PFN-Based Online Selection of Surrogate-Assisted CMA-ES Strategies

Vojtech Blaha[1,*,†], Vladyslav Furda[2,†] and Martin Holena[3]

[1]Department of Theoretical Computer Science and Mathematical Logic, Charles University, Prague, Czechia
[2]Faculty of Mathematics and Physics, Charles University, Prague, Czechia
[3]Institute of Computer Science, Czech Academy of Sciences, Prague, Czechia

Abstract

Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is a widely used algorithm for continuous black-box optimization, but its applicability to expensive optimization problems is often limited by the number of required objective evaluations. Surrogate-assisted variants address this issue by replacing a subset of expensive evaluations with surrogate model predictions. However, no single surrogate-assisted strategy consistently performs best across all optimization landscapes. To address this limitation, we propose a decision framework based on prior-data fitted networks (PFNs) that dynamically selects among multiple surrogate-assisted CMA-ES bundles during optimization. Each bundle combines a surrogate model with an evolution-control mechanism, while the decision model is trained on a dataset generated through counterfactual one-step lookahead evaluations that quantify the expected utility of alternative bundle choices from identical optimizer states. The proposed approach is evaluated on the 24 noiseless functions of the COCO BBOB benchmark in dimension 5 using a leave-one-function-out protocol over 15 problem instances. Results show that the PFN selector was most competitive at intermediate budgets. At 200 true evaluations, it achieved an overall target-hit ratio of 0.1787, compared with 0.1781 for its most frequently selected policy, LQ quadratic top-30, and 0.1843 for the strongest fixed policy. On f_{10}-f_{14} at the same budget, PFN improved over LQ quadratic from 0.0925 to 0.1038, although GP Matérn UNC remained stronger at 0.1176. These results show that learned switching can outperform state-free selection at early budgets and its dominant fixed action in specific regimes, but does not yet outperform the strongest fixed policy overall.

Keywords

CMA-ES, surrogate-assisted optimization, black-box optimization, dynamic surrogate selection, Prior-Data Fitted Networks, BBOB benchmark

CEUR-WS.org/Vol-4323/paper5.pdf

1. Introduction

Many real-world optimization problems involve objective functions that are expensive to evaluate, non-differentiable, or accessible only through black-box procedures. Examples include engineering design optimization, scientific simulations, automated parameter tuning, and materials discovery, where a single objective evaluation may require substantial computational or experimental resources [1, 2, 3]. In such scenarios, reducing the number of expensive objective evaluations is often more important than minimizing the computational cost of the optimization algorithm itself.

Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is one of the most successful algorithms for continuous black-box optimization [4, 5, 6]. Due to its adaptive covariance matrix updates and strong invariance properties, CMA-ES achieves robust performance across a wide range of optimization landscapes [7]. However, its practical applicability is limited by the potentially large number of black-box objective function evaluations required to reach high-quality solutions.

To alleviate this limitation, surrogate-assisted variants of CMA-ES employ predictive models that approximate the objective function using previously evaluated candidate solutions [8, 9]. Numerous surrogate models with various evolution-control mechanisms have been proposed, including local quadratic approximations [10], Gaussian-process-based approaches [11], random forests [12], and

ITAT'26: Information Technologies – Applications and Theory, September 25–29, 2026, Biele Karpaty, Slovakia
*Corresponding author.
†These authors contributed equally.
vojtech.blaha600@student.cuni.cz (V. Blaha)
0009-0006-7409-3598 (V. Blaha); 0009-0003-4722-1169 (V. Furda); 0000-0002-2536-9328 (M. Holena)
© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2026-11-09

ranking-based surrogate models [13, 14]. Although these methods can substantially reduce the number of expensive objective evaluations, no single surrogate-assisted strategy consistently performs best across all optimization problems. The effectiveness of a particular surrogate model and evolution-control mechanism strongly depends on the characteristics of the objective function and may vary during different stages of the optimization process.

This observation naturally motivates the use of online algorithm selection techniques capable of dynamically choosing the most suitable surrogate-assisted strategy during optimization. Instead of relying on a single surrogate model, one may consider a portfolio of surrogate-assisted CMA-ES variants and adaptively select the most promising strategy based on the current optimization state. Such an approach can potentially combine the strengths of multiple surrogate models while mitigating their individual weaknesses.

Recently, Prior-Data Fitted Networks (PFNs) have emerged as a promising framework for learning predictive inference directly from large collections of generated tasks [15]. However, their use for online decision making within evolutionary optimization algorithms remains largely unexplored.

In this paper, we propose an adaptive surrogate portfolio framework for CMA-ES driven by a PFN-based decision model. We employ a transformer-based PFN decision model that dynamically selects among a portfolio of surrogate-assisted CMA-ES policies. Each policy combines a surrogate model with a corresponding evolution-control strategy. The decision model is trained on a dataset generated through counterfactual one-step lookahead evaluations, where all candidate policies are assessed from identical CMA-ES states and scored according to their achieved improvement relative to the number of expensive objective evaluations consumed. In each generation of the evolutionary optimization, the trained PFN model selects the policy predicted to provide the highest optimization benefit in the current generation.

The proposed approach is evaluated on the noiseless COCO BBOB benchmark suite [16] using a leave-one-function-out evaluation protocol. The results show that PFN is competitive with strong fixed policies around 100–200 true evaluations. A matched-instance comparison shows that learned switching improves over the frequently selected LQ quadratic policy on f_{10}–f_{14} at 200 evaluations, although GP Matérn UNC remains the strongest fixed baseline. These findings demonstrate both the potential of state-dependent selection without hand-designed switching rules and the need to reduce the selector's concentration on a single policy.

2. Background and Related Work

2.1. Covariance Matrix Adaptation Evolution Strategy

Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is one of the most successful evolutionary algorithms for continuous black-box optimization [4, 5, 6]. The algorithm iteratively samples candidate solutions from a multivariate normal distribution and adapts its covariance matrix and global step size according to the observed search progress. This adaptation mechanism allows CMA-ES to learn dependencies between decision variables and efficiently exploit local structure of the search landscape.

A key advantage of CMA-ES is its invariance with respect to monotonic transformations of the objective function and affine transformations of the search space [5]. Owing to these properties, CMA-ES demonstrates robust performance across a broad range of continuous optimization problems [7]. However, its practical applicability to expensive optimization scenarios is often limited by the large number of objective evaluations required.

2.2. Surrogate-Assisted CMA-ES

Surrogate-assisted evolutionary optimization seeks to reduce the number of expensive objective evaluations by replacing some evaluations with predictions obtained from surrogate models trained on previously evaluated solutions [8, 9]. In the context of CMA-ES, surrogate models are commonly

combined with evolution-control mechanisms that determine which candidate solutions are evaluated using the true objective function and which receive surrogate-predicted fitness values.

Numerous surrogate-assisted CMA-ES variants have been proposed. Representative approaches include ranking-based surrogate models [13, 14], local quadratic approximations [10], Gaussian-process-based methods [11], and random-forest surrogates [12]. These methods differ in their modeling assumptions, computational complexity, uncertainty estimation capabilities, and evolution-control strategies.

Although surrogate-assisted CMA-ES algorithms can significantly reduce evaluation costs, no single surrogate model consistently outperforms all alternatives across different optimization problems. The suitability of a particular surrogate-assisted strategy often depends on the properties of the objective function as well as the current stage of the optimization process.

2.3. Algorithm Selection and Hyper-Heuristics

The observation that different optimization strategies perform well under different conditions can be interpreted as a specialization of the general algorithm selection problem introduced by Rice, which concerns the selection of a suitable algorithm for a given problem instance [17]. Related approaches include hyper-heuristics, which operate at a higher level by selecting or generating optimization heuristics during the search process [18].

Within evolutionary optimization, algorithm selection has been applied to adaptive operator selection, portfolio-based optimization, and dynamic configuration of optimization algorithms [19, 20]. Rather than relying on a single optimization strategy, these approaches seek to exploit complementary strengths of multiple methods.

The framework proposed in this paper follows a similar philosophy. However, instead of selecting among complete optimization algorithms, it dynamically selects among policies of surrogate-assisted CMA-ES variants, where each policy combines a surrogate model with a corresponding evolution-control mechanism.

Online algorithm selection is also closely related to adaptive operator selection and contextual-bandit methods, which update a selection rule from short-term rewards, and to reinforcement-learning approaches that model dynamic selection as a sequential decision problem. Our approach uses full-information counterfactual supervision: every portfolio action is evaluated from the same recorded state during data generation. It therefore avoids online exploration during deployment, but uses a myopic one-generation target rather than optimizing a multi-step return.

2.4. Prior-Data Fitted Networks

Prior-Data Fitted Networks (PFNs) are transformer-based models trained to approximate Bayesian predictive inference through exposure to a large collection of synthetically generated learning tasks [15]. Unlike conventional machine learning models, PFNs perform prediction in-context and do not require task-specific gradient-based training at inference time.

Recent work has demonstrated the effectiveness of PFN-based models in low-data learning and Bayesian optimization settings [21]. Their ability to perform inference directly from context makes them a promising candidate for decision-making tasks where only limited information is available.

While PFNs have been successfully used for in-context predictive inference and as surrogate models in Bayesian optimization, their use as decision models for online strategy selection within evolutionary optimization remains largely unexplored. This motivates the investigation presented in this paper.

3. Proposed Method

The proposed framework consists of three main components: a portfolio of surrogate policies, a counterfactual dataset-generation procedure, and a PFN-based decision model. During optimization, the decision model selects one policy from the portfolio based on the current CMA-ES state. The selected

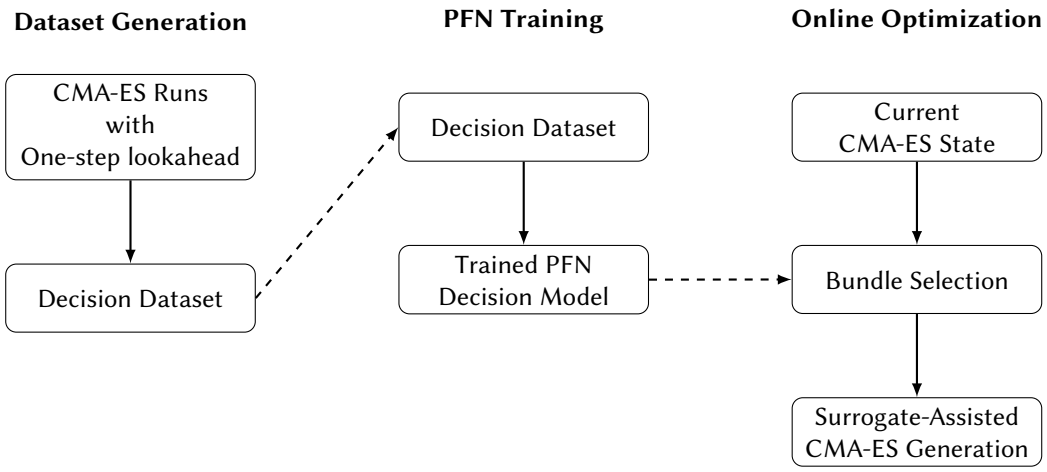


Figure 1: Overview of the proposed framework. Decision data are generated from CMA-ES runs and used to train a PFN-based decision model. During online optimization, the trained model selects a surrogate-assisted bundle for the current CMA-ES generation.

policy is subsequently used to perform one iteration of surrogate-assisted optimization in the CMA-ES state.

The decision model is trained offline using data obtained from counterfactual one-step lookahead evaluations. Figure 1 illustrates the overall workflow of the proposed approach.

3.1. Surrogate Portfolio

The proposed framework operates on a portfolio of surrogate-assisted CMA-ES policies. Each policy combines a surrogate model with an evolution-control mechanism responsible for determining which candidate solutions are evaluated using the true objective function and which receive surrogate-predicted fitness values.

Specifically, the portfolio includes representatives of the major surrogate-model families discussed in Section 2.2, including quadratic approximation models, Gaussian processes, random forests, and ranking-based support vector machines. These surrogate models are combined with several evolution-control mechanisms, including top-fraction evaluation, uncertainty-guided evaluation, adaptive model lifelength control, and doubly-trained surrogate control, which represent commonly used strategies for balancing surrogate predictions and expensive black-box function evaluations [8, 11, 14].

In addition to surrogate-assisted policies, the portfolio also contains a baseline strategy that evaluates all candidate solutions using the true objective function without employing surrogate modeling. This allows the decision model to fall back to standard CMA-ES whenever surrogate-assisted evaluation is predicted to be unbeneficial.

The selector's action space contains exactly seven predefined surrogate-control policies, listed in Table 1. The surrogate and evolution-control columns are not independently recombined; thus, only these seven complete policies, rather than their Cartesian product, were evaluated.

The portfolio is defined over complete surrogate–control policies rather than the Cartesian product of all surrogate models and evolution controls. Some controls require model-specific capabilities, such as predictive uncertainty, repeated model fitting, or persistent refresh state, while other combinations would represent closely related parameter variants. We therefore selected a compact set of compatible and representative policies so that every action could be evaluated counterfactually at every recorded state. Factorized selection of the surrogate and evolution control is left for future work.

3.2. State Representation

The PFN decision model receives a representation of the current optimization state and predicts the surrogate-assisted policy to be used in the current CMA-ES generation. The representation is constructed

Table 1
Surrogate-assisted policies used in the proposed framework.

Policy	Surrogate	Evolution Control
lq_linear_top50	Local Linear	Top 50%
lq_quadratic_top30	Local Quadratic	Top 30%
gp_matern_dts	GP Matérn	DTS
gp_matern_unc	GP Matérn	Uncertainty-Based
rf_lifelength	Random Forest	Adaptive Lifelength
svm_rank_top50	Rank-SVM	Top 50%
real_only	None	Evaluate All

from information available to the optimizer at the decision point and is designed to capture both the search history and the current state of the optimization process.

The optimization state at generation g is represented as

$$S^{(g)} = \left(H_x^{(g)}, H_y^{(g)}, P^{(g)}, x_{\text{best}}^{(g)}, y_{\text{best}}^{(g)}, A^{(g)}\right).$$

Here, $P^{(g)}$ is the candidate population sampled by CMA-ES for generation g. The history $H_x^{(g)}, H_y^{(g)}$ contains the at most $N_H = 128$ most recent points from the current run that have been evaluated by the true objective function, together with their objective values.

We use $N_H = 128$ as a fixed computational cap that bounds the memory and attention cost of the set-based model while emphasizing recent observations. The cap is inactive during the first 100 true evaluations, where the complete available history is retained. We did not perform an ablation of N_H, and its effect at larger budgets remains to be investigated.

The pair $(x_{\text{best}}^{(g)}, y_{\text{best}}^{(g)})$ denotes the incumbent solution, defined as the best solution observed by the true objective function before generation g. It is therefore not determined by surrogate predictions or by the ranking used by a particular evolution-control mechanism. Finally, $A^{(g)}$ contains auxiliary generation- and run-level descriptors, including the generation index, dimension, population size, current CMA-ES step size, and the number of true objective evaluations performed so far.

The amount of available optimization history and the number of current candidates can vary between generations. The PFN-style model therefore treats the evaluated history as a context set and the current population as a query set, rather than requiring a fixed-length feature vector. It combines information about the observations, candidates, incumbent solution, objective-value ranks, recency, optimization progress, and CMA-ES step size. We investigate whether this information is sufficient to select a suitable surrogate-control policy without hand-designed state summaries or online model updates.

Before being passed to the model, objective values are centered around the incumbent and standardized using the retained history. Rank features are included as well. This reduces sensitivity to shifts and positive rescaling of the objective values. However, the learned selector does not necessarily preserve all invariance properties of CMA-ES, and we did not evaluate these properties separately.

3.3. Counterfactual Dataset Generation

Training the decision model requires supervision indicating which surrogate-assisted policy should be selected in a given optimization state. Since such information is not directly available, we construct a decision dataset using counterfactual one-step lookahead evaluations.

For each recorded optimization state $S^{(g)}$, all policy from the portfolio are evaluated independently from an identical CMA-ES state. Each policy performs a surrogate-assisted generation, updates its internal optimizer copy, and produces a lookahead population that is subsequently evaluated using the true objective function. This procedure allows all policies to be compared under identical conditions without affecting the original optimization trajectory.

We use a one-generation horizon because it permits all actions to be evaluated from an identical optimizer state without specifying the policy that should be followed after the counterfactual action.

Extending the horizon would make each target dependent on subsequent policy choices and would introduce a branching cost that grows with both the portfolio size and rollout depth. The resulting utility should therefore be interpreted as a proxy for decision quality, not as a guarantee of long-horizon optimization performance.

The utility of policy b is quantified according to the improvement achieved during the lookahead step relative to the number of expensive objective evaluations consumed:

$$u_b = \frac{\max\left(y_{\text{best}} - y^{(b)}_{\text{lookahead}}, 0\right)}{n^{(b)}_{\text{eval}}},$$

where y_{best} denotes the incumbent objective value before applying the bundle, $y^{(b)}_{\text{lookahead}}$ is the best objective value obtained in the lookahead population generated by policy b, and $n^{(b)}_{\text{eval}}$ denotes the number of current-generation candidates evaluated using the true objective under policy b.

The resulting utility scores are used to construct the decision dataset for training the PFN decision model. For each state, let $u_{\max}$ and $u_{\min}$ denote the highest and lowest utility observed among all policies. A policy b is assigned a positive target label if

$$u_b \geq u_{\max} - \epsilon(u_{\max} - u_{\min}),$$

where $\epsilon = 0.1$ in our experiments. Thus, the target may include multiple policies whose utilities are close to the best observed one, reducing sensitivity to noise and allowing several policies to be considered suitable for the same optimization state.

Each record in the decision dataset consists of an optimization state $S^{(g)}$, the corresponding policy utility scores, and the derived target labels used for training the decision model.

3.4. PFN Decision Model

The bundle-selection task is formulated as a supervised learning problem. The selector is a custom PFN-style set model trained offline across prior optimization tasks. At inference, the evaluated history and objective values form a variable-size context set, the current CMA-ES population forms the query set, and learned action tokens represent the available policies. The model returns all policy scores in one forward pass and receives no gradient updates during an optimization run. In contrast to a conventional fixed-vector classifier, it operates directly on the variable-size history and candidate sets and shares the same learned inference procedure across held-out functions.

For each policy $b_k \in B$, the model predicts a suitability score

$$\hat{y}_k = f_\theta(S^{(g)})_k,$$

where f_θ denotes the PFN decision model and $S^{(g)}$ is the current optimization state.

The model is trained using a multi-label binary cross-entropy loss. During inference, scores within a fixed margin $\delta = 10^{-3}$ of the maximum are treated as ties. With policies indexed according to the fixed order in Table 1, the selected policy is

$$k^\star = \min\left\{k \,:\, \hat{y}_k \geq \max_j \hat{y}_j - \delta\right\}.$$

3.5. Optimization Procedure

Once trained, the decision model is integrated into the CMA-ES optimization loop. At the beginning of each generation, it predicts suitability scores for all policies and selects $b_{k^\star}$ using the deterministic margin rule defined in Section 3.4.

Depending on the selected bundle, a subset of candidate solutions is evaluated using the true objective function, while the remaining candidates receive surrogate-predicted fitness values. The resulting

fitness information is used to update the CMA-ES search distribution. This procedure is repeated until the evaluation budget is exhausted or another stopping criterion is satisfied.

Algorithm 1 summarizes the overall online optimization procedure.

Algorithm 1 PFN-Guided Surrogate Selection in CMA-ES

1: Initialize CMA-ES
2: **while** stopping criterion not satisfied **do**
3: Construct optimization state $S^{(g)}$
4: Predict policy scores using PFN
5: Select $k^\star = \min\{k : \hat{y}_k \geq \max_j \hat{y}_j - \delta\}$ and set $b^\star = b_{k^\star}$
6: Perform surrogate-assisted evaluation of the current generation using b^*
7: Update CMA-ES
8: **end while**
9: Return best solution found

4. Experimental Setup

We evaluated our approach on the noiseless single-objective BBOB suite from COCO, consisting of 24 scalable functions. All experiments were conducted in dimension $D = 5$. Here, dimension denotes the number of continuous decision variables. We selected $D = 5$ as a controlled low-dimensional setting because decision-data generation evaluates all seven policies counterfactually at every recorded state and LOFO training requires 24 separate checkpoints. The present results therefore do not establish scalability to higher-dimensional problems. We used the standard BBOB grouping into separable functions (f_1-f_5), low/moderate conditioning (f_6-f_9) and highly conditioned unimodal functions (f_{10}-f_{14}), multimodal functions with adequate global structure (f_{15}-f_{19}), and multimodal functions with weak global structure (f_{20}-f_{24}).

Decision data were generated from BBOB instances 1-20, while online evaluation used instances 1-15. We adopted a leave-one-function-out (LOFO) protocol: the model evaluated on function f_j was trained without any states originating from f_j, yielding 24 independently trained checkpoints.

Although the numeric instance ranges used for decision-data generation (1-20) and online evaluation (1-15) overlap, no state from the evaluated function is present in its training data: the checkpoint for f_j is trained only on states from functions f_k, $k \neq j$. Thus, an evaluation function-instance pair is never reused as a training task. The protocol evaluates transfer across functions; using disjoint instance identifiers and unseen benchmark families would provide a stronger test of distribution shift.

4.1. Decision-Data Generation

Trajectories were generated using CMA-ES with population size $\lambda = 10$, initial mean $x_0 = 0$, and initial step size $\sigma_0 = 2$. Each trajectory was limited to 1000 generations. At every recorded CMA-ES state, all seven surrogate–control pairs from Table 1 were evaluated using the counterfactual one-step lookahead procedure introduced in Section 3.3. The final pool contained about 551000 records.

4.2. Model Architecture and Training

The PFN used the state representation described in Section 3.2. The decision model used a transformer architecture with hidden dimension 128, eight attention heads, GELU activations, feed-forward expansion factor 4, and dropout 0.1. It consisted of four context-processing layers, two candidate-processing layers, and two action-processing layers. Each LOFO checkpoint was trained for 25 epochs using Adam with batch size 32, learning rate 10^{-3}, and weight decay 10^{-5}. These hyperparameters were fixed and used unchanged for every LOFO fold, no function-specific, budget-specific hyperparameter optimization was performed. The results therefore characterize this single model configuration rather than a full

potential of PFN architecture as such. Training used a 10000-record shuffle buffer and at most 10000 mini-batches per epoch.

4.3. Online Evaluation and Baselines

Each function was evaluated using its corresponding LOFO checkpoint. The PFN decision policy was compared with seven fixed policies corresponding exactly to the seven surrogate-control pairs in Table 1. We additionally evaluated two state-free dynamic baselines over the same policy portfolio. Uniform-random selection chose each policy independently with probability $1/7$ at every generation and was repeated with nine independent action-selection seeds (1000–1008). Round-robin selection cycled deterministically through the policies in the order listed in Table 1 as well. The action-selection random generator was independent of the CMA-ES initialization, for which all methods used optimizer seed 42. The evaluation comprised 2,880 PFN and fixed-policy runs, 3,240 uniform-random runs, and 360 round-robin runs, giving 6,480 runs in total.

4.4. Performance Measures

The primary performance measure was a fixed-budget target-hit ratio derived from COCO output. We used 51 logarithmically spaced target precisions Δf from 10^2 to 10^{-8}. For method m and raw evaluation budget B,

$$R_m(B) = \frac{1}{|\mathscr{F}|\,|\mathscr{I}|\,|\mathscr{T}|} \sum_{f\in\mathscr{F}} \sum_{i\in\mathscr{I}} \sum_{t\in\mathscr{T}} \mathbf{1}\{\tau_{m,f,i,t} \le B\}.$$

where $\tau_{m,f,i,t}$ is the first true-evaluation count at which method m reaches target t on function f and instance i. Unreached targets remain in the denominator, so higher values are better. We report budgets $B \in \{50, 100, 200, 500, 1000\}$, corresponding to $10, 20, 40, 100, 200 \times D$ true-evaluations.

In addition to optimization performance, we report decision-level diagnostics. On held-out counterfactual records, we measure top-1 agreement, where the selected surrogate–control pair has maximal utility, and acceptable agreement, where the selected surrogate–control pair belongs to the positive target set defined in Section 3.3. We also report the online selection distribution, i.e., the fraction of generations in which each surrogate–control pair is selected. For the two highlighted PFN-LQ quadratic comparisons in Section 5.1, we used 10,000 paired bootstrap resamples of function-instance target-hit ratios and report 95% percentile confidence intervals. Other comparisons, including the per-function heatmap, are descriptive. These diagnostics characterize the decision policy, while the fixed-budget target-hit ratio remains the main optimization measure.

Uniform-random results are reported as means over nine selector seeds, with 95% Student-t confidence intervals quantifying variability due to action selection. PFN, the fixed policies, and round robin used deterministic selectors and one matched CMA-ES seed, so these intervals should not be interpreted as complete algorithm-level uncertainty.

5. Results and Discussion

We compared the PFN selector with seven fixed policies, each using the same bundle throughout the run. In contrast, the PFN selector chose a bundle dynamically from the current optimization state, without function-specific configuration or predefined budget thresholds. This comparison evaluates whether learned state-dependent selection can compete with strong fixed policies without manually specifying when each surrogate-assisted bundle should be used.

PFN's relative performance depended strongly on the evaluation budget (Table 2). At 100 true evaluations, it ranked second with a target-hit ratio of 0.1401 versus 0.1447 for LQ quadratic top-30. At 200 evaluations, it again ranked second and nearly matched GP Matérn UNC (0.1787 versus 0.1843). The gap increased at 500 and 1000 evaluations, where PFN obtained 0.2402 and 0.3203, compared with

0.2875 and 0.4216 for GP Matérn UNC. Across all five budgets, GP Matérn UNC achieved the highest mean score of 0.2272, while PFN obtained 0.1949 and ranked fourth overall.

Table 2
Target-hit ratios over all 24 BBOB functions. Higher values are better. The final column is the unweighted mean.

Method	50	100	200	500	1,000	Mean
GP Matérn UNC	0.1034	0.1394	**0.1843**	**0.2875**	**0.4216**	**0.2272**
LQ quadratic top-30	0.0958	**0.1447**	0.1781	0.2494	0.3217	0.1980
GP Matérn DTS	**0.1040**	0.1387	0.1697	0.2464	0.3211	0.1960
PFN selector	0.0953	0.1401	0.1787	0.2402	0.3203	0.1949
Real only	0.0686	0.1164	0.1561	0.2425	0.3592	0.1886
RF lifelength	0.0857	0.1283	0.1630	0.2223	0.2589	0.1716
LQ linear top-50	0.0793	0.1303	0.1458	0.1835	0.2388	0.1555
SVM rank top-50	0.0715	0.1023	0.1273	0.1497	0.1862	0.1274

5.1. Performance by Function Group

Figure 2 presents the matched-instance results for the two conditioning-related function groups. On f_6–f_9, PFN obtained target-hit ratios of 0.0542 and 0.0971 at 50 and 100 evaluations, compared with 0.0575 and 0.1013 for the strongest fixed policy, LQ quadratic top-30. On f_{10}–f_{14}, PFN exceeded the complete fixed-surrogate range at 50 evaluations (0.0476 versus 0.0431 for the strongest fixed policy), but was slightly below LQ quadratic at 100 evaluations (0.0748 versus 0.0763). The 50-evaluation PFN–LQ difference was not conclusive under paired bootstrap resampling (95% CI $[-0.0013, 0.0133]$). Thus, the observed early-budget advantage was confined to one group and one budget rather than occurring consistently across both groups. Because PFN selected LQ quadratic in 65.6% of all online decisions, Figure 2 also shows this policy explicitly. The clearest evidence that dynamic selection was not equivalent to always using LQ quadratic occurs on f_{10}–f_{14} at 200 evaluations: PFN reached 0.1038, compared with 0.0925 for LQ quadratic, a difference of +0.0112. A paired bootstrap over the 75 function–instance units gave a 95% confidence interval of $[0.0024, 0.0204]$. Nevertheless, GP Matérn UNC remained the strongest fixed policy for this group at that budget with 0.1176. On individual functions, PFN strictly exceeded every fixed policy on four functions at 50 evaluations (f_7, f_{11}, f_{21}, f_{22}), three at 100 (f_7, f_{11}, f_{22}), three at 200 (f_2, f_{11}, f_{19}), and two at both 500 (f_{12}, f_{19}) and 1,000 evaluations (f_{12}, f_{16}).

Figure 3 places the function-specific results in context by reporting PFN's target-hit-ratio difference from both the strongest fixed policy and LQ quadratic top-30 at every budget. The gains over the strongest fixed policy are sparse and concentrated at particular function–budget pairs, whereas PFN exceeds LQ quadratic more often. This confirms that online switching can differ beneficially from always using its most frequent action, but does not support uniform superiority over the portfolio.

5.2. Comparison with State-Free Dynamic Baselines

Table 3 tests whether PFN's performance can be explained merely by switching among policies. PFN outperformed both state-free selectors at 50, 100, and 200 true evaluations. Its five-budget mean was 0.1949, compared with 0.1898 for uniform random and 0.1903 for round robin. Uniform random was slightly better at 500 evaluations, while the methods were comparable at 1,000 evaluations. On f_{10}–f_{14}, PFN also achieved a higher five-budget mean of 0.1444, versus 0.1356 for uniform random and 0.1346 for round robin. Thus, state-dependent selection provides measurable early-budget value beyond naive switching, although it remains below the fixed LQ quadratic policy overall.

5.3. Decision Diagnostics

As shown in Figure 4, across 86,506 online decisions, PFN selected LQ quadratic top-30 in 65.6% of generations, while no other bundle exceeded 8.8%. GP Matérn UNC was selected in only 8.7% of

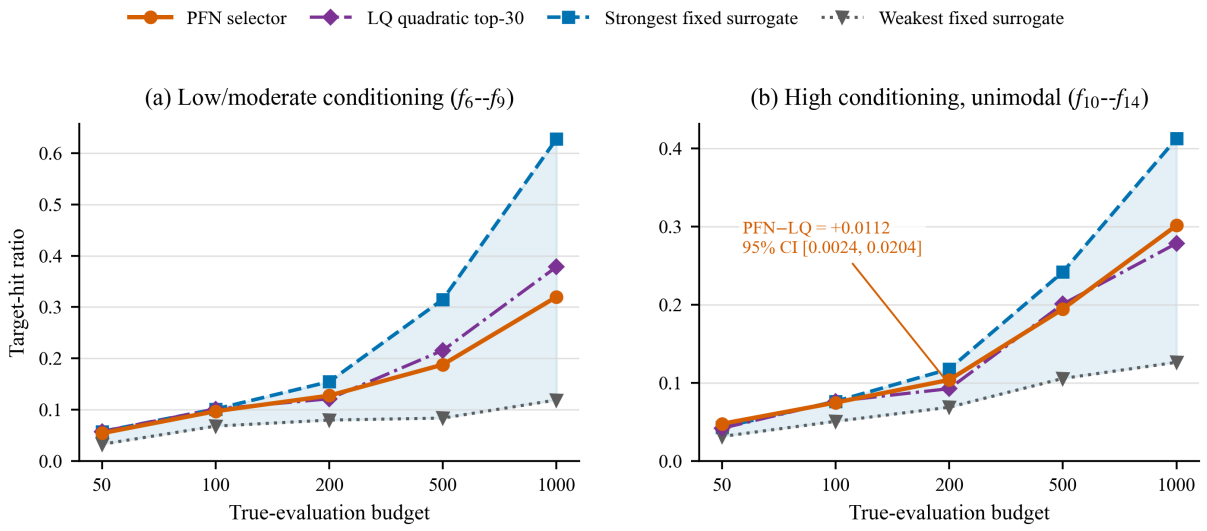


Figure 2: Group-average target-hit ratios on matched BBOB instances 1–15 for f_6–f_9 and f_{10}–f_{14}. The shaded region spans the six fixed surrogate-assisted policies from weakest to strongest at each budget; LQ quadratic top-30 is shown explicitly because it was the PFN selector's most frequent choice. The interval annotation is a paired bootstrap 95% confidence interval over function–instance units.

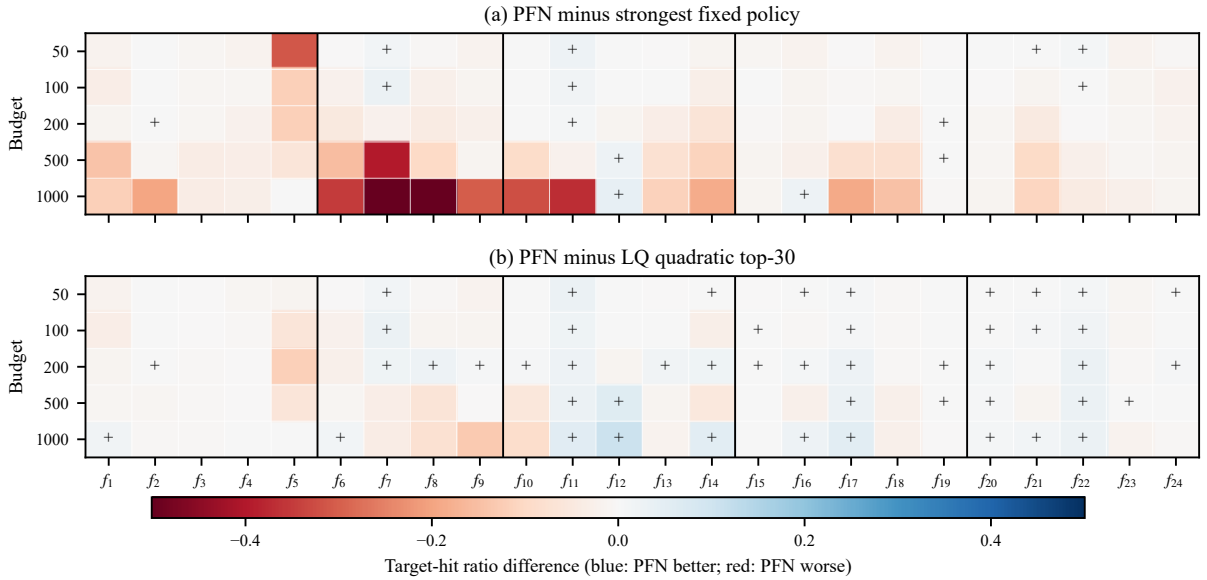


Figure 3: Per-function comparison on matched BBOB instances 1–15. Each cell shows the target-hit-ratio difference between PFN and (a) the strongest fixed policy or (b) LQ quadratic top-30 for one function and evaluation budget. Blue indicates an advantage for PFN, red an advantage for the comparator, and plus signs mark positive PFN differences. Vertical lines delimit the standard BBOB function groups.

generations despite achieving the best overall mean performance in Table 2. On 550,944 held-out logged states, exact oracle top-1 agreement was 0.154 and acceptable-label agreement was 0.551. LQ linear was oracle-best in 41.5% of states but predicted in only 2.8%, whereas LQ quadratic was oracle-best in 18.2% of states but predicted in 53.6%.

The low oracle frequency of GP Matérn UNC despite its strong fixed-policy performance reflects a difference between the two evaluation horizons. The offline oracle measures immediate one-generation improvement per true evaluation on recorded states, whereas the fixed-budget target-hit ratio accumulates the consequences of many decisions along an optimization trajectory. A policy can therefore

Table 3
Target-hit ratios for PFN, its dominant fixed policy, and state-free dynamic baselines. Uniform-random results are means over nine selector seeds; the second row gives 95% Student-t confidence intervals across those seeds. Higher values are better.

Method	50	100	200	500	1,000	Mean
PFN selector	0.0953	0.1401	0.1787	0.2402	0.3203	0.1949
LQ quadratic top-30	0.0958	0.1447	0.1781	0.2494	0.3217	0.1980
Uniform random	0.0831	0.1313	0.1703	0.2455	0.3189	0.1898
95% CI	[0.0788,0.0873]	[0.1298,0.1328]	[0.1687,0.1719]	[0.2415,0.2494]	[0.3134,0.3243]	[0.1877,0.1919]
Round robin	0.0821	0.1333	0.1725	0.2437	0.3200	0.1903

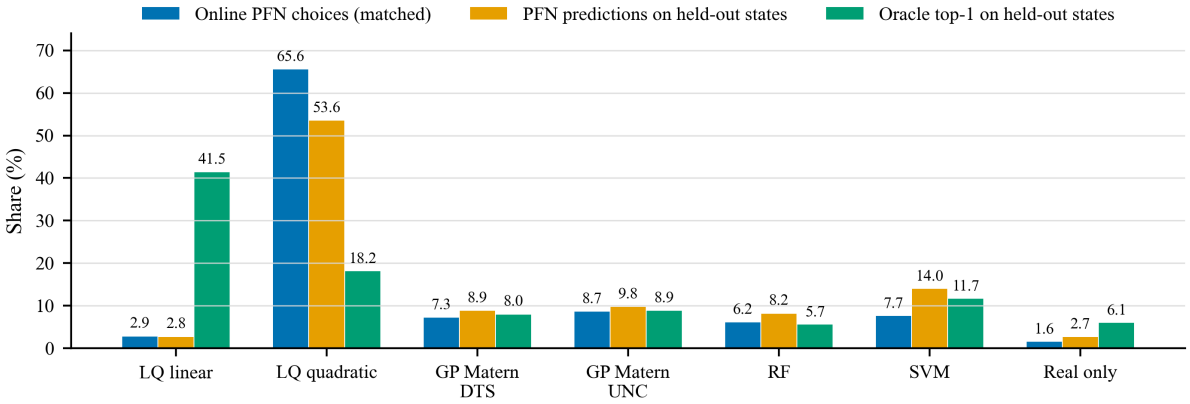


Figure 4: Online PFN choices, PFN predictions on held-out logged states, and oracle top-1 bundles on the same states.

have modest one-step oracle frequency yet provide robust long-run progress. This discrepancy, together with exact top-1 agreement of 0.154 and acceptable-label agreement of 0.551, indicates that both decision-model calibration and longer-horizon target construction deserve further investigation.

5.4. Discussion and Limitations

The results show that learned selection can approach the strongest fixed alternatives around 100–200 evaluations and can outperform its most frequently selected policy in specific regimes. The clearest case occurs on f_{10}–f_{14} at 200 evaluations, although the strongest fixed policy remains better. However, the strong concentration on LQ quadratic top-30 and the growing late-budget gap show that the learned policy is not yet robust over the full run. Class-balanced training, utility- or regret-aware targets, calibration, and explicit budget/progress features are natural next steps. The conclusions remain limited to noiseless five-dimensional BBOB, one CMA-ES seed, and 15 online instances per function.

Although LOFO excludes the evaluated function from training, the remaining training functions originate from the same BBOB suite and may share function-group structure. The experiment therefore evaluates within-benchmark transfer rather than out-of-distribution generalization, so performance on unseen benchmark families and real-world objectives remains a subject for future research.

6. Conclusion

We introduced a PFN-based framework for online selection among surrogate-assisted CMA-ES bundles. Trained using counterfactual one-step lookahead data, the selector requires neither function-specific bundle choices nor hand-designed switching rules. On 24 five-dimensional BBOB functions, the PFN selector was competitive with the strongest fixed policies at 100–200 evaluations.

In particular, on f_{10}–f_{14} at 200 evaluations, PFN achieved 0.1038 compared with 0.0925 for LQ quadratic. PFN also outperformed uniform-random and round-robin selection at 50–200 evaluations,

indicating that its early-budget performance cannot be attributed solely to switching among policies. However, GP Matérn UNC remained stronger, and the selector continued to choose LQ quadratic in 65.6% of generations.

Although the strongest fixed policies achieved better late-budget and overall performance, the diagnostics revealed a clear limitation: the PFN selector concentrated excessively on the LQ quadratic top-30 bundle. Class-balanced training, budget-aware features, and utility- or regret-aware objectives may therefore improve the robustness of the learned policy. Overall, the results demonstrate the promise of learned online bundle selection and provide a foundation for adaptive, evaluation-efficient black-box optimization.

Acknowledgments

The authors acknowledge the computational resources provided by MetaCentrum and the e-INFRA CZ infrastructure. This work was supported by the Grant Agency of Charles University, project No. 82126. Martin Holeňa's research was partially supported by the German Research Foundation (DFG) project funding no. 441926934.

Code and Data Availability.

The implementation and experiment scripts are publicly available at https://github.com/vojtechblaha/CMA-ES.

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT by OpenAI in order to: grammar and spelling correction, stylistic improvements of the manuscript, and generation of auxiliary LaTeX code for figures, tables, equations, and formatting. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] W. Kumagai, K. Yasuda, Black-Box Optimization and Its Applications, 2023, pp. 81–100. doi:10.1007/978-981-19-7776-3_6.

[2] K. Terayama, M. Sumita, R. Tamura, K. Tsuda, Black-box optimization for automated discovery, Accounts of Chemical Research (2021). doi:10.1021/acs.accounts.0c00713.

[3] D. Stenger, M. Ay, D. Abel, Robust parametrization of a model predictive controller for a CNC machining center using bayesian optimization, 2020. doi:10.1016/j.ifacol.2020.12.2778.

[4] N. Hansen, A. Ostermeier, Completely derandomized self-adaptation in evolution strategies., Evol. Comput 11 (2003) 159–195.

[5] N. Hansen, The CMA evolution strategy: A comparing review, volume 192, 2007, pp. 75–102. doi:10.1007/3-540-32494-1_4.

[6] N. Hansen, The CMA evolution strategy: a tutorial (2010). doi:10.48550/arXiv.1604.00772.

[7] N. Hansen, S. Finck, R. Ros, A. Auger, Real-parameter black-box optimization benchmarking 2009: Noisy functions definitions, 2009.

[8] Y. Jin, Surrogate-assisted evolutionary computation: Recent advances and future challenges, Swarm and Evolutionary Computation 1 (2011) 61–70. doi:10.1016/j.swevo.2011.05.001.

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

[10] N. Hansen, A global surrogate assisted CMA-ES, 2019, pp. 664–672. doi:10.1145/3321707.3321842.

[11] Z. Pitra, L. Bajer, J. Repický, M. Holeňa, Comparison of ordinal and metric gaussian process regression as surrogate models for CMA evolution strategy, in: Proceedings of the Genetic and Evolutionary Computation Conference Companion, GECCO '17, Association for Computing Machinery, New York, NY, USA, 2017, p. 1764–1771. URL: https://doi.org/10.1145/3067695.3084206. doi:10.1145/3067695.3084206.

[12] L. Bajer, Z. Pitra, M. Holeňa, Benchmarking gaussian processes and random forests surrogate models on the BBOB noiseless testbed, 2015. doi:10.1145/2739482.2768468.

[13] I. Loshchilov, M. Schoenauer, M. Sebag, Comparison-based optimizers need comparison-based surrogates, 2010. doi:10.1007/978-3-642-15844-5_37.

[14] I. Loshchilov, M. Schoenauer, M. Sebag, BI-population CMA-ES algorithms with surrogate models and line searches, 2013. doi:10.1145/2464576.2482696.

[15] S. Müller, N. Hollmann, S. Pineda, J. Grabocka, F. Hutter, Transformers can do bayesian inference, 2021. doi:10.48550/arXiv.2112.10510.

[16] N. Hansen, A. Auger, O. Mersmann, T. Tusar, D. Brockhoff, COCO: A platform for comparing continuous optimizers in a black-box setting, Optimization Methods and Software 36 (2016). doi:10.1080/10556788.2020.1808977.

[17] J. R. Rice, The algorithm selection problem 15 (1976) 65–118. URL: https://www.sciencedirect.com/science/article/pii/S0065245808605203. doi:https://doi.org/10.1016/S0065-2458(08)60520-3.

[18] E. Burke, M. Gendreau, M. Hyde, G. Kendall, et al., Hyper-heuristics: A survey of the state of the art, Journal of the Operational Research Society 64 (2013) 1695–1724. doi:10.1057/jors.2013.71.

[19] J. Maturana, Álvaro Fialho, F. Saubion, M. Schoenauer, et al., Adaptive operator selection and management in evolutionary algorithms, Autonomous Search (2012) 161. doi:10.1007/978-3-642-21434-9_7.

[20] P. Kerschke, H. H. Hoos, F. Neumann, H. Trautmann, Automated algorithm selection: Survey and perspectives (2018). URL: https://arxiv.org/abs/1811.11597. arXiv:1811.11597.

[21] S. Müller, M. Feurer, N. Hollmann, F. Hutter, PFNs4BO: In-context learning for bayesian optimization (2023).