ADAPTIVE FUNCTION VALUE WARPING FOR SURROGATE MODEL ASSISTED EVOLUTIONARY OPTIMIZATION

by

Amir Abbasnejad

Submitted in partial fulfillment of the requirements
for the degree of Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
July 2021

Table of Contents

List of Figures

Abstract

Invariance to strictly monotonic transformations of the objective function is an important feature in the design of black-box optimization algorithms. For this reason, ordinal surrogates have been established as an attractive class of models for integration into comparison-based optimizers. The recovery of this desirable property has not been explored for value-based surrogates. In this thesis, we adopt warping as a strategy to partially regain invariance lost by value-based models and propose a simple warped Gaussian process assisted covariance matrix adaptation evolution strategy. The algorithm is validated on families of parametrized, unimodal test problems and its performance compared with those of several related strategies. More intensive surrogate model exploitation is empirically demonstrated to benefit performance on ill-conditioned test problems. The simplicity and competitive performance of the proposed approach make it an appealing choice as a baseline for the evaluation of comparators on unimodal test problems.

Acknowledgements

I'm deeply grateful to my supervisor, Dr. Arnold, for his invaluable insight and guidance. Dr. Arnold has taught me the importance of asking simple questions when searching for answers. I'm also grateful to my parents, who nurtured my curiosity and brought me to Canada. Finally, I thank Kyly for her love and support.

Chapter 1

Introduction

1.1 Context

Surrogate models are mathematical abstractions of physical processes with many applications in the disciplines of science and engineering. Research and development in the field have enjoyed a surge in popularity that has paralleled the increase in computational power, as evidenced by a continuous rise in the number of publications reporting the use of the major surrogate modelling techniques [50]. In engineering, surrogate models have historically been referred to as *response surface methods*, whereby relationships between several *explanatory variables* are explored with respect to some *response variable* in a *designed experiment* [8]. The process typically involves the iterative refinement of some *response surface*, or model, via a sequence of designed experiments. The response surface is then used to explore the behaviour of some physical system or can be exploited to generate design configurations [8]. While the field started by primarily employing polynomial regression, other modelling techniques such as radial basis functions (RBF), support vector machines (SVM), and artificial neural networks are widely used [58]. In the field of geostatistics, surrogate models are referred to as spacial or spatio-temporal interpolation techniques and are employed in the study of geological and climatological features. Examples of such applications range from modelling ground water contamination [55] and mineral resource estimation [44], to approximating wind speeds and weather forecasting [76]. Perhaps the most popular modelling method in this discipline is *kriging*, which is essentially Gaussian process regression naturally focused on two- or three-dimensional input spaces [10]. The seminal work of Sacks et al. [67] popularized kriging beyond geostatistics by bringing it into the domain of surrogate modelling, which the authors termed the *design of computer experiments*. This precipitated the adoption of such models in a broad array of applications including aircraft design [11], nanomaterial property prediction [77], social systems modelling [13], supply chain network

optimization [14], financial modelling [12], etc. More recently, Gaussian processes have been explored as surrogates in discrete optimization [79] as well in quantum annealing [39].

Surrogate modelling techniques have been employed in the field of evolutionary algorithms to address black-box optimization problems where each objective function evaluation incurs significant cost. In practice, the definition of cost may range from the duration of a computer simulation to the financial expense of some physical experiment. Surrogate models can be constructed using information gained from evaluations of candidate solutions in past iterations. Low-order polynomials [46], ranking support vector machines [51], Gaussian process models [16, 5, 74], neural networks [43], and an ensemble of various models including radial basis functions [49] are among a few commonly employed model types. Surrogates can either provide cheap but inaccurate estimates to replace true objective function evaluations, or be used to determine potentially promising regions of the search space. In the former case, models can be broadly classified as either *value-based*, providing function value estimates, or *rank-based* (*ordinal*), approximating the relative quality of candidate solutions. Surrogate-assisted search strategies must tune their degree of reliance on their models, weighting the computational savings from avoiding costly function evaluations against the potentially poorer steps resulting from biased or inaccurate models [45]. Comprehensive reviews of surrogate model assisted evolutionary optimization have been prepared by Jin [42] and Loshchilov [50].

The incorporation of surrogate models in optimization algorithms naturally invokes a set of fundamental algorithm design decisions that may bear great impact on the resulting algorithm's performance. These can be broadly categorized into considerations of model definition and model exploitation. The construction of surrogate models must first consider the choice of a particular class of modelling technique, such as polynomial regression, Gaussian process regression, etc. [41]. Decisions regarding how to fit the model involve such questions as how to pick training points and how to select model parameters. For instance in the case of kernel-based models, an appropriate kernel must be selected. Questions around model training strategies, such as by means of heuristics derived from empirical observations or via more disciplined approaches such as cross-validation and maximum likelihood, must first define some

criteria by which to assess model quality. Characterizing surrogate model error is itself not a straight-forward task and has been the subject of research [42]. The definition of an error metric is often followed by the need to invoke a performant optimization strategy to reduce that error, again inviting intricate design considerations. Under the second general category of model exploitation, a key question involves how much to depend on the model. On one extreme, minimal trust can be afforded to a surrogate by formulating its role as a mere filter for seemingly poor candidate solutions suggested by the underlying strategy [46, 16], and on the other extreme the surrogate can be more intensely relied upon to direct sampling [9]. Moreover, the decision to employ rank-based instead of a value-based model may reflect a more conservative exploitation of the surrogate [27]. Each of the aforementioned considerations are but a few areas of inquiry in the field of surrogate-assisted optimization. The potential impact of these and many more design considerations on the performance of the resulting algorithms motivates their careful consideration and emphasizes the need for adoption of certain guiding principles.

In the domain of continuous black-box optimization, the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [35, 33] is considered to be competitive on several black-box optimization benchmarks [18, 29]. A contributing factor to the success of the CMA-ES is that, like all evolution strategies, it is a *comparison-based* optimizer. This implies that the algorithm relies solely on candidate solution ranking information and thus is unaffected by order-preserving objective function value transformations [35]. The competitive performance of the CMA-ES has identified it as a promising candidate for further enhancement via the integration of surrogate modelling approaches. Proposed algorithms [46, 54, 5, 25, 73] employ various modelling techniques and incorporate these models into the CMA-ES in different ways. Loshchilov [51, 53] argues that the integration of surrogates into optimization algorithms should ideally not disturb the desirable properties of the underlying search strategies. In particular, he takes inspiration from Runarsson et al. [66] to contend that comparison-based optimizers need rank-based surrogates in order retain their invariance with respect to order-preserving transformations of the objective function. The retention of this desirable invariance property has not yet been explored for value-based surrogates that assist comparison-based optimizers.

1.2 Motivation

Invariance is a desirable property in the analysis and design of optimization algorithms. The invariance of an algorithm with respect to some transformation of the problem domain confers robustness as it implies that any empirical or theoretical result observed on the problem instance generalizes to an entire associated invariance class induced by the transformation. Moreover, the retention of these properties as an algorithm design principle reduces the required number of parameters that need to be calibrated [51]. In the context of continuous optimization, two such invariance properties have been acknowledged as attractive, namely: invariance under transformations of the search space, and invariance under strictly increasing transformations of the objective function. The latter invariance type is characteristic of those algorithms which rely solely on comparisons between objective function values and has been theoretically demonstrated to be a source of robustness [19]. This, as well as robustness to affine transformations of the coordinate system, accounts for the notable success of the CMA-ES.

Integrating function value based surrogate modelling approaches into comparison-based optimizers removes the resulting optimizer's invariance with respect to order-preserving transformations of the objective function [51]. To address this limitation, Loshchilov et al. [51] argue that only the objective function ranks of candidate solutions are necessary from such models. They proceed to incorporate a rank-based support vector machine into the CMA-ES that additionally preserves the strategy's invariance with respect to search space transformations by exploiting the covariance matrix adapted by CMA-ES in the kernel of the SVM model. With the introduction of a mechanism for self-adaptive control of model hyper-parameters [53], as well as a procedure to tune the degree of reliance on its surrogate [54], Loshchilov et al. achieve state-of-the-art performance on a set of noiseless benchmark problems [23]. In further evolutions of the work, they combine their surrogate-assisted algorithm with the restart strategies introduced for CMA-ES and demonstrate its competitiveness on multi-modal test functions [52, 54]. The retention of these invariance properties is heavily credited by the authors for the success of their evolution strategy. Despite its remarkable success, the algorithm of Loshchilov et al. is complicated.

1.3 Contribution

In this thesis, we propose function value warping as a means of regaining some of the invariance lost when integrating value-based surrogates into comparison-based optimizers. Inspired by the work of Snelson et al. [69], we aim to adaptively obtain a representation of the objective function that is easier to model by the Gaussian process surrogate. To this end, an error measure based on a parametric warp is defined and a mechanism for its minimization presented. This warped Gaussian process is combined with CMA-ES based on the approach by Toal and Arnold [73]. The proposed algorithm is validated on a set of parametrized, unimodal test problems and shows promise for the partial recovery of invariance with respect to order-preserving transformations of the objective function. The simplicity and performance of the proposed algorithm make it an appealing choice as a basis of comparison for comparators on parametrized unimodal test problems.

1.4 Outline

The remainder of this work is organized as follows. In Chapter 2, we establish some background for the proposed algorithm and highlight related approaches. Chapter 3 describes the mechanism for deriving new transformations of objective function values and details its integration into the GP-CMA-ES [73]. Experimental validation of the algorithm and evaluation against comparators is presented in Chapter 4. Chapter 5 concludes the thesis by summarizing our contributions and providing directions for future inquiry.

Chapter 2

Preliminaries

This chapter provides the necessary background for the algorithm proposed in Chapter 3. First, the general problem setting is established via the definition of basic terminology in Section 2.1. Then, an introduction of the principles and algorithmic paradigm of evolutionary strategies is accompanied by an overview of the discipline's two canonical algorithms in Section 2.2. This is followed by a description of the class of surrogate model employed in this work in Section 2.3. Next, relevant aspects of a systematic approach to the empirical evaluation of stochastic optimization algorithms are discussed in Section 2.4. The chapter concludes with a brief summary of related works in Section 2.5.

2.1 Terminology

This section introduces basic terminology adopted in the thesis, including a specification of the optimization problem considered, as well as definitions of relevant invariance properties.

2.1.1 Black-Box Optimization

Optimization problems lie at the core of many disciplines. In general, given some objective function $f : \mathcal{X} \to \mathbb{R}$ to be optimized on some search space $\mathcal{X}$, the goal in optimization is to locate the setting $\mathbf{x}_{\text{opt}} \in \mathcal{X}$, called the *optimizer*, that corresponds to the optimal function value $f_{\text{opt}} = f(\mathbf{x}_{\text{opt}})$, known as *optimum*, by incurring the least cost. In a *black-box* scenario, no analytic definition of the objective function f is available; knowledge about f is restricted to the mere polling of a system whereby some input configuration $\mathbf{x} \in \mathcal{X}$ yields a related observable output $f(\mathbf{x})$. In particular, no specific assumptions, such as the availability of gradients, are made about the underlying optimization problem. The evaluation of $f(\mathbf{x})$ is often costly in practice, thus motivating the economy of objective function evaluations. This work considers,

without loss of generality, the minimization of the following real-valued black-box function:

$$f : \mathcal{X} \subseteq \mathbb{R}^n \to \mathbb{R}, \tag{2.1}$$

where there are no constraints imposed on the search domain $\mathcal{X} \subseteq \mathbb{R}^n$ or function value range, with cost quantified in terms of the number of objective function calls.

A location $\mathbf{x}' \in \mathbb{R}^n$ is a *local minimizer* of function f if there exists some constant $\epsilon \in \mathbb{R}_+$ such that for every point $\mathbf{x}$ that satisfies $\|\mathbf{x} - \mathbf{x}'\| < \epsilon$,

$$f(\mathbf{x}') \leq f(\mathbf{x}). \tag{2.2}$$

The definition for a *local maximizer* is given by inverting the comparison symbol of equation 2.2. A local optimum is the function value associated with a local minimizer or local maximizer. Functions that exhibit multiple local optimizers are said to be *multimodal*. A local optimum is also considered a *global optimum* iff the above definition holds for all $\epsilon \in \mathbb{R}_+$.

2.1.2 Invariance

Invariance properties generate equivalence classes between objective functions on which the cost of the optimization is identical [37]. Consequently, observations of algorithm performance on a single test function can be generalized to those on the entire problem class defined by the invariance property. For this reason, invariance is regarded as an important design principle of search algorithms.

The *state* s of a search algorithm $\mathcal{A}$ can be defined as a $v-$tuple that fully describes its configuration at a given iteration. The state encompasses the location in search space that the algorithm has reached, as well as all strategy internal parameters, such as step-size and covariance matrix. The *state space* S of a search algorithm is the set of all possible such $v-$tuples. With this definition of the state space S, we quote Hansen and Auger's [27] formal definition of invariance in search algorithms:

Definition 1 (Invariance [27])**.** Let $\mathcal{H}$ be a mapping from the set of all functions into its power set, $\mathcal{H} : \{\mathbb{R}^n \to \mathbb{R}\} \to \mathcal{P}(\{\mathbb{R}^n \to \mathbb{R}\}), f \to \mathcal{H}(f)$. Let S be the state space of the search algorithm, $s \in S$ and $\mathcal{A}_f : S \to S$ an iteration step of the algorithm under objective function f. The algorithm $\mathcal{A}$ is invariant under $\mathcal{H}$ (in other words:

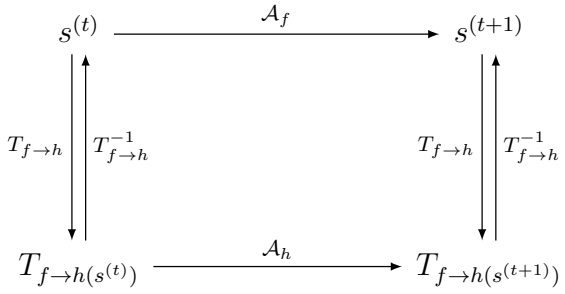


Figure 2.1: Commutative diagram for invariance (figure and caption modified from [27]). Horizontal arrows denote the transition between the state and function variables for one time step t of the algorithm $\mathcal{A}$. Vertical arrows signify the invertible transformation of state variables. The two possible paths between a state at time t and a state at time $t+1$ are equivalent in all four cases. When $f = h$, the diagram becomes trivial, with $T_{f\to h}$ denoting the identity transformation. One interpretation of the diagram is that any function h can be optimized like f given T^{-1}.

invariant under the exchange of f with elements of $\mathcal{H}(f)$) if for all $f \in \{\mathbb{R}^n \to \mathbb{R}\}$, there exists for all $h \in \mathcal{H}(f)$ a bijective state space transformation $T_{f\to h} : S \to S$ such that for all states $s \in S$

$$\mathcal{A}_h \circ T_{f\to h}(s) = T_{f\to h} \circ \mathcal{A}_f(s) \tag{2.3}$$

or equivalently

$$\mathcal{A}_h(s) = T_{f\to h} \circ \mathcal{A}_f \circ T_{f\to h}^{-1}(s). \tag{2.4}$$

If $T_{f\to h}$ is the identity for all $h \in \mathcal{H}(f)$, the algorithm is *unconditionally invariant* under $\mathcal{H}$. For randomized algorithms, the equalities hold in distribution. The set of functions $\mathcal{H}(f)$ is an invariance set of f for algorithm $\mathcal{A}$. □

The idea of invariance expressed by equation 2.3 is illustrated by the commutative diagram in Figure 2.1, depicting the equivalence of the two paths from the upper left to the lower right in the diagram. Equation 2.4 implies for all $t \in \mathbb{N}$ that

$$\mathcal{A}_h^{(t)}(s) = T_{f\to h} \circ \mathcal{A}_f^{(t)} \circ T_{f\to h}^{-1}(s), \tag{2.5}$$

where $\mathcal{A}^{(t)}$ denotes the t time steps of the algorithm starting from s [27]. Equation 2.5 implies that for all $h \in \mathcal{H}(f)$, the algorithm $\mathcal{A}$ optimizes the function h with initial state s just like the function f with initial state $T_{f\to h}^{-1}(s)$.

In the context of evolution strategies, two such invariance properties have been established as important: invariance to strictly increasing transformations of the

objective function, and invariance to affine transformations of the search space. These two invariance types are briefly discussed in the following subsections.

2.1.2.1 Invariance Under Function Value Transformations

A *strictly* increasing transformation is a one-to-one mapping $g : \mathbb{R} \to \mathbb{R}$ where

$$\forall x : \; g(x) < g(x') \iff x < x'. \tag{2.6}$$

Inverting the order symbol in equation 2.6 yields the corresponding concept of a *strictly decreasing* map. A function is *strictly monotonic* if it is either strictly increasing or strictly decreasing. A strictly increasing mapping g does not change the order of the input set and is thus referred to as *order-* or *rank-preserving*.

Invariance to order-preserving transformations of f is an example of unconditional invariance and is given by the following definition.

Definition 2 (Invariance to order-preserving transformations [27])**.** For all strictly increasing functions $g : \mathbb{R} \to \mathbb{R}$ and for all $f : \mathbb{R}^n \to \mathbb{R}$, algorithm $\mathcal{A}$ is invariant to order-preserving transformations of f if behaves identically on the objective function $\mathbf{x} \to f(\mathbf{x})$ and the objective function $\mathbf{x} \to g(f(\mathbf{x}))$. In other words, algorithm $\mathcal{A}$ is invariant under

$$\mathcal{H}_{\text{monotonic}} : f \to \{g \circ f \mid g \text{ is strictly increasing}\}. \tag{2.7}$$

□

Definition 2 implies that the cost of optimizing f is identical to that of optimizing $g \circ f$. Comparison-based optimizers that rely solely on candidate solution ranking information exhibit this invariance as g does not change this ranking information. For example, an optimizer with this invariance property would be indifferent to non-negative functions f, f^3, or $f^{1/2}$, faring equally well on convex functions as on non-convex ones. Figure 2.2 depicts three example functions on which comparison-based optimizers that rely solely on objective function value rankings would perform identically.

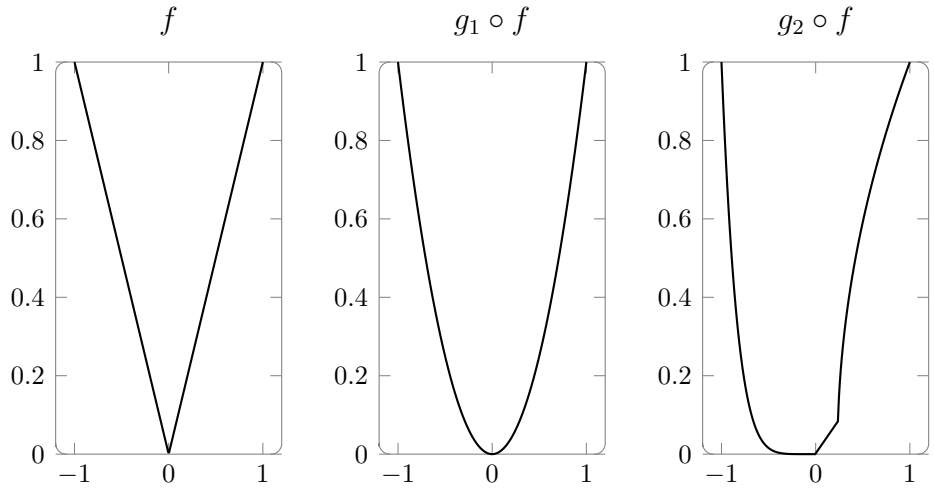


Figure 2.2: Example functions on which comparison-based optimizers that rely solely on candidate solution ranking information perform identically. As the strictly increasing function g preserves candidate solution ranks, such optimizers are unaffected by these transformations.

2.1.2.2 Invariance Under Search Space Transformations

An affine transformation is a one-to-one mapping $T : \mathbb{R}^n \to \mathbb{R}^n$ defined as

$$T(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{b}, \tag{2.8}$$

for a non-singular, matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ and a vector $\mathbf{b} \in \mathbb{R}^n$. For a symmetric matrix $\mathbf{A}$, the matrix multiplication encodes a search space rotation and an axis scaling while vector $\mathbf{b}$ represents a translation in search space.

Invariance to affine transformations of the search space can be defined as *translation invariance* to a shift vector $\mathbf{b}$ and a *general linear invariance* to a transformation matrix $\mathbf{A}$ as provided by the following two definitions.

Definition 3 (Translation invariance [27])**.** An algorithm $\mathcal{A}$ is translation invariant if it is invariant under

$$\mathcal{H}_{\text{translation}} : f \to \{h_b : \mathbf{x} \to f(\mathbf{x} - \mathbf{b}) \mid \mathbf{b} \in \mathbb{R}^n\}, \tag{2.9}$$

with the bijective state transformation, $T_{f \to h_b}$, that maps $\mathbf{x}$ to $\mathbf{x} + \mathbf{b}$. □

Definition 4 (General linear invariance [27])**.** Algorithm $\mathcal{A}$ is invariant under full rank linear transformations of the search space, meaning for each $f : \mathbb{R}^n \to \mathbb{R}$ invariant

under

$$\mathcal{H}_{\mathrm{GL}} : f \to \{f \circ \mathbf{A}^{-1} : \mathbf{x} \to f(\mathbf{A}^{-1}\mathbf{x}) \mid \mathbf{A} \text{ is a full rank } n \times n \text{ matrix}\}. \tag{2.10}$$

□

Invariance under affine search space transformations implies that the algorithm behaves similarly on the original coordinate system $\mathbf{x}$ and the coordinate system defined by $T(\mathbf{x})$, given a correspondingly transformed initial point. This category of invariance is considered crucial to the success of the CMA-ES in unconstrained black-box optimization [35, 33].

2.2 Evolution Strategies

Evolution strategies (ES) [63, 68] are general, nature-inspired search paradigms commonly applied to black-box optimization problems in continuous search spaces. They employ two fundamental evolutionary principles: variation to generate a collection of new candidate solutions from existing ones, and selection to iteratively direct them towards increasingly promising regions of the search space. In general, the objective of the search procedure is to locate a local optimum of the function, to within some arbitrary precision, using a small number of objective function evaluations.

The purpose of this section is to briefly introduce some important aspects of evolution strategies. First, basic notation commonly used to describe several aspects of ES is introduced. Next, a brief discussion of the generic algorithm prototype is presented in the nomenclature of biological evolution. This is followed by the simplest example of such prototype that embodies two major design philosophies of ES: unbiasedness, and invariance. The section concludes with an overview of the Covariance Matrix Adaptation Evolution Strategy (CMA-ES).

2.2.1 Algorithm Prototype

Evolutionary algorithms seek to lead collections of candidate solutions toward increasingly better regions of the search space. In the characteristic terminology of evolution, we refer to the objective function as the *fitness function*, to iterations as *generations*, to the collection of candidate solutions as a *population*, and to existing and newly

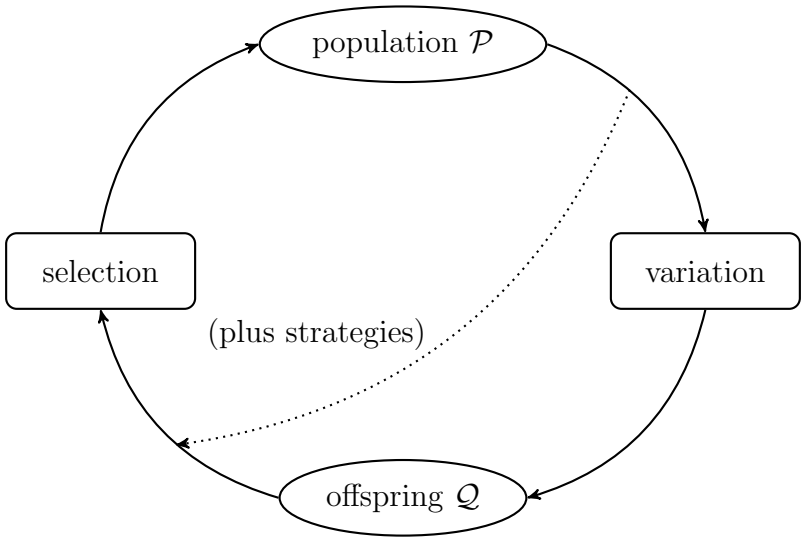


Figure 2.3: A graphical representation of the prototypical evolution loop (figure taken from [1]). A set of new candidate solutions $\mathcal{Q}$ is generated by applying variational operators to a parental population $\mathcal{P}$. Selection reduces the resulting population to its original size. Comma-strategies select candidate solutions from the pool of offspring $\mathcal{Q}$, while plus strategies select from the union $\mathcal{Q} \cup \mathcal{P}$ as indicated by the dashed line.

generated candidate solutions as *parents* and *offspring* respectively. A $(\mu/\rho^{+}, \lambda)$-ES employs a population $\mathcal{P}$ of μ candidate solutions. In each time-step t (denoted by superscript "$^{(t)}$" where necessary), a set $\mathcal{Q}^{(t)}$ of λ candidate solutions is created from $\mathcal{P}^{(t)}$ by means of variational operators of mutation and recombination. The symbol ρ indicates the number of parental candidate solutions involved in producing each offspring, with the "/" symbol signifying the operation of recombination. So-called environmental selection is used to maintain the population size of the next generation $\mathcal{P}^{(t+1)}$ at μ individuals. In the paradigm of fitness-based mating selection, an individual's fitness ranking as determined by its objective function value forms the basis of its selection. Depending on the selection type, $\mathcal{P}^{(t+1)}$ is chosen either from $\mathcal{P}^{(t)} \cup \mathcal{Q}^{(t)}$, or from $\mathcal{Q}^{(t)}$. The former is denoted as plus- and the latter as comma-selection. A graphical representation of this procedure is illustrated in Figure 2.3.

Variation introduces new information on the objective function into the search process and is achieved through mutation and recombination operators. In practice, this is instrumental for preventing stagnation of the evolutionary search. The mutation operator aims to add small, stochastic, and unbiased perturbations to an individual,

with the notion of small being determined by the so-called mutation strength σ. Recombination also introduces variation and may either depend on, or be independent of candidate solution fitness rankings. The former paradigm can be implemented by the uniformly random selection of $\rho \leq \mu$ parents for the computation of a centroid. Mutation would then consist of adding a normally distributed random vector to that centroid. For a population $\mathcal{P} = \{\mathbf{x}_1, \ldots, \mathbf{x}_\mu\}$, the set $\mathcal{Q}$ then consists of offspring

$$\langle \mathbf{x} \rangle_i = \frac{1}{\rho} \sum_{j=1}^{\rho} \mathbf{x}_{i_j} + \sigma \mathbf{z}_i \qquad \text{for } i = 1, \ldots, \lambda, \tag{2.11}$$

where the indices i_j are independently drawn, most often without replacement, and with equal probability from $\{1, \ldots, \mu\}$. Mutation vectors $\mathbf{z}_i$ consist of n independent components drawn from a zero-mean normal distribution with mutation strength (i.e. *step-size*) σ. In the case where $\rho = \mu$, recombination is deterministic as all parents are involved in the production of every offspring. This case is referred to as *global intermediate recombination* and is generalized by *weighted recombination*, where the contribution of each parent to the resultant centroid is scaled by rank-dependent weights.

Stationarity, or unbiasedness, under random selection is an important feature of randomized search algorithms and a design principle consistent with the imposition of least prior assumptions on the search procedure [35]. Stationarity of a parameter $\mathbf{v}$ under random selection implies that the expected value of the parameter remains unchanged across generations, that is $\mathbb{E}[\mathbf{v}^{(t+1)}|\mathbf{v}^{(t)}] = \mathbf{v}^{(t)}$. This aids generalizability of algorithm performance as prior assumptions that may be beneficial on one problem setting may be detrimental on another. Consequently, Hansen and Ostermeier [35] argue that only the current state and selection information should bias the behaviour of the search algorithm. From the viewpoint of information theory, the assumption of least additional information is known as the *maximum entropy principle*. For a given mean and covariance, the multivariate normal distribution has maximum entropy which, in addition to its mathematical tractability, makes it a natural choice as the search strategy's mutation operator. Informally, minimal variation bias via maximum entropy sampling implies maximum exploration of the immediate search space as the sampling distribution is least expected to generate a favourable outcome, and

most expected to reduce uncertainty about the search space given current information. Thus minimal variation bias, maximum entropy, and maximum exploration are mutually compatible notions that characterize the role of variation in evolutionary search.

While variation constitutes the exploratory aspect of evolutionary search, selection is its exploitative element and necessitates fitness evaluations of the candidate solution. Selection is a deterministic process that exploits information gained from objective function evaluations as well as that of the current state in order to bias the search procedure toward more promising regions of the search space. The $(\overset{+}{,})$ symbols denote two different selection mechanisms. In comma-selection, the lifespan of a candidate solution is constrained to one generation, with the μ best of λ offspring (for $\lambda \geq \mu$) in $\mathcal{Q}^{(t)}$ forming $\mathcal{P}^{(t+1)}$. Plus-selection picks the μ best of the $\mu + \lambda$ candidate solutions in $\mathcal{P}^{(t)} \cup \mathcal{Q}^{(t)}$ to form $\mathcal{P}^{(t+1)}$.

Principles of invariance and unbiasedness provide important guidelines in the application of variation and selection operators. Invariance confers the ability for the *progress rate*, defined as the expected distance change of the parental population centroid from the optimizer per time step [6], of ES observed on a single problem to generalize across a problem class induced by the invariance property. Aspects of these principles are exemplified by the $(1+1)$-ES with $1/5^{th}$ rule discussed in the following section.

2.2.2 (1+1)-ES

The $(1 + 1)$-ES is arguably the simplest evolution strategy. Algorithm 1 provides a simple implementation of the $(1 + 1)$-ES with $1/5^{th}$ success rule [63]. In each iteration, Line 2 generates a single offspring $\widetilde{\mathbf{x}}^{(t)}$ from the parent $\mathbf{x}^{(t)}$ via mutation by independently adding a zero-mean Gaussian perturbation to each component of the parent vector. The covariance of the Gaussian is set to the identity matrix, scaled by step-size σ. This leads to an isotropic sampling distribution which preserves the algorithm's invariance with respect to rotations of the coordinate system about its mean. Selection is performed in lines 4 to 8, where the new offspring $\widetilde{\mathbf{x}}^{(t)}$ is accepted only if its value is at least as good as that of the current parent. As objective function values are only compared against each other during selection, the algorithm retains

invariance to rank-preserving transformations of the fitness function.

Algorithm 1: The $(1+1)$-ES with $1/5^{th}$ Rule [26]

Given: $n \in \mathbb{N}_+, d \approx \sqrt{n+1}$

Initialize: $\mathbf{x} \in \mathbb{R}^n, \sigma > 0$

1: **while** *not terminated* **do**
2: $\quad \widetilde{\mathbf{x}}^{(t)} = \mathbf{x}^{(t)} + \sigma^{(t)} \times \mathcal{N}(\mathbf{0}, \mathbf{I})$;
3: $\quad \sigma^{(t+1)} \leftarrow \sigma^{(t)} \times \exp^{1/d}(\mathbb{1}_{f(\widetilde{\mathbf{x}}^{(t)}) \leq f(\mathbf{x}^{(t)})} - 1/5)$;
4: $\quad$ **if** $f(\widetilde{\mathbf{x}}^{(t)}) \leq f(\mathbf{x}^{(t)})$ **then**
5: $\quad\quad \mathbf{x}^{(t+1)} = \widetilde{\mathbf{x}}^{(t)}$;
6: $\quad$ **else**
7: $\quad\quad \mathbf{x}^{(t+1)} = \mathbf{x}^{(t)}$;
8: $\quad$ **end if**
9: **end while**
10: $t \leftarrow t + 1$

Parameter control is achieved by means of step-size adaptation. Rechenberg [63] was presumably the first to propose such a mechanism for the $(1+1)$-ES. He defined *success probability* p_s as the probability that an offspring is superior to its parent, and specified the notion *optimality* as achieving the largest expected approach of the optimum in a single iteration. For the objective functions he considered, he observed success probabilities of the $(1+1)$-ES with isotropic mutations and with optimally adjusted step-size to be approximately $1/5$. He also noted that increasing the step-size decreases the success probability, and vice versa. Thus Rechenberg recommended to approximate an average of success probabilities by monitoring multiple generations, and increasing the step-size if the observed estimate exceeds $1/5$, and decreasing it if the success probability is lower than $1/5$. This idea is implemented in Line 3 of Algorithm 1. If the offspring $\widetilde{\mathbf{x}}^{(t)}$ prevails, then the mutation strength increases by a factor of $e^{0.8/d}$, otherwise it decreases by a factor of $e^{-0.2/d}$. Parameter $d = \sqrt{1+n}$, suggested by Hansen et al. [26], dampens the change rate of the step-size. This implementation is consistent with Rechenberg's recommendation because if one out of every five offspring generated is successful, then the step-size updates neutralize each other on average and the logarithm of the step-size remains unchanged. If the success rate exceeds one fifth, then the step-size is enlarged more frequently, leading

to a systematic increase in step-size; the converse holds when the success rate is less than one fifth [46].

2.2.3 Covariance Matrix Adaptation Evolution Strategy (CMA-ES)

Perhaps the greatest drawback of the $(1 + 1)$-ES is its reliance on an isotropic mutation operator. Most objective functions in practice do not exhibit identical scales in different dimensions and are non-separable. Thus, correlated mutations as well as information about the underlying curvature of the objective function are often necessary to achieve satisfactory performance [35]. The CMA-ES addresses these and other issues by adapting to the local structure of the objective function [35, 33]. This is attained via adaptive control of its sampling distribution in the search space while preserving invariance with respect to order-preserving transformations of the objective function. The strategy independently adapts the scale and orientation of its mutation distribution, in part, by means of *cumulation*, which is a process whereby information about previously selected offspring is used to adjust current strategy parameters. Invariance to rank-preserving transformations of the objective is preserved via the strategy's reliance on offspring ranking information. The remainder of this section introduces the background for the CMA-ES and concludes with a condensed transcription of a single iteration of the algorithm.

2.2.3.1 Sampling

A population of λ new search points $\mathbf{x} \in \mathbb{R}^n$ is generated by sampling independently from a multivariate normal distribution

$$\mathbf{x}_i^{(t+1)} \sim \mathbf{m}^{(t)} + \sigma^{(t)} \times \mathcal{N}_i(\mathbf{0}, \mathbf{C}^{(t)}) \quad \text{for } i = 1, \ldots, \lambda, \tag{2.12}$$

where superscript $t \in \mathbb{N}$ indicates the time or generation index, $\mathbf{m}^{(t)} \in \mathbb{R}^n$ is the estimate of the optimizer at generation t, $\sigma \in \mathbb{R}_+$ denotes the step-size, and $\mathcal{N}(\mathbf{0}, \mathbf{C})$ is a multivariate normal distribution with zero mean and covariance matrix $\mathbf{C} \in \mathbb{R}^{n \times n}$. New candidate solutions $\mathbf{x}_i^{(t+1)}$ obey a multivariate normal distribution with expectation $\mathbf{m}^{(t)}$ and covariance $\sigma^2\mathbf{C}^{(t)}$. The covariance matrix determines the shape of the sampling distribution, where the level sets of the probability density function are hyper-ellipsoids. Equation 2.12 also implements the principle of unbiasedness as the

expected value of the normal distribution added to the parent is zero across generations. As mutation does not bias the mean of new offspring, the effect of sampling is decoupled from that of selection and improvements are made only subsequent to sampling.

Parameter λ determines the number of objective function evaluations per generation. A reasonable setting for this parameter cannot be entirely derived from first principles and is thus based on empirical investigations of the algorithm [27]. While a discussion of population size is beyond the scope of this thesis, two general bounding principles are outlined:

1. For large offspring populations $\lambda \gg n$, the likelihood of escaping local optima is increased and thus the probability of reaching the global optimum is greater. While the search strategy can thus be interpreted as being more global, this benefit is accompanied by the cost of increased function evaluations.

2. For small population sizes $\lambda \not\gg n$, the search process is relatively local and the algorithm can converge using comparatively fewer objective function calls. Because the mutation distribution is unbiased, new sampled solutions tend to be worse than previous best solutions. Therefore, the minimal value of $\lambda = 1$ is only viable if the best so-far evaluated candidate solution is always retained, (ie. under "+" selection) [32]. Comma-selection under the above sampling procedure must occur with $\lambda > 1$ and in practice $\lambda \geq 5$ is recommended [63].

A comparatively successful strategy is to pick a small default population size $\lambda \sim \mathcal{O}(log\ n)$ and conduct independent restarts of the algorithm with increasing population sizes [3] (see section 2.4.2.1).

2.2.3.2 Selection

The sampling process produces λ candidate solution pairs $\{\mathbf{x}_i, f(\mathbf{x}_i)\}_{i=1}^{i=\lambda}$ that provide information about the objective function. This information is exploited via selection to update the incumbent solution $\mathbf{m}^{(t)}$ and the distribution covariance matrix $\sigma^{(t)^2}\mathbf{C}^{(t)}$.

A heuristic principle is applied in selecting the new distribution mean: old information is disregarded. This principle embeds the assumption of locality, which is

intuitively sensible as with the convergence of the search procedure toward the optimizer, previously sampled points become increasingly farther away from the current region of interest [27]. Furthermore, disregarding old information also helps to avoid entrapment in local optima. In addition to these heuristic arguments, theoretical results have demonstrated that only slight improvements can be made by storing and using all previously sampled candidate solutions [72, 71].

Exploitation of the objective function is achieved, in part, via a form of truncation selection whereby candidate solutions with function values of poorer rank are discarded. Specifically, the algorithm considers the set of candidate function values $\{f_i\}_{i=1}^{i=\lambda}$ only with respect to its ordering signified via indices $i : \lambda$, for $i = 1, \ldots, \mu$, such that

$$f(\mathbf{x}_{1:\lambda}) \leq f(\mathbf{x}_{2:\lambda}) \leq \cdots \leq f(\mathbf{x}_{\lambda:\lambda}) \tag{2.13}$$

is satisfied. A weighted arithmetic mean of the μ best candidate solutions is then computed according to

$$\mathbf{m}^{(t+1)} = \frac{1}{\sum_{i=1}^{\mu} w_i} \sum_{i=1}^{\mu} w_i \mathbf{x}_{i:\lambda}^{(t+1)}, \tag{2.14}$$

where $\{w_i\}_{i=1}^{i=\mu} \in \mathbb{R}_+$ represent the set of recombination coefficients. Requiring a monotonically decreasing sequence of weights normalized such that

$$\sum_{i=1}^{\mu} w_i = 1, \quad w_1 \geq w_2 \geq \ldots \geq w_\mu > 0 \tag{2.15}$$

reduces equation 2.14 to

$$\mathbf{m}^{(t+1)} = \sum_{i=1}^{\mu} w_i \mathbf{x}_{i:\lambda}^{(t+1)}. \tag{2.16}$$

Equation 2.16 implements weighted intermediate recombination; the setting $w_i = \frac{1}{\mu}$ recovers intermediate recombination. The assignment of fitness-based weights also constitutes a means of selection since the computation of the resulting parental centroid $\mathbf{m}^{(t+1)}$ is more heavily influenced by the more promising regions of the search space [27].

Truncation and weighting occur based on candidate solution ranks. This implies that the exploitation of the objective function is rather conservative, potentially rendering the strategy less susceptible to deception [27]. Furthermore, the availability of

function values is not explicitly assumed, which is beneficial in settings such as the optimization of game-playing algorithms, where only ordinal outputs may be available. Importantly, the strategy's sole reliance on ranking information makes it invariant under strictly increasing transformations of the objective function.

A parental population size of $\mu \approx \lfloor \frac{\lambda}{2} \rfloor$ is considered reasonable. This is in part justified by the consideration that on a linear function, about half of the new candidate solutions are expected to be superior to the existing solution $\mathbf{m}^{(t)}$. Empirical observations of the CMA-ES on isotropic quadratic functions as well as on isotropic multimodal functions suggest up to a factor of half as being beneficial [27].

2.2.3.3 Covariance Matrix Adaptation (CMA)

The covariance matrix represents the variation parameters. Consistent with the paradigm of evolution, a sensible design principle is the reinforcement of successful variations in a manner which preserves unbiasedness. Hansen [27] defines successful variations as

$$\mathbf{x}_{i;\lambda} - \mathbf{m}^{(t)}, \quad \text{for } i = 1, \ldots, \mu. \tag{2.17}$$

Notably, successful variation does not imply $f(\mathbf{x}_{i;\lambda}^{(t+1)}) < f(\mathbf{m}^{(t)})$, which is neither necessary nor desirable in general as it would often result in an excessively small step-size [27]. Preservation of unbiasedness is consistent with building the least number of additional assumptions into the search procedure. This principle is also implemented in the update of the mean parameter, where successful candidate solutions are reinforced by means of weighting and truncation selection.

This section outlines how candidate solution ranking information is used to update the strategy's covariance matrix $\mathbf{C}$. The remainder of this subsection is divided into three parts. First, a mechanism used to reinforce successful intragenerational variations, known as the rank-μ update [33], is discussed. A second procedure referred to as the rank-one update exploits correlations between successful intergenerational variations. The final covariance matrix update combines both processes in a manner which preserves stationarity of the covariance matrix update.

Rank-μ Update

The μ-best variations of a single generation, normalized by step-size $\sigma^{(t)}$,

$$\frac{\mathbf{x}_{i;\lambda}^{(t+1)} - \mathbf{m}^{(t)}}{\sigma^{(t)}} \tag{2.18}$$

for $i = 1, \ldots, \mu$ are used in a weighted sum of outer products to update the existing covariance matrix. The update is computed according to:

$$\mathbf{C}^{(t+1)} = (1 - c_\mu)\mathbf{C}^{(t)} + c_\mu \sum_{i=1}^{\mu} w_i \underbrace{\left(\frac{\mathbf{x}_{i;\lambda}^{(t+1)} - \mathbf{m}^{(t)}}{\sigma^{(t)}}\right)\left(\frac{\mathbf{x}_{i;\lambda}^{(t+1)} - \mathbf{m}^{(t)}}{\sigma^{(t)}}\right)^{\mathrm{T}}}_{\text{outer product}}, \tag{2.19}$$

where constant $0 \leq c_\mu \leq 1$ weights the influence of prior generations on the covariance matrix update. Covariance matrices from multiple generations are comparable as the outer product is normalized by ${\sigma^{(t)}}^2$ in equation 2.19. The recursive construction of a weighted sum, parametrized by learning rate c_μ constitutes an exponential smoothing of past covariances. For learning rate $c_\mu = 1$, no prior information is retained, while for $c_\mu = 0$ no learning takes place. The choice of c_μ is crucial as too small values lead to slow learning, while too large values lead to degeneration of the covariance estimate [24]. Although the best choice of c_μ remains an open question, empirical tuning of the parameter on the quadratic sphere ($f_{\text{sphere}} : \mathbf{x} \to \sum_{i=1}^{i=n} x_i^2$; see subsection 2.4.2.2) has proven effective on non-noisy objective functions and is inferred to be largely independent of the function to be optimized [24]. Observing that the sum of outer products in 2.19 contains $\min(\mu, n)$ linearly independent vectors, Hansen et al. [33] termed this covariance matrix update the *rank-μ update*. Defining

$$\mathbf{A}^{(t)} \coloneqq {\mathbf{C}^{(t)}}^{1/2} \tag{2.20}$$

as the principal square root of $\mathbf{C}^{(t)}$ and

$$\mathbf{z}_{i;\lambda}^{(t)} \coloneqq {\mathbf{C}^{(t)}}^{-1/2} \frac{\mathbf{x}_{i;\lambda}^{(t+1)} - \mathbf{m}^{(t)}}{\sigma^{(t)}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{2.21}$$

equation 2.19 can be reformulated:

$$\mathbf{C}^{(t+1)} = (1 - c_\mu)\mathbf{C}^{(t)} + c_\mu \mathbf{A}^{(t)} \left(\sum_{i=1}^{\mu} w_i \, \mathbf{z}_{i;\lambda}^{(t+1)} {\mathbf{z}_{i;\lambda}^{(t+1)}}^{\mathrm{T}} \right) {\mathbf{A}^{(t)}}^{\mathrm{T}} \tag{2.22}$$

to express the update in the natural coordinate system defined by linear transformation matrix $\mathbf{A}$ applied to isotropically distributed mutation vectors $\mathbf{z}$. This can

be seen by considering the eigendecomposition of the covariance matrix $\mathbf{C}$ into a set of orthonormal eigenvectors $\mathbf{B}$ and a diagonal matrix of eigenvalues $\mathbf{D}^2$ such that $\mathbf{C} = \mathbf{B}\mathbf{D}^2\mathbf{B}^{\mathrm{T}}$. Thus $\mathbf{C}$'s principal square root $\mathbf{A} = \mathbf{B}\mathbf{D}\mathbf{B}^{\mathrm{T}}$ since $\mathbf{A}\mathbf{A} = \mathbf{B}\mathbf{D}\mathbf{B}^{\mathrm{T}}(\mathbf{B}\mathbf{D}\mathbf{B}^{\mathrm{T}}) = \mathbf{B}\mathbf{D}^2\mathbf{B}^{\mathrm{T}} = \mathbf{C}$, encodes an axis rotation by $\mathbf{B}$ and an axis scaling via $\mathbf{D}$.

Jastrebski and Arnold [40] introduced *active* covariance matrix adaptation, which generalizes the update rule in 2.22 from μ to λ candidate solutions by employing negative weights for the remaining trial steps $\mathbf{x}_{\mu+1;\lambda}, \ldots, \mathbf{x}_{\lambda:\lambda}$ in order to decrease variances in less promising regions of the search space. The update in 2.22 can thus be written

$$\mathbf{C}^{(t+1)} = (1 - c_\mu \textstyle\sum_{i=1}^{\lambda} w_i)\mathbf{C}^{(t)} + c_\mu \mathbf{A}^{(t)} \left(\textstyle\sum_{i=1}^{\lambda} w_i \, \mathbf{z}_{i:\lambda}^{(t+1)} \mathbf{z}_{i:\lambda}^{(t+1)^{\mathrm{T}}} \right) \mathbf{A}^{(t)^{\mathrm{T}}}, \tag{2.23}$$

where $\{w_i\}_{i=1}^{i=\lambda} \in \mathbb{R}$ form a monotonically decreasing sequence such that $w_1 \geq \ldots \geq w_\mu \geq 0 \geq w_{\mu+1} \geq \ldots \geq w_\lambda$, with $\sum_{i=1}^{\mu} w_i = 1$ and $\sum_{i=1}^{\lambda} w_i \approx 0$. Negative recombination weights can compromise the positive definiteness of the resulting covariance matrix and thus mechanisms outlined in [24] have been introduced [40, 36] to address this issue. The update has been demonstrated to be beneficial on a noiseless testbed of problems [36] and implemented in a variant of the $(1 + 1)$-CMA-ES [2].

Closer inspection of 2.19 reveals that using selected steps $\mathbf{y}_{i:\lambda}^{(t+1)} = (\mathbf{x}_{i:\lambda}^{(t+1)} - \mathbf{m}^{(t)})/\sigma^{(t)}$ in an outer product renders the sign of these steps irrelevant as $\mathbf{y}_{i:\lambda}^{(t+1)}\mathbf{y}_{i:\lambda}^{(t+1)^{\mathrm{T}}} = (-\mathbf{y}_{i:\lambda}^{(t+1)})(-\mathbf{y}_{i:\lambda}^{(t+1)})^{\mathrm{T}}$. The remainder of this subsection explains the mechanism used to reintroduce the sign information into the covariance matrix update $\mathbf{C}^{(t+1)}$.

Rank-one Update

In order to recover the sign information, the technique of cumulation can be employed. An elegant means of achieving this end is to sum successive steps taken by the strategy in previous generations into a so-called *evolution-* or *search-path* [20, 34]. To define an unbiased estimator of the sum of successive steps, the evolution path can be interpreted as the realization of a random vector $\mathbf{s}_c^{(t)}$ from a maximum entropy distribution in accordance with the stationarity condition

$$\mathbf{s}_c^{(t+1)} \sim \mathbf{s}_c^{(t)}. \tag{2.24}$$

The direction and length of such vector can be encoded by the orientation and scale of the covariance matrix of a zero-mean multivariate normal distribution.

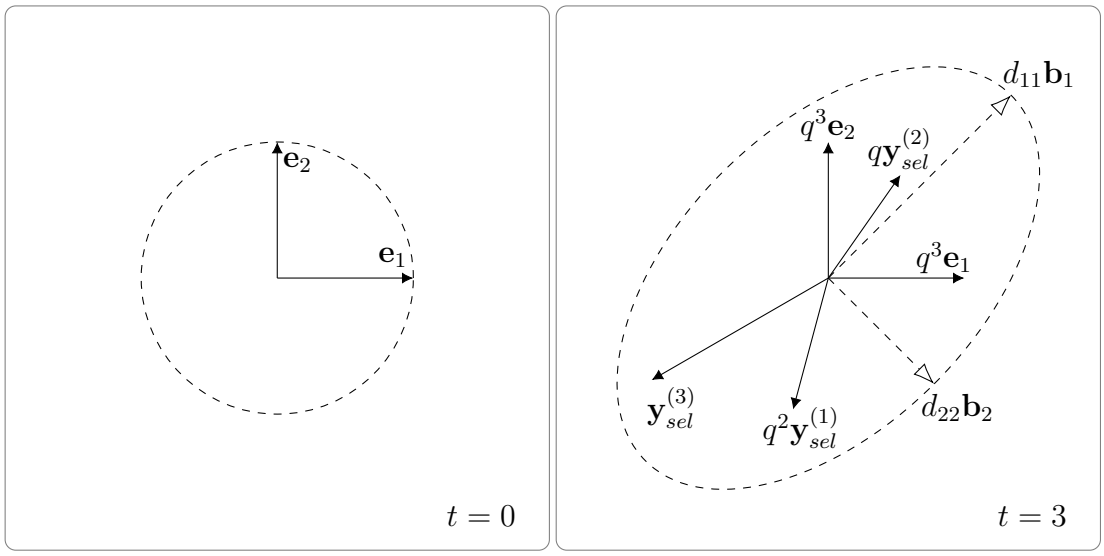


Figure 2.4: Construction of the search distribution using a weighted sum of selected steps (figure and caption modified from [35]). The initial search distribution at generation $t = 0$ is isotropic and described by orthogonal search vectors $\mathbf{e}_1$ and $\mathbf{e}_2$. The selected steps $y^{(t)}_{sel}$ are consecutively added in subsequent generations, with old vectors multiplied by $q = 0.9$ in each generation. The schematic shows how $\mathcal{N}(\mathbf{0}, \mathbf{C}^{(1)})$ tends to reproduce $\mathbf{y}^{(1)}_{sel}$ with a larger probability than the initial distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$; distribution $\mathcal{N}(\mathbf{0}, \mathbf{C}^{(2)})$ tends to reproduce $\mathbf{y}^{(2)}_{sel}$ with a larger probability than $\mathcal{N}(\mathbf{0}, \mathbf{C}^{(1)})$, and so on.

In practice, the evolution path $\mathbf{s}_c \in \mathbb{R}^n$ is constructed as an exponentially fading record of previous steps parametrized by constant $0 \leq c_c \leq 1$, with initial setting $\mathbf{s}^{(0)}_c = \mathbf{0}$. Normalizing by standard deviations $\sigma^{(t)}$ of steps, the equation

$$\mathbf{s}^{(t+1)}_c = (1 - c_c)\mathbf{s}^{(t)}_c + \sqrt{c_c(2 - c_c)\mu_{\text{eff}}}\frac{\mathbf{m}^{(t+1)} - \mathbf{m}^{(t)}}{\sigma^{(t)}}, \tag{2.25}$$

describes the update of the evolution path, where $\mu_{\text{eff}} = 1/\sum_{i=1}^{\mu} w_i^2$ and the remaining terms ensure that condition 2.24 is satisfied under random selection.

The evolution path of 2.25 is utilized to construct the covariance matrix update as in 2.19, yielding [20]:

$$\mathbf{C}^{(t+1)} = (1 - c_1)\mathbf{C}^{(t)} + c_1\mathbf{s}^{(t)}_c\mathbf{s}^{(t)^{\mathrm{T}}}_c. \tag{2.26}$$

A few observations can be made about 2.26. When $c_c = 1$ and $\mu = 1$, equations 2.25 and 2.19 are identical. Furthermore, as $\mathbf{s}^{(t)}_c\mathbf{s}^{(t)^{\mathrm{T}}}_c$ has a rank of one, the covariance matrix update in 2.26 is termed the *rank-one update*. A learning rate of $c_1 \approx 2/n^2$ is empirically validated [24]. When weights w_i are selected such that μ_{eff} is small, using

the evolution path for the update of $\mathbf{C}$ has been found to significantly improve 2.19 as it results in the exploitation of correlations between successive steps [22]. Figure 2.4 illustrates a simple 2-dimensional scenario for $t = 3$ generations where the distribution in each generation tends to reproduce the selected step $\mathbf{y}_{sel}^{(t)}$ with a larger probability than the previous generation's distribution. The combination of the rank-one and rank-μ covariance matrix updates is discussed in the remainder of the subsection.

The covariance Matrix Update

The rank-one update in 2.26 and rank-μ update of 2.19 are combined to produce the final covariance matrix update

$$\mathbf{C}^{(t+1)} = (1 - \underbrace{c_1 - c_\mu \textstyle\sum_{i=1}^{\lambda} w_i}_{\approx 0})\mathbf{C}^{(t)} + c_1 \underbrace{\mathbf{s}_c^{(t+1)}\mathbf{s}_c^{(t+1)^{\mathrm{T}}}}_{\text{rank-one update}} + c_\mu \underbrace{\sum_{i=1}^{\lambda} w_i \, \mathbf{y}_{i:\lambda}^{(t+1)}(\mathbf{y}_{i:\lambda}^{(t+1)})^{\mathrm{T}}}_{\text{rank-}\mu\text{ update}} \quad (2.27)$$

where $\mathbf{y}_{i:\lambda}^{(t+1)} = (\mathbf{x}_{i:\lambda}^{(t+1)} - \mathbf{m}^{(t)})/\sigma^{(t)}$. According to the recommendation by Hansen [24], the sum of weights is selected such that the sum of positive weights is equal to one and the sum of all weights $\sum_{i=1}^{\lambda} w_i \approx -c_1/c_\mu$, bringing the multiplicative factor on $\mathbf{C}^{(t)}$ close to one. This implies no decay on $\mathbf{C}^{(t)}$, to which the positive update adds, in expectation, an amount of variance that is removed by the negative weights [24].

Equation 2.27 combines the advantages of 2.19 and 2.26, reducing to 2.19 for $c_1 = 0$ and to 2.26 for $c_\mu = 0$. The former is important in large populations, where it is desirable to efficiently use information from the entire population while the latter is particularly consequential in small populations, where correlations between generations are exploited by means of an evolution path [35, 24]. Computing the expected value of 2.27 would confirm that the update equation obeys the stationarity condition $\mathbb{E}(\mathbf{C}^{(t+1)}|\mathbf{C}^{(t)}) = \mathbf{C}^{(t)}$ under random selection [27].

2.2.3.4 Cumulative Step-Size Adaptation (CSA)

The covariance matrix adaptation procedure described in the previous subsection fails to explicitly account for the overall scale, or step-size, of the search distribution. Hansen and Ostermeier [35] note the importance of adding such a mechanism as being motivated by at least two requirements. First, achieving satisfactory progress rates on the simplest of objective function classes, such as the quadratic sphere, would require that the overall variance is adapted on a timescale proportional to n. This is

contrasted by the n^2 timescale necessary for adapting the overall shape of the search distribution due to its $\mathcal{O}(n^2)$ parameters [35]. Secondly, a poor initial setting of the search distribution's overall scale, such as at too small a step-size, would result in the problem "appearing" linear and adaptation erroneously increasing the variance in one direction. Under such a regime, the overall variance must be adapted more quickly than the distribution shape to prevent this failure [35].

In the field of evolution strategies, well-known heuristics for step-size adaptation include:

— *self-adaptation* [68], whereby the strategy's step-size σ is determined stochastically via sampling candidate solutions with varying σ and retaining the step-size of the selected candidate solutions;

— the $1/5^{th}$ rule [63] (described in section 2.2.2), whereby step-size is adjusted such that the success rate across generations is approximately equal to 1/5;

— *cumulative step-size adaptation* (CSA) [60], which controls the length of the evolution path, taken over a number of generations.

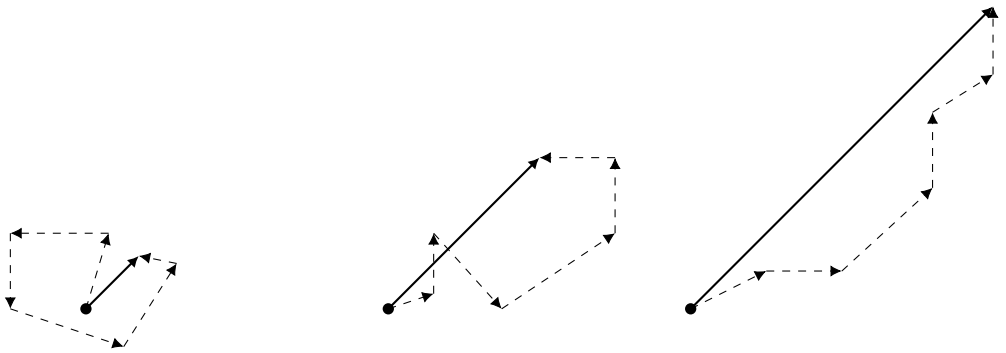
The CMA-ES employs cumulative step-size adaptation to adjust its step-size. As its design principle, CSA aims to achieve perpendicularity of successive steps. Perpendicularity as an optimality criterion for step-size control can be intuitively understood by considering the following three cases:

1. If successive steps are negatively correlated, then they tend to be anti-parallel. Thus successive steps cancel each other out and a similar trajectory could be covered by fewer, shorter steps. Hence the step-size is assumed to be "too long".

2. If successive steps are positively correlated, then they tend to be parallel. Thus multiple consecutive steps can ideally be replaced by fewer, but longer steps. In this scenario it is reasonable to assume that the step length is "too short".

3. If successive steps are uncorrelated, then they tend to be roughly perpendicular. It is deduced that step length is neither too short nor too long.

Figure 2.5 illustrates these three cases.



(a) decrease σ (b) keep σ (c) increase σ

Figure 2.5: Intuition behind step-size adaptation (figure and caption modified from [27]). The diagram shows three prototypical evolution paths in the search space, each using six consecutive steps. While the lengths of the single (dashed) steps are similar, the length of the evolution paths differ greatly. The situation in (a) depicts single steps that are anti-parallel and thus cancel each other out, resulting in a short evolution path. Part (b) illustrates steps that are on average perpendicular and a resulting evolution path that is neither too short nor too long. Part (c) displays correlated steps, and the resulting long evolution path. The length of the evolution path is a good proxy for the optimality of the step-size.

Successive steps are defined in terms of an evolution path

$$\mathbf{m}^{(t)} - \mathbf{m}^{(t-j)} \tag{2.28}$$

where, again, the search algorithm's solution estimate at generation t is represented by $\mathbf{m}^{(t)}$ and the solution corresponding to j previous generations denoted by $\mathbf{m}^{(t-j)}$. Parameter j can be interpreted as the length of the evolution path, which is constructed in a manner similar to equation 2.25. Applying exponential smoothing parametrized by constant $0 \leq c_\sigma \leq 1$, the evolution path is given by

$$\mathbf{s}_\sigma^{(t+1)} = (1 - c_\sigma)\mathbf{s}_\sigma^{(t)} + \sqrt{c_\sigma(2 - c_\sigma)\mu_{\text{eff}}}\mathbf{C}^{(t)^{-\frac{1}{2}}} \frac{\mathbf{m}^{(t+1)} - \mathbf{m}^{(t)}}{\sigma^{(t)}}, \tag{2.29}$$

where $\mathbf{C}^{(t)^{-\frac{1}{2}}}$, denoting the principal square root of $\mathbf{C}^{(t)^{-1}}$, is the matrix which transforms the selected step into the coordinate system wherein the sampling distribution is isotropic. The remaining coefficients are added to maintain unit variance of the path update. Consequently, starting with the initial evolution path $\mathbf{s}_\sigma^{(0)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and under random selection of $\mathbf{x}_{i:\lambda}$ in all succeeding generations, for any sequence of realized covariance matrices $\mathbf{C}^{(0)}, \mathbf{C}^{(1)}, \ldots, \mathbf{C}^{(t)}$ the following stationarity condition

$$\mathbf{s}_\sigma^{(t)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \quad \text{for } t = 1, 2, 3, \ldots, \tag{2.30}$$

is satisfied [21].

Perpendicularity is sought by comparing the length of the evolution path $\|\mathbf{s}_\sigma^{(t+1)}\|$ with its expected length under random selection $\mathbb{E}\|\mathcal{N}(\mathbf{0},\mathbf{I})\|$ in order to detect whether the evolution path is long or short. In the ideal situation, selection does not bias the length of the evolution path and the length of the path equals its expected length under random selection. This would result in consecutive steps that are independent and therefore uncorrelated. If selection biases the evolution path to be longer than expected, then σ should increase, and vice versa. The following construction achieves the aforementioned requirements:

$$\sigma^{(t+1)} = \sigma^{(t)} \exp\left(\frac{\|\mathbf{s}^{(t+1)}\|}{\mathbb{E}\|\mathcal{N}(\mathbf{0},\mathbf{I})\|} - 1\right). \tag{2.31}$$

When the path length equals its length under random selection, $\frac{\|\mathbf{s}_\sigma^{(t+1)}\|}{\mathbb{E}\|\mathcal{N}(\mathbf{0},\mathbf{I})\|} = 1$, the argument of the exponential function evaluates to zero, retaining the current step-size $\sigma^{(t)}$. When the ratio of expected lengths is greater than one, the argument to the exponential function becomes positive and the step-size $\sigma^{(t)}$ is increased. The converse holds when the ratio of expected lengths is less than one. This control mechanism attempts to achieve unbiasedness under random selection on the logarithmic scale as $\mathbb{E}[\log_e \sigma^{(t+1)} | \sigma^{(t)}] = \log_e \sigma^{(t)}$ and can be further interpreted as striking a balance between exploration and exploitation. When the step-size is too short, success probability is increased and thus exploitation is thought to be excessively favoured, motivating its counteraction with an increase in step-size. The converse is presumed to hold when the step-size is too long.

2.2.3.5 Algorithm Overview

Algorithm 2 highlights the main steps of the $(\mu/\mu_w, \lambda)$-CMA-ES with weighted recombination in a single generation, where the adoption of the assignment operator $\leftarrow$ eliminates the need for superscript t. The state of the algorithm at any time step is described by the parameter set $\{\mathbf{m}, \sigma, \mathbf{C}, \mathbf{s}_\sigma, \mathbf{s}_c\}$. The inputs to the algorithm consist of a problem-dependent initial point $\mathbf{m} \in \mathbb{R}^n$ and step-size $\sigma \in \mathbb{R}_+$ as well as an initial population size λ with default setting $4 + \lfloor 3\log_e n \rfloor$. Evolution paths $\mathbf{s}_\sigma$ and $\mathbf{s}_c$ are initialized to $\mathbf{0}$ and the covariance matrix defaults to $\mathbf{C} = \mathbf{I}$ at the start.

Algorithm 2: $(\mu/\mu_w, \lambda)$-CMA-ES [24]

Input: $\mathbf{m} \in \mathbb{R}^n,\ \sigma \in \mathbb{R}_+,\ \lambda$

Initialize: $\mathbf{C} = \mathbf{I},\ \mathbf{s}_c = \mathbf{0},\ \mathbf{s}_\sigma = \mathbf{0}$

Set: $c_c \approx 4/n,\ c_\sigma \approx 4/n,\ c_1 \approx 2/n^2,\ c_\mu \approx \mu_{\text{eff}}/n^2,\ c_1 + c_\mu \leq 1,\ d_\sigma \approx 1 + \sqrt{\frac{\mu_{\text{eff}}}{n}}$

1: **while** *not terminated* **do**

2: Compute the principal square root of the covariance matrix

$$\mathbf{A} \leftarrow \mathbf{C}^{\frac{1}{2}} \tag{2.32}$$

3: Sample a new population of search points, for $i = 1, \ldots, \lambda$

$$\mathbf{z}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{2.33}$$

$$\mathbf{x}_i = \mathbf{m} + \sigma \mathbf{A} \mathbf{z}_i \quad \sim \mathcal{N}(\mathbf{m}, \sigma^2 \mathbf{C}) \tag{2.34}$$

4: Select and recombine offspring

$$\langle \mathbf{z} \rangle = \sum_{i=1}^{\mu} w_i \mathbf{z}_{i;\lambda},\ \text{where}\ \sum_{i=1}^{\mu} w_i = 1, w_i > 0 \text{ for } i = 1, \ldots, \mu \tag{2.35}$$

$$\mathbf{m} \leftarrow \mathbf{m} + \sigma \mathbf{A} \langle \mathbf{z} \rangle \text{ equals } \sum_{i=1}^{\mu} w_i \mathbf{x}_{i;\lambda} \tag{2.36}$$

5: Adapt step-size

$$\mathbf{s}_\sigma \leftarrow (1 - c_\sigma)\mathbf{s}_\sigma + \sqrt{c_\sigma(2 - c_\sigma)\mu_{\text{eff}}}\ \langle \mathbf{z} \rangle \tag{2.37}$$

$$\sigma \leftarrow \sigma \times \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|\mathbf{s}_\sigma\|}{\mathbb{E}\|N(\mathbf{0}, \mathbf{I})\|} - 1\right)\right) \tag{2.38}$$

6: Adapt covariance matrix

$$\mathbf{s}_c \leftarrow (1 - c_c)\mathbf{s}_c + \sqrt{c_c(2 - c_c)\mu_{\text{eff}}}\ \mathbf{A}\langle \mathbf{z} \rangle \tag{2.39}$$

$$w_i^\circ = w_i \times \left(1 \text{ if } w_i \geq 0 \text{ else } \frac{n}{\|z_{i;\lambda}\|^2}\right) \tag{2.40}$$

$$\mathbf{C} \leftarrow \mathbf{C} + c_1 \mathbf{s}_c \mathbf{s}_c^{\mathrm{T}} + c_\mu \mathbf{A}\left(\sum_{i=1}^{\lambda} w_i^\circ\ \mathbf{z}_{i:\lambda} \mathbf{z}_{i:\lambda}^{\mathrm{T}}\right)\mathbf{A}^{\mathrm{T}} \tag{2.41}$$

7: **end while**

Algorithm 2 is expressed in terms of a linear transformation $\mathbf{A}$ of the random mutation vector $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, which can be implemented in *Matlab* using the built-in `randn` function. Equation 2.32 of Line 2 computes this transformation by taking the principal square root of the covariance matrix $\mathbf{C}$. Equation 2.33 of Line 3 samples λ trial steps $\mathbf{z}_i$ from a zero-mean normal distribution with covariance $\mathbf{C}$. Equation 2.34 then transforms $\mathbf{z}_i$ such that the distribution of the resulting candidate solutions $\sim \mathcal{N}(\mathbf{m}, \sigma^2 \mathbf{C})$. Line 4 employs weighted averaging via equation 2.35 to compute

the new population centroid where $\mathbf{z}_{i:j}$ notation signifies the i^{th} best of j mutation vectors.

The global step-size update is performed in Line 5 by comparing the length of the evolution path $\mathbf{s}_\sigma$ maintained via equation 2.37 with its expected length under random selection. This evolution path accumulates steps in the coordinate system where the mutation distribution is isotropic and which can be derived by adjusting its scale only. For more suitable change rates, Hansen [21, 24] has added the factor c_σ/d_σ, with d_σ as the damping parameter to equation 2.38.

The covariance matrix update in Line 6 relies on the evolution path $\mathbf{s}_c$ given by equation 2.39, which performs cumulation in the given coordinate system. Equation 2.41 performs a rank-one update using $\mathbf{s}_c$ as well as a rank-μ update using transformation matrix $\mathbf{A}$ and mutation vectors $\mathbf{z}_{i;\lambda}$. This rank-μ update is performed with λ recombination weights w_i° provided by equation 2.40, which is an implementation by Hansen [24] that attempts to scale all negative weights such that the coefficient of $\mathbf{C}$ becomes one while guaranteeing the positive definiteness of $\mathbf{C}$. The specific setting for these weights is given in Table 1 of [24]. The remaining parameters are set according to the recommendation by Hansen [24], with a thorough motivation for their setting offered in [35, 33, 24].

2.3 Gaussian Processes

Evolution strategies generate candidate solutions via unbiased probabilistic sampling. Generally, the majority of candidate solutions are relatively poor and thus discarded. Efficiency can be gained by substituting evaluations of the expensive objective function with calls to cheaper, though potentially inaccurate, approximate models of the objective. Such models can be constructed using information gained from prior objective function evaluations and are referred to as *surrogate models*. Various surrogate modelling approaches have been proposed and surveyed [42, 50] in the context of evolutionary algorithms.

The remainder of the section provides some background on the particular choice of model employed in this work. First, a mathematical definition of Gaussian processes (GP) is presented. Next, the problem of model selection is addressed from both Bayesian and frequentist viewpoints. The section concludes by briefly covering a

generalization that allows for non-Gaussian processes.

2.3.1 Model Definition

A stochastic process is a collection of random variables, $\{f(\mathbf{x}_t) : t \in \mathcal{T}\}$, indexed by elements from a set $\mathcal{T}$, known as the index set. A Gaussian Process is a stochastic process such that any finite subset of random variables has a multivariate Gaussian distribution [62]. Specifically, a collection of random variables $\{f(\mathbf{x}_t) : \mathbf{x} \in \mathbb{R}^n, t \in \mathcal{T}\}$ is said to be drawn from a Gaussian process with *mean function* $m(\cdot)$ and covariance function $k(\cdot, \cdot)$ if for any finite set of elements $\mathbf{x}_1, \ldots, \mathbf{x}_t$, the associated finite set of random variables $f(\mathbf{x}_1), \ldots, f(\mathbf{x}_t)$ have joint Gaussian distribution

$$\begin{bmatrix} f(\mathbf{x}_1) \\ \vdots \\ f(\mathbf{x}_t) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(\mathbf{x}_1) \\ \vdots \\ m(\mathbf{x}_t) \end{bmatrix}, \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_2) & \ldots & k(\mathbf{x}_1, \mathbf{x}_t) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_t, \mathbf{x}_1) & \ldots & k(\mathbf{x}_t, \mathbf{x}_t) \end{bmatrix} \right). \tag{2.42}$$

The mean and variance functions are suitably named since for any pair $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^n$ in the finite collection of inputs $\mathbf{x}_1, \ldots, \mathbf{x}_t$, the *marginalization property* of multivariate normal distributions implies that:

$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})], \tag{2.43}$$

$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[\big(f(\mathbf{x}) - m(\mathbf{x})\big)\big(f(\mathbf{x}') - m(\mathbf{x}')^{\mathrm{T}}\big)]. \tag{2.44}$$

This property determines that if, for example, the GP specifies $p(f(\mathbf{x}_1), f(\mathbf{x}_2)) \sim \mathcal{N}(\mu, \Sigma)$, then it must also specify $p(f(\mathbf{x}_1)) \sim \mathcal{N}(\mu_1, \Sigma_{11})$, where Σ_{11} is the relevant sub-matrix of Σ. The marginalization property, also known as a *consistency* requirement, is automatically satisfied if the covariance function specifies the entries of the covariance matrix [62]. A standard result in probability theory states that for a valid GP, $k(\cdot, \cdot)$ must be positive semi-definite. This is plausible intuitively as variance is by definition greater than, or equal to zero. A positive semi-definite matrix is also referred to as a *kernel* [56] and is used interchangeably with the term covariance matrix.

The definition of GPs as having any finite number of variables characterized by a multivariate Gaussian distribution makes the model readily interpretable and analytically tractable for predictions with data. For a training dataset of observations $\mathcal{D} =$

$\{\mathbf{x}_i, f_i\}_{i=1}^{i=m}$, as well as test locations with corresponding values $\mathcal{D}^* = \{\mathbf{x}_i^*, f_i^*\}_{i=1}^{i=m^*}$, the joint distribution of training outputs and test outputs according to the prior is multivariate Gaussian distributed

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}^* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(\mathbf{x}) \\ m(\mathbf{x}^*) \end{bmatrix}, \begin{bmatrix} k(\mathbf{x},\mathbf{x}) & k(\mathbf{x},\mathbf{x}^*) \\ k(\mathbf{x}^*,\mathbf{x}) & k(\mathbf{x}^*,\mathbf{x}^*) \end{bmatrix} \right), \tag{2.45}$$

where $\mathbf{x} \in \mathbb{R}^{n \times m}$ and $\mathbf{x}^* \in \mathbb{R}^{n \times m^*}$ are the first component of $\mathcal{D}$ and $\mathcal{D}^*$ respectively, with n denoting dimension. Introducing more compact notation: $k(\mathbf{x}, \mathbf{x}^*)$ as $\mathbf{k}_*$, $k(\mathbf{x}^*, \mathbf{x}^*)$ as $\mathbf{k}_{**}$, and $k(\mathbf{x}, \mathbf{x})$ as $\mathbf{K}$, and noting that the conditional probability density function of a joint Gaussian distribution is itself Gaussian [62], the distribution of test outputs given training data is

$$p\left(\mathbf{f}^* | \mathbf{x}^*, \mathbf{x}, \mathbf{f}\right) \sim \mathcal{N}\big(m(\mathbf{x}^*) + \mathbf{k}_*^{\mathrm{T}}\mathbf{K}^{-1}(\mathbf{f} - m(\mathbf{x})),\ \mathbf{k}_{**} - \mathbf{k}_*^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{k}_*\big). \tag{2.46}$$

The proof of this property is omitted; interested readers are referred to [75]. The predictive mean and covariance can be read from 2.46:

$$\mathbb{E}[f^* | \mathbf{x}^*, \mathbf{x}, f] = m(\mathbf{x}^*) + \mathbf{k}_*^{\mathrm{T}}\mathbf{K}^{-1}(\mathbf{f} - m(\mathbf{x})), \tag{2.47}$$

$$\mathbb{V}[f^* | \mathbf{x}^*, \mathbf{x}, f] = \mathbf{k}_{**} - \mathbf{k}_*^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{k}_*. \tag{2.48}$$

A wide range of functions can be modelled by specifying the covariance function of a GP. Linear regression splines and Kalman filters are examples of GPs with particular kernels [62]. There are many possible choices of kernel function. One common example is the *squared exponential*

$$k(\mathbf{x}_p, \mathbf{x}_q) = \theta_0^2 \exp\big(-\frac{1}{2\theta_1^2}\|\mathbf{x}_p - \mathbf{x}_q\|^2\big) + \theta_2^2 \delta_{pq}, \tag{2.49}$$

where the parameters $\theta = \{\theta_0^2, \theta_1^2, \theta_2^2\}$ are referred to *signal variance*, *length-scale*, and *noise variance* in order of appearance, with δ_{pq} denoting the Kronecker delta, which is one iff $p = q$ and zero otherwise. Informally, the length-scale characterizes the "wiggliness" of a function; the signal variance controls the uncertainty about the mean function; and the noise variance takes into account observation noise. Figure 2.6 demonstrates the effect of these kernel parameters on a one dimensional case of Gaussian process regression.

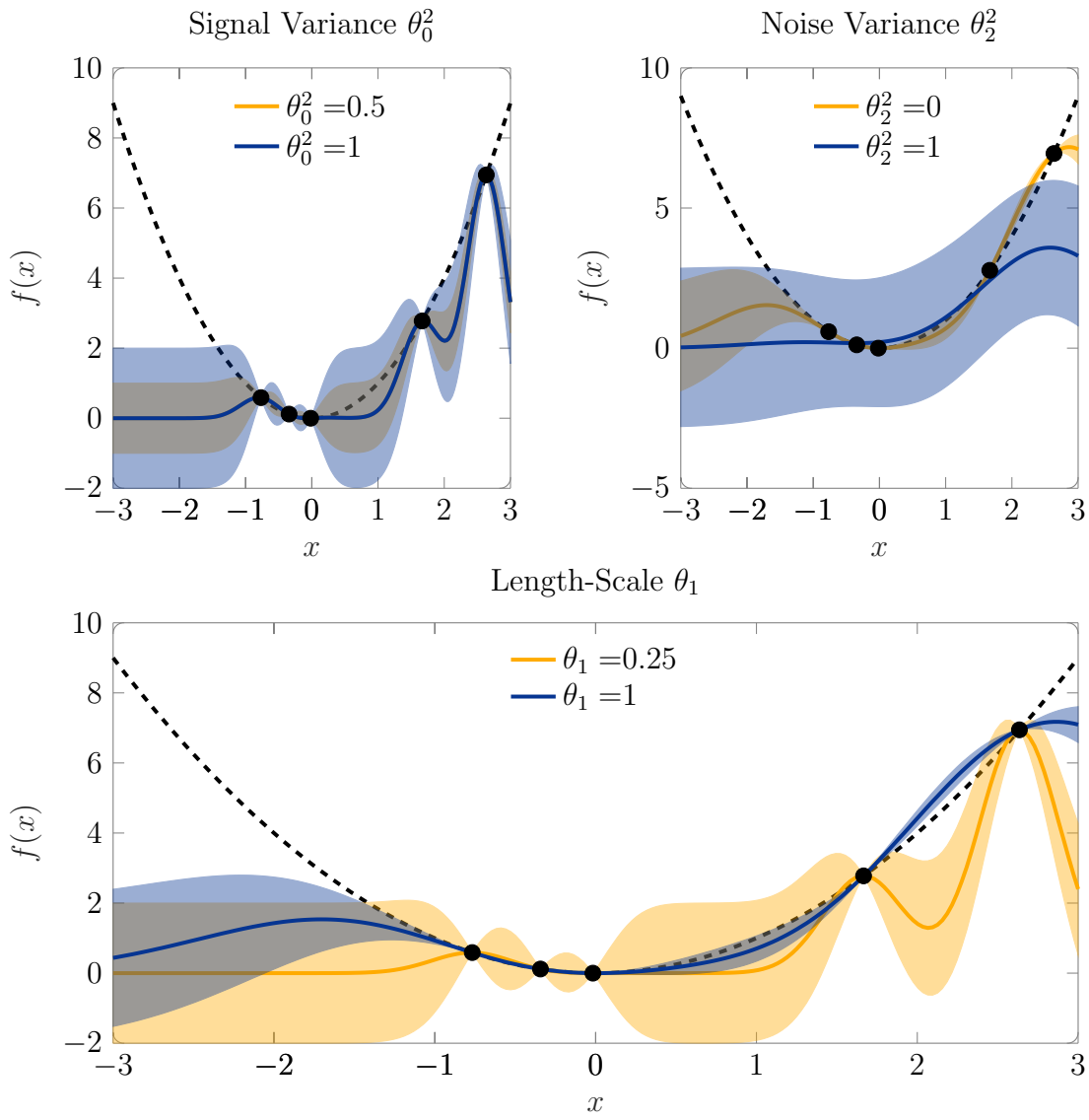


Figure 2.6: Impact of the squared exponential kernel parameters θ on one-dimensional GP mean and variance functions. The dashed line indicates the function $f(x) = x^2$ being modelled and from which five samples, denoted by black dots, are randomly selected. The coloured lines illustrate the mean functions for various parameter settings and the shades highlight two standard deviations about the mean values. Each sub-figure displays the GP prediction when one component of θ varies while the others remain fixed.

2.3.2 Model Selection

Model selection refers to the problem of specifying model parameters. Two philosophically distinct approaches to inference can be pursued to this end: Bayesian and frequentist. While in the former paradigm the problem is formulated as maximizing the probability of the model for some given observations, the latter approach involves minimizing an estimate of generalization error. This section briefly describes how model selection can be performed in the Bayesian sense via *maximum likelihood estimation* and in the frequentist sense by means of *cross-validation*.

2.3.2.1 Maximum Likelihood Estimation

Bayesian principles provide a systematic framework for inference. A full Bayesian treatment of model selection can involve the application of these principles at various hierarchical levels: from over the discrete space of covariance function choices, to the continuous space of covariance function parameters [62]. However the application of these principles in practice often requires computation of the analytically intractable *marginal likelihood* of a dataset, which are integrals over the parameter space for a given model. This makes deriving good approximations difficult. By virtue of their definition, GP regression models provide analytically tractable marginal likelihoods which can be maximized to infer model parameters. In addition to enabling automatic model construction, the marginal likelihood of a dataset given a particular model, allows for comparisons between models, balancing between model complexity and data fit [62]. As these assessments constitute important aspects of model selection, the Bayesian framework is considered to be persuasive.

Let $\mathbf{x} \in \mathbb{R}^n$ be input locations, $f : \mathbb{R}^n \to \mathbb{R}$, $\mu \in \mathbb{R}^m$, and $\Sigma \in \mathbb{R}^{m \times m}$. The marginal likelihood under a GP prior of a set of function values $[f(\mathbf{x}_1), f(\mathbf{x}_2), \ldots, f(\mathbf{x}_m)]^{\mathrm{T}} := \mathbf{f}(\mathbf{x})$ at locations $\mathbf{X} \in \mathbb{R}^{n \times m}$ is given by:

$$p(\mathbf{f}(\mathbf{X})|\mathbf{X}, \mu(\cdot), k(\cdot,\cdot)) = \mathcal{N}\big(\mathbf{f}(\mathbf{X})|\mu(\mathbf{X}), k(\mathbf{X},\mathbf{X})\big), \tag{2.50}$$

which can be written as:

$$(2\pi)^{-\frac{m}{2}} \times |k(\mathbf{X},\mathbf{X})|^{-\frac{1}{2}} \times \exp\bigg(-\frac{1}{2}(\mathbf{f}(\mathbf{X}) - \boldsymbol{\mu}(\mathbf{X}))^{\mathrm{T}} k(\mathbf{X},\mathbf{X})^{-1}(\mathbf{f}(\mathbf{X}) - \boldsymbol{\mu}(\mathbf{X}))\bigg). \tag{2.51}$$

Taking the logarithm of equation 2.51 and dropping $\mathbf{X}$ for compactness, the *log marginal likelihood* can be written as

$$\log p(\mathbf{f}|\mathbf{X},\theta) = -\frac{m}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{1}{2}(\mathbf{f}-\boldsymbol{\mu})^{\mathrm{T}}\mathbf{K}^{-1}(\mathbf{f}-\boldsymbol{\mu}), \tag{2.52}$$

where $\mathbf{K}$ is the covariance matrix for the targets $\mathbf{f}$. The logarithmic form facilitates differentiation with respect to the parameters which can be useful in its maximization. Readers are referred to [62] for a derivation of partial derivatives of equation 2.52 with respect to θ. The first term in equation 2.52 is a normalization constant; the second term imposes a complexity penalty as it depends only on the covariance function and the input locations; and the third term controls the data fit as it is the only term involving observed data [62]. Informally, complexity refers to the degree of the Taylor polynomial required to sufficiently approximate the underlying function being modelled about a given point. In this sense, a linear function would be considered simple, while a "wiggly" function would be regarded as more complex. Maximization of the marginal likelihood favours an increase in the variable terms of equation 2.52, which requires a less wiggly model defined by a larger length-scale for the second term and a more wiggly model defined by a smaller length scale for the last term. Thus the second term can be interpreted as penalizing model complexity, while the last term can be regarded as encouraging it, leading to a better data fit.

From a Bayesian viewpoint, a complex model is capable of accounting for a wider range of possible data sets than a simple one. Figure 2.7 illustrates the behaviour of the marginal likelihood for three different model complexities, given a fixed number of data points m and input locations $\mathbf{X}$. The horizontal axis is an idealized depiction of all possible vectors of targets $\mathbf{f}$ and the vertical axis represents the marginal likelihood $p(\mathbf{f},\mathbf{X}|H_i)$ for a given model H_i. The marginal likelihood is a probability distribution over $\mathbf{f}$ that must, by definition, normalize to unity. This constraint implies that the marginal likelihood of a simple model that accounts for a limited range of possible sets of data is larger for those target values which the model can explain. Conversely, a complex model, being more flexible, accounts for a wider range of possible data sets and would necessarily attain a smaller marginal likelihood on the same set of targets. As improving model fit and lowering model complexity are opposing requirements, maximizing the marginal likelihood thus implies balancing these two factors. In this way, the marginal likelihood can be interpreted as tending to favour

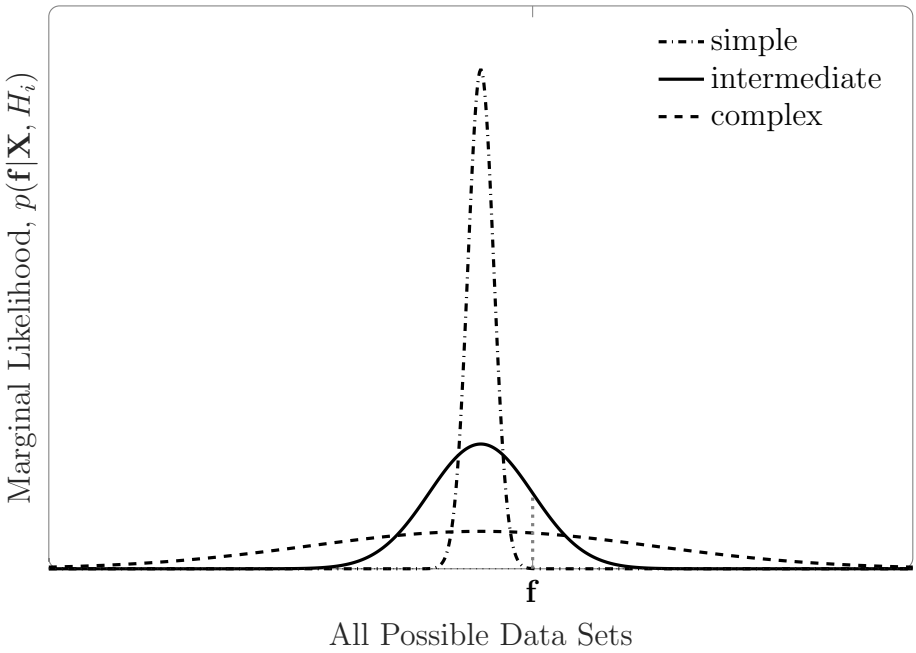


Figure 2.7: Data fit versus complexity trade-off (figure and caption modified from [62]). The marginal likelihood is $p(\mathbf{f}|\mathbf{X}, H_i)$ is the probability of the data, given some model H_i. The horizontal axis is an idealized representation of of all possible vectors of targets $\mathbf{f}$. The number of data points m and the inputs $\mathbf{X}$ are fixed and not shown. Marginal likelihoods for models of three different complexities are plotted and must normalize to unity. For a particular data set indicated by $\mathbf{f}$ (dotted line), a model of intermediate complexity over too simple or too complex alternatives is preferred by this measure.

the simplest model that best explains the data. This trade-off is a consequence of GP model definition and is thus automatic, making this an appealing approach to model selection.

2.3.2.2 Cross-Validation

Cross-validation (CV) involves partitioning the collection of observations into two disjoint sets, using one to train and the other to monitor the performance of the model. Model performance on the latter set (*validation set*) is used as a proxy for generalization error. Model selection is thus performed by minimizing this error.

A shortcoming of CV is that only a fraction of the full data can be used for training, and if the validation set is small, the performance estimate obtained may be highly variable. To address this, data is split into k disjoint, equally sized subsets;

validation is done on a single subset and training is done on the union of the remaining $k-1$ subsets. This procedure is repeated k times, until all subsets have been used for validation. Thus, all cases appear as validation cases and a large fraction of the data is used for training. This procedure is referred to as $k-$*fold cross-validation* and its drawback is the requirement to build k models instead of one. When $k = m$, where m the number of training cases, this is known as leave-one-out cross-validation (LOO-CV).

Cross-validation can be performed with any loss function. Although the squared error loss is the most common choice for regression, for probabilistic models such as GPs it is natural to consider also cross-validation using the *negative log probability loss*, known as the *log pseudo-likelihood*. The predictive log probability when leaving out training case i is

$$\log p\left(f_i|\mathbf{X}, \mathbf{f}_{-i}, \theta\right) = -\frac{1}{2}\log \sigma_i^2 - \frac{(f_i - \mu_i)^2}{2\sigma_i^2} - \frac{1}{2}\log 2\pi, \tag{2.53}$$

where f_i is the prediction at point $\mathbf{x}_i$, $\mathbf{f}_{-i}$ refers to all but the i^{th} target in which the training sets are taken to be $(\mathbf{x}_{-i}, \mathbf{f}_{-i})$, and μ_i and σ_i^2 are computed according to equations 2.47 and 2.48 respectively. Accordingly, the LOO log predictive probability is

$$L_{LOO}(\mathbf{X}, \mathbf{f}, \theta) = \sum_{i=1}^{m} \log p(f_i|\mathbf{X}, \mathbf{f}_{-i}, \theta). \tag{2.54}$$

The training cost of building m models is often prohibitive, but for GP regression, there are cheaper computational alternatives. In each of the m folds, the most computationally demanding aspect of inference in a GP model with fixed parameters is a matrix inversion costing $\mathcal{O}(m^3)$. When repeatedly applying the predictions from equations 2.47 and 2.48, the expressions are almost identical, with the i^{th} row and column removed in each fold. Thus a single inverse of the covariance matrix can be computed using inversion by partitioning, yielding the following expressions for the LOO-CV predictive mean and variance [70]:

$$\begin{aligned} \mu_i &= f_i - \frac{[\mathbf{K}^{-1}\mathbf{f}]_i}{[\mathbf{K}^{-1}]_{ii}}, \\ \sigma_i^2 &= \frac{1}{[\mathbf{K}^{-1}]_{ii}}. \end{aligned} \tag{2.55}$$

The expense of computing these quantities includes a single matrix inversion costing $\mathcal{O}(m^3)$ in addition to the m vector multiplications of $\mathcal{O}(m^2)$ cost (once $\mathbf{K}^{-1}$ is known).

Plugging these equations into 2.53 and 2.54 produces a performance estimator that can be optimized to perform model selection. The partial derivatives of L_{LOO} with respect to θ are given in [62] and can be employed in the error minimization.

2.3.3 Model Warping

While the presumption of a joint multivariate Gaussian distribution of data enables predictions using simple matrix manipulations, it is often an unreasonable assumption in practice. One such example occurs when observations consist of positive quantities varying over multiple orders of magnitude. In this case, it is common to apply a logarithmic transformation to the data before modelling [69]. However the logarithm is but one instance of non-linear transformation, or *warp*, that could be employed to render the observation space more suitable for a GP. Further, the ad-hoc choice of function can be replaced by a principled, probabilistic approach to defining a warp. Snelson et al. [69] introduce warped Gaussian processes (WGP) that generalize GPs to non-Gaussian observation spaces within a Bayesian framework.

Given a dataset $\mathcal{D}$ consisting of input locations $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{i=m}$ and corresponding real-valued targets $\{f_i\}_{i=1}^{i=m}$, the goal is to predict f^* at some new input location $\mathbf{x}^*$. Instead of placing a prior directly on the function space by assuming that any finite selection of points $\mathbf{x}$ gives rise to a joint normal distribution over function values $\mathbf{f}$, Snelson et al. [69] consider a vector of so-called *latent targets* $\mathbf{z}$ that is well-modelled by a GP such that

$$\log p(\mathbf{z}|\mathbf{X}, \theta) = -\frac{m}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{1}{2}\mathbf{z}^{\mathrm{T}}\mathbf{K}^{-1}\mathbf{z}, \tag{2.56}$$

where θ parametrizes the covariance $\mathbf{K}$ of latent targets. To obtain $\mathbf{z}$, each component f_i of the target vector $\mathbf{f}$ is mapped from the observation space to a component z_i in the latent space via a strictly increasing function $\Omega : \mathbb{R} \to \mathbb{R}$ such that

$$z_i = \Omega(f_i - \mu(\mathbf{X}); \omega) \quad \forall i = 1, \ldots, m, \tag{2.57}$$

with ω parametrizing the transformation. To conserve the probability measure, Ω is by definition required to be a *measurable transformation*, meaning that the inverse image of an admissible set $\mathbb{R}$ in the range space must be admissible in the domain space. Moreover, to permit warps from the latent space back into observation space,

Ω is required to be invertible and again, in order to preserve the probability measure in that reverse direction, Ω^{-1} is also required to be measurable. This implies that in addition to being strictly monotonic, the bijection Ω on $f \in \mathbb{R}$ must map onto the entirety of the real line. Including the Jacobian term to account for the transformation of random variables, the resulting log likelihood in observation space becomes:

$$\log p(\mathbf{f}|\mathbf{X},\theta,\omega) = -\frac{m}{2}\log 2\pi - \frac{1}{2}\log|\mathbf{K}| - \frac{1}{2}\Omega\big(\mathbf{f}-\mu(\mathbf{X});\omega\big)^{\mathrm{T}}\mathbf{K}^{-1}\Omega\big(\mathbf{f}-\mu(\mathbf{X});\omega\big) + \sum_{i=1}^{m}\log\frac{\delta\Omega\big(f-\mu(\mathbf{X});\omega\big)}{\delta f}\bigg|_{f_i}. \tag{2.58}$$

The log marginal likelihood of the WGP given by equation 2.58 can be optimized with respect to both θ and ω in order to perform model selection. This enables the simultaneous learning of both the kernel parameters and the non-linear transformation within the same Bayesian framework. The computational cost incurred by the additional differentiation is negligible compared to the cubic cost of inverting $\mathbf{K}$, as long as the derivatives of Ω are easy to compute.

The predictive distribution at a new point z^* in latent variable space is just as for a regular GP, calculated according to equation 2.47 for the mean μ_{z^*} and 2.48 for variance because

$$p(z^*|\mathbf{x}^*,\mathcal{D},\theta) \sim \mathcal{N}(\mu_{z^*}(\theta),\sigma^2_{z^*}(\theta)). \tag{2.59}$$

In observation space, that Gaussian is passed through the non-linear warping function, yielding:

$$p(f^*|\mathbf{x}^*,\mathcal{D},\theta,\omega) = \frac{\Omega'(f^*;\omega)}{\sqrt{2\pi}\sigma_{z^*}}\exp\left[-\frac{1}{2}\left(\frac{\Omega(f^*;\omega)-\mu_{z^*}}{\sigma_{z^*}}\right)^2\right]. \tag{2.60}$$

While the shape of the distribution defined by 2.60 depends on the warping function Ω, in general it may be asymmetric and multimodal [69]. Figure 2.8 visualizes the space transformations that compose warped GP predictions.

Warping functions should ideally be flexible, allowing for classes of transformations ranging from linear to highly non-linear. The authors [69] give the example of a neural network style sum of `tanh` functions:

$$\Omega(f;\omega) = f + \sum_{i=1}^{I} a_i \mathtt{tanh}(b_i(f+c_i)) \quad a_i, b_i \geq 0 \ \ \forall i, \tag{2.61}$$

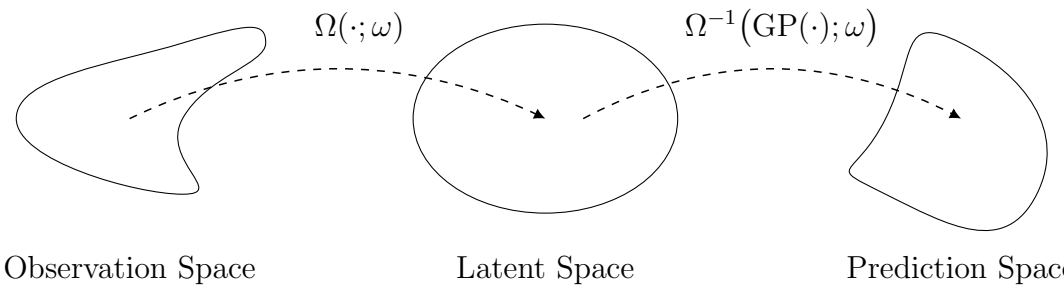


Figure 2.8: A schematic of the space transformations involved in generating warped Gaussian process predictions. Points in the observation space undergo a transformation Ω into a so-called latent space that renders them more amenable to modelling by Gaussian Processes. Points from the latent space are then provided as input to a Gaussian process model whose predictions are "unwarped" by applying the inverse transformation.

where $\omega = \{\mathbf{a},\mathbf{b},\mathbf{c}\}$ and for each `tanh` component: **a** controls the vertical scale, **b** controls the horizontal scale, and **c** controls the horizontal position. The desired flexibility of the warping function is controlled by parameter I and f is added to make Ω unbounded on $\mathbb{R}$ in order to lead to a proper probability density in the observation space. Many other warping functions are possible [69].

There are drawbacks to the parametric formulation of warps in WGPs as maximum likelihood can sometimes fail under low data regimes, data with censored values, etc. [48] Lázaro-Gredilla [48] addresses this problem by introducing Bayesian warped Gaussian processes (BWGP) to demonstrate the viability of non-parametric warps in GP regression.

2.4 Empirical Evaluation

Rigorous quantitative evaluation of algorithm performance is an important aspect in the research and development of optimization algorithms. This section provides a brief overview of the components of empirical evaluation related to this work. After a brief description of the role of benchmarks, the types of difficulties posed by test problems as well as related function families are discussed. The section concludes by explaining significant elements of empirical evaluation such as initialization, termination, and comparison criteria.

2.4.1 Benchmarks

While at first glance evaluating algorithm performance may appear simple, in practice it is surprisingly tedious and riddled with intricacies. Benchmarking entails a series of tasks including: selecting a set of objective functions; defining objective function transformations; identifying relevant performance criteria; and communicating data in an interpretable manner. Each task involves subtle decisions crucial for the validity of the outcome. The goal is to generate results that lead to deeper insights beyond the standard performance ranking on a problem set.

Benchmarks aim to provide a framework for the systematic assessment of algorithm performance to facilitate an understanding of algorithm behaviour. The COCO platform [30] ("COCO" standing for COmparing Continuous Optimizers) provides a comprehensive set of such tools that enable systematic experimentation in the field of evolution strategies. Among other utilities, COCO offers the Black Box Optimization Benchmark (BBOB) *test suites*, which are collections of test functions used to systematically evaluate an algorithm's robustness to problem characteristics such as noise, ill-conditioning, multi-modality, etc. The BBOB offers multiple test-beds, each containing numbered problems with corresponding analytic definitions. A subset of the noiseless test-bed [31] is employed in this investigation.

2.4.2 Test Functions

The choice of test function is motivated by the desire to understand algorithm performance with respect to some specific problem characteristic. A function may be selected to model a relevant practical difficulty or a simple topology, such as a linear slope, that every search algorithm should in principle be able to overcome. The following subsections describe a few common classes of difficulty that may be posed by an objective function as well as a natural grouping by function definitions referred to as *function families*.

2.4.2.1 Characteristics

Multi-modality

Multi-modal functions possess more than one local optimizer. For strategies that rely on neighbourhood information to converge to the global optimum, local optima present a challenge as convergence to such solutions may stop progress altogether.

Optimization algorithms often employ restart strategies to overcome this challenge. Two prominent variants of the CMA-ES that use restart strategies are the IPOP- [3] and BIPOP-CMA-ES [23]. In IPOP-CMAES, CMA-ES is restarted with double the population size each time the termination criteria are satisfied. The BIPOP-CMA-ES operate under two regimes. In the large population regime, CMA-ES doubles its population size at each restart; in the small population mode, it varies its population size uniformly at random between $[\lambda_{\text{default}}/2, \lambda_{\text{default}}]$ at each restart, where parameter λ_{default} denotes the default population size of the CMA-ES adapted for unimodal functions. BIPOP-CMA-ES start with λ_{default} and in each restart pick the mechanism that has incurred the fewest function evaluations thus far. Loshchilov et al. [53, 54] employ both strategies in their surrogate-assisted CMA-ES variant [50] to achieve competitive results on multimodal functions.

Ill-conditioning

After multi-modality, *Ill-conditioning* is perhaps the most common challenge in real parameter optimization [30]. The *condition number* of a matrix $\mathbf{H}$ refers to the ratio of the largest to the smallest eigenvalues of $\mathbf{H}$. In the case of convex quadratic functions, *conditioning* can be defined as the condition number of the function's Hessian matrix $\mathbf{H}$. Level sets of convex quadratic functions are ellipsoids; the condition number of these functions corresponds to the square of the ratio between the largest and smallest axes of the ellipsoidal level sets. For more general functions, conditioning informally refers to the ratio the longest and shortest principal axes of the functions. Problems that exhibit condition numbers in excess of 10^5 are considered to be highly ill-conditioned [28]. The BBOB [31] test-bed contains ill-conditioned functions with a typical conditioning of 10^6.

Separability

A problem is considered to be separable if it can be decomposed into a sequence of n

independent optimization procedures:

$$\arg\min_{\mathbf{x}\in\mathcal{X}^n} f(\mathbf{x}) = \left(\arg\min_{x_1} f(\mathbf{x}), \ldots, \arg\min_{x_n} f(\mathbf{x})\right), \tag{2.62}$$

where $\mathcal{X}^n$ is the n-dimensional search space. Non-separable problems are considered to be much more difficult than separable ones because the difficulty of separable problems scales linearly in problem dimension n, as the search space volume increases exponentially in n. Thus most benchmark functions are designed to be non-separable in order to prevent exploitation of such problem structure. A problem may be non-separable in the original variables and with a rotation of the coordinates it becomes separable in the different coordinate system, which an algorithm may be able to learn. Other problems aren't separable in any coordinate system. The well-established technique for generating non-separable functions from separable ones is the application of a random rotation matrix $\mathbf{R}$ to the function's input coordinates [30]. However this technique does not work in some cases, such as on functions that feature a perfectly isotropic topology.

2.4.2.2 Families

The set of test functions considered in this thesis falls into the following four families: sphere functions, convex quadratic functions, quartic functions, and power sum functions. This section describes each family.

Sphere functions take the following form:

$$f(\mathbf{x}) = \left(\mathbf{x}^{\mathrm{T}}\mathbf{x}\right)^{\alpha/2}, \tag{2.63}$$

where $\mathbf{x} \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}_+$. The *quadratic sphere* is obtained for $\alpha = 2$ in equation 2.63. Spheres exhibit identical scales in all directions and are thus perfectly suited to the isotropic mutation operator. The (1+1)-ES without surrogate model assistance achieves the same convergence rate on all sphere functions. However, unless using comparison-based surrogate models as proposed by Loshchilov et al. [51], surrogate models may exhibit different degrees of accuracy depending on α.

Spheres are a good candidate for theoretical and empirical analyses of evolution strategies. Mathematical analysis of ES is difficult and analytically feasible only on simple objective functions [1]. In practice, an optimization algorithm that is

unsuccessful on an objective function as simple as the sphere cannot reasonably be expected to perform well on more complicated test functions. Moreover, a search strategy is ideally able to adapt to the fitness landscape such that locally, the problem is rescaled into a spherical one [1]. Such local adaptation demand is hoped to be met by the CMA-ES, which linearly transform the problem into a spherical one on convex quadratic functions [35]. Analyses of the performance of ES on the sphere can thus be expected to generalize to other objective functions.

Convex quadratic functions are of the form:

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{H}\mathbf{x}, \tag{2.64}$$

where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{H} \in \mathbb{R}^{n\times n}$ is a positive definite matrix. Matrix $\mathbf{H}$ can be set to vary the type of adaptation demand imposed on the optimizer. For $\mathbf{H} = \mathtt{diag}(\beta, 1, \ldots, 1)$, this family includes both *cigar* ($\beta \ll 1$) and *discus* functions ($\beta \gg 1$). In the *ellipsoid* function, $\beta^{\frac{(i-1)}{(n-1)}}, i = 1, ..., n$, with $\beta \approx 10^6$ in the COCO benchmark [30]. When $\mathbf{H}$ equals the identity matrix $\mathbf{I}$, the quadratic sphere is recovered. On convex quadratic functions, the covariance matrix of the CMA-ES approximately converges to the inverse Hessian $\mathbf{H}^{-1}$ during its run [24].

Generalized quartic functions

$$f(\mathbf{x}) = \sum_{i=1}^{n-1} \left[\gamma(x_{i+1} - x_i^2)^2 + (1 - x_i)^2\right] \tag{2.65}$$

are less than perfectly conditioned and non-quadratic. These functions feature a bent ridge whose conditioning increases by increasing γ, thus intensifying the requirement for constant relearning of the axis scales. At $\gamma = 100$, this family includes the generalized Rosenbrock function, which becomes tedious to optimize without CMA. The global optimizer for these functions is located at $\mathbf{x} = \mathbf{1}$, however for $n \geq 4$ they exhibit a local optimum.

Power summations of the form:

$$f(\mathbf{x}) = \sqrt{\sum_{i=1}^{n} |x_i|^{2+4\frac{i-1}{n-1}}} \tag{2.66}$$

combine the properties of being highly ill-conditioned and non-quadratic. The BBOB [28] benchmark calls equation 2.66 the *Sum of Different Powers* (f_{diffpow}). This family,

as the generalized quartic functions, cannot be linearly transformed into a spherical problem. Ill-conditioning continually increases while approaching the optimum. A strategy that relies on local information is presumed to pursue a progressively narrowing ridge on this class of test problems [35].

2.4.3 Experiment Settings

Decisions regarding how to initialize, terminate, and compare the performance of optimizers form an important part of their empirical evaluation. This subsection provides some background on these considerations.

2.4.3.1 Initialization Criteria

Requisite inputs to optimization algorithms typically include: the objective function of interest, an initial n-dimensional point, and applicable search space constraints. All functions in the COCO benchmark are defined everywhere in $\mathbb{R}^n$ and have their global optimum in $[-5, 5]^n$, with most having their global optimum in $[-4, 4]^n$ [30]. A reasonable setting for the starting location is to provide a point selected uniformly at random from the interval $[-4, 4]^n$ [28]. Functions are typically initialized via the provision of random shifts and rotation matrices, which test the solvers' invariance to translations and rotations of the search space.

2.4.3.2 Termination Criteria

Termination criteria are typically expressed in terms of either: a function evaluation budget, a target function value, a distance to the optimizer, or a combination thereof. The specific choice of termination criteria depends on the aspect of the problem under evaluation and is limited by considerations of numerical accuracy. The number of function evaluations required to reach a target value is known as the *run-time* of an algorithm. This measure can be aggregated from several targets for a given problem in order to study how algorithm performance scales with respect to problem difficulty.

Target function values f_{target} are parametrized as the sum of the optimal function value f_{opt} and target precision f_{Δ}. The optimal function value f_{opt} differs by objective function. Problem difficulty does not necessarily scale uniformly with decrease in target function value and may result in varying characteristics of the problem to

be solved. The power sum functions are such an example, where ill-conditioning increases with decreasing distance to the optimizer. However, smaller target values on the same function are invariably more difficult to reach. Thus the choice of f_Δ depends on the experiment's objectives. Reasonable values change by simple modifications in the function definition, such as an order preserving transformation. The lower-bound on target precision is determined by the computing environment's floating point precision. A value of no less than $f_\Delta = 10^{-8}$ is used in the COCO benchmark [30].

2.4.3.3 Comparison Criteria

Performance metrics are ideally quantitative, interpretable, and as simple as possible [30]. Cost incurred in satisfying termination criteria forms the basis for comparing algorithm performance. Trial runs terminate when either the budget of function evaluations calls is exceeded, or when the optimum f_{opt} is reached to within some predefined precision f_Δ. The former termination criterion is characterized as a *fixed-budget* and the latter as a *fixed-target* scenario [28].

The fixed-budget scenario is graphically represented by a vertical line on convergence graphs as seen in Figure 2.9, with cost being measured in the number of objective function calls. The function evaluation budget is often determined by practical constraints such as CPU time, with larger budgets being generally expected to lead to better solutions. This mode of measuring cost solely provides a ranking of comparators; it does not allow quantitatively interpretable comparisons as there is no a priori evidence regarding the scaling of problem difficulty with respect to function value. Thus, interpretation of resulting data requires a thorough understanding of the underlying test function. Values from different test functions are in general incompatible and therefore not amenable to aggregation. Moreover, for algorithms that are invariant under transformations of the function value, fixed-budget measures can be made invariant to these transformations by transforming all resulting function values.

In the fixed-target scenario, cost is quantified by counting the number of function evaluations until the target $(f_{\text{opt}} + f_\Delta)$ is reached. This can be pictured as a horizontal line on convergence graphs (see Figure 2.9) and is the preferred setting

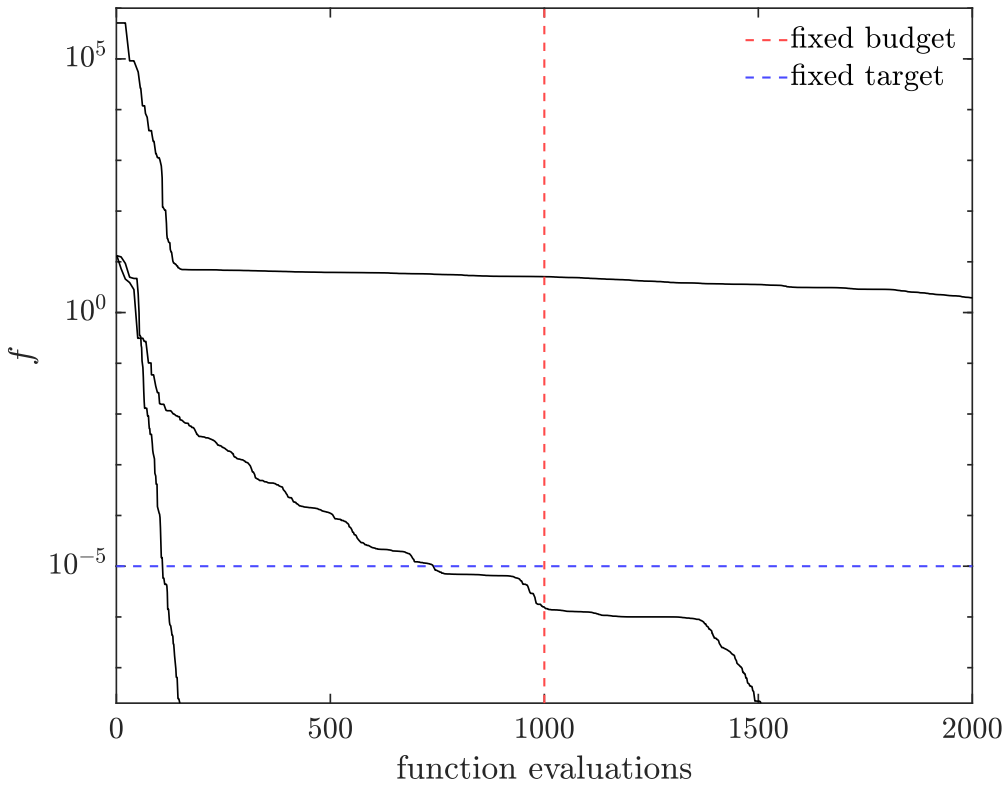


Figure 2.9: Fixed target and fixed budget evaluation scenarios. The fixed budget criteria depicted by the red vertical line can fail to detect good results, while under the fixed-target scenario illustrated by the blue horizontal line, bad results can be missed. Running times are quantitatively meaningful; unless objective function values are quantitatively meaningful as well, the fixed budget approach is preferable.

for benchmarking as it enables quantitative and interpretable comparisons between algorithms. The interpretation of cost in the fixed-target scenario is independent of the test function; hence results from different runs can be aggregated. Fixed function value targets can be made invariant with respect to rank-preserving transformations of the objective function simply by transforming the chosen target function value. Alternatively, parametrizing termination criteria in terms of a target distance to the optimizer, such as $\|\mathbf{x} - \mathbf{x}_{\text{opt}}\|$, avoids the requirement for this adjustment as this metric is invariant to function value transformations.

The median number of function evaluations required to reach a target function value in a batch of trials is a reliable means of measuring running-time for unimodal test problems, which do not in principle require restarts. This median is computed

over N_{trial} runs for a single setup (test function, dimensionality, etc.) and can be a good choice as it is robust to outliers and permits quantitative comparisons. The default value for N_{trial} is fifteen in the COCO benchmark [30] as this number is considered sufficiently large to render relevant performance differences statistically significant, while being small enough be computationally feasible [28].

2.5 Related Work

Surrogate modelling techniques have been applied extensively to black-box optimization problems across various disciplines. In the domain of evolution strategies, multiple such algorithms have been proposed and covered in comprehensive reviews [42, 61, 5, 50]. This section highlights model-assisted CMA-ES that are deemed competitive on the COCO benchmark and considers a few algorithm variants directly related to the current work.

Kern et al. [47] developed one of the first variants of surrogate-assisted CMA-ES, named lmm-CMA-ES ("lmm" standing for local meta model). They exploit the strategy's covariance matrix to construct locally-weighted quadratic models for each offspring that scale the influence of previously evaluated training points on candidate solution estimates. In each generation, the strategy produces a population of candidate solutions, ranks them using its surrogate, and selects a fraction of its seemingly best candidate solutions for evaluation on the true objective function. This fraction is adapted according to rank changes from adding data to the model across generations, an idea inspired by the work of Runarsson [66] who termed the notion "approximate ranking". Bouzarkouna et al. [7] further refine this algorithm to be more effective for larger population sizes.

Loshchilov et al. propose the ACM-ES [51] which employ support vector machines that use the covariance matrix adapted by CMA-ES within a Gaussian kernel. The surrogate is used to rank candidate solutions, thus preserving the strategy's invariance with respect to order-preserving transformations of the objective. They emphasize the importance of retaining this property in the surrogate models that assist comparison-based optimizers. In a further evolution of the approach, they integrate a second CMA-ES to adjust the surrogate's hyper-parameters throughout the run, as well as a mechanism to adaptively refine the number of generations required to retrain a new

model. The algorithm was termed ^{s*}ACM-ES [53] ("^{s*}" standing for Self-Adaptive Surrogate-Assisted) and was later upgraded in subsequent work by the authors [54] to include a procedure for increased exploitation of the model. A greater reliance on the surrogate was empirically demonstrated to benefit performance on unimodal test problems while increasing the tendency to fail on multimodal functions.

Bajer et al. [5] focus on Gaussian process assisted CMA-ES variants and use the COCO benchmark to systematically compare these algorithms against several other evolutionary black-box optimizers. They find ^{s*}ACM-ES and lmm-CMA-ES to be more competitive than published GP-assisted CMA-ES variants. The authors propose the Doubly Trained Surrogate CMA-ES (DTS-CMA-ES) and find that they perform similarly to the lmm-CMA-ES on many COCO benchmark problems, faring well especially on some classes of multimodal test functions. On unimodal test problems, ^{s*}ACM-ES are shown to be the most competitive algorithm, performing similarly to DTS-CMA-ES in five dimensions, but significantly outperforming it in twenty dimension. They attribute this performance advantage of the ^{s*}ACM-ES in higher dimensions to the strategy's invariance with respect to strictly increasing transformations of the objective function.

Hansen [25] presents the lq-CMA-ES ("lq" denoting linear/quadratic), which extend the lmm-CMA-ES in multiple ways. The lq-CMA-ES employ a single global surrogate, instead of multiple local models, to perform weighted regression based on a distance in function space. The surrogate model produces fitness rankings that are compared to the true ranks, rather than to rank changes from adding data, in order to determine when to accept the new model. The surrogate's optimizer is subsequently used to compute a candidate direction for the next iteration. Model complexity is adjusted depending on the number of previously evaluated points in the strategy's archive. Hansen also integrates active covariance matrix adaptation [40], which exploits information about both successful and unsuccessful candidate solutions in order to decrease variances of the mutation distribution in unpromising search space directions. Hansen utilizes the COCO benchmark to systematically compare this algorithm against several surrogate-assisted CMA-ES variants, including lmm-CMA-ES, DTS-CMA-ES, and ^{s*}ACM-ES. These comparisons reveal that despite their relative

simplicity, lq-CMA-ES perform similarly to the other surrogate model assisted algorithms and appear to surpass their comparators when function evaluation budgets are up to ten times the number of dimensions of the test problems.

Kayhani and Arnold [45] design a simple Gaussian process assisted (1+1)-ES that use their surrogate model to filter out potentially unpromising candidate solutions prior to their evaluation on the true objective function. They apply a step-size adaptation mechanism based on the implementation of the $1/5^{th}$ rule [63] by Kern et al. [46]. Specifically, they decrease the step-size if either the model-predicted or true objective function value of the offspring is inferior to that of the parent, and increase it if the offspring is successful. Yang and Arnold [78] generalize Kayhani's algorithm for increased exploitation of the model via preselection. In each iteration, the strategy generates a population of candidate solutions, uses intermediate recombination to compute the centroid of a fraction of trial points deemed best by the surrogate model, then assesses this point on the surrogate to determine whether it should be rejected or evaluated on the true objective function. They find that more intensive exploitation of the model requires different rates of change for the step-size adaptation mechanism. Like Loshchilov et al. [54], Yang and Arnold also observe that increased exploitation of the model is more often beneficial on unimodal problems.

Toal and Arnold [73] introduce the GP-CMA-ES which combine the surrogate-assisted evolution strategy of Yang and Arnold [78] with the approach to covariance matrix adaptation. They compare the performance of the resulting algorithm with competing surrogate-assisted CMA-ES on several families of parametrized unimodal test problems. They observe that the GP-CMA-ES, despite their simplicity, display at least the second best performance of all surrogate model assisted CMA-ES variants on the problems considered. Their comparisons on a set of parametrized unimodal test problems reveal that lq-CMA-ES excel predominantly on quadratic test problems, while ^{s*}ACM-ES become increasingly competitive in higher dimensions. They highlight the invariance of Loshchilov's algorithm [54] with respect to rank-preserving transformations of the objective as a source of its robustness to ill-conditioning.

Regis [64] proposes the Constrained Optimization by Radial Basis Function (COBRA) strategies that construct one RBF surrogate model for each constraint function

as well as the objective function. The author employs a piecewise logarithmic transformation (termed *plog*) introduced in an earlier work [65], which is strictly monotonic and applied to the function values of select test problems in order to render them more suitable for modelling by the RBF surrogates. Bagheri et al. [4] build on the work of Regis [64] to present SACOBRA ("SA" standing for Self-Adaptive), which among other additions, provide a mechanism for the adaptive application of the plog transformation to objective function values. In each iteration of the algorithm, the population of candidate solutions is used to train a pair of RBF surrogates, one on plog-transformed and another on untransformed function values. Models of the objective function are then employed at every t iterations to produce a fitness estimate for a new candidate solution to be added to the training archive. The inverse plog transform is applied to the fitness estimate of the RBF that is trained on the plog-transformed archive, and a model prediction error is computed by comparing both models' fitness estimate against the true function value of the new candidate solution. The model deemed superior is selected as the surrogate for the next t iterations. The authors evaluate SACOBRA on the G-function benchmark [17, 57] of constrained test problems, consuming fewer than one thousand function evaluations on all but one test problem. The algorithm of Bagheri et al. [4] toggles the application of a warp rather than attempting to find the most suitable transformation at every iteration; the latter approach is adopted in this work.

Chapter 3

Algorithm

This chapter highlights the main contribution of this thesis. The remainder of this chapter describes the proposed algorithm in detail and motivates the choice of parameters.

3.1 Warped Gaussian Process Assisted CMA-ES (wGP-CMA-ES)

Algorithm 3 shows a single iteration of the wGP-CMA-ES and closely follows the pseudo-code provided by Toal et al. [73] with modifications necessitated by the introduction of a warp. A value or model prediction that is warped with function $\Omega(\cdot;\omega)$ is distinguished with the hat $\widehat{(\cdot)}$ symbol. The state of the algorithm is fully described by candidate solution $\mathbf{x} \in \mathbb{R}^n$, step-size parameter $\sigma \in \mathbb{R}_+$, positive definite matrix $\mathbf{C} \in \mathbb{R}^{n \times n}$ referred to as the covariance matrix, vector $\mathbf{s} \in \mathbb{R}^n$ referred to as the evolution path, and an archive $\mathcal{A}$ containing m previously evaluated candidate solutions along with their objective function values. In Line 1, a collection of previously evaluated points are used to construct a warped surrogate model $\hat{f}_\varepsilon : \mathbb{R}^n \to \mathbb{R}$ that approximates the objective function near the previously evaluated points and is much cheaper to evaluate than the true objective. The details of its construction are provided in subsection 3.1.3. The algorithm continues in Line 2 by computing the positive definite principal square root of the covariance matrix $\mathbf{C}$. Lines 3 to 5 implement the variant of preselection introduced by Yang and Arnold in [78]: Line 3 samples λ mutations from a Gaussian distribution with zero mean and unit covariance; the surrogate model is then used to evaluate the λ trial points $\mathbf{y}_i = \mathbf{x} + \sigma\mathbf{A}\mathbf{z}_i$ in Line 4; and Line 5 computes the mutation centroid using rank-based weights. Line 6 generates candidate solution $\mathbf{y} = \mathbf{x} + \sigma\mathbf{A}\mathbf{z}$ using the resultant centroid $\mathbf{z}$ from the prior line and approximates its objective function value using the warped surrogate model. In Line 7, the warping function is used to transform the parent value prior to comparing it to the warped candidate solution estimate $\hat{f}_\varepsilon$ in Line 8.

Algorithm 3: Single Iteration of the wGP-CMA-ES

Required: $\mathbf{x} \in \mathbb{R}^n,\ \sigma \in \mathbb{R}_+,\ \mathbf{C} \in \mathbb{R}^{n\times n},\ \mathbf{s} \in \mathbb{R}^n,\ \mathcal{A} = \{(\mathbf{x}_k, f(\mathbf{x}_k))|k = 1, 2\ldots, m\},\ \Omega(\cdot, \omega)$

1: Build a warped surrogate model from archive $\mathcal{A}$.
2: Compute $\mathbf{A} = \mathbf{C}^{1/2}$.
3: Generate trial step vectors $\mathbf{z}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{n\times n}),\ i = 1, 2\ldots, \lambda$.
4: Evaluate $\mathbf{y}_i = \mathbf{x} + \sigma\mathbf{A}\mathbf{z}_i$ using the warped surrogate model, yielding $\hat{f}_\varepsilon(\mathbf{y}_i)$.
5: Let $\mathbf{z} = \sum_{j=1}^{\lambda} w_j \mathbf{z}_{j;\lambda}$, where $j;\lambda$ is the index of the j^{th} smallest of the $\hat{f}_\varepsilon(\mathbf{y}_i)$.
6: Evaluate $\mathbf{y} = \mathbf{x} + \sigma\mathbf{A}\mathbf{z}$ using the warped surrogate model, yielding $\hat{f}_\varepsilon(\mathbf{y})$.
7: Warp parent value $f(\mathbf{x})$ with warp function Ω, yielding $\hat{f}(\mathbf{x})$.
8: **if** $\hat{f}_\varepsilon(\mathbf{y}) > \hat{f}(\mathbf{x})$ **then**
9: Let $\sigma \leftarrow \sigma e^{-d_1/D}$.
10: **else**
11: Evaluate $\mathbf{y}$ using the objective function, yielding $f(\mathbf{y})$.
12: Add $(\mathbf{y}, f(\mathbf{y}))$ to $\mathcal{A}$.
13: Update warp parameters ω.
14: **if** $f(\mathbf{y}) > f(\mathbf{x})$ **then**
15: Let $\sigma \leftarrow \sigma e^{-d_2/D}$.
16: **else**
17: Let $\mathbf{x} \leftarrow \mathbf{y}$ and $\sigma \leftarrow \sigma e^{d_3/D}$.
18: Update $\mathbf{s}$ and $\mathbf{C}$.
19: **end if**
20: **end if**

If the candidate solution is suggested by the surrogate model to be inferior to that of the parent, then the step-size is reduced, otherwise the point is evaluated on the true objective function and added to the archive. In Line 13 the warping parameters are updated using this newly formed training archive. The true objective function value is then used to rule out false positives: if the candidate solution is indeed worse than the parent, the step-size is reduced and the iteration completed, otherwise the candidate solution replaces the parent, the step-size is increased, and the covariance matrix $\mathbf{C}$ and cumulation path $\mathbf{s}$ are updated as described in section 3.1.2. At most one

evaluation of the objective function is performed in each iteration of the algorithm. Moreover, except for the use of a warped surrogate model and the associated lines 7 and 13, Algorithm 3 is identical to that described by Toal and Arnold [73].

3.1.1 Parameter Settings

Parameter λ determines the degree of surrogate model exploitation. Yang and Arnold [78] discuss how this parameter affects the change rates for step-size adaptation. The impact of parameter λ is experimentally explored in Chapter 4 for the warped surrogate setting. Following in the steps of Toal and Arnold [73], the remaining parameters of the algorithm are classified according to the setting for parameter λ as follows:

— if $\lambda = 1$, then the only weight used in Line 5 is $w_1 = 1$. The constants which control step-size adaptation rates in Lines 9, 15, and 17 are fixed at $d_1 = 0.05$, $d_2 = 0.2$, and $d_3 = 0.6$ following the recommendation by Kayhani and Arnold [45].

— if $\lambda > 1$, then the recommendation by Hansen [24] is used to set the rank based weights in Line 5. Weights w_1 through $w_{\lfloor \lambda/2 \rfloor}$ establish a strictly decreasing sequence of positive values that sum to one and the remaining weights are set to zero. The constants that determine step-size adaptation rates are set to $d_1 = 0.2$, $d_2 = 1.0$, and $d_3 = 1.0$ based on the recommendation by Yang and Arnold [78].

Parameter D, which dampens the rates of the step-size parameter changes, is fixed at $\sqrt{1+n}$ according to previous work [45, 78].

3.1.2 Covariance Matrix Adaptation

The covariance matrix update proposed by Hansen et al. [35, 33], with some modifications provided in [24], is employed in Line 18 of the algorithm for $\lambda > 1$. The cumulation path $\mathbf{s}_c$, which implements an exponentially moving average of previously selected steps is updated according to equation 2.39:

$$\mathbf{s}_c \leftarrow (1 - c_c)\mathbf{s}_c + \sqrt{c_c(2 - c_c)\mu_{\text{eff}}}\mathbf{A}\langle \mathbf{z} \rangle, \tag{3.1}$$

where μ_{eff} is computed from rank-based weights as described by Hansen [24]. This evolution path is used in the covariance matrix update according to equation 2.41:

$$\mathbf{C} \leftarrow \mathbf{C} + c_1 \mathbf{s}_c \mathbf{s}_c^{\mathrm{T}} + c_\mu \mathbf{A} \left(\sum_{i=1}^{\lambda} w_i^\circ \mathbf{z}_{i;\lambda} \mathbf{z}_{i;\lambda}^{\mathrm{T}} \right) \mathbf{A}^{\mathrm{T}}. \tag{3.2}$$

The covariance matrix update incorporates information from present and prior covariance matrices as well as from new candidate solutions. Settings for parameters c_c, c_1, and c_μ as well as the weights w_i° are as described by Hansen [24] and preserve the positive definiteness of the covariance matrix $\mathbf{C}$ despite some of the weights w_i° being negative. In the single offspring case with $\lambda = 1$, the covariance matrix update described by Igel et al. [38] with corresponding parameter settings is utilized. The above settings are identical to those used in Toal and Arnold's GP-CMA-ES [73].

3.1.3 Surrogate Model

Gaussian processes are described in Section 2.3 of this thesis, with a detailed treatment of the subject provided by Rasmussen and Williams [62]. In order to build a surrogate model from prior observations $\{\mathbf{x}_k, f(\mathbf{x}_k)\}_{i=1}^{i=m}$, the strategy's covariance matrix $\sigma^2\mathbf{C}$ is used to compute Mahalanobis distances in the squared exponential kernel described by 2.49. In Line 1 of Algorithm 3, an $m \times m$ matrix $\mathbf{K}$ with entries $k_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ is generated using the kernel function

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(- \frac{(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}} \mathbf{C}^{-1} (\mathbf{x}_i - \mathbf{x}_j)}{2\theta_1^2 \sigma^2} \right), \tag{3.3}$$

with unit signal variance $\theta_0 = 1$, length-scale parameter $\theta_1 = 10n$, and zero noise variance $\theta_2 = 0$. The choice of the Mahalanobis distance is inspired by the work of Kern et al [47] who perform polynomial regression using the same distance measure. The use of the underlying strategy's covariance matrix in the kernel of the GP makes the surrogate model robust with respect to affine transformations of the search space with no additional expense. Parameter $\theta_0 = 1$ as θ_0 only affects uncertainty predictions during inference and this measure is not employed in Algorithm 3. Experiments with maximum likelihood estimation and cross-validation to select parameter θ_1 did not result in reliable improvements compared to the simple heuristic employed for the length-scale parameter. The constant of proportionality chosen here for θ_1 is greater than that used by Toal and Arnold [73], who set $\theta_1 = 8n$; this is motivated by

empirical observations of larger constants of proportionality leading to lower model prediction error in the limit of numerical precision. Kernel parameter setting $\theta_3 = 0$ is reasonable as observations are not corrupted by noise.

The warping function family is chosen to be the power functions with two degrees of freedom: an exponent for power transformations, and a constant for translations according to:

$$\Omega(f;\omega) = (f-q)^p; \quad \omega \in \{(p,q)|q \leq f_{1;m}, p \in \mathbb{R}_+, q \in \mathbb{R}\}. \tag{3.4}$$

Warp parameters ω are selected by maximizing the ranking consistency measured by Kendall's correlation coefficient τ between predicted $\hat{\mathbf{f}}_{\text{pred}}(\omega)$ and true $\mathbf{f}$ function values of points in the archive:

$$\omega^* = \text{argmax}_\omega \, \tau\big(\mathbf{f}, \hat{\mathbf{f}}_{\text{pred}}(\omega)\big). \tag{3.5}$$

For a given ω, $\hat{\mathbf{f}}_{\text{pred}}$ is computed by means of LOO-CV according to equation 2.55, with kernel function given by equation 3.3. This approach to selecting ω bypasses the need to compute an inverse warp and thus foregoes the requirement to define a bijective map on $\mathbb{R}$. The procedure employed to find an appropriate warp is outlined in Algorithm 4; we use the compact notation $\tau(\omega)$ in lieu of that used in equation 3.5 to denote rank correlations.

Algorithm 4 implements a simple variant of two-dimensional coordinate search in the parameters $\omega = (p, q)$. The success criterion of the warp optimization process controls the frequency of warp parameter updates and is defined as a rank correlation value of $\tau \geq 0.9$. Based on empirical investigation across the test problems of varying dimension presented in the next section, this value was calibrated to balance higher model accuracy and the greater update frequency observed to be necessary in higher dimensions with the computational overhead of running Algorithm 4. In each invocation of Algorithm 4, if warp parameters are deemed to be poor, then k_q points are enumerated linearly in the interval $[f_{1;m}, f_{2;m}]$, where m is the number of training points in the archive and the $i;j$ convention refers to the i^{th} smallest of j elements. When rank correlations for each shift component of the warp parameter in the specified range are enumerated, the maximizer of τ is selected; if the resulting warp is unsuccessful (i.e. rank correlation $\tau < 0.9$), then k_p power values are enumerated logarithmically in the interval $[10^{-1}, 10^{1}]$ and the corresponding maximizer of τ

is selected. If neither adjustment is successful, then no warp is used by the surrogate model by defaulting to $\omega = (1, 0)$. The number of enumeration points $k_p = k_q = 101$ are tuned empirically, balancing the trade-off between computational expense and precision. The range for power values p is constrained to within two orders of magnitude in order to prevent numerical accuracy issues. The interval for shift values is adapted based on the difference measure between the two smallest function values in the training set. This helps to maintain an appropriate enumeration density for the shift variable throughout the run of the optimizer.

Algorithm 4: Warp Update Routine

Required: Archive of points $\mathcal{A} = \{(\mathbf{x}_k, f(\mathbf{x}_k))|k = 1, 2 \ldots, m\}$, function value predictions of warped GP model $\hat{\mathbf{f}}_{\text{pred}}$, warp parameter ω,

if $\tau(\omega) \geq 0.9$ **then**
    Keep the current warp.
else
    Search for a better shift q by enumeration and set $\widetilde{\omega} \leftarrow (p, q_{new})$.
    if $\tau(\widetilde{\omega}) \geq 0.9$ **then**
        Update warp $\omega \leftarrow \widetilde{\omega}$.
    else
        Search for a better power p by enumeration and set $\widetilde{\omega} \leftarrow (p_{new}, q)$.
        if $\tau(\widetilde{\omega}) \geq 0.9$ **then**
            Update warp $\omega \leftarrow \widetilde{\omega}$.
        else
            Default to no warp $\omega \leftarrow (1, 0)$.
        end if
    end if
end if

Selection of the class of power functions for Ω was driven by the need for a simple, non-linear transformation with low computational cost of optimizing model parameters. While the choice of Ω is motivated by the requirement to represent a complex set of transformations, it is tempered by the number free parameters necessitated by the model as optimization of these parameters can be computationally expensive and numerically tedious. Gradient descent approaches based on finite differencing

methods are slow and expensive while quasi-Newton methods can suffer from premature convergence. Popular numerical solvers such as *Matlab's* `fminsearch`, which implements Nelder-Mead's simplex method [59], can fail to consistently provide good warps. As model expressiveness and computational cost are conflicting requirements, a balance needs to be struck. The `tanh` function proposed by Snelson et al. [69] has four degrees of freedom and is thus more tedious to optimize. Power functions allow for non-linear transformations while keeping a small number of warping parameters. Furthermore, the formulation of the optimization error criterion given by equation 3.5 as a rank-based metric circumvents the need to compute the inverse warp Ω^{-1}.

Once a suitable warp is identified, function values in the archive are warped $\hat{\mathbf{f}} = \Omega(\mathbf{f};\omega)$ to perform predictions. In order to evaluate these models in Lines 4 and 6 of Algorithm 3 at a point $\mathbf{y} \in \mathbb{R}^n$, an $m \times 1$ vector $\mathbf{k}$ with entries $k_i = k(\mathbf{x}_i, \mathbf{y})$ is computed. The warped model prediction becomes

$$\hat{f}_\varepsilon(\mathbf{y}) = \hat{f}(\mathbf{x}) + \mathbf{k}^{\mathrm{T}}\mathbf{K}^{-1}\hat{\mathbf{f}}, \tag{3.6}$$

where $\hat{\mathbf{f}}$ is an $m \times 1$ vector with warped entries $\hat{f}_i = \hat{f}(\mathbf{x}_i) - \hat{f}(\mathbf{x})$. This implies that the Gaussian process's prior mean is set to the warped parental function value. In contrast to Bajer et al. [5] and consistent with Toal and Arnold [73], the uncertainty prediction of the Gaussian process is not used in our algorithm.

The maximum length of the archive is constrained to the $m = 6n$ points most recently evaluated on the objective function. This helps to avoid increasing computational costs with a growing archive size, resulting from the need to invert matrix $\mathbf{K}$ when building surrogate models. Toal and Arnold [73] construct surrogate models based on a maximum of the most recent $m = (n+2)^2$ points from the archive in order to acquire an amount of information that scales linearly in the number of covariance matrix variables. While we find their motivation compelling, we observe larger model sizes to decrease model accuracy in the wGP-CMA-ES for $\lfloor m/n \rfloor > 6$. For instance on the sphere, archive lengths larger than this frequently cause convergence failures as the fast progress rate of the wGP-CMA-ES quickly leads to an ill-conditioned kernel whose inversion causes numerical accuracy issues.

Figure 3.1 illustrates select convergence plots of the wGP-CMA-ES on three function value transformed spheres for $n = 16$. The sub-figure on the right illustrates

similar running times under linear convergence of the algorithm for α-adjusted termination criteria $f_{\text{target}} = (10^{-8})^{\alpha}$. The top left sub-figure confirms that the warp search mechanism is working as expected, with rank correlations of $\tau \geq 0.9$. The bottom left sub-figure depicts the procedure's recovery of a warp of approximately $p = 2/\alpha$ on the test functions. In effect, the warping mechanism is shown to convert the optimization of a quartic (or linear) sphere into that of a quadratic one. This observation is consistent with the findings of Yang and Arnold [78], whose GP-assisted ES, as compared to an unassisted model, enjoyed the greatest speed-ups on the quadratic sphere.

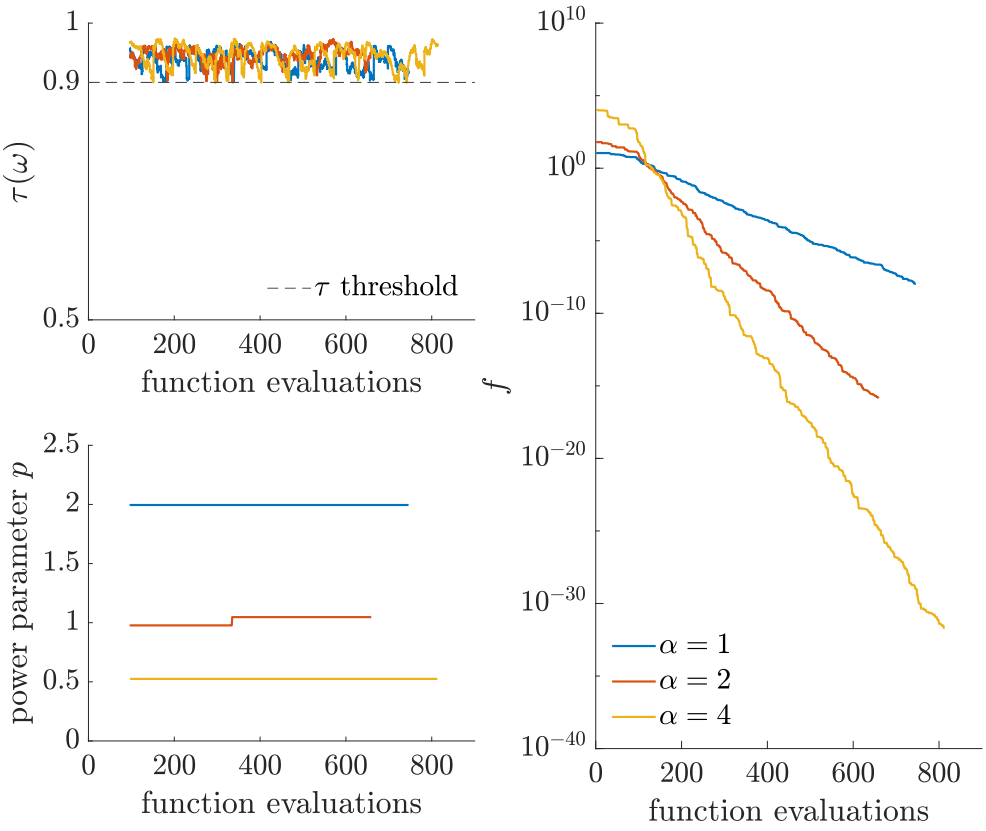


Figure 3.1: Convergence plots of the wGP-CMA-ES on select 16−dimensional spheres under varying increasing transformations parametrized by α. The sub-figure on the right shows roughly similar running times of the wGP-CMA-ES on the three different spheres. The sub-figure on the top left displays run-time rank correlation values $\tau(\omega)$ of no less than 0.9, as stipulated by Algorithm 4. The bottom left sub-figure demonstrates the warp search strategy of Algorithm 4 recovers the setting of approximately $\frac{2}{\alpha}$ for power parameter p on each test function.

3.1.4 Initialization and Start-Up

Covariance matrix $\mathbf{C}$ is initialized to the identity matrix; evolution path $\mathbf{s}_c$ is set to the zero vector. The initialization of $\mathbf{x}$ and σ are problem-dependent. As with the algorithm of Toal and Arnold [73], minimum archive length is set to $2n$ to ensure that surrogate models are only constructed once sufficient data is available. This is achieved by defaulting $\lambda = 1$ and $\hat{f}_\varepsilon(\cdot)$ to $-\infty$ while the archive contains fewer than $2n$ points. Once $2n$ points have been recorded in the archive, the initial warp parameter ω_0 is selected by enumerating roughly one thousand points on the $2-$dimensional parameter grid using $k_p = k_q = 31$. This setting is a compromise between the accuracy of ω_0 and its computation cost. A visualization of this procedure is provided in Figure 3.2. The surrogate model is employed immediately after this point. During the start-up phase, the algorithm is equivalent to the model-free (1+1)-CMA-ES, whereby the parameters controlling step-size adaption rate are set to $d_2 = -0.2$ and $d_3 = 0.8$.

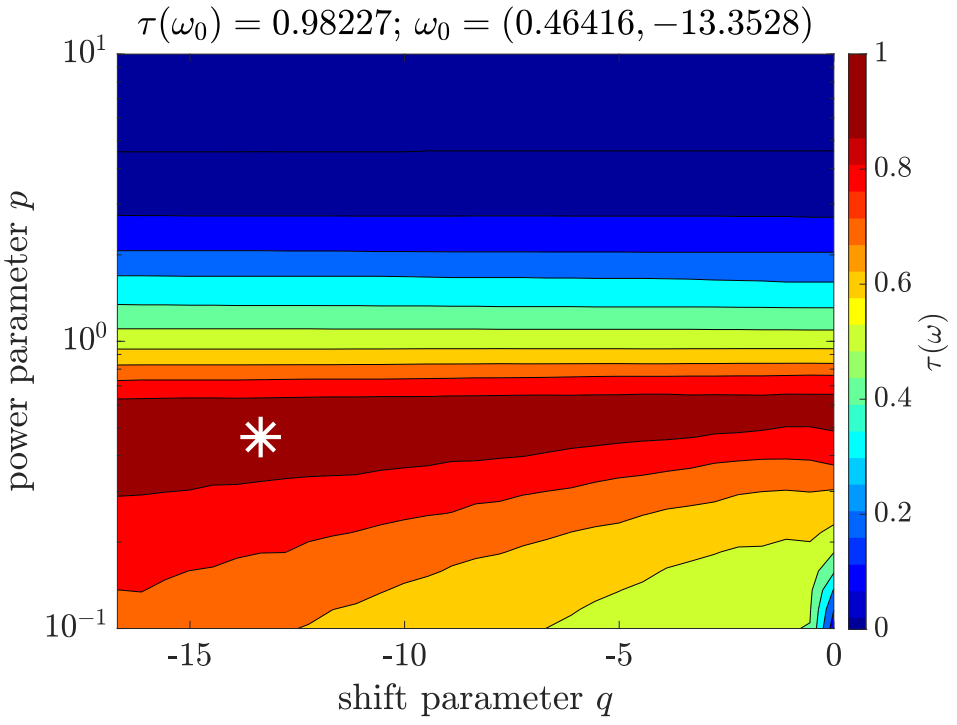


Figure 3.2: Selection of initial warping parameters $\omega_0 = (p, q)$ by enumeration on the 8-dimensional quartic sphere. After function values in the archive $\mathcal{A}$ are subtracted by the parental function value, $k_q = 31$ points are enumerated linearly such that $f_{1;m} \leq q \leq f_{2;m}$, where the subscript $i; j$ denotes the i^{th} best (smallest) of j elements. Likewise, $k_p = 31$ points are enumerated logarithmically in the interval $p \in [10^{-1}, 10^1]$. The star (*) symbol locates the parameter pair corresponding to the empirical maximum of rank correlations $\tau(\omega)$.

Chapter 4

Evaluation

This section presents an experimental evaluation of the performance of wGP-CMA-ES and compares it to several related algorithms. Section 4.1 highlights the comparator algorithms, Section 4.2 describes the experimental setting, and Section 4.3 reports the empirical results along with a discussion of the findings.

4.1 Comparator Algorithms

We examine the performance of wGP-CMA-ES with $\lambda \in \{1, 10, 20\}$ as well as that of four other algorithms:

— CMA-ES Version 3.61.beta; we use the *Matlab* implementation without surrogate model assistance provided by Hansen at `cma.gforge.inria.fr`.

— ^{s*}ACM-ES Version 2; we use the *Matlab* implementation by Loshchilov [54] at `loshchilov.com`.

— lq-CMA-ES Version 3.0.3; we use the *Python* implementation by Hansen [25] on `GitHub`.

— GP-CMA-ES; we use the *Matlab* implementation provided Toal and Arnold [73]

Algorithm specific parameters for all but the GP-CMA-ES were fixed at their default settings. The GP-CMA-ES [54] were run with a linear memory size $m = 8n$ instead of their default $m = (n+2)^2$, as well as a length scale of $8 * n$. As the goal of empirical evaluations is to examine the effect of warping, this setting is selected to rule out the impact of memory size on algorithm performance. All evolution strategies, except for lq-CMA-ES, are invariant with respect to rotations of the search space coordinates. While both CMA-ES and ^{s*}ACM-ES are invariant to strictly monotonic transformations of objective function values, GP-CMA-ES and lq-CMA-ES are not. We exclude DTS-CMA-ES from the list of comparators as its advantages are primarily

on multimodal test functions. Moreover, we consider the performance of lq-CMA-ES to surpass that of lmm-CMA-ES and so omit the latter from our experiments.

4.2 Test Environment

We employ the following four families of parametrized unimodal test problems in order to assess the relative strengths of the five approaches considered:

— Spherically symmetric functions corresponding to f_1 in the BBOB noiseless test-bed [31] when parameter $\alpha = 2$.

— Convex quadratic functions for $\beta \in \{10^2, 10^4, 10^6\}$, corresponding to f_2 of the BBOB noiseless benchmark [31] at parameter setting $\{\alpha = 2, \beta = 10^6\}$.

— Generalized quartic functions for $\gamma \in \{10^0, 10^1, 10^2\}$, corresponding to f_8 in the BBOB noiseless benchmarks [31] at parameter setting $\{\alpha = 2, \gamma = 100\}$.

— Power sum functions corresponding to f_{14} in the BBOB noiseless test-bed [31].

Test problems are not rotated. As problem separability may confer an advantage only in the case of the lq-CMA-ES, and the goal of experiments is to study the effect of warping, it is reasonable to employ the separable version of the functions in algorithm evaluation. The optimal function value for all of the functions considered is zero. All algorithms are tested on each function in dimensions $n \in \{2, 4, 8, 16\}$. Values of α ranged between $\alpha \in [1, 4]$ with 9 logarithmically-spaced values for all but the power sum family of functions.

Runs are initialized by sampling starting points uniformly at random from the interval $\mathbf{x} \in [-4, 4]^n$. The step-size parameter was initialized to $\sigma = 2$ for all runs of the algorithms. Termination criteria for all strategies under consideration were set to an objective function value no larger than $\left(10^{-8}\right)^{\alpha/2}$. As we consider function value based run times in a fixed target termination scenario, the terminal objective function value must be transformed according to α in order to ensure fair comparisons between runs. This transformation of the terminal function value can be avoided if the stopping criterion is reformulated in terms of the distance (L_2 norm) of the solution to the function's global optimizer; however we opted for the function-value based metric

in accordance with related work [51, 25]. The default stopping criteria of the CMA-ES, lq-CMA-ES, and ^{s*}ACM-ES were adjusted to ensure that the aforementioned objective function value would be the first enacted termination criterion. For $n \geq 4$, the generalized quartic functions exhibit a second local minimizer. Runs that converged to this merely local minimizer were discarded.

4.2.1 Rationale for the Choice of Objective Functions

The choice of functions is motivated by the goal of studying the behaviour of the proposed warping mechanism and to evaluate the potential benefit of its integration into the GP-CMA-ES. We wish to systematically examine the scaling of algorithm performance with respect to problem difficulties including function value transformations, ill-conditioning due to axis scaling, and dimensionality. A secondary goal is to observe how performance is affected with increasing exploitation of the surrogate model. The wGP-CMA-ES is expected to perform at least as well as the GP-CMA-ES while being more robust to function value transformations. In order to contextualize the advantage of the proposed approach, we provide a comparison to related surrogate-assisted CMA-ES variants deemed competitive on the COCO Benchmark [30].

Spherically symmetric functions form a natural baseline for the evaluation of the wGP-CMA-ES as they are the simplest of function families. The sphere's isotropic level sets do not necessitate the learning of objective function axis scales and thus eliminate the need for covariance matrix adaptation. As compared with the GP-CMA-ES, these functions test only the ability of wGP-CMA-ES to learn a suitable warp; hence poor performance on the sphere is expected to generalize to more difficult problems. If warping works as expected, then performance on all spheres should be invariant under strictly increasing function value transformations, yielding a constant speed-up over the CMA-ES for varying α.

The primary challenge posed by ellipsoids is their ill-conditioning, which necessitates learning the problem's axis scales. The degree of ill-conditioning is controlled by parameters α and β, with the functions being globally quadratic at $\alpha = 2$ posing less difficulty for algorithms that internally build quadratic surrogate models. An objective on this test function is to rule out any major interference of the warping procedure with the mechanism of covariance matrix adaptation on a problem for

which the CMA-ES can learn the axis scales during its run. After this adaptation phase, algorithm behaviour is expected to be identical to that on the sphere family of functions, as is observed with the CMA-ES.

Generalized quartic functions combine the properties of being less than perfectly conditioned and being non-quadratic. The Hessian matrix of these functions changes throughout the runs of the algorithms and thus optimizers need to constantly relearn the problem's axis scales. This class of problems is more difficult than the convex quadratic family as the covariance matrix is never fully learned throughout the run of the ES. It is thus important to test whether the effectiveness of the warping mechanism is compromised by this demand. As with the convex quadratic functions, parameter γ is varied to gradually intensify the severity of this challenge for a range of function value transformations parametrized by α. While CMA's robustness to axis scaling is expected to provide similar conditions of operation for the warping mechanism, this assumption must be tested explicitly.

The family of power sum functions are highly ill-conditioned, with the degree of ill-conditioning increasing with the approach toward their optimum. As with the generalized quartic functions, the axis scales on this problem must be constantly relearned by the optimizer. We examine whether warping provides an advantage this test function without function value transformations. While the primary challenge posed by this problem is one of search space adaptation, we explore whether a warped GP confers an advantage over the non-warped surrogate of the GP-CMA-ES.

4.3 Results

Fifteen runs of each algorithm were conducted for every problem instance. The figures plot the number of function evaluations divided by dimension with respect to function value transformations of the respective objective functions. The lines in each figure connect medians and the error bars reflect the full range of values observed on the respective problem instance. For strategies where multiple population sizes were tested, various line styles are used to distinguish each setting (solid for $\lambda = 1$, dashed for $\lambda = 10$, and dotted for $\lambda = 20$). Results are presented and discussed in the following subsections.

4.3.1 Sphere Functions

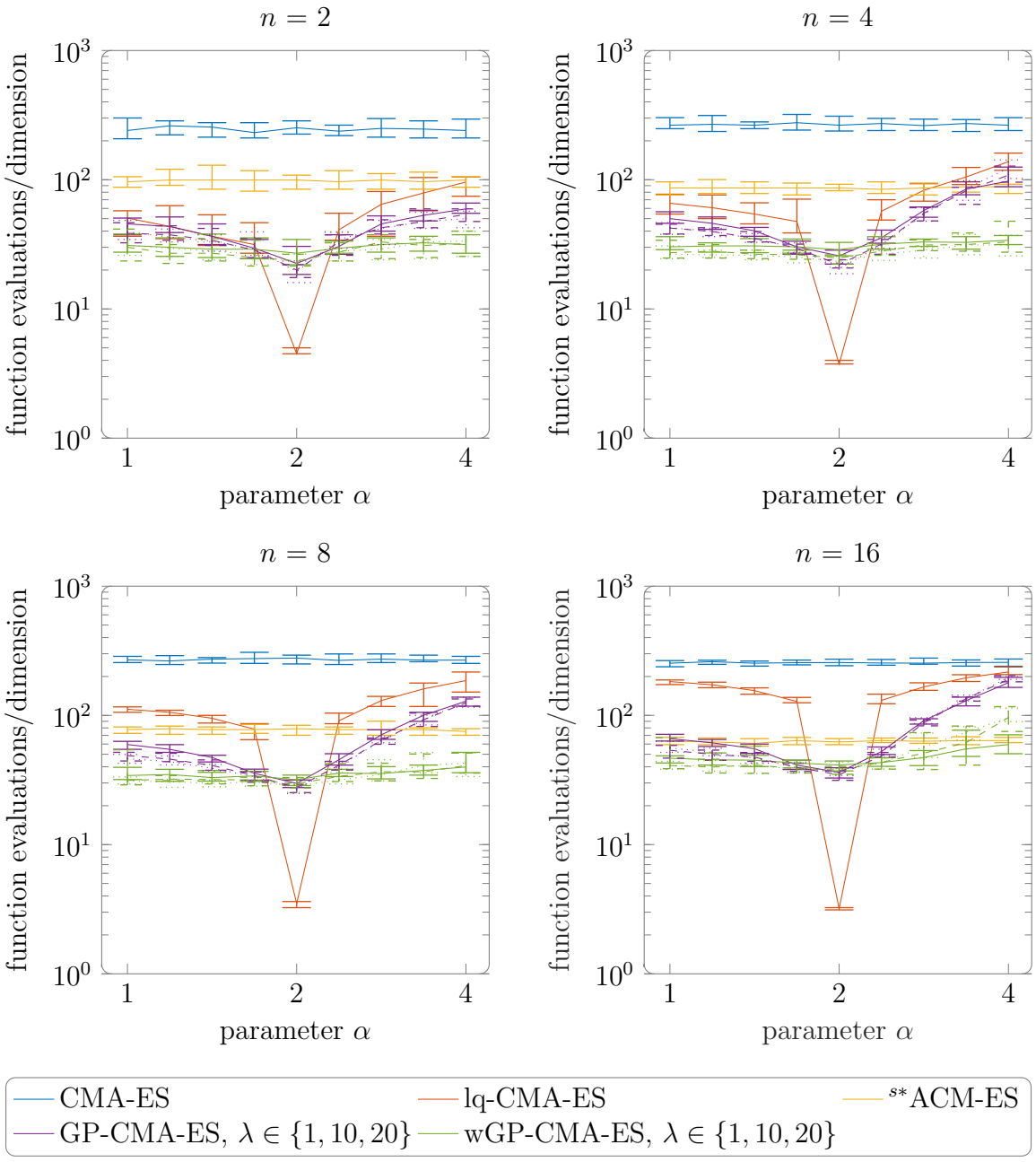


Figure 4.1: Number of objective function evaluations per dimension required to optimize the sphere functions with parameter $\alpha \in [1, 4]$. The lines connect median values; the error bars reflect the full range of values observed for the respective algorithms.

Figure 4.1 shows the number of objective function evaluations per dimension required to optimize the sphere functions for $\alpha \in [1, 4]$. CMA-ES without surrogate model assistance show invariance with respect to strictly monotonic transformations of the objective functions and require the largest number of function evaluations as expected. The ^{s*}ACM-ES also display invariance to rank-preserving function value transformations and exhibit a constant speed-up that rises with increasing dimension of the problem being tested. Both GP-CMA-ES and lq-CMA-ES are observed to perform best when $\alpha = 2$, with the lq-CMA-ES significantly surpassing all other algorithms on this test problem as the lq-CMA-ES' surrogate model perfectly matches the objective function. The speed-ups attained by both of these algorithms decline as function value transformations diverge from $\alpha = 2$, with the GP-CMA-ES having a significant edge over lq-CMA-ES except in the case of the quadratic sphere. As in Toal and Arnold [73], we attribute the large performance advantage of the lq-CMA-ES over (w)GP-CMA-ES for $\alpha = 2$ to both the lower model accuracy of Gaussian processes on quadratic functions, as well as to the manner in which the respective surrogates are exploited. For $n \geq 8$ and α in excess of 2, the ^{s*}ACM-ES eventually overtake the GP-CMA-ES. This observation is consistent with that by Toal and Arnold [73]. The performance advantage of the wGP-CMA-ES over that of GP-CMA-ES is explained by the effect of warping.

The wGP-CMA-ES enjoy a near-constant speed-up over the ^{s*}ACM-ES for the α values tested, highlighting the utility of warping in regaining invariance lost on the function value based Gaussian process surrogate. The algorithm using a batch size of $\lambda = 1$ performs the best, or similarly to the most competitive algorithm, for all $\alpha \neq 2$. In most cases, larger batch sizes yield a minor speed-up on the wGP-CMA-ES, with increasing λ causing a small slow-down on larger α values in 16 dimensions. We hypothesize that this is due to sub-optimal parameter settings that can be remedied with further tuning of the algorithm. Rates of step-size adaptation for the GP-CMA-ES, which were adopted by the warped variant, were calibrated for a quadratic memory size with respect to n. Thus further adjustments of these vales are expected to generate improvements in higher dimensions. The slight difference in performance relative to GP-CMA-ES on the quadratic sphere is attributable to imperfect optimization of the warp parameters throughout the run of the wGP-CMA-ES.

4.3.2 Ellipsoid Functions

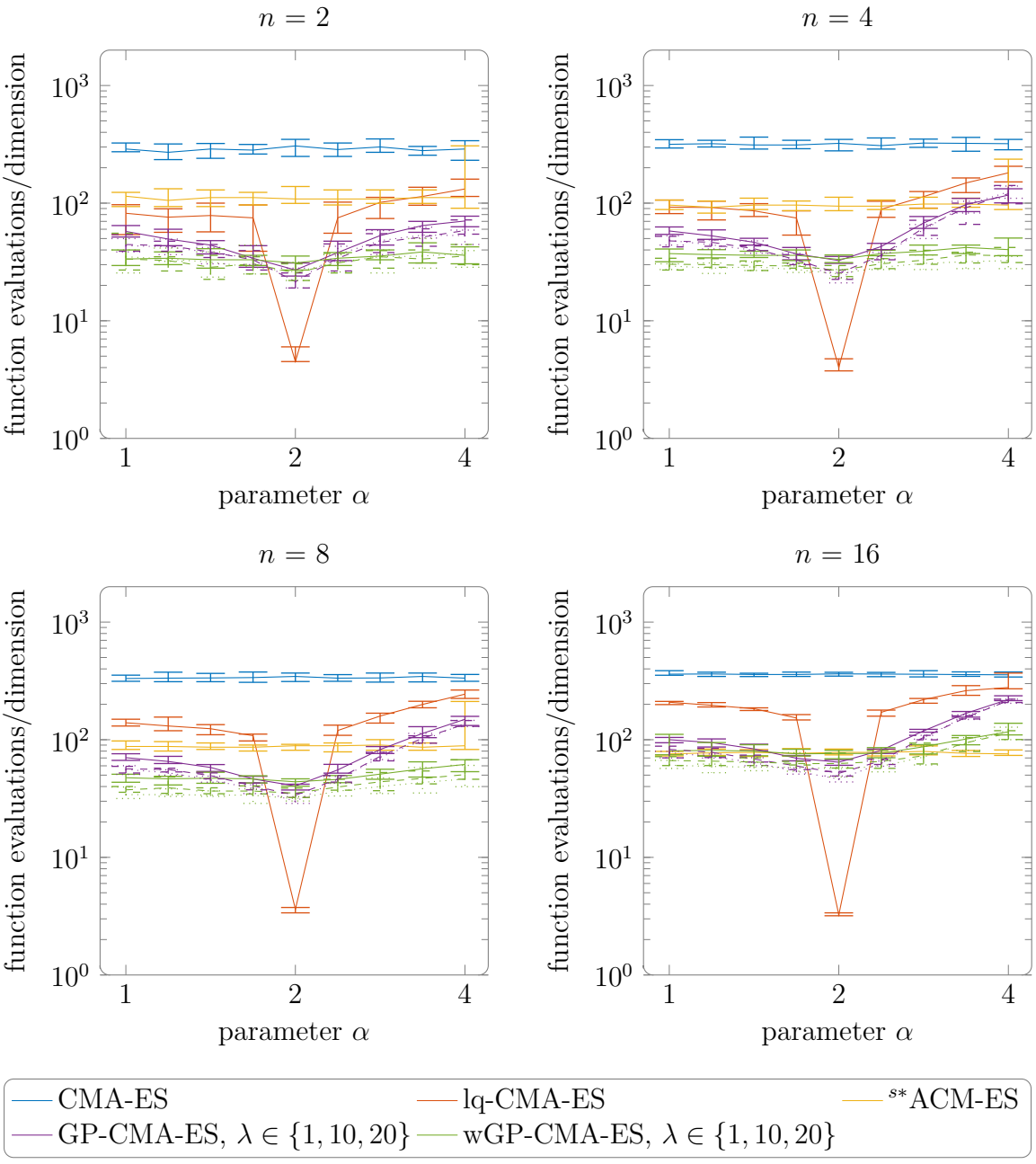


Figure 4.2: Number of objective function evaluations per dimension required to optimize the function value transformed ellipsoid functions with parameters $\alpha \in [1, 4]$ and $\beta \in 10^2$. The lines connect median values; the error bars reflect the full range of values observed for the respective algorithms.

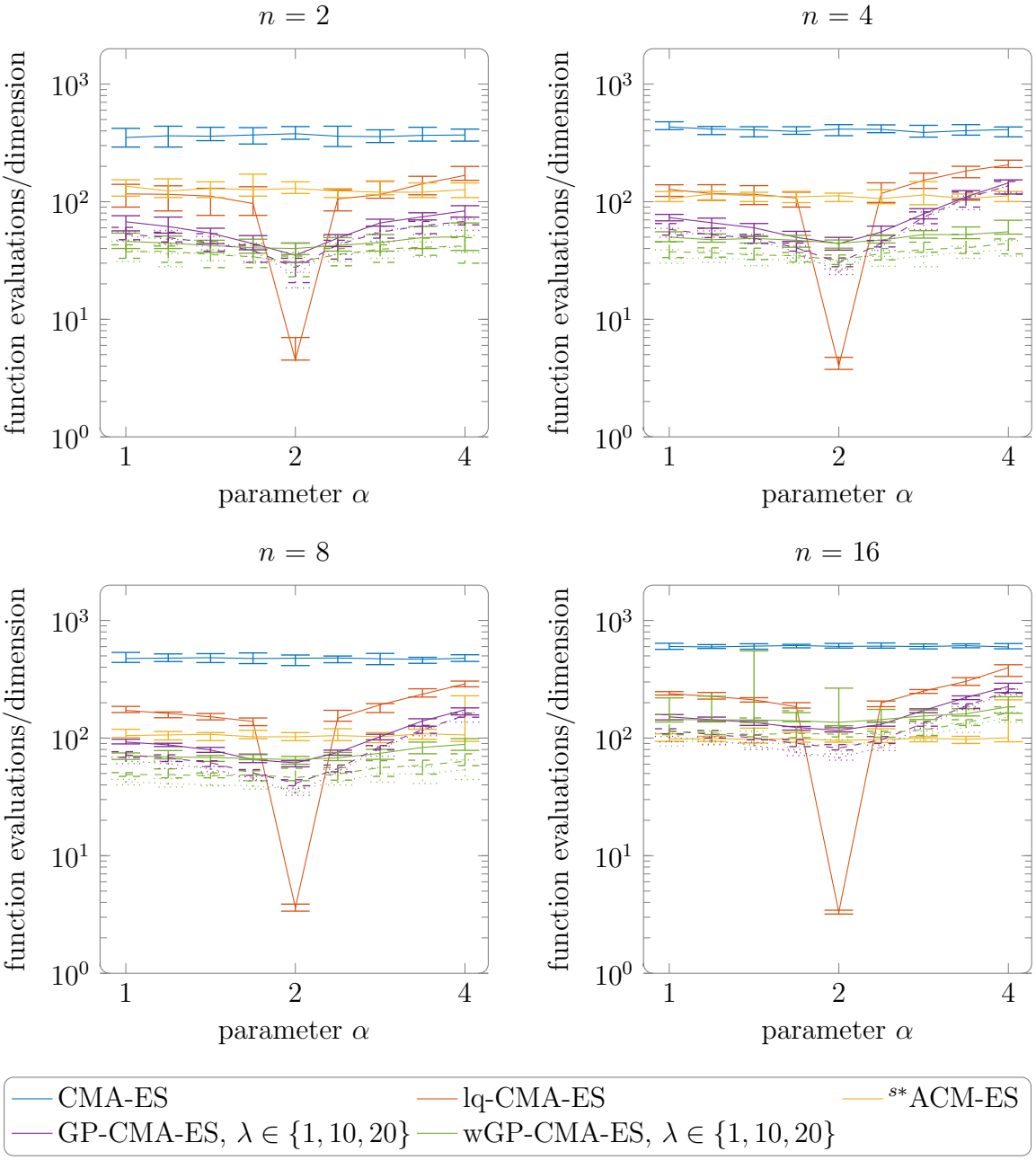


Figure 4.3: Number of objective function evaluations per dimension required to optimize the function value transformed ellipsoid functions with parameters $\alpha \in [1, 4]$ and $\beta \in 10^4$. The lines connect median values; the error bars reflect the full range of values observed for the respective algorithms.

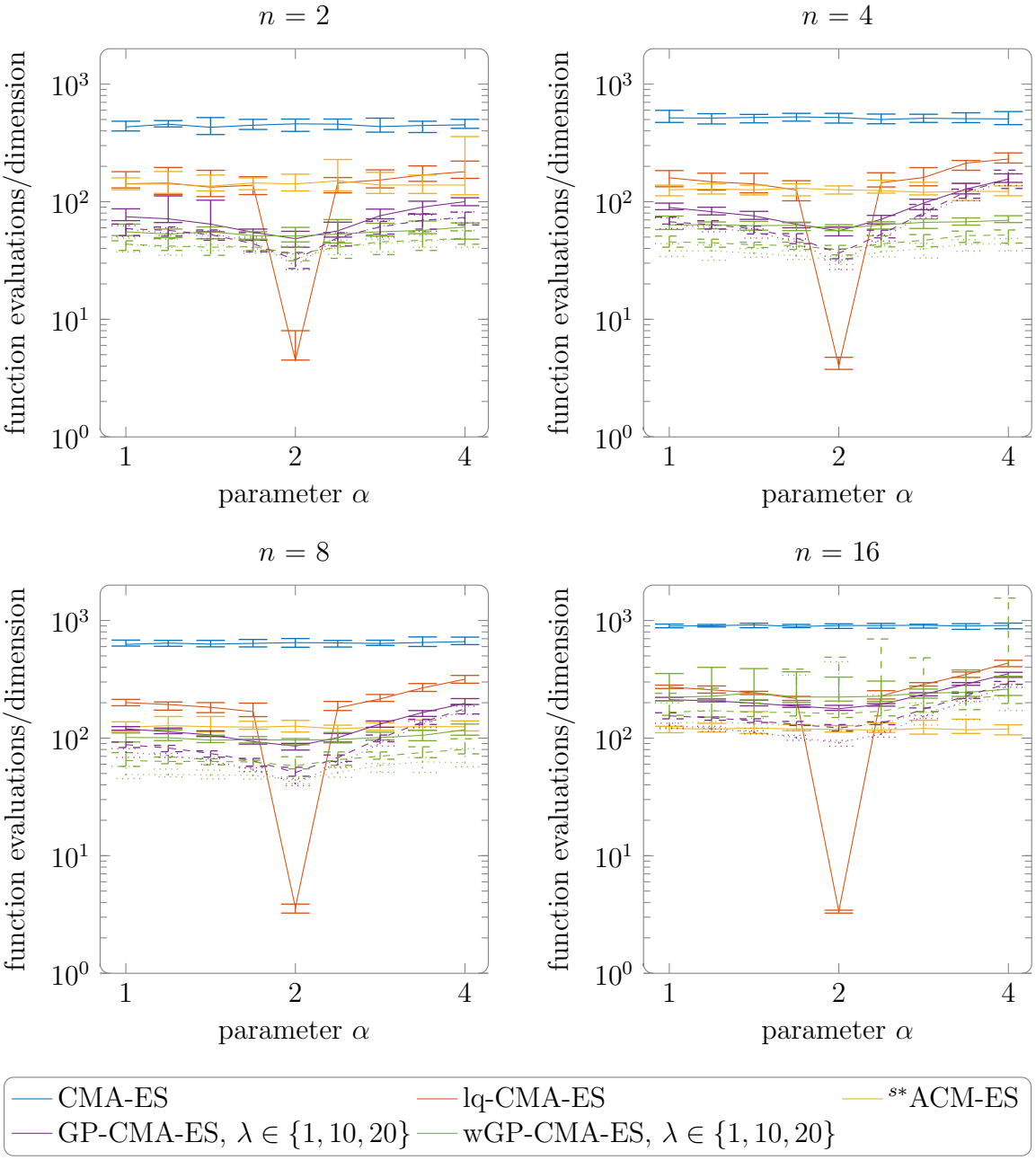


Figure 4.4: Number of objective function evaluations per dimension required to optimize the function value transformed ellipsoid functions with parameters $\alpha \in [1, 4]$ and $\beta \in 10^6$. The lines connect median values; the error bars reflect the full range of values observed for the respective algorithms.

Figures 4.2 to 4.4 plot the number of objective function evaluations divided by dimension for function value transformed ellipsoids with parameter $\alpha \in [1, 4]$. Each figure corresponds to a different degree of ill-conditioning controlled by parameter

$\beta \in \{10^2, 10^4, 10^6\}$. In Figure 4.2, where the ellipsoid being optimized is mildly ill-conditioned, the performance advantage of the ^{s*}ACM-ES over the CMA-ES widens with increasing dimension. This is due to the well-calibrated exploitation of their surrogate model and is notable as the performance advantage of the comparators over the CMA-ES either remains unchanged, or slightly deteriorates with increasing dimension. As with Toal and Arnold [73], more intensive exploitation of the surrogate model with larger population sizes is not seen to be detrimental and appears to especially benefit performance of both GP-CMA-ES and the warped variant with increasing β.

As with the sphere, the lq-CMA-ES are able to locate the function optimizer within the fewest number of objective function iterations for $\alpha = 2$. The performance of the algorithm appears insensitive to the Ellipsoid's degree of ill-conditioning controlled by parameter β as the median number of function evaluations for a given dimension and a fixed parameter α remains constant across Figures 4.2 to 4.4. We attribute this to the internally quadratic models which match the quadratic objective function and are used almost exclusively to locate the optimizer once sufficiently many training points have been evaluated on the true objective. Although for $n = 2$ lq-CMA-ES fare similarly or better than the ^{s*}ACM-ES, their sensitivity to rank-preserving transformations of the objective function constrains their performance between ^{s*}ACM-ES on the lower and CMA-ES on the upper limits of the range of parameter $\alpha \neq 2$ under consideration.

The GP-CMA-ES generally exhibit an advantage over ^{s*}ACM-ES, but this advantage fades with $n \geq 8$ and $\alpha > 2$. We ascribe this diminishing performance advantage to three factors. First the invariance of ^{s*}ACM-ES' surrogate to order-preserving transformations of the objective function ensures that their performance is consistent for all α values considered. This is confirmed as warping, which is intended to regain some of this invariance for the value based Gaussian process model, is seen to significantly narrow the gap in performance between the ^{s*}ACM-ES and the GP-CMA-ES. Secondly, the ^{s*}ACM-ES adapt their population size with respect to the dimensionality of the problem being optimized which enable it to more fully exploit their surrogate model. As can be seen with (w)GP-CMA-ES, using larger batch sizes reduces the number of objective function evaluations. Third, as was confirmed by

Toal and Arnold [73], the GP-CMA-ES with an archive size that is quadratic relative to problem dimensionality, may consistently prevail over ^{s*}ACM-ES on the highly ill-conditioned ellipsoids. Because in our experiments both GP-CMA-ES and wGP-CMA-ES employed archive lengths linear in the number of dimensions, we would expect larger training set sizes to be generally beneficial on this problem in higher dimensions.

In dimension $n = 16$, the wGP-CMA-ES exhibit a more variable range in performance across across all α values tested, with the median number of required objective function evaluations being slightly higher than that of GP-CMA-ES for each batch size at values of $\alpha < 3$. We attribute this to the update frequency of the shift parameter in the warp optimization process, where the negative impact of a poor shift is more pronounced in higher dimensions.

4.3.3 Generalized Quartic Functions

The ^{s*}ACM-ES show a constant speed-up relative to the CMA-ES, which do not use surrogate models. Variations from the median are much larger for this class of functions than other test problems considered. The invariance of the ^{s*}ACM-ES to strictly monotonic transformations of the objective function is again observed on this family of functions, but this and all comparators illustrate a sensitivity to ill-conditioning as a result of larger γ values. The lq-CMA-ES perform better than ^{s*}ACM-ES in lower dimensions, but their advantage decreases with increasing dimension and vanishes for $n = 16$. In the low-conditioned generalized quartic function with $\gamma = 10^0$, the lq-CMA-ES display the familiar dip at $\alpha = 2$ and perform similarly to the GP-assisted comparators for dimensions $n \in \{4, 8\}$. However this advantage deteriorates with progressively larger values of γ, paralleling its decrease in sensitivity to parameter α.

Figures 4.5 to 4.7 plot the number of objective function evaluations divided by dimension for function value transformed generalized quartic functions with parameter $\alpha \in [1, 4]$. Each figure corresponds to a different value of parameter $\gamma \in [10^0, 10^2]$, with the original Rosenbrock function located in the middle of the third figure.

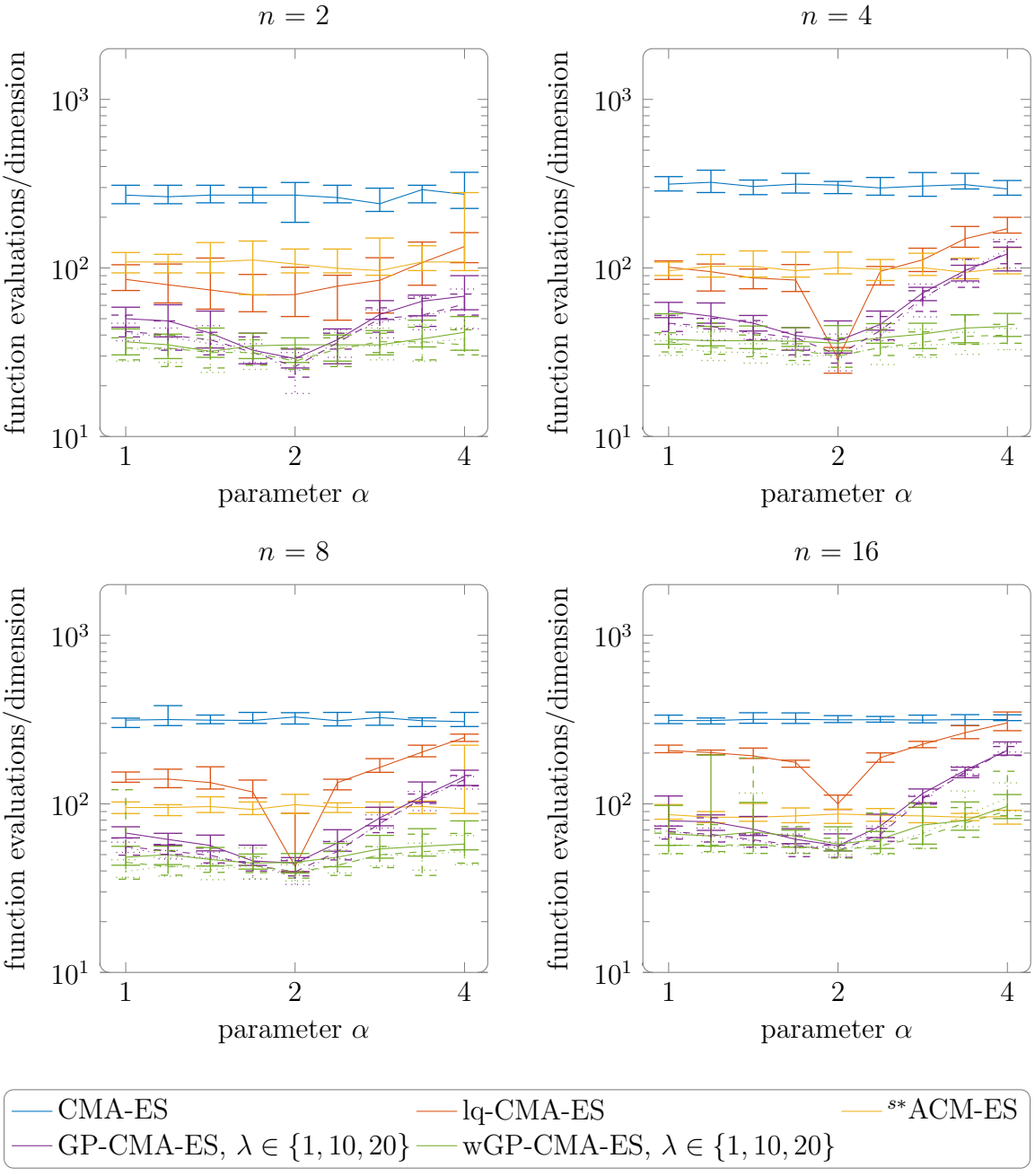


Figure 4.5: Number of objective function evaluations per dimension required to optimize the generalized quartic functions with parameters $\alpha \in [1, 4]$ and $\gamma \in 10^0$. The lines connect median values; the error bars reflect the full range of values observed for the respective algorithms.

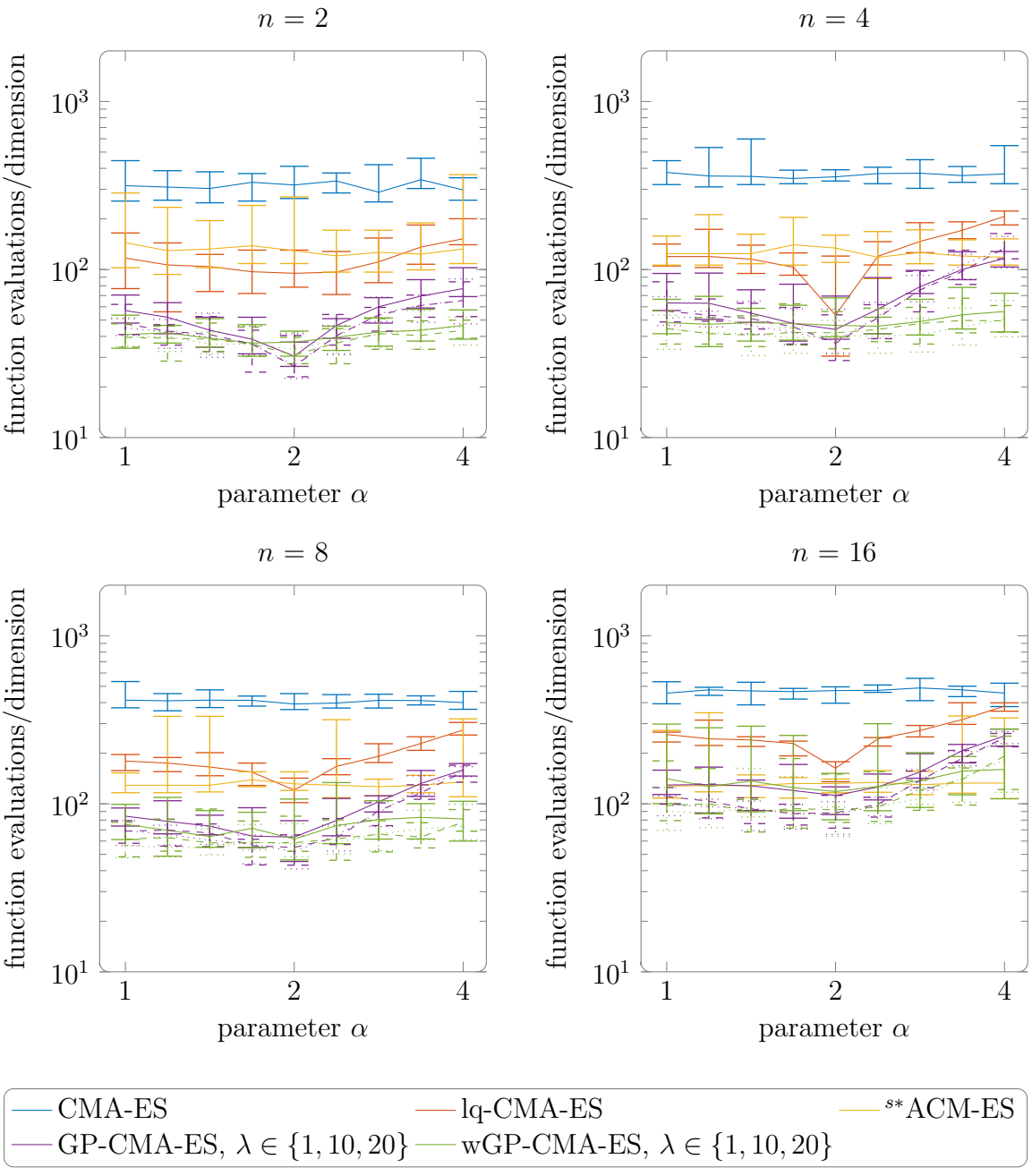


Figure 4.6: Number of objective function evaluations per dimension required to optimize the generalized quartic function functions with parameters $\alpha \in [1, 4]$ and $\gamma \in 10^1$. The lines connect median values; the error bars reflect the full range of values observed for the respective algorithms.

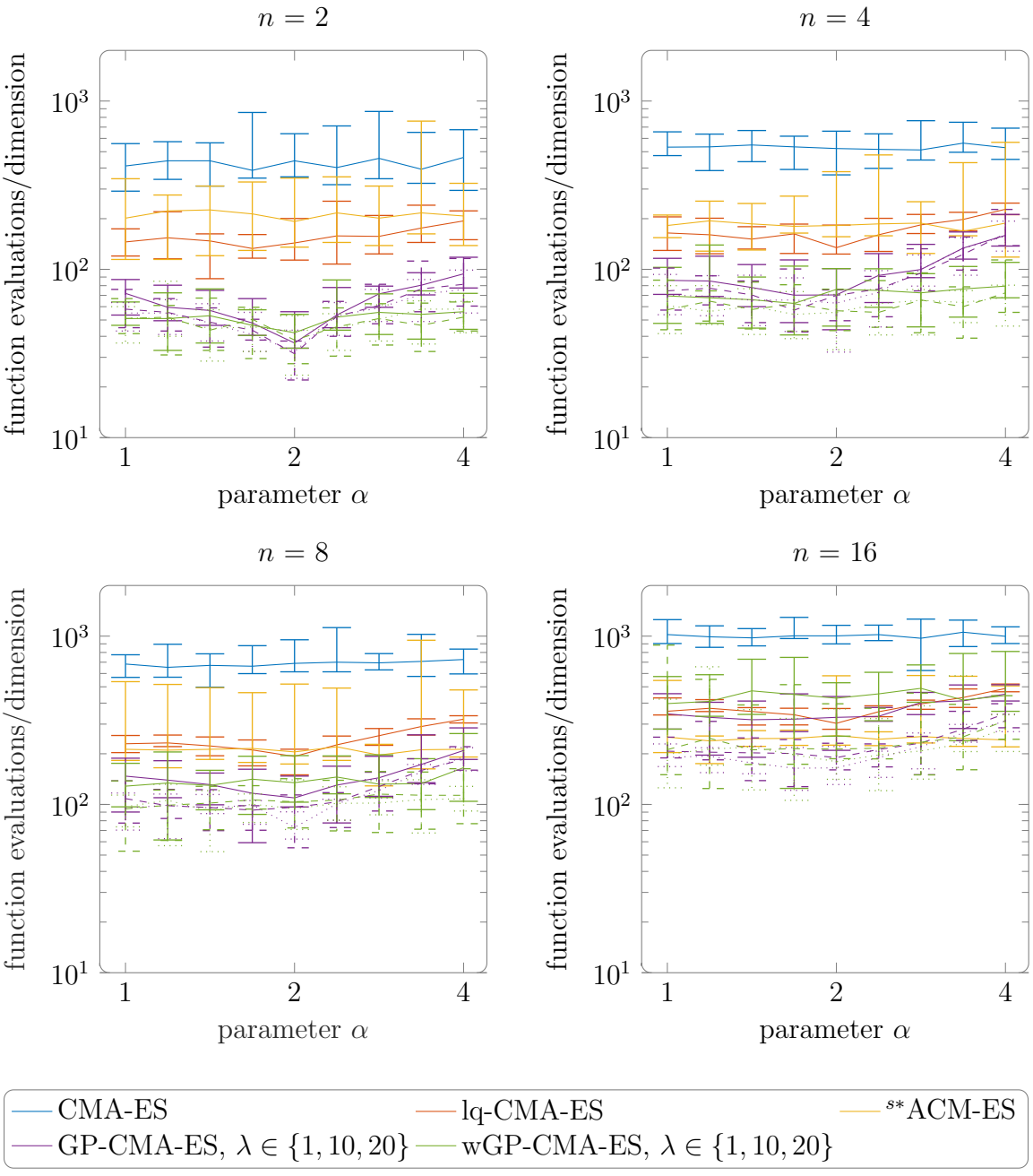


Figure 4.7: Number of objective function evaluations per dimension required to optimize the function value transformed generalized quartic function functions with parameters $\alpha \in [1, 4]$ and $\gamma \in 10^2$. The lines connect median values; the error bars reflect the full range of values observed for the respective algorithms.

The GP-CMA-ES perform similarly, or better than, ^{s*}ACM-ES for $\alpha > 3$ and $n \leq 8$. In sixteen dimensions, increasing γ values cause the gap in performance to narrow and disappear between ^{s*}ACM-ES and GP-CMA-ES. Warping successfully

regains the invariance lost from the GP surrogate and, with the exception of $\alpha \approx 4$, enables the wGP-CMA-ES to exhibit at least the second best performance on this problem class. We hypothesize that the degradation in performance at $\alpha \approx 4$ can be mitigated with a more accurate warp optimization procedure. A more intensive exploitation of the model for both GP-assisted strategies is observed to be more advantageous in problems with larger γ.

4.3.4 Power Sum Functions

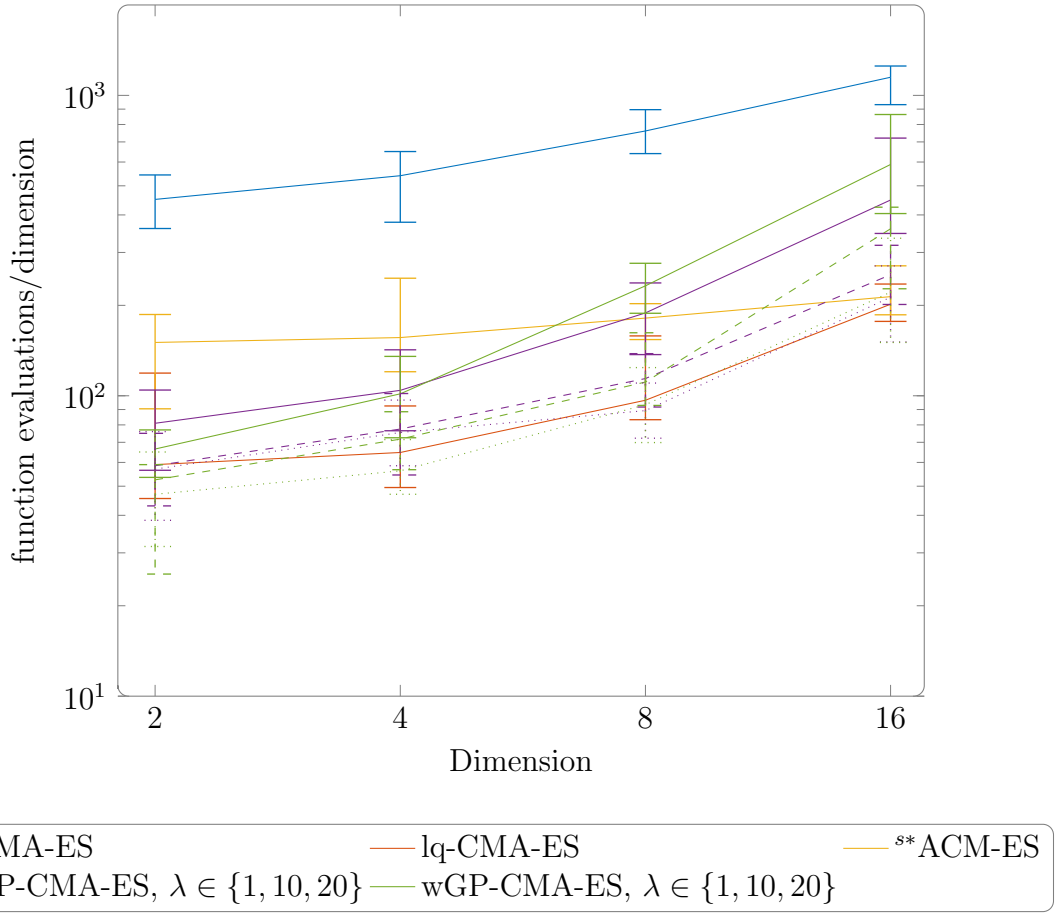


Figure 4.8: Number of objective function evaluations per dimension required to optimize the Sum of Different Powers functions with respect to problem dimension. The lines connect median values; the error bars reflect the full range of values observed for the respective algorithms.

Figure 4.8 illustrates the performance of the evolution strategies on f_{diffpow} in various dimensions. Notably, the ^{s*}ACM-ES are the only comparator whose speed-up

increases with increasing dimension. The performance of the lq-CMA-ES is highly competitive on this function and comparable to that of the GP-CMA-ES with $\gamma = 20$. Both Gaussian process surrogate model assisted algorithms benefit form increased exploitation of the model. The advantage of warping diminishes with increasing dimensionality likely due to the imperfect warp produced by the warp search procedure. Although the advantage of a rank-preserving surrogate is more palpable in dimensions $n < 8$, the combination of warping and more intensive exploitation of the model enables the wGP-CMA-ES to achieve competitive performance on this test problem.

Chapter 5

Conclusions

This chapter highlights the importance of warping for value-based surrogates, summarizes the main contributions of the thesis, and offers future directions for research.

5.1 Summary

Invariance is a major design principle of evolution strategies. Adaptation to affine transformations of the search space as well as invariance with respect to strictly monotonic transformations of the objective function are thus important features of the CMA-ES that account for its wide-ranging success. Surrogate models can replace expensive objective function evaluations by making use of information gathered in prior iterations of the optimizer to approximate the objective function in the vicinity of the current solution. This approximation can then be used in lieu of the true objective function, leading to potential savings of costly evaluations. However, the introduction of value-based surrogate models into comparison-based optimizers can compromise the resulting strategy's invariance to rank-preserving transformations of the objective function. In this work, we have proposed warping as an approach to regaining some of the invariance lost when employing such surrogate models in comparison-based optimizers. We have first proposed an error metric that can be used to learn an appropriate transformation of function values before these values are employed by the surrogate to construct an approximation. We have then combined this error metric with a previously proposed approach to Gaussian process assisted CMA-ES [73]. The advantage of warping was demonstrated on four families of parametrized, unimodal test problems. The proposed wGP-CMA-ES is competitive with more complicated strategies, but on highly ill-conditioned problems in higher dimensions, warping is observed to be a less effective alternative to rank-based surrogate models. More intensive surrogate model exploitation is shown to provide moderate speed-ups on ill-conditioned problems.

5.2 Future Directions

Several directions of future work are identifiable. First, strategies for more reliable and faster learning of the warp parameters are of immediate interest. We expect poorly conditioned objective functions in high dimensions to benefit from such improvement. Additionally, an adaptive mechanism for the selection of archive size is desirable. While some functions such as the quadratic sphere may require smaller archive sizes, others may benefit from larger training archives whose size scales quadratically with respect to the number of problem dimensions. This would increase model accuracy and could be beneficial beyond the unimodal set of test problems. Moreover, the batch size may be adapted to control the degree of exploitation of the model according to dimensions. As was noted earlier, part of the success of the ^{s*}ACM-ES in higher dimension is attributable to the ability to tune the degree of reliance on their surrogate model. Finally, a Bayesian treatment of model selection, which has shown promise on multimodal functions in the works of [15, 74, 16, 5], may be a more theoretically disciplined approach that may also prove beneficial in practice.

Bibliography

[1] Dirk V Arnold. *Noisy Optimization with Evolution Strategies*, volume 8 of *Genetic algorithms and evolutionary computation*. KAP, Dordrecht, The Netherlands, 2002.

[2] Dirk V. Arnold and Nikolaus Hansen. Active covariance matrix adaptation for the (1+1)-CMA-ES. In *Conference on Genetic and Evolutionary Computation*, GECCO '10, page 385–392, New York, NY, USA, 2010. ACM.

[3] Anne Auger and Nikolaus Hansen. A restart CMA evolution strategy with increasing population size. In *IEEE Congress on Evolutionary Computation*, volume 2, pages 1769–1776, New York, NY, USA, 2005. IEEE.

[4] Samineh Bagheri, Wolfgang Konen, Michael Emmerich, and Thomas Bäck. Self-adjusting parameter control for surrogate-assisted constrained optimization under limited budgets. *Applied Soft Computing*, 61:377–393, 2017.

[5] Lukás Bajer, Zbynek Pitra, Jakub Repický, and Martin Holena. Gaussian process surrogate models for the CMA evolution strategy. *Evolutionary Computation*, 27(4):665–697, 2019.

[6] Hans-Georg Beyer. *The theory of evolution strategies*. Natural computing series. Springer, New York, NY, USA, 2001.

[7] Zyed Bouzarkouna, Anne Auger, and Didier Yu Ding. Investigating the local-meta-model CMA-ES for large population sizes. In Cecilia D Chio, et al., editor, *Applications of Evolutionary Computation - EvoApplicatons*, volume 6024 of *Lecture Notes in Computer Science*, pages 402–411, New York, NY, USA, 2010. Springer.

[8] George E P Box and Norman R Draper. *Response Surfaces, Mixtures, and Ridge Analyses*. Wiley Series in Probability and Statistics. Hoboken, NJ, USA, New York, NY, USA, 2 edition, 2007.

[9] Dirk Büche, Nicol N Schraudolph, and Petros Koumoutsakos. Accelerating evolutionary algorithms with gaussian process fitness function models. *IEEE Transactions on Systems, Man, and Cybernetics - SMC*, 35(2):183–194, 2005.

[10] Jean-Paul Chilès and Nicolas Desassis. Fifty years of kriging. In B S Daya Sagar et al., editor, *Handbook of Mathematical Geosciences: Fifty Years of IAMG*, pages 589–612. Springer, New York, NY, USA, 2018.

[11] Hyoung-seog Chung, Hyoung-seog Chung, Alonso Juan J, and Juan J Alonso. Design of a low-boom supersonic business jet using cokriging approximation models. In *Symposium on Multidisciplinary Analysis and Optimization*, pages 2002–5598, Red Hook, NY, USA, 2002. Curran Associates Inc.

[12] Areski Cousin, Hassan Maatouk, and Didier Rullière. Kriging of financial term-structures. *European Journal of Operational Research*, 255(2):631–648, 2016.

[13] Marco Aurélio de Oliveira, Osmar Possamai, Luiz V O Dalla Valentina, and Carlos Alberto Flesch. Modeling the leadership – project performance relation: radial basis function, gaussian and kriging methods as alternatives to linear regression. *Expert Systems with Applications*, 40(1):272–280, 2013.

[14] Vijaya Dixit, Navaneeth Seshadrinath, and M.K. Tiwari. Performance measures based optimization of supply chain network resilience: A nsga-ii+co-kriging approach. *Computers and Industrial Engineering*, 93:205–214, 2016.

[15] Michael Emmerich, Alexios Giotis, Mutlu Özdemir, Thomas Bäck, and Kyriakos Giannakoglou. Metamodel-assisted evolution strategies. In Juan J M Guervós et al., editor, *Parallel Problem Solving from Nature - PPSN VII*, pages 361–370, Berlin, Heidelberg, 2002. Springer.

[16] Michael T 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(4):421–439, 2006.

[17] Christodoulos A Floudas and Panos M Pardalos. *A Collection of Test Problems for Constrained Global Optimization Algorithms*. Springer, Berlin, Heidelberg, 1990.

[18] Salvador García, Daniel Molina, Manuel Lozano, and Francisco Herrera. A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization. *Journal of Heuristics*, 15(6):617–644, 2009.

[19] Sylvain Gelly, Sylvie Ruette, and Olivier Teytaud. Comparison-based algorithms are robust and randomized algorithms are anytime. *Evolutionary Computation*, 15(4):411–434, 2007.

[20] N. Hansen and A. Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation. In *Proceedings of IEEE International Conference on Evolutionary Computation*, pages 312–317, New York, NY, USA, 1996. IEEE Press.

[21] Nikolaus Hansen. *Verallgemeinerte individuelle Schrittweitenregelung in der Evolutionsstrategie - eine Untersuchung zur entstochastisierten, koordinatensystemunabhängigen Adaptation der Mutationsverteilung*. Mensch und Buch, Berlin, Germany, 1998.

[22] Nikolaus Hansen. *The CMA Evolution Strategy: A Comparing Review*, volume 192 of *Studies in Fuzziness and Soft Computing*, pages 75–102. Springer, Berlin, Heidelberg, 2006.

[23] Nikolaus Hansen. Benchmarking a bi-population CMA-ES on the bbob-2009 function testbed. In *Genetic and Evolutionary Computation Conference - GECCO*, GECCO '09 Companion, pages 2389–2396, New York, NY, USA, 2009. ACM.

[24] Nikolaus Hansen. The CMA evolution strategy: A Tutorial. *CoRR*, abs/1604.0, 2016.

[25] Nikolaus Hansen. A global surrogate assisted CMA-ES. In Anne Auger et al., editor, *Genetic and Evolutionary Computation Conference - GECCO*, pages 664–672, Cambridge, MA, USA, 2019. ACM.

[26] Nikolaus Hansen, Dirk V Arnold, and Anne Auger. Evolution strategies. In Janusz Kacprzyk and Witold Pedrycz, editors, *Springer Handbook of Computational Intelligence*, Springer Handbooks, pages 871–898. Springer, Berlin, Heidelberg, 2015.

[27] Nikolaus Hansen and Anne Auger. Principled design of continuous stochastic search: From theory to practice. In Yossi Borenstein and Alberto Moraglio, editors, *Theory and Principled Methods for the Design of Metaheuristics*, Natural Computing Series, pages 145–180. Springer, Berlin, Heidelberg, 2014.

[28] Nikolaus Hansen, Anne Auger, Steffen Finck, and Raymond Ros. Real-parameter black-box optimization benchmarking 2010: Experimental setup. Research Report RR-7215, INRIA, 2010.

[29] Nikolaus Hansen, Anne Auger, Raymond Ros, Steffen Finck, and Petr Pošík. Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009. In *Genetic and Evolutionary Computation - GECCO*, GECCO '10, page 1689–1696, New York, NY, USA, 2010. Association for Computing Machinery.

[30] Nikolaus Hansen, Anne Auger, Raymond Ros, Olaf Mersmann, Tea Tusar, and Dimo Brockhoff. COCO: a platform for comparing continuous optimizers in a black-box setting. *Optimization Methods and Software*, 36(1):114–144, 2021.

[31] Nikolaus Hansen, Steffen Finck, Raymond Ros, and Anne Auger. Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions. Research Report RR-6829, INRIA, 2009.

[32] Nikolaus Hansen, Andreas Gawelczyk, and Andreas Ostermeier. Sizing the population with respect to the local progress in $(1, \lambda)$-evolution strategies — a theoretical analysis. In *Proceedings of IEEE International Conference on Evolutionary Computation*, volume 1, pages 80–, New York, NY, USA, 1995. IEEE Press.

[33] Nikolaus Hansen, Sibylle D. Müller, and Petros Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evolutionary Computation*, 11(1):1–18, 2003.

[34] Nikolaus Hansen and Andreas Ostermeier. Convergence properties of evolution strategies with the derandomized covariance matrix adaptation: The $(\mu/\mu_i, \lambda)$-CMA-ES. In *European Congress on Intelligent Techniques and Soft Computing*, EUFIT'97, pages 650–654, Aachen, Germany, 1997.

[35] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[36] Nikolaus Hansen and Raymond Ros. Benchmarking a weighted negative covariance matrix update on the bbob-2010 noiseless testbed. In *Conference Companion on Genetic and Evolutionary Computation*, GECCO '10, page 1673–1680, New York, NY, USA, 2010. ACM.

[37] Nikolaus Hansen, Raymond Ros, Nikolas Mauny, Marc Schoenauer, and Anne Auger. Impacts of invariance in search: When CMA-ES and PSO face ill-conditioned and non-separable problems. *Applied Soft Computing*, 11:5755–5769, 2011.

[38] Christian Igel, Thorsten Suttorp, and Nikolaus Hansen. A computational efficient covariance matrix update and a (1+1)-CMA for evolution strategies. In Mike Cattolico, editor, *Genetic and Evolutionary Computation Conference - GECCO*, pages 453–460, Cambridge, MA, USA, 2006. ACM.

[39] Syun Izawa, Koki Kitai, Shu Tanaka, Ryo Tamura, and Koji Tsuda. Continuous black-box optimization with quantum annealing and random subspace coding. *arXiv:2104.14778*, 2021.

[40] Grahame A. Jastrebski and Dirk V. Arnold. Improving evolution strategies through active covariance matrix adaptation. In *IEEE International Conference on Evolutionary Computation*, pages 2814–2821, New York, NY, USA, 2006. IEEE.

[41] Yaochu Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing*, 9(1):3–12, 2005.

[42] Yaochu Jin. Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, 1(2):61–70, 2011.

[43] Yaochu Jin and Bernhard Sendhoff. Reducing fitness evaluations using clustering techniques and neural network ensembles. In Kalyanmoy Deb, editor, *Genetic and Evolutionary Computation*, GECCO '04, pages 688–699, Berlin, Heidelberg, 2004. Springer.

[44] A G Journel and C J Huijbregts. *Mining Geostatistics*. Blackburn Press, Caldwell, NJ, USA, 2003.

[45] Arash Kayhani and Dirk V Arnold. Design of a surrogate model assisted (1+1)-es. In Anne Auger et al., editor, *Parallel Problem Solving from Nature - PPSN XV*, volume 11101 of *Lecture Notes in Computer Science*, pages 16–28, New York, NY, USA, 2018. Springer.

[46] Stefan Kern, Nikolaus Hansen, and Petros Koumoutsakos. Local meta-models for optimization using evolution strategies. In Thomas P Runarsson et al., editor, *Parallel Problem Solving from Nature- PPSN IX*, volume 4193 of *Lecture Notes in Computer Science*, pages 939–948, New York, NY, USA, 2006. Springer.

[47] Stefan Kern, Sibylle D Müller, Nikolaus Hansen, Dirk Büche, Jiri Ocenasek, and Petros Koumoutsakos. Learning probability distributions in continuous evolutionary algorithms - a comparative review. *Natural Computing*, 3(1):77–112, 2004.

[48] Miguel Lázaro-Gredilla. Bayesian warped gaussian processes. In *Conference on Neural Information Processing Systems - NeurIPS*, volume 1, pages 1619–1627, Red Hook, NY, USA, 2012. Curran Associates Inc.

[49] Dudy Lim, Yew-Soon Ong, Yaochu Jin, and Bernhard Sendhoff. A study on metamodeling techniques, ensembles, and multi-surrogates in evolutionary computation. In *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, GECCO '07, page 1288–1295, New York, NY, USA, 2007. ACM.

[50] Ilya Loshchilov. *Surrogate-Assisted Evolutionary Algorithms*. Theses, Université Paris Sud, 2013.

[51] Ilya Loshchilov, Marc Schoenauer, and Michèle Sebag. Comparison-based optimizers need comparison-based surrogates. In Robert Schaefer et al., editor, *Parallel Problem Solving from Nature - PPSN XI*, volume 6238 of *Lecture Notes in Computer Science*, pages 364–373, New York, NY, USA, 2010. Springer.

[52] Ilya Loshchilov, Marc Schoenauer, and Michele Sebag. Black-box optimization benchmarking of ipop-saacm-es and bipop-saacm-es on the bbob-2012 noiseless testbed. In *Genetic and Evolutionary Computation Conference - GECCO*, GECCO '12, pages 175–182, New York, NY, USA, 2012. ACM.

[53] Ilya Loshchilov, Marc Schoenauer, and Michèle Sebag. Self-adaptive surrogate-assisted covariance matrix adaptation evolution strategy. In Terence Soule and Jason H Moore, editors, *Genetic and Evolutionary Computation Conference - GECCO*, pages 321–328, New York, NY, USA, 2012. ACM.

[54] Ilya Loshchilov, Marc Schoenauer, and Michele Sèbag. Bi-population CMA-ES algorithms with surrogate models and line searches. In *Genetic and Evolutionary Computation - GECCO*, pages 1177–1184, New York, NY, USA, 2013. ACM.

[55] M I McLean, L Evers, A W Bowman, M Bonte, and W R Jones. Statistical modelling of groundwater contamination monitoring data: A comparison of spatial and spatiotemporal methods. *Science of the Total Environment*, 652:1339–1346, 2019.

[56] James Mercer. Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 209:415–446, 1909.

[57] Zbigniew Michalewicz and Marc Schoenauer. Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary Computation*, 4(1):1–32, 1996.

[58] Raymond H Myers, Douglas C Montgomery, and Christine M Anderson-Cook. *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*. Wiley Series in Probability and Statistics. Wiley, Hoboken, NJ, USA, 3 edition, 2009.

[59] John A. Nelder and Roger Mead. A Simplex Method for Function Minimization. *The Computer Journal*, 7(4):308–313, 01 1965.

[60] Andreas Ostermeier, Andreas Gawelczyk, and Nikolaus Hansen. Step-size adaptation based on non-local use of selection information. In Yuval Davidor, Hans-Paul Schwefel, and Reinhard Männer, editors, *Parallel Problem Solving from Nature*, pages 189–198, Berlin, Heidelberg, 1994. Springer.

[61] Zbynek Pitra, Lukás Bajer, Jakub Repický, and Martin Holena. Overview of surrogate-model versions of covariance matrix adaptation evolution strategy. In Peter A N Bosman, editor, *Genetic and Evolutionary Computation Conference - GECCO*, pages 1622–1629, New York, NY, USA, 2017. ACM.

[62] Carl Edward Rasmussen and Christopher K I Williams. *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, Cambridge, MA, USA, 2006.

[63] Ingo Rechenberg. *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Problemata (Stuttgart). Frommann-Holzboog, Stuttgart, Germany, 1973.

[64] Rommel G Regis. Constrained optimization by radial basis function interpolation for high-dimensional expensive black-box problems with infeasible initial points. *Engineering Optimization*, 46(2):218–243, 2014.

[65] Rommel G Regis and Christine A Shoemaker. A quasi-multistart framework for global optimization of expensive functions using response surface models. *Journal of Global Optimization*, 56(4):1719–1753, 2013.

[66] Thomas Philip Runarsson. Constrained evolutionary optimization by approximate ranking and surrogate models. In Xin Yao et al., editor, *Parallel Problem Solving from Nature - PPSN VIII*, pages 401–410, Berlin, Heidelberg, 2004. Springer.

[67] Jerome Sacks, William J Welch, Toby J Mitchell, and Henry P. Wynn. Design and analysis of computer experiments. *Statistical Science*, 4(4):409 – 423, 1989.

[68] Hans-Paul Schwefel. *Numerische Optimierung von Computermodellen mittels der Evolutionsstrategie*, volume 26. Wiley, Hoboken, NJ, USA, 1977.

[69] Edward Snelson, Carl Edward Rasmussen, and Zoubin Ghahramani. Warped gaussian processes. In Sebastian Thrun et al., editor, *Conference on Neural Information Processing Systems - NeurIPS*, pages 337–344, Cambridge, MA, USA, 2003. MIT Press.

[70] Sellamanickam Sundararajan and Sathiya K. Selvaraj. Predictive approaches for choosing hyperparameters in gaussian processes. *Neural Computation*, 13(5):1103—1118, 2001.

[71] Olivier Teytaud and Hervé Fournier. Lower bounds for evolution strategies using vc-dimension. In Günter Rudolph, Thomas Jansen, Nicola Beume, Simon Lucas, and Carlo Poloni, editors, *Parallel Problem Solving from Nature – PPSN X*, pages 102–111, Berlin, Heidelberg, 2008. Springer.

[72] Olivier Teytaud and Sylvain Gelly. General lower bounds for evolutionary algorithms. In Thomas P Runarsson et al., editor, *Parallel Problem Solving from Nature - PPSN IX*, volume 4193 of *Lecture Notes in Computer Science*, pages 21–31, New York, NY, USA, 2006. Springer.

[73] Lauchlan Toal and Dirk V Arnold. Simple surrogate model assisted optimization with covariance matrix adaptation. In Thomas Bäck et al., editor, *Parallel Problem Solving from Nature - PPSN XVI*, volume 12269 of *Lecture Notes in Computer Science*, pages 184–197, New York, NY, USA, 2020. Springer.

[74] Holger Ulmer, Felix Streichert, and Andreas Zell. Model-assisted steady-state evolution strategies. In Erick Cantú-Paz et al., editor, *Genetic and Evolutionary Computation Conference - GECCO*, volume 2723 of *Lecture Notes in Computer Science*, pages 610–621, New York, NY, USA, 2003. Springer.

[75] Richard Von Mises and Hilda Geiringer. *Mathematical Theory of Probability and Statistics*, chapter 9, pages 427–429. Academic Press, New York, NY, USA, 1964.

[76] R. Webster and M.A. Oliver. *Geostatistics for Environmental Scientists*. Statistics in Practice. Wiley, Hoboken, NJ, USA, 2007.

[77] J W Yan, K M Liew, and L H He. A mesh-free computational framework for predicting buckling behaviors of single-walled carbon nanocones under axial compression based on the moving kriging interpolation. *Computer Methods in Applied Mechanics and Engineering*, 247-248:103–112, 2012.

[78] Jingyun Yang and Dirk V Arnold. A surrogate model assisted (1+1)-ES with increased exploitation of the model. In Anne Auger and Thomas Stützle, editors, *Genetic and Evolutionary Computation Conference - GECCO*, pages 727–735, New York, NY, USA, 2019. ACM.

[79] Martin Zaefferer. *Surrogate Models for Discrete Optimization Problems*. PhD thesis, Technische Universität Dortmund, 2018.