of CMA-ES with margin as a meta-algorithm for tuning the parameters of CMA-ES. We conduct experiments on several benchmark optimization problems and compare the performance of CMA-ES with margin to that of SMAC, TPE, and random search. First, we provide an overview of CMA-ES, its parameters, and its variants (Section 2.1). In addition, we briefly describe the margin extension, which specifically addresses mixed-integer optimization problems (Section 2.2). We then describe the experimental setup and the software implementation employed in our study (Section 3). Finally, we present the results obtained from our experiments and engage in a comprehensive discussion of these results (Section 4).

# 2 CMA-ES

## 2.1 Parameters and Variants

The Covariance Matrix Adaptation Evolution Strategy (CMA-ES) (Hansen, 2016; Hansen and Ostermeier, 1996) is a group of iterative heuristic algorithms designed to solve continuous optimization problems with a single objective. In each generation $g$, a population denoted as $x$ is generated, consisting of $\lambda$ offspring. These offspring are sampled from a multivariate normal distribution characterized by a mean value $m^{(g)} \in \mathbb{R}^n$, a covariance matrix $C^{(g)} \in \mathbb{R}^{n\times n}$, and a standard deviation $\sigma^{(g)} \in \mathbb{R}_{>0}$:

$$x_k^{(g+1)} \sim m^{(g)} + \sigma^{(g)} \mathcal{N}(0, C^{(g)}) \quad \forall k = 1, ..., \lambda. \tag{1}$$

Then, the best $\mu$ individuals are selected from the population to compute the new mean value $m^{(g+1)}$ with the given weights $w_i$:

$$m^{(g+1)} = \sum_{i=1}^{\mu} w_i x_{i:\lambda}^{(g+1)}, \tag{2}$$

$$\sum w_i = 1, \quad w_1 \geq w_2 \geq ... \geq w_\mu. \tag{3}$$

The covariance matrix $C^{(g)}$ is updated with the evolution path $p_c^{(g)} \in \mathbb{R}^n$:

$$\begin{aligned} C^{(g+1)} &= (1 - c_1 - c_\mu \sum_{i=1}^{\lambda} w_i) C^{(g)} \\ &+ c_1 \underbrace{p_c^{(g+1)} p_c^{(g+1)^T}}_{\text{rank-one update}} + \underbrace{c_\mu \sum_{i=1}^{\lambda} w_i \, y_{i:\lambda}^{(g+1)} (y_{i:\lambda}^{(g+1)})^T}_{\text{rank-}\mu\text{ update}}, \end{aligned} \tag{4}$$

$$\begin{aligned} p_c^{(g+1)} &= (1 - c_c) p_c^{(g)} \\ &+ \sqrt{c_c(2 - c_c)\mu_{eff}} \frac{m^{(g+1)} - m^{(g)}}{\sigma^{(g)}}, \end{aligned} \tag{5}$$

$$\mu_{eff} = (\sum_{i=1}^{\mu} w_i^2)^{-1}, \quad y_{i:\lambda}^{(g+1)} = \frac{x_{i:\lambda}^{(g+1)} - m^{(g)}}{\sigma^{(g)}}, \tag{6}$$

and the standard deviation $\sigma^{(g)}$ is updated with the conjugate evolution path $p_\sigma^{(g)} \in \mathbb{R}^n$ and a damping parameter $d_\sigma$:

$$\sigma^{(g+1)} = \sigma^{(g)} \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\left\| p_\sigma^{(g+1)} \right\|}{E\left\| \mathcal{N}(0, I) \right\|} - 1\right)\right), \tag{7}$$

$$\begin{aligned} p_\sigma^{(g+1)} &= (1 - c_\sigma) p_\sigma^{(g)} \\ &+ \sqrt{c_\sigma(2 - c_\sigma)\mu_{eff}} C^{(g)^{-\frac{1}{2}}} \frac{m^{(g+1)} - m^{(g)}}{\sigma^{(g)}}. \end{aligned} \tag{8}$$

The optimization behavior of CMA-ES is determined by the parameters $\lambda$, $\mu$, $c_1$, $c_c$, $c_\mu$, and $c_\sigma$, which can be tuned for specific functions or sets of functions (Andersson et al., 2015; Zhao and Li, 2018). Moreover, several variations of the CMA-ES were developed (Bäck et al., 2013). In this study, we examine the following variants within modular CMA-ES (de Nobel et al., 2021; van Rijn et al., 2016): Active Update (Jastrebski and Arnold, 2006), Elitism (van Rijn et al., 2016), Mirrored Sampling (Brockhoff et al., 2010), Orthogonal Sampling (Wang et al., 2014), Threshold Convergence (Piad-Morffis et al., 2015), Weighted Recombination (Hansen and Ostermeier, 2001), Restart with increasing population (IPOP) (Auger and Hansen, 2005) or bi-population (BIPOP) (Hansen, 2009), Bound Correction (Caraffini et al., 2019).

### 2.2 CMA-ES with Margin

The canonical CMA-ES is designed for continuous problems. CMA-ES can be applied to discrete problems by rounding the continuous values from CMA-ES to the allowed discrete values, resulting in plateaus between the rounded values of size $\rho$ (Hansen, 2011; Thomaser et al., 2023a). However, its effectiveness decreases. This limitation arises from the self-adaption mechanism of the CMA-ES, which can cause the variance of the mutation distribution to become smaller than the granularity of the discretization. In other words, when the mutation step is smaller than the plateau size $\rho$, the optimization tends to remain on the plateau.

To address this issue, Hamano et al. (Hamano et al., 2022) introduced a modification to CMA-ES known as CMA-ES with margin (CMA-ESwM). This approach involves incorporating a diagonal matrix, denoted as $\mathbf{A}$, into the mutation distribution $\mathcal{N}(\mathbf{m}, \sigma^2 \mathbf{ACA}^T)$. The purpose of this modification is to ensure that the marginal probabilities of the mutation distribution are lower bounded, guaranteeing a minimum probability $\alpha$ that the mutation steps are larger than the plateau size $\rho$. The adaption of $\mathbf{A}$ and $\mathbf{m}$ is performed in each generation.

In the proposed CMA-ES with margin, Hamano et al. suggest using $\alpha = \frac{1}{\lambda n}$ as the default margin value. Experimental results on the bbob-mixint testbed (Tušar et al., 2019) demonstrate that CMA-ESwM outperforms several other methods, especially in higher-dimensional scenarios.

## 3 EXPERIMENTAL SETUP

### 3.1 CMA-ES Performance Assessment

To optimize the performance of CMA-ES in solving the original problem, a performance metric is needed. Tuning the parameters of an optimization algorithm with a fixed budget will yield optimal parameters only for that specific budget (Thomaser et al., 2023c). Hence, to assess the effectiveness of an optimization algorithm in terms of anytime performance, we utilize the area under the curve (AUC) of its empirical cumulative distribution function (ECDF) as a measure, as suggested by Ye et al. (Ye et al., 2022). To compute the ECDF curves, we consider 81 target values logarithmically distributed from $10^8$ to $10^{-8}$. The objective is to maximize the AUC value.

As the original optimization problems, we utilize four from the black-box optimization benchmark suite (BBOB) (Hansen et al., 2009). These functions, namely $F1, F4, F20$, and $F21$, serve as benchmarks for evaluating the effectiveness of CMA-ES. While $F1$ and $F4$ have a global structure and are separable, $F20$ and $F21$ have no global structure and are not separable. $F1$ is unimodal and $F4, F20, F21$ are multimodal. To reduce the computational effort, we consider the functions in two dimensions only.

In each run of the Covariance Matrix Adaptation Evolution Strategy (CMA-ES), we allocate a maximum evaluation budget of 400 for the BBOB function $F1$, and 2000 for the BBOB functions $F4, F20, F21$. The reason for the smaller budget in the case of $F1$ is that, unlike the other three functions, $F1$ is unimodal. An optimization problem involves four instances of the same BBOB function. We evaluate the first four instances of each BBOB function by performing 25 runs per instance, for a total of 100 CMA-ES runs. This procedure is used to evaluate the effectiveness of a CMA-ES configuration by calculating the AUC.

### 3.2 Parameters and Meta-Algorithms

Table 1 presents an overview of the parameters and variants of CMA-ES considered for tuning in this study. The learning rates $c_1$, $c_c$, $c_\mu$, and $c_\sigma$ are continuous, while the population size $\lambda$ is an integer, and the remaining variables are categorical. The values considered represent a realistic problem faced by a user who wants to find a well performing configuration of CMA-ES. Tuning these CMA-ES parameters is a mixed-integer optimization problem.

To solve the mixed-integer parameter tuning optimization problem with CMA-ES itself, we use the margin extension from (Hamano and Saito, 2022). Furthermore, we combine the margin extension with the modular CMA-ES (de Nobel et al., 2021; van Rijn et al., 2016). This allows us to leverage variants such as mirrored sampling within CMA-ESwM as a meta-algorithm. Previous studies (Thomaser et al., 2023c; Wang et al., 2014; Wang et al., 2019) have shown that mirrored and orthogonal sampling generally improve the exploration of CMA-ES. Increasing the ini-

Table 1: Parameter space for tuning CMA-ES.

| Parameter | Description | Variants and Parameters |
|---|---|---|
| $c_1$ | Learning rate rank-one update | ]0, 1] |
| $c_c$ | Learning rate covariance matrix adaption | ]0, 1] |
| $c_\mu$ | Learning rate rank-$\mu$ update | ]0, 1] |
| $c_\sigma$ | Learning rate step size control | ]0, 1[ |
| $\lambda$ | Number of children derived from parents | {4,6,..,20} |
| $\mu_r$ | Ratio of parents selected from population | {0.3, 0.5, 0.7} |
| $\sigma_0$ | Initial standard deviation | {0.2, 0.4, 0.6, 0.8} |
| Bound correction | Correction if individual out of bounds | {saturate, unif, COTN, toroidal, mirror} |
| Active update | Covariance matrix update variation | {on, off} |
| Elitism | Strategy of the evolutionary algorithm | $\{(\mu,\lambda),(\mu+\lambda)\}$ |
| Mirrored sampling | Mutations are the mirror image of another | {on, off} |
| Orthogonal | Orthogonal sampling | {on, off} |
| Threshold | Length threshold for mutation vectors | {on, off} |
| Weights | Weights for recombination | {default, equal, $\frac{1}{2}^{\lambda}$} |
| Restart | Local restart of CMA-ES | {IPOP, BIPOP} |

tial standard deviation $\sigma_0$ and the population size $\lambda$ can also lead to better global performance. Therefore, we increase the population size from 12 to 18 and the initial standard deviation from 0.2 to 0.6 compared to the default parameter values of modular CMA-ES.

We compare two versions of the CMA-ESwM as meta-algorithm: one using the default values from the modular CMA-ES, and another using a modified CMA-ESwM with adjusted parameter values, as described above. Both versions use saturation as the bound correction method.

To handle categorical and integer values, a transformation is required when using CMA-ESwM. To accomplish this, we use the ordinal encoder and min-max scaler provided by scikit-learn (Pedregosa et al., 2011). First, integer and categorical values are ordinal encoded, followed by scaling to the range $[-5,5]$, which are the default values for the lower and upper bounds within modular CMA-ES. Continuous parameters are only scaled to the same range of $[-5,5]$. To illustrate, the three considered values $\{0.3,0.5,0.7\}$ for the selection ratio $\mu_r$ are first ordinal encoded to $\{0,1,2\}$ and then scaled to $\{-5,0,5\}$.

For the purpose of comparison with the modular CMA-ESwM, we used several other meta-algorithms, namely SMAC3 (version 2.0.0) (Lindauer et al., 2022), Optuna's TPE sampler and Random sampler (version 3.2.0) (Akiba et al., 2019), each using their default configurations. Our evaluation budget for the meta-algorithm was set at 3 000, and we performed 50 full parameter tuning runs on each BBOB function for each meta-algorithm.

# 4 RESULTS

Figure 2 illustrates the average performance of CMA-ES across 50 parameter tuning runs for each meta-algorithm considered, on the four BBOB functions $F1, F4, F20, F21$, which serve as the original optimization problems. The objective is to maximize the AUC. Each evaluation of a CMA-ES configuration involves 100 optimization runs on the original problem.

The results show that the majority of performance improvements in CMA-ES parameters can be achieved within the first 1 000 evaluations for all four BBOB functions. Subsequent improvements are relatively small. Both the modified CMA-ESwM and the TPE exhibit similar progressions over the evaluations, with TPE performing slightly better in the early stages (up to 500 evaluations), and CMA-ESwM mostly outperforming TPE thereafter. Around 500 evaluations, SMAC may initially appear slower in discovering good solutions compared to the other algorithms. However, its performance steadily improves over time, reaching a similar performance compared to the other meta-algorithms mentioned above. In contrast, the random search stagnates and its progress decreases significantly after 500 evaluations. CMA-ESwM with default parameters shows a worse performance compared to the modified CMA-ESwM. This emphasizes parameter tuning of CMA-ES, not only for optimizing the original optimization problem but also as a meta-algorithm for tuning itself.

To ensure a more accurate assessment of the best configuration found by a meta-algorithm, we rerun the same configurations again 50 times for validation and calculate the median. Figure 3 shows these validated AUC values.

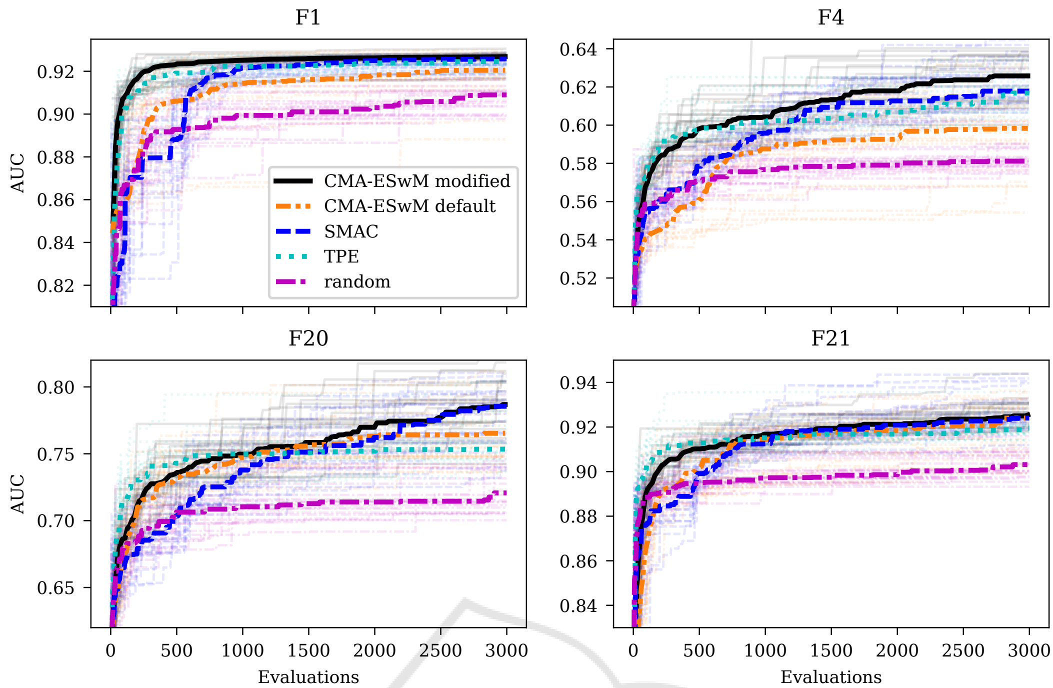


Figure 2: Median AUC values over evaluations of 50 runs (single runs transparent) for the different meta-algorithms considered for tuning CMA-ES parameters on the four 2-dimensional BBOB functions $F1, F4, F20, F21$.

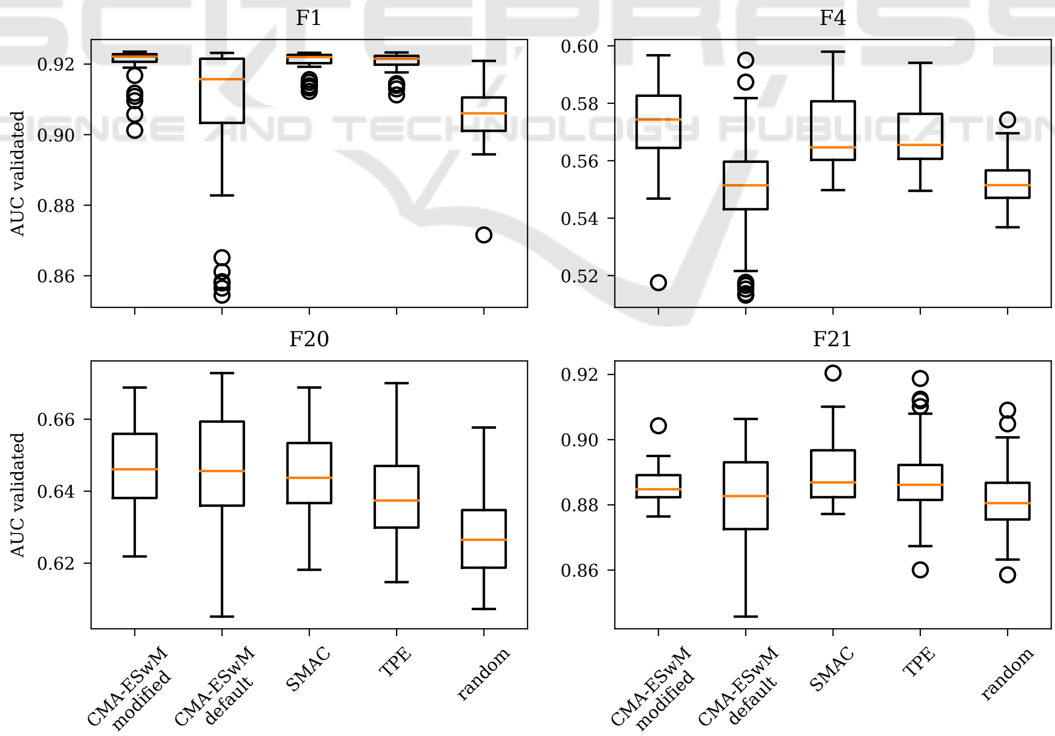


Figure 3: Boxplot of the validated AUC values of the best CMA-ES configurations found by the different meta-algorithms on each of the four BBOB functions considered. For each meta-algorithm and BBOB function, 50 parameter tuning runs were performed. Each of the configurations found in this process is in turn validated by 50 validation runs.

While CMA-ESwM with default parameters can find comparably good configurations, many are worse than the solutions found by random search, especially in the case of the two functions $F1$ and $F4$. In the median, the modified CMA-ESwM finds the best configuration for $F1$, $F4$, and $F20$. For $F21$ SMAC finds the best configuration, followed by TPE and the modified CMA-ESwM. In summary, the modified CMA-ESwM performs best in three out of four cases.

To further investigate whether the differences in performance between the meta-algorithms are statistically significant, we employ the Mann-Whitney U test (Mann and Whitney, 1947) within SciPy (Virtanen et al., 2020) with the alternative hypothesis *greater*. We compare the considered meta-algorithms pairwise and for each considered function (Figure 4). If the p-value is below 0.05, we reject the null hypothesis in favor of the alternative hypothesis, thus the performance of the algorithm (y-axis) is greater than that of the other algorithm (x-axis).

The p-value of the modified CMA-ESwM, SMAC, and TPE when compared with random search is far below 0.05 for each function considered (last column in Figure 4). Thus, based on the Mann-Whiteney U test, our results show that the modified CMA-ESwM, SMAC, and TPE outperform random search as a meta-algorithm.

Moreover, regarding the Mann-Whitney U test, the modified CMA-ESwM performs significantly better than TPE on $F1$, $F4$, $F20$, and SMAC on $F4$. Only on $F21$ SMAC performs significantly better than the modified CMA-ESwM.

However, the modified CMA-ESwM, SMAC, and TPE show similar performance but differ in their wall clock times. On average, CMA-ESwM is the fastest of the three. This advantage is due to its ability to parallelize the population within a single generation, while the others evaluate configurations sequentially. This leads to the result that although the computational cost of evaluating a new configuration within random search is negligible, random search takes about 50 % more time than CMA-ESwM to complete a parameter tuning run. In contrast, both the SMAC and TPE algorithms require about two to three times more time than CMA-ESwM. This increased time is due not only to their sequential evaluation procedures but also to the additional internal computations and model training involved in these methods, which will not be lowered even if SMAC and TPE are implemented with more parallelization.

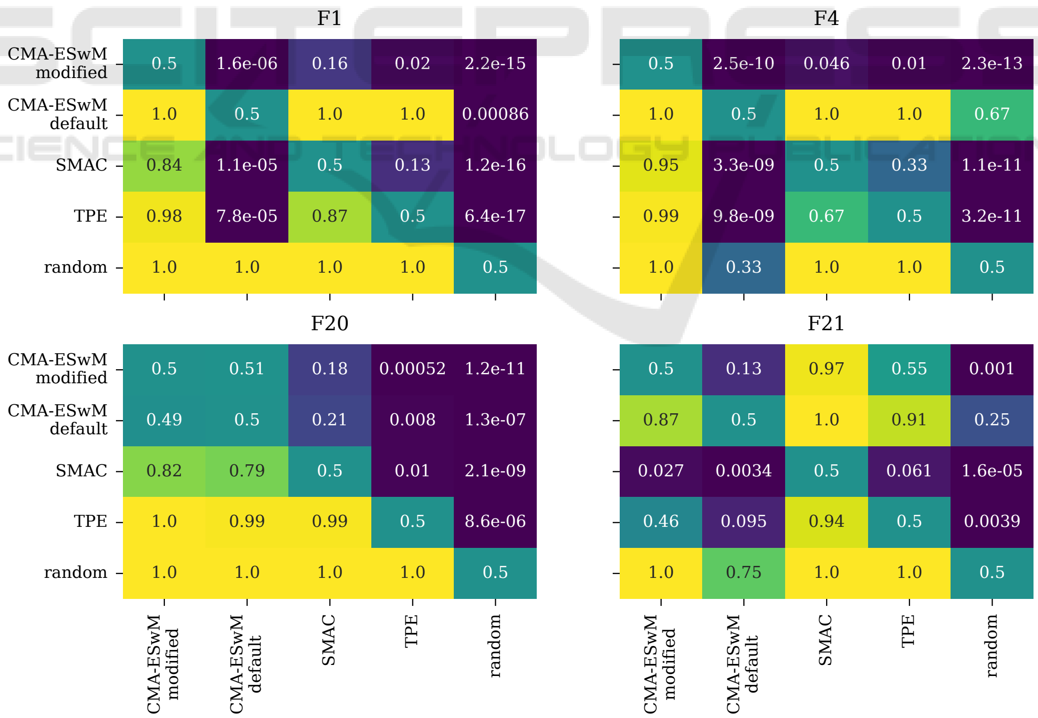


Figure 4: P-values from the Mann-Whitney U test (Mann and Whitney, 1947) with the alternative hypothesis *greater* when comparing the performance of the five meta-algorithms considered pairwise with each other. The meta-algorithm on the y-axis is compared to the meta-algorithm on the x-axis. If the p-value is below 0.05, the null hypothesis can be rejected in favor of the alternative, thus the performance of the meta-algorithm on the y-axis is greater than the performance of the other algorithm on the x-axis. To assess the performance of a meta-algorithm, 50 parameter tuning runs were performed.

## 5 CONCLUSION

We have demonstrated significant improvements in the efficiency and effectiveness of CMA-ES by tuning its parameters. To handle the mixed-integer meta-optimization problem of parameter tuning, we used CMA-ES with margin, which effectively handles the discrete parameters. In addition, we combined the margin extension with modular CMA-ES with orthogonal mirrored sampling activated and with increased default population size and initial standard deviation to improve global exploration. As a result, our CMA-ES configuration for parameter tuning competes with state-of-the-art algorithms such as SMAC and TPE.

In terms of wall clock time, CMA-ES outperforms SMAC and TPE due to its parallelization capability and internal efficiency. This advantage further highlights the potential of CMA-ES in various domains.

It is worth noting that even with a simple random search, we can find a very good configuration. Random search is particularly advantageous in situations where fully parallel execution is feasible.

Future research can focus on expanding the range of original optimization problems considered and extending the study to other BBOB functions or benchmark sets. In addition, exploring the possibility of tuning CMA-ES as a meta-algorithm with a third optimization algorithm holds the potential for further performance improvement.

The Python code to reproduce the described results has been made available on our Zenodo repository (Thomaser et al., 2023b). This repository also contains the data of the results and additional code to re-create the presented figures.

## ACKNOWLEDGEMENTS

This paper was written as part of the project newAIDE under the consortium leadership of BMW AG with the partners Altair Engineering GmbH, divis intelligent solutions GmbH, MSC Software GmbH, Technical University of Munich, TWT GmbH. The project is supported by the Federal Ministry for Economic Affairs and Climate Action (BMWK) on the basis of a decision of the German Bundestag.

## REFERENCES

Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. (2019). Optuna: A Next-Generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, pages 2623–2631, New York, NY, USA. Association for Computing Machinery.

Andersson, M., Bandaru, S., Ng, A. H., and Syberfeldt, A. (2015). Parameter Tuned CMA-ES on the CEC'15 Expensive Problems. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 1950–1957.

Auger, A. and Hansen, N. (2005). A Restart CMA Evolution Strategy With Increasing Population Size. In *Proceedings of the IEEE Congress on Evolutionary Computation*, volume 2, pages 1769–1776.

Bäck, T. (1994). Parallel Optimization of Evolutionary Algorithms. In Goos, G., Hartmanis, J., Leeuwen, J., Davidor, Y., Schwefel, H.-P., and Männer, R., editors, *Parallel Problem Solving from Nature — PPSN III*, volume 866 of *Lecture Notes in Computer Science*, pages 418–427. Springer Berlin Heidelberg, Berlin, Heidelberg.

Bäck, T., Foussette, C., and Krause, P. (2013). *Contemporary Evolution Strategies*. Natural Computing Series. Springer Berlin, Heidelberg, Berlin, Heidelberg, 1st ed. edition.

Bergstra, J., Bardenet, R., Bengio, Y., and Kégl, B. (2011). Algorithms for Hyper-Parameter Optimization. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc.

Breiman, L. (2001). Random Forests. *Machine Learning*, 45(1):5–32.

Brockhoff, D., Auger, A., Hansen, N., Arnold, D. V., and Hohm, T. (2010). Mirrored Sampling and Sequential Selection for Evolution Strategies. In Schaefer, R., Cotta, C., Kołodziej, J., and Rudolph, G., editors, *Parallel Problem Solving from Nature, PPSN XI*, Lecture Notes in Computer Science, pages 11–21. Springer, Berlin.

Caraffini, F., Kononova, A. V., and Corne, D. (2019). Infeasibility and structural bias in differential evolution. *Information Sciences*, 496:161–179.

de Nobel, J., Vermetten, D., Wang, H., Doerr, C., and Bäck, T. (2021). Tuning as a Means of Assessing the Benefits of New Ideas in Interplay with Existing Algorithmic Modules. Technical report.

Eiben, A. E. and Smit, S. K. (2011). Parameter tuning for configuring and analyzing evolutionary algorithms. *Swarm and Evolutionary Computation*, 1(1):19–31.

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., and Hutter, F. (2015). Efficient and Robust Automated Machine Learning. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Grefenstette, J. (1986). Optimization of Control Parameters for Genetic Algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 16(1):122–128.

Hamano, R. and Saito, S. (2022). CMA-ES with Margin.

Hamano, R., Saito, S., Nomura, M., and Shirakawa, S. (2022). CMA-ES with Margin: Lower-Bounding

Marginal Probability for Mixed-Integer Black-Box Optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '22, pages 639–647, New York, NY, USA. Association for Computing Machinery.

Hansen, N. (2009). Benchmarking a BI-Population CMA-ES on the BBOB-2009 Function Testbed. In *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*, ACM Conferences, pages 2389–2396, New York, NY, USA. Association for Computing Machinery.

Hansen, N. (2011). A CMA-ES for Mixed-Integer Nonlinear Optimization: Research Report. Technical Report RR-7751, INRIA.

Hansen, N. (2016). The CMA Evolution Strategy: A Tutorial. Technical report.

Hansen, N., Finck, S., Ros, R., and Auger, A. (2009). Real-Parameter Black-Box Optimization Benchmarking 2009: Noiseless Functions Definitions. Technical Report RR-6829, INRIA.

Hansen, N. and Ostermeier, A. (1996). Adapting Arbitrary Normal Mutation Distributions in Evolution Strategies: The Covariance Matrix Adaptation. In *Proceedings of the IEEE International Conference on Evolutionary Computation*, pages 312–317.

Hansen, N. and Ostermeier, A. (2001). Completely Derandomized Self-Adaptation in Evolution Strategies. *Evolutionary Computation*, 9(2):159–195.

Hutter, F., Hoos, H. H., and Leyton-Brown, K. (2011). Sequential Model-Based Optimization for General Algorithm Configuration. In Coello, C. A. C., editor, *Learning and Intelligent Optimization*, volume 6683 of *Lecture Notes in Computer Science*, pages 507–523. Springer Berlin Heidelberg, Berlin, Heidelberg.

Jastrebski, G. A. and Arnold, D. V. (2006). Improving Evolution Strategies through Active Covariance Matrix Adaptation. In *IEEE International Conference on Evolutionary Computation*, pages 2814–2821.

Lindauer, M., Eggensperger, K., Feurer, M., Biedenkapp, A., Deng, D., Benjamins, C., Ruhkopf, T., Sass, R., and Hutter, F. (2022). SMAC3: A Versatile Bayesian Optimization Package for Hyperparameter Optimization. *Journal of Machine Learning Research*, 23(54):1–9.

Mann, H. B. and Whitney, D. R. (1947). On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*, pages 50–60.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12(85):2825–2830.

Piad-Morffis, A., Estévez-Velarde, S., Bolufé-Röhler, A., Montgomery, J., and Chen, S. (2015). Evolution Strategies with Thresheld Convergence. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 2097–2104.

Thomaser, A., de Nobel, J., Vermetten, D., Ye, F., Bäck, T., and Kononova, A. V. (2023a). When to be Discrete: Analyzing Algorithm Performance on Discretized Continuous Problems. Technical report.

Thomaser, A., Vogt, M.-E., Bäck, T., and Kononova, A. V. (2023b). Optimizing CMA-ES with CMA-ES - Data and Code. https://doi.org/10.5281/zenodo.8256601.

Thomaser, A., Vogt, M.-E., Kononova, A. V., and Bäck, T. (2023c). Transfer of Multi-objectively Tuned CMA-ES Parameters to a Vehicle Dynamics Problem. In Emmerich, M., Deutz, A., Wang, H., Kononova, A. V., Naujoks, B., Li, K., Miettinen, K., and Yevseyeva, I., editors, *Evolutionary Multi-Criterion Optimization*, pages 546–560, Cham. Springer Nature Switzerland.

Tušar, T., Brockhoff, D., and Hansen, N. (2019). Mixed-Integer Benchmark Problems for Single- and Bi-Objective Optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '19, pages 718–726, New York, NY, USA. Association for Computing Machinery.

van Rijn, S., Wang, H., van Leeuwen, M., and Bäck, T. (2016). Evolving the structure of Evolution Strategies. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, VanderPlas, Jake, Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Wang, H., Emmerich, M., and Bäck, T. (2014). Mirrored Orthogonal Sampling with Pairwise Selection in Evolution Strategies. In *Proceedings of the 29th Annual ACM Symposium on Applied Computing*, SAC '14, pages 154–156, New York, NY, USA. Association for Computing Machinery.

Wang, H., Emmerich, M., and Bäck, T. (2019). Mirrored Orthogonal Sampling for Covariance Matrix Adaptation Evolution Strategies. *Evolutionary Computation*, 27(4):699–725.

Ye, F., Doerr, C., Wang, H., and Bäck, T. (2022). Automated Configuration of Genetic Algorithms by Tuning for Anytime Performance. *IEEE Transactions on Evolutionary Computation*, page 1.

Zhao, M. and Li, J. (2018). Tuning the hyper-parameters of CMA-ES with tree-structured Parzen estimators. In *2018 Tenth International Conference on Advanced Computational Intelligence (ICACI)*, pages 613–618.