Achieving Highly Scalable Evolutionary Real-Valued Optimization by Exploiting Partial Evaluations

Anton Bouter Anton.Bouter@cwi.nl
Centrum Wiskunde & Informatica (CWI), Amsterdam, The Netherlands

Tanja Alderliesten T.Alderliesten@amsterdamumc.nl
Amsterdam UMC, University of Amsterdam, Amsterdam, The Netherlands

Peter A.N. Bosman Peter.Bosman@cwi.nl
Centrum Wiskunde & Informatica (CWI), Amsterdam, The Netherlands

https://doi.org/10.1162/evco_a_00275

Abstract

It is known that to achieve efficient scalability of an Evolutionary Algorithm (EA), dependencies (also known as linkage) must be properly taken into account during variation. In a Gray-Box Optimization (GBO) setting, exploiting prior knowledge regarding these dependencies can greatly benefit optimization. We specifically consider the setting where partial evaluations are possible, meaning that the partial modification of a solution can be efficiently evaluated. Such problems are potentially very difficult, for example, non-separable, multimodal, and multiobjective. The Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA) can effectively exploit partial evaluations, leading to a substantial improvement in performance and scalability. GOMEA was recently shown to be extendable to real-valued optimization through a combination with the real-valued estimation of distribution algorithm AMaLGaM. In this article, we definitively introduce the Real-Valued GOMEA (RV-GOMEA), and introduce a new variant, constructed by combining GOMEA with what is arguably the best-known real-valued EA, the Covariance Matrix Adaptation Evolution Strategies (CMA-ES). Both variants of GOMEA are compared to L-BFGS and the Limited Memory CMA-ES (LM-CMA-ES). We show that both variants of RV-GOMEA achieve excellent performance and scalability in a GBO setting, which can be orders of magnitude better than that of EAs unable to efficiently exploit the GBO setting.

Keywords

Linkage, gray-box optimization, real-valued optimization, multiobjective optimization.

1 Introduction

Evolutionary Algorithms (EAs) are frequently used to solve optimization problems that are considered too difficult for more efficient algorithms such as local search in case of discrete problems or gradient descent in case of real-valued problems. This is the case when, for example, the problem is multimodal or when plateaus in the problem landscape prevent gradient information from directing the search toward the global optimum. Furthermore, EAs are known to be among the state-of-the-art for the optimization of Multi-Objective (MO) optimization problems (Deb, 2001). In most cases, the optimization problem is considered to be a Black-Box Optimization (BBO)

Manuscript received: 8 July 2019; revised: 25 May 2020 and 22 April 2020; accepted: 29 May 2020.

problem, where only function evaluations can reveal information about the structure of the optimization problem at hand.

Considering problems in a BBO setting is, however, not always a necessity, and the exploitation of problem-specific information, when possible, could substantially increase the performance of an EA. Some exploitable form of problem information can be available even when the optimization problem is considered very difficult. For example, an EA with problem-specific recombination and mutation operators has been able to find near-optimal solutions for a binary optimization problem with a billion variables (Deb and Myburgh, 2016). Near-linear scalability in the number of problem variables was achieved, despite the problem being non-separable due to a combination of equality and inequality constraints. In comparison, conventional integer linear programming solvers were unable to solve a 2000-variable version of this problem.

Partition crossover (Whitley et al., 2009) is a more general instance of a custom binary crossover operator that exploits problem information. This operator has access to a variable interaction graph, describing whether or not interactions exist between pairs of problem variables. Partition crossover was previously applied to the Traveling Salesman Problem (TSP) (Whitley et al., 2009) and NK-landscapes (Tinós et al., 2015). It substantially outperformed state-of-the-art local-search solvers for the Maximal Satisfiability (MAXSAT) problem (Chen et al., 2018).

Partial evaluations are a way to exploit problem-specific information in the domain of either discrete or real-valued optimization, allowing efficient evaluation of the objective value(s) of a solution for which only a small number of problem variables have been modified. When such partial evaluations are possible, we speak of a Gray-Box Optimization (GBO) setting.

In real-valued optimization, which our work is focused on, proper use of partial evaluations has been shown to lead to a substantial increase in performance and scalability on a wide range of benchmark problems (Bouter, Alderliesten et al., 2017), as well as the real-world problems of medical deformable image registration (Bouter, Alderliesten, and Bosman, 2017) and the optimization of brachytherapy treatment plans for prostate cancer (Luong, Alderliesten et al., 2018), when combined with a real-valued version of GOMEA (RV-GOMEA) (Bouter, Alderliesten, Witteveen et al., 2017; Bouter, Luong et al., 2017). The two aforementioned real-world problems are both instances of non-separable, multimodal, multiobjective problems, making them very suitable for optimization with an EA. Despite the complexity of these problems, their definitions allow for the application of partial evaluations, leading to substantial improvements in performance and scalability. To the best of our knowledge, RV-GOMEA is the first real-valued EA to exploit partial evaluations.

In this article, we highlight the benefits of a GBO setting in which partial evaluations are possible, and the requirements for such a setting. Furthermore, we definitively introduce RV-GOMEA, extending previously published work (Bouter, Alderliesten, Witteveen et al., 2017; Bouter, Luong et al., 2017) by comparisons with state-of-the-art large-scale optimization algorithms including the well-known gradient-based optimization method L-BFGS (Liu and Nocedal, 1989), and the Limited-Memory CMA-ES (LM-CMA-ES) (Loshchilov, 2014), a large-scale variant of CMA-ES, on various types of single-objective and multiobjective problems. Furthermore, we introduce a novel variation of RV-GOMEA, by making a new combination with the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) (Hansen and Ostermeier, 2001), arguably the most well-known and state-of-the-art EA for real-valued optimization.

The remainder of this article is organized as follows. In Section 2, we discuss work on Model-Based EAs (MBEAs) related to GOMEA. Section 3 then gives a general introduction to GOMEA, and Section 4 describes how linkage structure is modeled in GOMEA. Section 5 describes the GBO setting that we consider, which allows for the use of partial evaluations, and the benefits and restrictions of such a setting are discussed. The extension of GOMEA to the real-valued domain, that is, RV-GOMEA, is introduced in Section 6, after which two instantiations (one novel instance and one previously introduced) of RV-GOMEA are discussed in Section 7. Multiobjective variants of these algorithms are subsequently designed in Section 8. In Section 9, the performance of all discussed algorithms, combined with a variety of different linkage models, is then analyzed through scalability experiments on a set of well-known benchmark problems. Results are then discussed in Section 10, followed by stating a number of conclusions drawn from said experiments in Section 11.

2 Related Work

In the standard Genetic Algorithm (GA) (Holland, 1975), variation is performed completely at random. In comparison, the goal of MBEAs is to perform variation in a more informed manner, so as to substantially improve performance. MBEAs exploit the problem structure or the optimization landscape throughout the optimization process guided by an explicit model that describes key characteristics of the problem at hand. This model may be learned online, while optimization is being performed, or it may be (partially) instantiated with problem-specific knowledge.

In the domain of real-valued optimization, which this work focuses on, a Gaussian distribution is frequently used to describe how to generate the values for the problem variables of new solutions. This is done in, e.g., the Covariance Matrix Adaptation Evolution Strategies (CMA-ES) (Hansen and Ostermeier, 2001), the Adapted Maximum-Likelihood Gaussian Model Iterated Density Estimation Algorithm (AMaLGaM-ID𝔼A or AMaLGaM for short) (Bosman et al., 2013), and Natural Evolution Strategies (NES) (Wierstra et al., 2014).

A key aspect of problem structure is the linkage structure, which describes the structure of an optimization problem in terms of the dependencies between problem variables. Linkage information, which is generally derived from the population or known beforehand due to problem-specific knowledge, is used in many MBEAs to estimate a statistical model that captures the underlying linkage structure of the optimization problem. In Estimation of Distribution Algorithms (EDAs) (Lozano et al., 2006; Pelikan et al., 2007), a specific type of MBEA, a probability distribution is used to model the linkage structure. In discrete optimization, the Bivariate Marginal Distribution Algorithm (BMDA) (Pelikan and Mühlenbein, 1999), an extension of the Univariate Marginal Distribution Algorithm (UMDA) (Mühlenbein, 1997), incorporates bivariate dependencies between problem variables in order to model dependencies. Models of larger capacity were later introduced, with the Extended Compact Genetic Algorithm (ECGA) (Harik et al., 2006) using a marginal product model, and the Estimation of Bayesian Network Algorithm (EBNA) (Etxeberria and Larrañaga, 1999), Bayesian Optimization Algorithm (BOA) (Pelikan, 2005a), and the hierarchical Bayesian Optimization Algorithm (hBOA) (Pelikan, 2005b) using Bayesian networks to model linkage structure. In the aforementioned Gaussian-based MBEAs, CMA-ES (Hansen and Ostermeier, 2001), AMaLGaM (Bosman et al., 2013), and NES (Wierstra et al., 2014), a covariance matrix describes the linkage structure.

Recently, there has been an increasing focus on MBEAs explicitly modeling and exploiting linkage information to guide mixing operators, rather than using the linkage information to guide a statistical model that samples new problem variables. This trend was first observed for discrete optimization (Thierens, 2010; Hsu and Yu, 2015; Thierens and Bosman, 2011; Goldman and Punch, 2014), and more recently for real-valued optimization (Bouter, Alderliesten, Witteveen et al., 2017). Of these new types of MBEAs, we focus on GOMEA (Thierens and Bosman, 2011) here, because GOMEA is so far the only one among these MBEAs to be extended beyond discrete optimization.

3 Gene-Pool Optimal Mixing

GOMEA has its roots in the Linkage Tree Genetic Algorithm (LTGA) (Thierens, 2010) for binary variables, which was later generalized and renamed (Thierens and Bosman, 2011). Subsequent efficiency enhancements, restart mechanisms, and multiobjective extensions, adapted GOMEA to its current state (Bosman and Thierens, 2013; Luong, La Poutré et al., 2018), which is considered to be among the state-of-the-art for discrete optimization.

A central concept in GOMEA that is the key to its performance, is to exploit linkage information by applying variation not to all problem variables simultaneously, but instead to apply variation to small subsets of dependent variables, and accepting a variation operation only if it leads to an improved offspring solution. This latter concept is called optimal mixing. A so-called linkage model is used to describe (small) subsets of variables that are considered to be dependent. Moreover, when the entire population is used as potential donor information during optimal mixing, the procedure is generally called Gene-pool Optimal Mixing (GOM).

After its introduction in GOMEA, optimal mixing was included also in other state-of-the-art EAs, such as DSMGA-II (Hsu and Yu, 2015) and P3 (Goldman and Punch, 2014). Moreover, even though GOMEA was originally introduced in the domain of discrete optimization, the concept of linkage information exploitation of GOMEA is more widely applicable, as was recently shown through RV-GOMEA (Bouter, Alderliesten, Witteveen et al., 2017). RV-GOMEA combines key aspects of GOMEA with those of the real-valued EDA known as AMaLGaM (Bosman et al., 2013). The dependency model of GOMEA is used in order to exploit linkage information, and the distribution-based sampling methods used by AMaLGaM are used in order to match the continuous nature of the search space.

The use of AMaLGaM as the method of sampling new problem-variable values is, however, not an absolute necessity. Instead, other real-valued EAs could be combined with GOMEA in order to benefit from its model-building capabilities. This not only applies to real-valued optimization, but could apply to EAs in other domains of optimization. Indeed, novel variants of GOMEA have recently appeared, for example, for permutation spaces (Bosman et al., 2016) and genetic programming (Virgolin et al., 2017).

4 Modeling Linkage Structure

In GOMEA, the linkage structure of a problem is modeled as a Family Of Subsets (FOS), denoted $\mathcal{F}$, which is a subset of the power set of $\mathcal{I}$, which contains the indices of all ℓ problem variables, that is, $\mathcal{F} \subseteq \mathcal{P}(\mathcal{I})$, where $\mathcal{I} = \{0, 1, \ldots, \ell - 1\}$. Each element $\mathcal{F}_i \in \mathcal{F}$ contains a number of indices of problem variables, all of which are considered to be mutually dependent by this linkage model. Such an element of a FOS is named a linkage set, as it describes a set of problem variables between which linkage information is modeled. Virtually any FOS can be used to model linkage structure, though generally speaking, each index $i \in \mathcal{I}$ appears in at least one linkage set $\mathcal{F}_j$.

The linkage model can either be determined prior to optimization, and be fixed throughout optimization, or it can be learned based on the population at the start of every generation. We refer to linkage models that are fixed as static linkage models, and to linkage models that change throughout optimization as dynamic linkage models.

In the remainder of this section, we discuss four potentially useful FOS models. Depending on the optimization problem, and how much domain-specific knowledge is available, one can select the linkage model that is the most appropriate.

4.1 Marginal Product FOS

The marginal product FOS is a FOS where each problem variable index exists in exactly one linkage set, that is, $\mathcal{F}_i \cap \mathcal{F}_j = \emptyset$ for each $\mathcal{F}_i, \mathcal{F}_j \in \mathcal{F}$ with $i \neq j$, and $\bigcup_{\mathcal{F}_i \in \mathcal{F}} \mathcal{F}_i = \mathcal{I}$. A simple instance of a marginal product FOS is the univariate FOS, which contains each problem variable index in a separate linkage set, that is, $\mathcal{F} = \{\{0\}, \{1\}, \ldots, \{\ell - 1\}\}$.

We also define the k-block FOS as a specific marginal product FOS where each linkage set consists of blocks of k subsequent variables, that is, $\mathcal{F} = \{\{0, 1, \ldots, k-1\}, \{k, k+1, \ldots, 2k-1\}, \ldots\}$. This FOS structure is defined to match the problem structure of the Sum of Rotated Ellipsoid Blocks (SoREB) benchmark problem discussed in Section 9.1.

4.2 Linkage Tree

The linkage tree FOS, first introduced in Thierens (2010), is a hierarchical linkage model. It contains linkage sets of varying sizes, including all linkage sets of one problem variable, and the linkage set that includes all problem variables. Moreover, each linkage set with a size larger than one consists of the elements of two smaller linkage sets combined, that is, for any $\mathcal{F}_i \in \mathcal{F}$ with $|\mathcal{F}_i| > 1$ there exist $\mathcal{F}_j, \mathcal{F}_k \in \mathcal{F}$ such that $i \neq j \neq k$, $\mathcal{F}_i = \mathcal{F}_j \cup \mathcal{F}_k$, and $\mathcal{F}_j \cap \mathcal{F}_k = \emptyset$.

The linkage tree FOS can be learned through agglomerative bottom-up hierarchical clustering. For this, a similarity notion is required between problem variables. In the dynamic case, an often used similarity notion is mutual information (Kraskov et al., 2004), which is estimated from the population. In real-valued optimization, when estimating Gaussian distributions, the mutual information between a pair of problem variables can for instance be estimated through the Pearson product-moment correlation coefficient. The Unweighted Pair Grouping Method with Arithmetic-mean (UPGMA) clustering approach, whereby the similarity between two sets is the average similarity of all pairwise combinations, is then used for the construction of the linkage tree. This approach has a complexity of $\mathcal{O}(\ell^2)$ (Gronau and Moran, 2007).

4.3 Bounded Fixed Linkage Tree

For certain problems, and arguably many large-scale real-valued problems, using a linkage tree that models dependencies up to the set of all problem variables is a waste of resources, because only relatively low-order dependencies are present in the problem. In this case, the Bounded Fixed Linkage Tree (BFLT) model, which was introduced by Bouter, Alderliesten, Witteveen et al. (2017), can be used. The BFLT model is constructed in the same way as any linkage tree, but any merge of two linkage sets that would lead to a linkage set of a size larger than k is avoided, and the construction of the BFLT is terminated when no two available linkage sets can be merged into a new linkage set of size up to k.

A BFLT can be determined using a problem-specific notion of similarity between problem variables, for example, Euclidean distance in various real-world problems (Bouter, Alderliesten, and Bosman, 2017; Luong, Alderliesten et al., 2018). For this reason, a BFLT is not learned at the start of each generation, but prior to optimization.

Even though bounding the linkage tree can have a large impact on the efficiency of optimization, the construction of a BFLT with any $k > 1$ has the same complexity as the construction of a full linkage tree, i.e., $\mathcal{O}(\ell^2)$, as the calculation of all pairwise similarities is still required.

5 Partial Evaluations in Gray-Box Optimization

We speak of a GBO setting when the optimization problem allows for the application of partial evaluations, because in contrast to the BBO setting, this requires some knowledge of the optimization problem. A partial evaluation efficiently calculates the objective function(s) of a solution after the modification of a small number of problem variables, by subtracting the contribution of these variables to the objective function before modification, and adding their contribution after modification.

5.1 Definition

An optimization problem must be suitable for the application of partial evaluations. This is the case when it consists of k subfunctions, and it is known which variables each subfunction depends on. A given $\mathbf{I} = \{\mathcal{I}_0, \mathcal{I}_1, \ldots, \mathcal{I}_{k-1}\}$ defines that subfunction f_j^M has a dependency on all variables for which the index is included in $\mathcal{I}_j$. Other than this, each subfunction is considered to be a black box. A function to which partial evaluations can be applied, can be formulated in the following way (Bouter et al., 2018):

$$f(\boldsymbol{x}) = f^P\left(f_0^M(\boldsymbol{x}|_{\mathcal{I}_0}) \oplus f_1^M(\boldsymbol{x}|_{\mathcal{I}_1}) \oplus \ldots \oplus f_{k-1}^M(\boldsymbol{x}|_{\mathcal{I}_{k-1}})\right) = f^P\left(\bigoplus_{j=0}^{k-1} f_j^M(\boldsymbol{x}|_{\mathcal{I}_j})\right), \quad (1)$$

where for each $j \in [0, \ldots, k-1]$, $f_j^M : \mathbb{R}^{|\mathcal{I}_j|} \rightarrow \mathbb{R}$ is a function of $\boldsymbol{x}|_{\mathcal{I}_j}$, a restricted number of elements of $\boldsymbol{x}$. The indices of $\boldsymbol{x}$ that $\boldsymbol{x}|_{\mathcal{I}_j}$ is restricted to are defined by a given $\mathcal{I}_j \subseteq \mathcal{I} = \{0, 1, \ldots, \ell - 1\}$. The operator $\oplus$ can be any commutative binary operator for which we know an inverse operator $\ominus$, for example the summation or multiplication operators, though using the multiplication operator requires extra care to avoid division by zero (Bouter et al., 2018).

In the case that $f^P : \mathbb{R} \rightarrow \mathbb{R}$ is the identity function, the partial evaluation of a solution $\boldsymbol{x}'$ that is the result of a solution $\boldsymbol{x}$ following the modification of $\boldsymbol{x}_i$, requires the calculation of each subfunction that has a dependency with $\boldsymbol{x}_i$, that is:

$$f^{\texttt{part}}(\boldsymbol{x}, f_{\boldsymbol{x}}, \boldsymbol{x}', i) = f_{\boldsymbol{x}} \ominus \bigoplus_{\mathcal{I}_j \ni i} f_j^M(\boldsymbol{x}|_{\mathcal{I}_j}) \oplus \bigoplus_{\mathcal{I}_j \ni i} f_j^M(\boldsymbol{x}'|_{\mathcal{I}_j}), \quad (2)$$

with $f_{\boldsymbol{x}}$ the given objective function of $\boldsymbol{x}$, and $\mathcal{I}_j \ni i$ shorthand for $\{\mathcal{I}_j \in \mathbf{I} | i \in \mathcal{I}_j\}$.

If f^P is any non-invertible function, the application of a partial evaluation requires maintaining the sum of all subfunctions Σ in memory. This Σ is updated following each partial evaluation. Given a solution $\boldsymbol{x}$ and

$$\Sigma = \bigoplus_{j=0}^{k-1} f_j^M(\boldsymbol{x}|_{\mathcal{I}_j}), \quad (3)$$

a partial evaluation following the modification of $\boldsymbol{x}_i$ can be performed as follows:

$$f^{\texttt{part}}(\boldsymbol{x}, \Sigma, \boldsymbol{x}', i) = f^P\left(\Sigma \ominus \bigoplus_{\mathcal{I}_j \ni i} f_j^M(\boldsymbol{x}|_{\mathcal{I}_j}) \oplus \bigoplus_{\mathcal{I}_j \ni i} f_j^M(\boldsymbol{x}'|_{\mathcal{I}_j})\right). \quad (4)$$

Note that the definition of a function can reveal information about its separability, because a variable interaction graph can be built based on $\mathbf{I}$.

5.2 Application

Because variation is applied to small subsets of variables by GOMEA, followed by an evaluation to determine whether the variation step should be accepted or rejected, the possibility of applying partial evaluations substantially improves the performance of GOMEA. Partial evaluations are used to efficiently update the objective value of a parent solution that is modified during GOM. If the parent solution is now in a state that achieves a better objective value than before its modification, this state of the parent solution is maintained in the population. Otherwise, the parent solution is returned to the state before its modification.

Knowing the separability of a problem, and how partial evaluations can be applied, does, however, not necessarily mean that the optimal linkage structure is immediately clear. For example, some non-separable problems can best be solved with a marginal product FOS due to relatively weak interactions between variables.

In contrast to GOMEA, state-of-the-art EAs such as CMA-ES (Hansen and Ostermeier, 2001) and some of its variants (Ros and Hansen, 2008; Loshchilov, 2014) cannot take full advantage of the possibility of partial evaluations, because solutions are only ever evaluated after all of their variables have been sampled anew. CMA-ES can benefit from a GBO scenario by using a covariance matrix that is restricted based on the decomposability of the problem, as this can be derived from the GBO problem definition. However, this benefit is marginal compared to the full potential of partial evaluations, because sep-CMA-ES (Ros and Hansen, 2008) performed multiple orders of magnitude worse than RV-GOMEA on various decomposable benchmark problems in a GBO setting, in terms of time and number of function evaluations (Bouter, Alderliesten, Witteveen et al., 2017).

In Section 9.1, we discuss how partial evaluations can be applied to a series of benchmark problems. This includes problems that are non-separable, multimodal, or multiobjective. Furthermore, partial evaluations have previously been applied to the real-world problems of medical deformable image registration (Bouter, Alderliesten, and Bosman, 2017) and the optimization of brachytherapy treatment plans for prostate cancer (Luong, Alderliesten et al., 2018). Both of these problems are non-separable, multiobjective, and multimodal, making them very well suited for optimization with an EA. The possibility of applying partial evaluations to these problems then gives GOMEA a substantial advantage compared to different algorithms that cannot benefit as much from partial evaluations.

6 Real-Valued GOMEA

RV-GOMEA (Bouter, Alderliesten, Witteveen et al., 2017) was introduced as a combination of GOMEA (Thierens and Bosman, 2011) and AMaLGaM (Bosman et al., 2013). The use of AMaLGaM as the method of sampling new problem-variable values is however not an absolute necessity. Instead, other real-valued EAs could be combined with GOMEA in order to benefit from its model-building capabilities. In this section, we present a general outline of RV-GOMEA, and we show how a different EA can be combined with GOMEA to form variations of RV-GOMEA. In Section 7.2, we then apply this to combine GOMEA with CMA-ES.

In RV-GOMEA, a population of solutions is maintained, and a linkage model is used to describe the linkage structure of the optimization problem (see Section 4). Until any one of the termination criteria are satisfied, possibly a budget in terms of time or number of evaluations, generations are performed, each consisting of variation and

Algorithm 1 RV-GOMEA

```
1:  procedure RV-GOMEA($n$)
2:      $\mathcal{P} \leftarrow$ InitializeAndEvaluatePopulation($n$)
3:      $\mathcal{F} \leftarrow$ InitializeLinkageModel()
4:      for $j \in \{0, \ldots, |\mathcal{F}| - 1\}$ do
5:          $\mathcal{M}_j \leftarrow$ InitializeSamplingModel($\mathcal{F}_j$)
6:      while $\neg$TerminationCriterionSatisfied() do
7:          $\mathcal{P}_0 \leftarrow \boldsymbol{x}^{\text{elitist}}$
8:          for $j \in \{0, \ldots, |\mathcal{F}| - 1\}$ do                ▷ Random order
9:              for $\boldsymbol{x} \in \mathcal{P}_{[1 \ldots n-1]}$ do
10:                 GOM($\boldsymbol{x}, \mathcal{F}_j, \mathcal{M}_j$)
11:             IntermediateUpdate($\mathcal{M}_j$)
12:         for $\boldsymbol{x} \in \mathcal{P}_{[1 \ldots n_{\text{AMS}}]}$ do AnticipatedMeanShift($\boldsymbol{x}$)
13:         for $\boldsymbol{x} \in \mathcal{P}_{[1 \ldots n-1]}$ do
14:             if NIS($\boldsymbol{x}$) > NIS$^{\text{MAX}}$ then
15:                 ForcedImprovement($\boldsymbol{x}$)
16:         for $j \in \{0, \ldots, |\mathcal{F}| - 1\}$ do
17:             UpdateSamplingModel($\mathcal{M}_j$)
```

selection. The general outline of a generation of RV-GOMEA consists of applying GOM with each linkage set to each solution in the population $\mathcal{P}$ of size n, apart from the one elitist solution from the previous generation that is copied to the current generation. At the end of each generation, the parameters of the sampling model, for example, the covariance matrix, are updated in a way defined by the sampling model itself. The application of GOM with some linkage set $\mathcal{F}_j$ to some parent solution $\boldsymbol{x}$ then consists of sampling a new partial solution $\boldsymbol{o}$, which contains values for all variables described by linkage set $\mathcal{F}_j$. All values of the partial solution $\boldsymbol{o}$ are then inserted into the parent $\boldsymbol{x}$, and the parent is evaluated by applying partial evaluations.

Partial evaluations can lead to numerical errors in the objective values of solutions, of which the magnitude depends on the optimization problem and the dimensionality of the instance being solved. Therefore, whenever a solution seems to have reached the Value-To-Reach (VTR), it is completely reevaluated to remove any numerical errors caused by partial evaluations. Moreover, errors in the objective values of solutions can have an impact on the overall optimization process. For this reason, the entire population is reevaluated every 50 generations by default.

Pseudocode of the general outline of RV-GOMEA is shown in Algorithm 1, and pseudocode of the GOM procedure is given in Algorithm 2. How GOM is performed, and how a sampling model is initialized and updated, strictly depends on how this sampling model is defined. The Anticipated Mean Shift (AMS) (see Section 7.3) and forced-improvement procedures (see Section 7.4) are used for the real-valued instantiation of GOMEA, as these procedures are specifically targeted at real-valued optimization, and proved to be successful (Bouter, Alderliesten, Witteveen et al., 2017). The forced-improvement procedure is only applied to solutions that have not been improved for a number of generations larger than the maximum No-Improvement Stretch (NIS) (Bosman et al., 2013).

7 Single-Objective Optimization

In this section, we describe the application of two different sampling models to RV-GOMEA. The first application uses a sampling model based on AMaLGaM, as was

Algorithm 2 Gene-pool optimal mixing

```
1:  procedure GOM($\boldsymbol{x}, \mathcal{F}_j, \mathcal{M}_j$)
2:      for $u \in \{0, \ldots, |\mathcal{F}_j| - 1\}$ do
3:          $\boldsymbol{b}[u] \leftarrow \boldsymbol{x}[\mathcal{F}_j[u]]$
4:      $\boldsymbol{o} \leftarrow$ SamplePartialSolution($\mathcal{M}_j$)
5:      for $u \in \{0, \ldots, |\mathcal{F}_j| - 1\}$ do
6:          $\boldsymbol{x}[\mathcal{F}_j[u]] \leftarrow \boldsymbol{o}[u]$
7:      $\boldsymbol{f_o} \leftarrow$ PartialEvaluation($\boldsymbol{x}, \boldsymbol{f_x}, \mathcal{F}_j$)
8:      if $\boldsymbol{f_o} < \boldsymbol{f_x}$ or $\mathcal{U}(0,1) < p^{\text{accept}}$ then
9:          $\boldsymbol{f_x} = \boldsymbol{f_o}$
10:     else
11:         for $u \in \{0, \ldots, |\mathcal{F}_j| - 1\}$ do
12:             $\boldsymbol{x}[\mathcal{F}_j[u]] \leftarrow \boldsymbol{b}[u]$
```

introduced in Bouter, Alderliesten, Witteveen et al. (2017), and the second application uses a sampling model based on CMA-ES (Hansen and Ostermeier, 1996, 2001). We refer to the former as RV-GOMEA$^{\text{A}}$, and to the latter as RV-GOMEA$^{\text{C}}$. The combination of GOMEA and CMA-ES was selected because CMA-ES is a well-known algorithm that is arguably the state-of-the-art of EAs for real-valued BBO. Both in AMaLGaM and in CMA-ES a multivariate normal probability distribution is used to sample new variables, but a different approach to determining the parameters of this probability distribution is taken in these algorithms. The AMS and forced improvement procedures are applied to both variants of RV-GOMEA, as displayed in Algorithm 1.

7.1 AMaLGaM Sampling Model

We now define the structure and parameters of the sampling model used by this instance of RV-GOMEA, based on AMaLGaM (Bosman et al., 2013). For the sampling model of each FOS element, a multivariate normal probability distribution is maintained, with each of the dimensions of this probability distribution corresponding to one of the variables included in the linkage set. This probability distribution $\mathcal{N}\left(\boldsymbol{\mu}_j, \boldsymbol{C}_j\right)$ is defined in terms of the mean vector $\boldsymbol{\mu}_j$ and the covariance matrix $\boldsymbol{C}_j$, describing the means of all variables in $\mathcal{F}_j$, and the covariances of all pairs of variables in $\mathcal{F}_j$, respectively. Each such a probability distribution is estimated with maximum-likelihood based on the selection, and then multiplied by the distribution multiplier $c_j^{\texttt{Mul}}$, which is described in Section 7.1.3. The selection $\mathcal{S}$ consists of the τn best solutions in the population, with $\tau = 0.35$, as in Bosman et al. (2013). Selection and the estimation of the probability distribution are performed during the initialization of each sampling model, and during the update procedure of each sampling model. Pseudocode of this instance of RV-GOMEA is included in supplementary material, available at https://www.mitpressjournals.org/doi/suppl/10.1162/evco_a_00275.

7.1.1 Gene-Pool Optimal Mixing

The normal probability distribution that is maintained, is used to sample new partial solutions, that is, values for subsets of variables, during the GOM phase. In this phase, for each linkage set, GOM is applied to all but $n^{\texttt{elitist}} = 1$ solutions in the population. During the GOM phase of a linkage set $\mathcal{F}_j$, partial variation is applied to all solutions in the population by sampling new partial solutions describing new values for all variables in $\mathcal{F}_j$. In order to sample from the distribution $\mathcal{N}\left(\boldsymbol{\mu}_j, \boldsymbol{C}_j\right)$, a Cholesky decomposition is applied to $\boldsymbol{C}_j$ to find $\boldsymbol{C}_j = \boldsymbol{L}_j \boldsymbol{L}_j^*$. A sample $\boldsymbol{o} \sim \mathcal{N}\left(\boldsymbol{\mu}_j, \boldsymbol{C}_j\right)$ is then obtained through

$$\boldsymbol{o} = \boldsymbol{\mu}_j + \boldsymbol{L}_j \cdot \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}). \tag{5}$$

The resulting offspring solution is then evaluated, and the modification of the parent solution is accepted if the offspring has a better objective value than the parent. If no better objective value is achieved by the offspring, the modification is accepted with a probability of $p^{\texttt{accept}} = 0.05$, based on preliminary experiments, because this encourages exploration and increases the likelihood of escaping local optima.

7.1.2 Anticipated Mean Shift

A fraction 0.5τ, corresponding to Bosman et al. (2013), of newly sampled partial solutions is subject to the AMS, moving them in the direction of generational improvement. This is done by adding the difference between the means of consecutive generations, times a multiplicative factor, to the partial solution $\boldsymbol{o}$, that is,

$$\boldsymbol{o}^{\texttt{AMS}} = \boldsymbol{o} + \delta^{\texttt{AMS}} c_j^{\texttt{Mul}} \left(\boldsymbol{\mu}_j^{(g)} - \boldsymbol{\mu}_j^{(g-1)}\right), \tag{6}$$

where $\delta^{\texttt{AMS}} = 2$, $\boldsymbol{\mu}_j^{(g)}$ is the mean vector of all variables in $\mathcal{F}_j$ in generation g, and $c_j^{\texttt{Mul}}$ is the distribution multiplier that is described in Section 7.1.3.

7.1.3 Adaptive Variance Scaling

Each covariance matrix is scaled by a distribution multiplier that is used to adaptively control the kernel size of the probability distribution. The procedure of adapting the distribution multiplier is called Adaptive Variance Scaling (AVS) (Bosman et al., 2013). Parameters of the AVS are set corresponding to Bosman et al. (2013). The value of the distribution multiplier is initialized to 1 and scaled by either the factor $\eta^{\texttt{DEC}} = 0.9$ or the factor $\eta^{\texttt{INC}} = 1/\eta^{\texttt{DEC}}$ at the end of each generation, depending on the success of the optimization procedure. More specifically, to update the distribution multiplier $c_j^{\texttt{Mul}}$ of a sampling model $\mathcal{M}_j$, we first find the set of solutions $\boldsymbol{X}_j^{\texttt{Improved}}$. This set contains the offspring solutions that were produced by the GOM phase of $\mathcal{M}_j$ and obtained an objective value better than that of the elitist solution at the start of the GOM phase of $\mathcal{M}_j$. If no such improvements were found, i.e., $\boldsymbol{X}_j^{\texttt{Improved}} = \emptyset$, $c_j^{\texttt{Mul}}$ is multiplied by the factor $\eta^{\texttt{DEC}}$ to narrow down the size of the search space.

If $\boldsymbol{X}_j^{\texttt{Improved}}$ is non-empty, $c^{\texttt{Mul}}$ is first reset to 1 if it was smaller than 1. Subsequently, $c^{\texttt{Mul}}$ is adapted if improvements were found far away from the sample mean, determined by the Standard Deviation Ratio (SDR) approach. For this purpose, the vector $\boldsymbol{x}_j^{\texttt{Avg-imp}}$ defines, for all variables in $\mathcal{F}_j$, the average values of all solutions in $\boldsymbol{X}_j^{\texttt{Improved}}$. A transformation is applied to $\boldsymbol{x}_j^{\texttt{Avg-imp}}$ in order to find the vector of standard deviation ratios $\boldsymbol{z}_j^{\texttt{Avg-imp}} = \boldsymbol{L}_j^{-1}\left(\boldsymbol{x}_j^{\texttt{Avg-imp}} - \boldsymbol{\mu}_j\right)$, which describes the average direction of improvement in terms of the standard normal probability distribution $\mathcal{N}(0, 1)$ for each variable. If any dimension of $\boldsymbol{z}_j^{\texttt{Avg-imp}}$ is larger than $\theta^{\texttt{SDR}} = 1$ in absolute terms, the average improvement is considered to be far away from the sample mean, resulting in the multiplication of $c_j^{\texttt{Mul}}$ by the factor $\eta^{\texttt{INC}}$, because the kernel of the probability distribution should be enlarged to find solutions far away from the mean.

7.2 CMA-ES Sampling Model

In this section we describe our implementation of a CMA-ES-based (Hansen and Ostermeier, 1996, 2001) sampling model in RV-GOMEA. Pseudocode of this instance of RV-GOMEA is included in supplementary material.

Each sampling model $\mathcal{M}_j$ consists of all parameters of CMA-ES as in Hansen and Ostermeier (2001), modeling the probability distribution of the set of variables in $\mathcal{F}_j$. Each of these sampling models based on CMA-ES maintains the parameters that define the $\mathcal{F}_j$-dimensional normal probability distribution $\mathcal{N}(\boldsymbol{\mu}_j, \sigma_j^2 \boldsymbol{C}_j)$ from which new solutions are sampled, where $\boldsymbol{\mu}_j$ is the mean vector, $\boldsymbol{C}_j$ is the covariance matrix, and σ_j is the step size parameter. All these parameters are updated at the start of each generation, separately for each sampling model.

7.2.1 Parameter Adaptation

In order to update μ and $\boldsymbol{C}$, truncation selection is performed to find the selection $\mathcal{S}$ consisting of the best $\lfloor \tau n \rfloor$ individuals in $\mathcal{P}$. The vectors $\langle \boldsymbol{x} \rangle_w$ and $\langle z \rangle_w$ are the weighted average of the solutions in $\mathcal{S}$ and the weighted average of the $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$-distributed samples z of these individuals, respectively, which are computed based on the selection. The weight of a solution with rank $0 \leq r \leq n-1$ is $\log(|\mathcal{S}|+1) - \log(r+1)$, and all weights are normalized such that their sum is equal to 1.

The evolution path $\boldsymbol{p}_c$ is then adapted, which determines how the covariance matrix $\boldsymbol{C}$ is adapted. This is done according to Hansen and Ostermeier (2001), that is,

$$\boldsymbol{p}_c^{(g+1)} = (1-c_c) \cdot \boldsymbol{p}_c^{(g)} + c_c^u \cdot \frac{c_w}{\sigma^{(g)}} \left(\langle \boldsymbol{x} \rangle_w^{(g+1)} - \langle \boldsymbol{x} \rangle_w^{(g)} \right), \tag{7}$$

$$\boldsymbol{C}^{(g+1)} = (1-c_{\text{cov}}) \cdot \boldsymbol{C}^{(g)} + c_{\text{cov}} \cdot \boldsymbol{p}_c^{(g+1)} \left(\boldsymbol{p}_c^{(g+1)} \right)^{\mathrm{T}}, \tag{8}$$

with c_c the cumulation constant, and c_c^u and c_w normalization factors. These parameters are set corresponding to an $|\mathcal{F}_j|$-dimensional instance of CMA-ES Hansen and Ostermeier (2001).

The step size parameter σ is adapted based on a separate evolution path $\boldsymbol{p}_\sigma$ in the following way, according to Hansen and Ostermeier (2001):

$$\boldsymbol{p}_\sigma^{(g+1)} = (1-c_\sigma) \cdot \boldsymbol{p}_\sigma^{(g)} + c_\sigma^u \cdot c_w \boldsymbol{B}^{(g)} \langle z \rangle_w^{(g+1)}, \tag{9}$$

$$\sigma^{(g+1)} = s\sigma^{(g)} \cdot \exp\left(\frac{1}{d_\sigma} \cdot \frac{\|\boldsymbol{p}_\sigma^{(g+1)}\| - \widehat{\chi}_j}{\widehat{\chi}_j} \right), \tag{10}$$

with c_σ the cumulation constant, c_σ^u a normalization factor, d_σ the damping parameter, and $\widehat{\chi}_j$ (an approximation of) the expected length of a vector drawn from the $|\mathcal{F}_j|$-dimensional normal distribution $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. These parameters are set corresponding to an $|\mathcal{F}_j|$-dimensional instance of CMA-ES (Hansen and Ostermeier, 2001).

The update rule for the evolution path of σ is dependent on $\boldsymbol{B}\langle z \rangle_w$, that is, the rotation matrix $\boldsymbol{B}$ and the weighted mean of the sample vector z, because this indicates the direction of improvement. However, $\boldsymbol{B}$ is updated every generation, while z could have been sampled more than one generation ago, because later samples were rejected. Additionally, the weight of each solution that contributes to $\langle z \rangle_w$ changes every generation, because it depends on the relative quality of the solution. This means that, if $\boldsymbol{B}$ and z originate from a different generation, the product $z\boldsymbol{B}$ results in a vector unrelated to the direction of improvement. Instead z and $\boldsymbol{B}$, both originating from the same generation, would have to be stored until all values of z have been replaced by newer

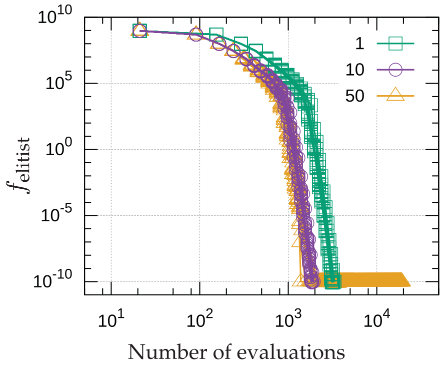


Figure 1: Convergence plots of typical runs of RV-GOMEA$^{\text{C}}$ on the 80960-dimensional sphere problem, with different settings for the number of generations after which the complete population is reevaluated.

samples. This would increase the overall memory complexity to $\mathcal{O}(n\ell^2)$, which should be avoided. Therefore we set the value of z to 0 if the sample in the current generation is rejected, because this would lead to a mismatch between z and $\boldsymbol{B}$.

7.2.2 Gene-Pool Optimal Mixing

To sample from the distribution with covariance matrix $\boldsymbol{C}$, an eigendecomposition is applied to find $\boldsymbol{C} = \boldsymbol{B}\boldsymbol{D}^2\boldsymbol{B}^T$, where $\boldsymbol{B}$ is an orthogonal matrix that determines the coordinate system, and $\boldsymbol{D}$ is a diagonal matrix that scales the dimensions of this coordinate system accordingly. For a sampling model $\mathcal{M}_j$, a $|\mathcal{F}_j|$-dimensional partial solution $\boldsymbol{o}$ is sampled from $\mathcal{N}(\boldsymbol{\mu}, \sigma^2\boldsymbol{C})$ as follows:

$$\boldsymbol{o}^{(g+1)} = \langle \boldsymbol{x} \rangle_w^{(g)} + \sigma^{(g)}\boldsymbol{B}^{(g)}\boldsymbol{D}^{(g)}\boldsymbol{z}^{(g+1)}, \tag{11}$$

with $\boldsymbol{z}^{(g+1)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$.

As discussed in Section 6, the population has to be periodically reevaluated to avoid large numerical errors. However, reevaluating the population once every 50 generations, as is done in RV-GOMEA$^{\text{A}}$, appeared insufficient in preliminary experiments to efficiently solve high-dimensional problems with RV-GOMEA$^{\text{C}}$, because the numerical errors have too much of an influence on the optimization process. This can be seen in the convergence plots in Figure 1. Figure 1 shows runs of RV-GOMEA$^{\text{C}}$ with three different settings for the number of generations after which the complete population is reevaluated. Note that all runs took fewer than 50 generations, so the population was never reevaluated for the setting with reevaluation every 50 generations. If the population is reevaluated every 50 generations, RV-GOMEA$^{\text{C}}$ gets "stuck" near the VTR, because each solution that appears to reach the VTR is reevaluated, but then turns out to be worse than the VTR due to numerical errors. This is the cause of a large inefficiency, most notably in highly dimensional problems, and can even cause premature convergence. For RV-GOMEA$^{\text{C}}$ we therefore choose to reevaluate the population every 10 generations, as this is a reasonable setting given the results in Figure 1.

7.3 Anticipated Mean Shift

Similar to the AMS applied to partial solutions in Section 7.1.2, AMS is applied to all variables of a fraction 0.5τ of the population directly following GOM, with the purpose of moving solutions in the direction of generational improvement for all problem variables simultaneously. This application of AMS is accepted only when it leads to an improvement, or with a probability of p^{accept}. AMS is applied as follows:

$$\boldsymbol{x}^{\texttt{AMS}} = \boldsymbol{x} + \delta^{\texttt{AMS}} \left(\boldsymbol{\mu}^{(g)} - \boldsymbol{\mu}^{(g-1)}\right), \tag{12}$$

where $\delta^{\texttt{AMS}} = 2$, and $\boldsymbol{\mu}_j^{(g)}$ is the mean vector of all variables in generation g.

7.4 Forced Improvements

A phase of forced improvements was introduced in the discrete GOMEA to force solutions out of local optima if they have not improved for a certain number of generations. For each solution $\boldsymbol{x}$, the number of generations of no improvement NIS$(\boldsymbol{x})$ is kept track of. This counter is reset whenever GOM or AMS resulted in an improvement of $\boldsymbol{x}$. Note that NIS$(\boldsymbol{x})$ is not reset when a step of GOM is accepted due to $p^{\texttt{accept}}$. Whenever NIS$(\boldsymbol{x})$ reaches NIS$^{\texttt{MAX}} = 100$, the forced improvement phase is applied to $\boldsymbol{x}$. This phase consists of multiple rounds, during each of which GOM is applied to $\boldsymbol{x}$ for each linkage set. However, all new partial solutions that are inserted into $\boldsymbol{x}$ are a weighted average of the variables of $\boldsymbol{x}$ and the elitist solution $\boldsymbol{x}^{\texttt{elitist}}$, where the weight of $\boldsymbol{x}$ is α and the weight of $\boldsymbol{x}^{\texttt{elitist}}$ is $1 - \alpha$. If the modification of $\boldsymbol{x}$ leads to an improvement of its objective value, this modification is accepted and the forced improvement phase terminates. After one round of GOM operations has been applied to $\boldsymbol{x}$, but no improvements have been found, the value of α is multiplied by 0.5, and the next round of GOM operations starts. This process continues until α has reached a value below 0.01, at which point a copy of $\boldsymbol{x}^{\texttt{elitist}}$ replaces $\boldsymbol{x}$ in the population.

8 Multiobjective Optimization

Optimization problems often involve multiple conflicting objective functions that require optimization. Again, without loss of generality, we assume that all objective functions need to be minimized. A weighted average of multiple objective functions could be optimized by a single-objective algorithm, but such an approach might not always result in the desired outcome, because such weights can be difficult to tune. Instead, the goal of multiobjective optimization is finding a set of solutions, each of which has a different optimal trade-off between each of the objective functions.

In the setting of multiobjective optimization, a solution $\boldsymbol{x}$ is considered to dominate a solution $\boldsymbol{y}$, denoted $\boldsymbol{x} \succ \boldsymbol{y}$, if it is better than $\boldsymbol{y}$ in at least one objective function, and at least as good as $\boldsymbol{y}$ in all other objective functions. If neither $\boldsymbol{x} \succ \boldsymbol{y}$ nor $\boldsymbol{y} \succ \boldsymbol{x}$ holds, $\boldsymbol{x}$ and $\boldsymbol{y}$ are said to be non-dominating. A solution is Pareto optimal if no solution exists that dominates it. All Pareto-optimal solutions comprise the so-called Pareto set. In objective space, the solutions in a Pareto set form a so-called Pareto front. EAs are known to be among the state-of-the-art in multiobjective optimization (Deb, 2001). Multiobjective EAs produce a so-called approximation set, a set of solutions that is as close as possible to the Pareto front, and is spread across all regions of the Pareto front.

8.1 Multiobjective RV-GOMEA

Extending the RV-GOMEA introduced in Section 3 from single-objective optimization to multiobjective optimization is relatively simple with the multiobjective framework introduced by Bosman and Alderliesten (2012). This framework employs selection based on non-dominated sorting and a clustering procedure in order to spread the search effort across different regions of the objective space, because the goal of multiobjective optimization is to find many solutions, presumably consisting of widely varying problem variable values, near each region of the Pareto front. Clustering was previously shown to be beneficial for finding a good spread of solutions across the regions of the Pareto front (Pelikan et al., 2005). Bosman and Alderliesten (2012) applied the aforementioned multiobjective framework to AMaLGaM, resulting in the Multiobjective AMaLGaM (MAMaLGaM) where basically each cluster is provided its own AMaLGaM estimation and sampling procedure. Cluster registration, that is, the one-to-one matching of clusters in generation g and those in generation $g+1$, is performed, in order to apply various mechanisms that rely on parameter settings from multiple generations, such as AMS and AVS.

Similar to how the multiobjective framework is applied to AMaLGaM, we can apply it to RV-GOMEA by maintaining a sampling model for each cluster. Each cluster should also have the possibility of having a different linkage model, because different dependencies could be of importance in different regions of the Pareto front. Moreover, even if the same linkage model is used by each cluster, the optimal distribution variable values of a specific linkage set are presumably different at different regions of the Pareto front. Therefore, a model $\mathcal{M}_{ij}$ is maintained for each linkage set $\mathcal{F}_j$ in each cluster $\mathcal{C}_i$.

8.1.1 Clustering

To distinguish the different regions of the Pareto front that are to be approached by different clusters, the selection $\mathcal{S}$ is clustered into q possibly overlapping clusters, each consisting of $c = 2\tau\frac{n}{q}$ solutions. All distances in the clustering procedure are Euclidean distances in objective space, where each dimension is normalized by the range of the current selection. A so-called single-objective cluster is first created for each objective, each consisting of the best c solutions in that objective. To establish the remaining $q-m$ clusters, $q-m$ far apart cluster leaders are heuristically chosen from the selection. The first cluster leader is a solution that is the maximum in a randomly chosen dimension. All $q-m-1$ remaining cluster leaders are then iteratively selected by choosing the solution that is furthest away from all previously selected cluster leaders. The process of selecting a subset of far apart solutions is further on referred to as scattered subset selection. Cluster members are then determined by assigning the closest c solutions in the selection to each cluster leader, meaning that some solutions in the selection can be assigned to more than one cluster, or to no cluster.

In contrast to MAMaLGaM, where only the selection $\mathcal{S}$ is clustered, the application of the multiobjective framework to RV-GOMEA will require the clustering of the entire population, because each solution in the population must receive new samples from a sampling model that is directly associated with a cluster. After clusters consisting of solutions in $\mathcal{S}$ have been established, all solutions in the population (including the selection) are assigned to exactly one cluster. To ensure a minimum size of each cluster, we first assign $c = 2\tau\frac{n}{q}$ solutions to each cluster in a round robin fashion. In each round, for each cluster, the closest (with respect to the cluster mean) non-assigned solution is assigned to the cluster, starting with the single-objective clusters. This process is repeated until c solutions are assigned to each cluster. All non-assigned solutions are then assigned to their closest (again with respect to the cluster mean) cluster.

8.1.2 Elitist Archive

An adaptive elitist archive, introduced in Luong and Bosman (2012), is used to keep track of a set of non-dominated solutions with a desired target size. This elitist archive is set to a maximum capacity of 125% of its target size, and the elitist archive is adapted when the maximum capacity is exceeded. This adaptation condition is checked at the end of each generation. While the elitist archive has not been adapted yet, this condition is also checked after each application of GOM, to prevent the elitist archive from growing substantially beyond its maximum capacity. The objective space of the elitist archive is then discretized into a regular grid, and no more than one solution is allowed in each grid cell. Different grid resolutions are attempted in a binary search manner to find a grid resolution such that the size of the elitist archive is close to 75% of its target size. Whenever a solution would be added to the elitist archive, and it would be located in an already occupied grid cell, either the new solution or the solution preexisting in this cell is selected to be stored in the archive. If either of the solutions is dominating the other, the dominating solution is selected. Otherwise, the preexisting solution remains in the archive.

8.1.3 Mixing

Accepting or rejecting the modification of a solution during GOM is based on whether the modification resulted in an improvement of the solution in question. Contrary to single-objective optimization, however, an improvement is not clearly defined in multiobjective optimization. A modified solution that dominates the previous state of the solution clearly counts as an improvement, but little can be said when the two solutions are non-dominating. A modification leading to a non-dominating state could very well lead to a solution that is further from the Pareto front. We therefore adopt the acceptation criteria that were previously used in the (discrete) Multi-Objective GOMEA (MO-GOMEA) (Luong et al., 2014). This means that a modification is accepted only if it leads to a solution that dominates the parent solution, or if the resulting solution is not dominated by any solution in the elitist archive. In the multiobjective framework, we set $p^{\texttt{accept}} = 0$, as the acceptation of sideways (non-dominating) steps serves the same purpose.

8.2 AMaLGaM Sampling Model

The adaptation of GOMEA for multiobjective real-valued optimization was previously introduced by Bouter, Luong et al. (2017) as the Multi-Objective RV-GOMEA (MO-RV-GOMEA) and uses the AMaLGaM sampling model.

8.3 CMA-ES Sampling Model

Extending RV-GOMEA$^{\mathrm{C}}$ to its multiobjective variant, MO-RV-GOMEA$^{\mathrm{C}}$, is done through the straightforward application of the multiobjective framework discussed in Section 8.1 and thus in the same way that RV-GOMEA$^{\mathrm{A}}$ is extended to MO-RV-GOMEA$^{\mathrm{A}}$. A separate linkage model is used for each cluster, and each linkage model in each cluster maintains a sampling model for each of its linkage sets. All methods discussed in Section 8.1 are then applied.

9 Experiments

9.1 Benchmark Problems

This section introduces the set of single-objective and multiobjective benchmark problems used in this article. Definitions and further details of these problems are included as supplementary material.

9.1.1 Single-Objective

The set of single-objective optimization problems consists of the sphere (De Jong, 1975), Rosenbrock (Rosenbrock, 1960), Rastrigin (Rastrigin, 1974), Michalewicz (Michalewicz and Janikow, 1991), SoREB (Bouter, Alderliesten, Witteveen et al., 2017) functions, and a step function, all subject to minimization. The partial evaluation of these functions is generally performed by subtracting the previous contribution of the modified variables from the objective value, and adding the new contribution of such variables to the objective value.

From this set of functions, the sphere function is clearly the easiest to optimize, because it has no (non-global) local optima, and it is completely dimension-wise decomposable. The second benchmark problem that we consider is the Rosenbrock function, because this is a relatively difficult function to which partial evaluations can be applied despite the fact that it is not dimension-wise decomposable. The Rastrigin and Michalewicz functions are multimodal problems that are dimension-wise decomposable. The SoREB function, previously used by Bouter, Alderliesten, Witteveen et al. (2017), is also used, as this is a decomposable problem with relatively difficult subproblems. Finally, we use a step-function variant of the sphere function, because the landscape of this function has many plateaus, a landscape characteristic that may well occur in real-world problems that should still be possible to overcome with population-based optimizers, but pose difficulty for gradient-based solvers.

9.1.2 Multiobjective

The set of multiobjective benchmark problems includes the convex generalized Multiple Euclidean Distances (genMED) problem (Bosman, 2012), the well-known ZDT1 and ZDT3 problems (Deb, 1999), and a multiobjective variant of the SoREB function, named MOSoREB.

The genMED problem is a relatively simple problem with a convex Pareto front. We specifically selected the ZDT1 and ZDT3 problems for their distinct properties, as the Pareto fronts for these problems are convex and discontinuous, respectively. Both the ZDT1 and ZDT3 problems are non-separable (Li et al., 2016).

Finally, the MOSoREB problem consists of the objective function f_0 for which each value is optimal, and an objective function f_1 that is similar to the SoREB function, but conflicts with f_0. The Pareto front of the MOSoREB problem is a straight line between (0,1) and (1,0).

9.2 Experimental Setup

For all variants of GOMEA, we use the Interleaved Multi-start Scheme (IMS) (Bouter, Alderliesten, Witteveen et al., 2017) to avoid tuning the population size and the number of clusters. With this scheme, one run of an EA consists of several independent instances of the EA with population sizes exponentially growing in size. The generations of these independent instances are interleaved, such that the instance with population size $2n$ performs one generation for each c^{IMS} generations of the instance with population size n. For the multiobjective EAs, the number of clusters is increased by 1 for each new population that is started. The IMS starts with one instance of an EA with a relatively small population size n^{base} and q^{base} clusters (in MO optimization). Each consecutive instance that is started has a population size that is a factor 2 larger, and has one more cluster (in MO optimization). For all experiments $c^{\text{IMS}} = 8$, and $q^{\text{base}} = 5$ (for MO problems) are used. For the SoREB problem we use $n^{\text{base}} = 50$, and for the MOSoREB problem we use $n^{\text{base}} = 50q^{\text{base}}$, conform to the AMaLGaM population-size guideline for a 5-dimensional problem (Bosman et al., 2013). For all other Single-Objective (SO) experiments $n^{\text{base}} = 10$ is used, and for all other MO experiments $n^{\text{base}} = 10q^{\text{base}}$ is used.

Random restarts are used with L-BFGS, that is, a run is restarted when convergence is observed while the computational budget has not yet been expended.

An initialization range of $[-115, -100]$ is used for all SO benchmark problems, as this range does not bracket the optimum. An initialization range of [0,1] is used for all MO benchmark problems, as some of these problems are constrained to this range. Boundary repair is used as a constraint handling mechanism (Orvosh and Davis, 1994).

For the sphere, Rosenbrock, Rastrigin, and SoREB problems we use a VTR of 10^{-10}. For the Michalewicz problem the VTR is set to 95% of the global optimum. In MO optimization, we use the $D_{\mathcal{P}_f \to S}$ metric (Bosman and Thierens, 2003), also known as the inverted generational distance, to evaluate the quality of an approximation set. The $D_{\mathcal{P}_f \to S}$ is the average distance in objective space computed over each point in an approximation set of 5000 equally spread out points on the Pareto front $\mathcal{P}_f$ with its associated nearest point in a given approximation set S. Using this metric requires that the Pareto front is known, but this is the case for all used MO benchmark problems. For all MO benchmark problems, we use a VTR of $D_{\mathcal{P}_f \to S} < 5 \cdot 10^{-3}$. An adaptive elitist archive is used with a target size of 1000.

For all problems except the SoREB and MOSoREB problems, the partial evaluation of k variables is counted as a fraction k/ℓ of an evaluation, because the computational effort for such a partial evaluation takes $\mathcal{O}(k/\ell)$. For the SoREB and MOSoREB problems, the computational effort of a partial evaluation depends on the number of blocks that have to be evaluated. Therefore, the partial evaluation of any number of variables in k different blocks of size β is counted as $k\beta/\ell \leq 1$ evaluations. Note that the partial evaluation of a set of variables consists of the calculation of their contribution to the objective value before modification, and the calculation of their contribution to the objective value after modification. However, if the objective function is computationally expensive, one can choose to save partial contributions to the objective function in memory, which has no impact on the overall memory complexity. For this reason, a partial evaluation is counted as a fraction k/ℓ of an evaluation instead of $2k/\ell$.

Experimental results for RV-GOMEA$^{\text{A}}$ were obtained by running experiments on an Intel(R) Core(TM) i7-2600 CPU @ 3.40 GHz. Experimental results for MO-RV-GOMEA$^{\text{A}}$, (MO-)RV-GOMEA$^{\text{C}}$, L-BFGS, and LM-CMA-ES were obtained by running experiments on an Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20 GHz. Reported times of RV-GOMEA$^{\text{A}}$ are scaled by a constant to compensate for the difference in CPU clock speed.

9.3 Comparison to Gradient-Based Optimization

If each subfunction has a dependency with up to k problem variables, the gradient can be estimated in up to k evaluations in a GBO setting. This makes gradient-based optimization methods, such as L-BFGS (Liu and Nocedal, 1989), very efficient in a GBO setting. However, though a GBO setting provides information on the separability of the optimization problem, each subfunction is considered as a black box. As any such subfunction can have a landscape that is arbitrarily difficult to optimize, gradient-based methods can still have difficulties optimizing such problems.

Figure 2 shows convergence plots of L-BFGS for the 40-dimensional Rastrigin problem, given different step sizes for the estimation of the gradient. The open source Python library SciPy (Jones et al., 2001) was used for the implementation of L-BFGS. Figure 2 shows that L-BFGS is unable to reach the global optimum of the Rastrigin function, caused by the multimodality of this problem. In contrast, EAs are generally well-suited

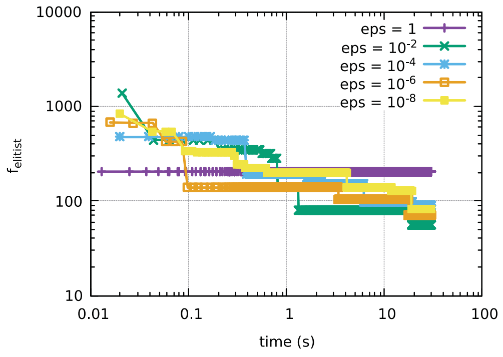


Figure 2: Convergence plot for the 40-dimensional Rastrigin problem using L-BFGS with different step size (eps) values for the estimation of the gradient. The vertical axis shows the best objective value found following random restarts of L-BFGS. A total time limit of 30 seconds was used.

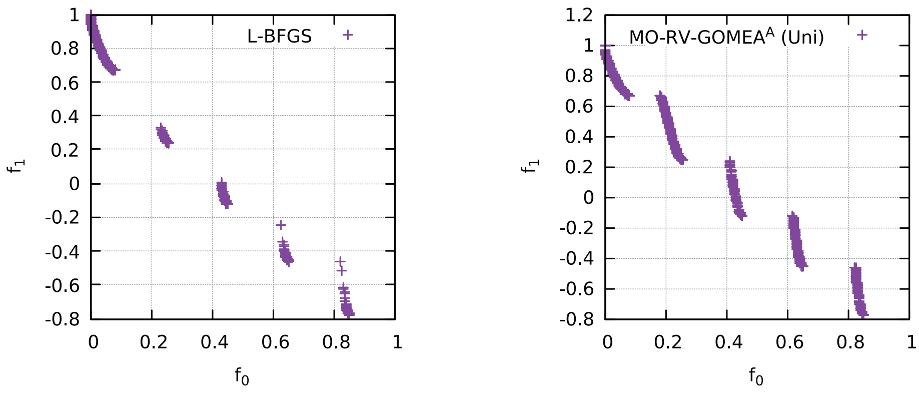


Figure 3: Resulting approximation fronts for L-BFGS and MO-RV-GOMEA^A for the 10-dimensional ZDT3 problem. A time limit of 60 seconds was used. The front found by L-BFGS was constructed through repeated optimization of linear combinations of the two objectives.

for such multimodal problems. This is also the case for the variants of GOMEA, as shown in Figure 4.

L-BFGS can also be used for MO through repeated optimization of linear combinations of the objectives. This, however, makes it difficult to achieve good coverage of the Pareto front for problems such as ZDT3, as shown in Figure 3. In this case, weights of the initial two runs were set to 0.99 and 0.01, and weights of subsequent runs were set to the middle of the largest range of weights not yet covered, as to perform the optimization with a wide range of different weights. Additionally, also in MO, L-BFGS has trouble optimizing difficult problem landscapes, for example, multimodal problem landscapes.

9.4 Scalability Analysis

As computing power increases and the demand for the optimization of large-scale problems increases, it becomes increasingly important to analyze how well algorithms scale to high-dimensional optimization problems. For this reason, we perform a scalability analysis of all discussed algorithms in a GBO setting using different linkage models on a set of well-known single-objective and multiobjective benchmark problems.

We analyze the scalability of RV-GOMEA^A, RV-GOMEA^C, MO-RV-GOMEA^A, and MO-RV-GOMEA^C in a GBO setting by applying these algorithms to all benchmark problems introduced in Section 9.1, with exponentially increasing dimensionality. We compare these variants of GOMEA to the Limited Memory CMA-ES (LM-CMA-ES) (Loshchilov, 2014), as it was shown that this EA scales the best to high dimensions compared to various large-scale variants of CMA-ES (Varelas et al., 2018). All default parameter settings are used, and the EA is restarted with double the population size when the convergence criterion has been reached. The comparison to LM-CMA-ES should give an indication of the benefits of a GBO setting compared to a BBO setting in terms of scalability, as LM-CMA-ES is also designed to perform well on large-scale optimization problems. We also compare to L-BFGS (Liu and Nocedal, 1989), as implemented in the SciPy (Jones et al., 2001) library. This implementation was modified to allow for the more efficient estimation of the gradient through partial evaluations. For the application of L-BFGS to MO benchmark problems, we repeatedly optimize weighted linear combinations of the objective functions.

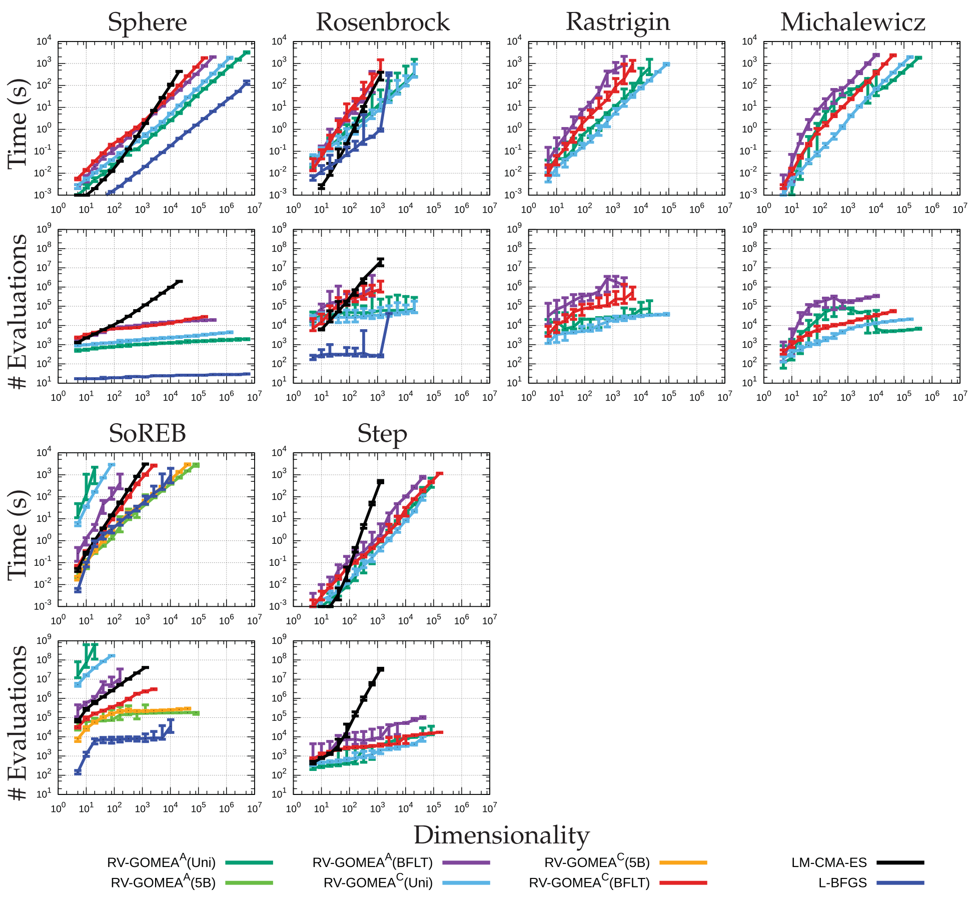
All variants of GOMEA are tested with the Univariate factorization (Uni) and a BFLT linkage model (see Section 4.1) with an upper bound of 100 on the size of each linkage set. The LT model is not used, because including the linkage set of size ℓ should be avoided to achieve good scalability. A 5-Block FOS (5B) linkage model is also tested on the SoREB and MOSoREB problems. For each dimensionality, 30 independent runs of each EA are performed with a time limit of one hour. A run is considered successful when the EA finds the VTR within this time limit. As long as an EA is successful in all 30 runs for a problem of a certain dimensionality, we keep increasing the problem dimensionality by a factor 2 until the algorithm is no longer successful in all 30 runs, up to a dimensionality of 10^7 for the SO benchmark problems and 10^5 for the MO benchmark problems.

9.4.1 Results

The results of all SO and MO scalability experiments are displayed in Figure 4.

RV-GOMEA^A and RV-GOMEA^C achieve very similar performance on most benchmark problems. Some differences in scalability can be observed for the Michalewicz and SoREB problems.

On the Michalewicz problem, the univariate RV-GOMEA^A scales relatively poorly up to approximately 10^3-dimensional problems. On higher-dimensional instances, RV-GOMEA^A, however, achieves scalability similar to the sphere problem. This is caused by the fact that a VTR equal to a fraction (95%) of the global optimum is used, which becomes easier to reach as the dimension of the Michalewicz problems increases, counteracting the fact that the optimization problem becomes more difficult as its dimensionality increases. A changing trend in the scalability of RV-GOMEA^C is not observed, because RV-GOMEA^C is able to consistently solve the Michalewicz problem with a relatively small population size, whereas RV-GOMEA^A requires a larger population size to efficiently solve this problem for medium-sized dimensionalities. This can also be observed through generally smaller variance in the results of RV-GOMEA^C compared to RV-GOMEA^A.



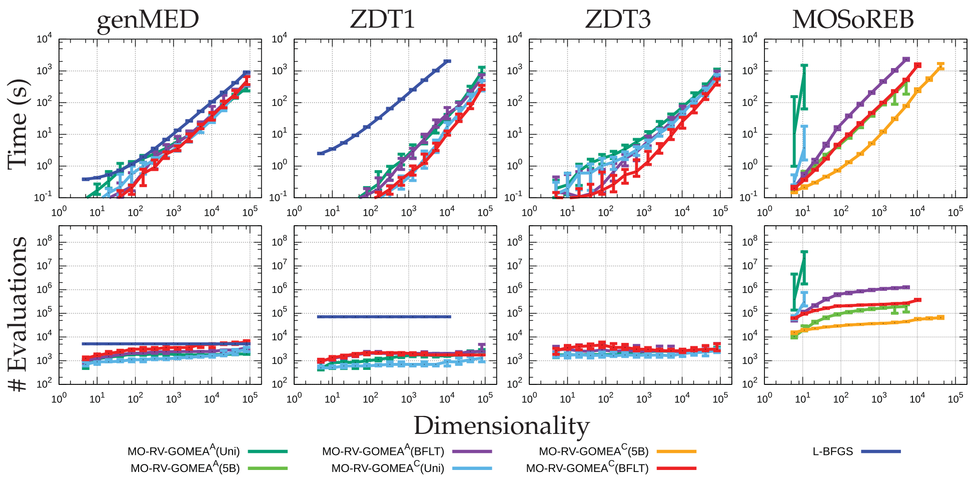


Figure 4: Medians and interdecile ranges of all GBO scalability experiments with each data point being the median of 30 successful runs.

Table 1: Statistical comparison of RV-GOMEA^A and RV-GOMEA^C for all single-objective benchmark problems, for the highest dimensionality achieved by both algorithms in Figure 4. A cell marked by a letter (A or C) indicates that this variant of RV-GOMEA performed statistically significantly better (in terms of function evaluations) than the other variant, for the given linkage model and benchmark problem. The number of asterisks indicates the level of significance, with 0 to 3 asterisks indicating p-values smaller than 10^{-2}, 10^{-3}, 10^{-4}, and 10^{-5}, respectively.

	Sphere	Rosenbrock	Rastrigin	Michalewicz	SoREB	Step
Uni	A***		C***	A***	C***	C
BFLT	A***		C***	C***	C***	C***
B5					A***	

On the SoREB problem, the scalability of RV-GOMEA^A and RV-GOMEA^C is largely the same when a 5B linkage model is used. RV-GOMEA^C, however, scales better than RV-GOMEA^A when a BFLT model is used, due to the fact that CMA-ES inherently scales better than AMaLGaM in case a full covariance matrix is used and the BFLT has linkage sets up to 100 variables.

LM-CMA-ES was unsuccessful at consistently solving the Rastrigin and Michalewicz problems for any dimensionality. Both of these problems are highly multimodal. On all other problems, the scalability of LM-CMA-ES was substantially worse than that of all GOMEA variants, showing the benefits of a GBO setting for GOMEA.

L-BFGS unsurprisingly had the best performance on the sphere function, but failed to solve the Rastrigin, Michalewicz and step functions for any number of dimensions. On the Rosenbrock problem, L-BFGS was very efficient, but suffered from errors in numerical precision for high-dimensional problems. The performance of L-BFGS on the SoREB problem was very similar to RV-GOMEA variants with a 5B FOS in terms in time. Note that a Python implementation of L-BFGS is used, whereas the competing algorithms are implemented in C, affecting computation time results in an absolute sense, but not in terms of scalability. A more efficient implementation of L-BFGS may therefore be expected to achieve better computation times than RV-GOMEA on the SoREB, genMED, and possibly ZDT1 problems.

On the set of MO benchmark problems, MO-RV-GOMEA^C achieves the best performance on the genMED, ZDT1, and ZDT3 problems. MO-RV-GOMEA^C and MO-RV-GOMEA^A generally achieve the same scalability when the same linkage model is used. MO-RV-GOMEA^A achieves slightly better performance than MO-RV-GOMEA^C on the MOSoREB problem, but both achieve the same scalability. Though the ZDT1 and ZDT3 problems are non-separable (Li et al., 2016), the performance of different linkage models is very similar for these problems.

L-BFGS performs very well on the easy genMED problem, but is outperformed by GOMEA on the ZDT1 and ZDT3 problems. L-BFGS was unable to reach the VTR on the ZDT3 and MOSoREB problems for any tested problem dimensionality. These results show that EAs are generally better at optimizing MO problems than repeated SO approaches.

In Tables 1 and 2 we show a statistical comparison of results shown in Figure 4. In particular, we compare the RV-GOMEA^A and RV-GOMEA^C variants of RV-GOMEA

Table 2: Statistical comparison of RV-GOMEA$^{\text{A}}$ and RV-GOMEA$^{\text{C}}$ for all multiobjective benchmark problems, for the highest dimensionality achieved by both algorithms in Figure 4. A cell marked by a letter (A or C) indicates that this variant of RV-GOMEA performed statistically significantly better (in terms of function evaluations) than the other variant, for the given linkage model and benchmark problem. The number of asterisks indicates the level of significance, with 0 to 3 asterisks indicating p-values smaller than 10^{-2}, 10^{-3}, 10^{-4}, and 10^{-5}, respectively.

	genMED	ZDT1	ZDT3	MOSoREB
Uni	A***	C***	C*	C***
BFLT	A***	C***		C***
B5				C***

given identical linkage models. We restrict the analysis to these comparisons, because comparisons to L-BFGS and LM-CMA-ES are generally directly clear from Figure 4. For each linkage model and each benchmark problem, a t-test is performed to compare the number of evaluations of the two variants of RV-GOMEA for the highest dimensionality reached by both algorithms. Though the absolute differences in the number of evaluations are generally small between the two variants, as shown in Figure 4, RV-GOMEA$^{\text{C}}$ is statistically significantly better ($p < 0.01$) than RV-GOMEA$^{\text{A}}$ in 13 of 22 cases. In 6 of 22 cases RV-GOMEA$^{\text{A}}$ is statistically significantly better ($p < 0.01$) than RV-GOMEA$^{\text{C}}$.

10 Discussion

First and foremost, we have shown that the exploitation of partial evaluations in a GBO setting can substantially improve the performance and scalability of EAs compared with a BBO setting. For this reason, exploiting what a GBO setting has to offer should be strongly considered if possible when optimizing a real-world problem, as substantial gains in performance can be achieved if the problem supports partial evaluations. Such partial evaluations are certainly not restricted to trivial problems, as they have previously been applied to real-world problems that are multiobjective, non-separable, and multimodal (Luong, Alderliesten et al., 2018; Bouter, Alderliesten, and Bosman, 2017). For such difficult problems, gradient-based optimizers are likely to converge to a local optimum. In contrast, EAs are considered state-of-the-art for such optimization problems.

In particular, we have introduced two variants of the real-valued version of GOMEA, based on previous work (Bouter, Alderliesten, Witteveen et al., 2017; Bouter, Luong et al., 2017), which is designed to exploit partial evaluations in a GBO setting. To the best of our knowledge, RV-GOMEA is the only real-valued EA capable of exploiting partial evaluations in a GBO setting. Different real-valued EAs are not directly suitable to get the most out of a GBO setting. For example, variants of CMA-ES can use a restricted covariance matrix to account for available problem information. Using the optimal restriction of the covariance matrix however still leads to worse performance than RV-GOMEA in a GBO setting (Bouter, Alderliesten, Witteveen et al., 2017).

Based on the results of comparisons between (MO-)RV-GOMEA$^{\text{A}}$ and (MO-)RV-GOMEA$^{\text{C}}$, we conclude that both algorithms are valid and attractive alternatives to employ for optimization in a GBO setting. Our results show that the core principles of GOMEA have been successfully applied to CMA-ES, as we were able to achieve results similar, and in some cases superior, to the results of (MO-)RV-GOMEA$^{\mathrm{A}}$ that were previously shown to be excellent compared to the state-of-the-art (Bouter, Alderliesten, Witteveen et al., 2017; Bouter, Luong et al., 2017).

Future work includes research into automatically determining the best linkage structure for GBO problems. Though this may be trivial for problems such as SoREB, it may be difficult for others. For instance, we found that a univariate linkage structure achieved the best performance on the non-separable Rosenbrock problem. It is further not immediately clear what the best linkage structure would be, for example, when all pairs of variables have some degree of dependence, but some relations are clearly stronger than others. Various techniques could be considered to learn problem structure, such as techniques used in cooperative coevolution (Potter and Jong, 2000). Such techniques analyze interactions between problem variables to decompose the optimization problem, for example using adaptive weighting (Yang et al., 2008), delta grouping (Omidvar et al., 2010), or differential grouping (Omidvar et al., 2014).

Further future work consists of trying to improve the performance of RV-GOMEA in a BBO setting. This could require fundamental changes to GOM, because GOM requires evaluations to be done after only a small number of variables have been modified. For this reason, GOM is very efficient in a GBO setting, but not as efficient in a BBO setting. Potentially, combinations with surrogate models that are efficient to evaluate could have an impact here.

11 Conclusions

Effectively exploiting linkage information can lead to substantial improvements in the performance of EAs, including for problems with real-valued variables. In this article, we showed that exploiting such information in a GBO setting, specifically one that allows for partial evaluations, can lead to substantial improvements in performance and scalability.

We described combinations of the state-of-the-art real-valued EAs AMaLGaM and CMA-ES (Hansen and Ostermeier, 2001), with GOMEA, a state-of-the-art EA framework designed around effectively mixing partial solutions. The new GOMEA variants, referred to as RV-GOMEA, were also extended to be suitable for multi-objective optimization. Scalability analyses were performed on a set of well-known benchmark problems.

The performance of RV-GOMEA was compared to that of LM-CMA-ES, a state-of-the-art EA for large-scale BBO, and L-BFGS, a well-known gradient-based optimization method. This showed that using RV-GOMEA in a GBO setting leads to substantially better performance than the state-of-the-art EAs in BBO and the ability to scalably solve a much richer class of optimization problems than L-BFGS, while achieving similar, or for MO, even better scalability in a GBO setting. Therefore, RV-GOMEA can be considered a valuable addition to the field of EAs, in particular for the purpose of solving complex (large-scale) optimization problems when facing a GBO scenario that permits efficient partial evaluations.

Acknowledgments

This work is part of the research program IPPSI-TA (project number 628.006.003), financed by the Netherlands Organisation for Scientific Research (NWO) and Elekta.

References

Bosman, P. A. N. (2012). On gradients and hybrid evolutionary algorithms for real-valued multi-objective optimization. *IEEE Transactions on Evolutionary Computation*, 16(1):51–69.

Bosman, P. A. N., and Alderliesten, T. (2012). Incremental Gaussian model-building in multi-objective EDAs with an application to deformable image registration. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 241–248.

Bosman, P. A. N., Grahl, J., and Thierens, D. (2013). Benchmarking parameter-free AMaLGaM on functions with and without noise. *Evolutionary Computation*, 21(3):445–469.

Bosman, P. A. N., Luong, N. H., and Thierens, D. (2016). Expanding from discrete Cartesian to permutation gene-pool optimal mixing evolutionary algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 637–644.

Bosman, P. A. N., and Thierens, D. (2003). The balance between proximity and diversity in multi-objective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 7(2):174–188.

Bosman, P. A. N., and Thierens, D. (2013). More concise and robust linkage learning by filtering and combining linkage hierarchies. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 359–366.

Bouter, A., Alderliesten, T., Bel, A., Witteveen, C., and Bosman, P. A. N. (2018). Large-scale parallelization of partial evaluations in evolutionary algorithms for real-world problems. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 1199–1206.

Bouter, A., Alderliesten, T., and Bosman, P. A. N. (2017). A novel model-based evolutionary algorithm for multi-objective deformable image registration with content mismatch and large deformations: Benchmarking efficiency and quality. In *Proceedings of SPIE*, p. 1013312, Vol. 10133.

Bouter, A., Alderliesten, T., Witteveen, C., and Bosman, P. A. N. (2017). Exploiting linkage information in real-valued optimization with the real-valued gene-pool optimal mixing evolutionary algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 705–712.

Bouter, A., Luong, N. H., Witteveen, C., Alderliesten, T., and Bosman, P. A. N. (2017). The multi-objective real-valued gene-pool optimal mixing evolutionary algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 537–544.

Chen, W., Whitley, D., Tinós, R., and Chicano, F. (2018). Tunneling between plateaus: Improving on a state-of-the-art MAXSAT solver using partition crossover. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 921–928.

De Jong, K. A. (1975). *Analysis of the behavior of a class of genetic adaptive systems*. Technical Report. University of Michigan, Ann Arbor, Michigan.

Deb, K. (1999). Multi-objective genetic algorithms: Problem difficulties and construction of test problems. *Evolutionary Computation*, 7(3):205–230.

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*. Hoboken, NJ: John Wiley & Sons, Inc.

Deb, K., and Myburgh, C. (2016). Breaking the billion-variable barrier in real-world optimization using a customized evolutionary algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 653–660.

Etxeberria, R., and Larrañaga, P. (1999). Global optimization using Bayesian networks. In *Second Symposium on Artificial Intelligence*, pp. 332–339.

Goldman, B. W., and Punch, W. F. (2014). Parameter-less population pyramid. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 785–792.

Gronau, I., and Moran, S. (2007). Optimal implementations of UPGMA and other common clustering algorithms. *Information Processing Letters*, 104(6):205–210.

Hansen, N., and Ostermeier, A. (1996). Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Proceedings of IEEE International Conference on Evolutionary Computation*, pp. 312–317.

Hansen, N., and Ostermeier, A. (2001). Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195.

Harik, G. R., Lobo, F. G., and Sastry, K. (2006). Linkage learning via probabilistic modeling in the extended compact genetic algorithm (ECGA). In M. Pelikah, K. Sastry, and E. Cantú Paz (Eds.), *Scalable optimization via probabilistic modeling*, volume 33 of *Studies in Computational Intelligence*, pp. 39–61. New York: Springer.

Holland, J. (1975). *Adaptation in natural and artificial systems*. Ann Arbor: University of Michigan Press.

Hsu, S.-H., and Yu, T.-L. (2015). Optimization by pairwise linkage detection, incremental linkage set, and restricted/back mixing: DSMGA-II. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 519–526.

Jones, E., Oliphant, T., Peterson, P., et al. (2001). SciPy: Open source scientific tools for Python. Retrieved from http://www.scipy.org/

Kraskov, A., Stögbauer, H., and Grassberger, P. (2004). Estimating mutual information. *Physical Review E*, 69(6):066138.

Li, K., Omidvar, M. N., Deb, K., and Yao, X. (2016). Variable interaction in multi-objective optimization problems. In *International Conference on Parallel Problem Solving from Nature*, pp. 399–409.

Liu, D. C., and Nocedal, J. (1989). On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1-3):503–528.

Loshchilov, I. (2014). A computationally efficient limited memory CMA-ES for large scale optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 397–404.

Lozano, J. A., Larrañaga, P., Inza, I., and Bengoetxea, E. (2006). *Towards a new evolutionary computation: Advances on estimation of distribution algorithms, volume 192 of Studies in Fuzziness and Soft Computing*. New York: Springer.

Luong, H. N., and Bosman, P. A. N. (2012). Elitist archiving for multi-objective evolutionary algorithms: To adapt or not to adapt. In *International Conference on Parallel Problem Solving from Nature*, pp. 72–81.

Luong, N. H., Alderliesten, T., Bel, A., Niatsetski, Y., and Bosman, P. A. N. (2018). Application and benchmarking of multi-objective evolutionary algorithms on high-dose-rate brachytherapy planning for prostate cancer treatment. *Swarm and Evolutionary Computation*, 40:37–52.

Luong, N. H., La Poutré, H., and Bosman, P. A. N. (2014). Multi-objective gene-pool optimal mixing evolutionary algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 357–364.

Luong, N. H., La Poutré, H., and Bosman, P. A. N. (2018). Multi-objective gene-pool optimal mixing evolutionary algorithm with the interleaved multi-start scheme. *Swarm and Evolutionary Computation*, 40:238–254.

Michalewicz, Z., and Janikow, C. Z. (1991). Genetic algorithms for numerical optimization. *Statistics and Computing*, 1(2):75–91.

Mühlenbein, H. (1997). The equation for response to selection and its use for prediction. *Evolutionary Computation*, 5(3):303–346.

Omidvar, M. N., Li, X., Mei, Y., and Yao, X. (2014). Cooperative co-evolution with differential grouping for large scale optimization. *IEEE Transactions on Evolutionary Computation*, 18(3):378–393.

Omidvar, M. N., Li, X., and Yao, X. (2010). Cooperative co-evolution with delta grouping for large scale non-separable function optimization. In *IEEE Congress on Evolutionary Computation*, pp. 1–8.

Orvosh, D., and Davis, L. (1994). Using a genetic algorithm to optimize problems with feasibility constraints. In *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence*, pp. 548–553.

Pelikan, M. (2005a). Bayesian optimization algorithm. In *Hierarchical Bayesian optimization algorithm, volume 170 of Studies in Fuzziness and Soft Computing*, pp. 31–48. New York: Springer.

Pelikan, M. (2005b). Hierarchical Bayesian optimization algorithm. In *Hierarchical Bayesian optimization algorithm, volume 170 of Studies in Fuzziness and Soft Computing*, pp. 105–129. New York: Springer.

Pelikan, M., and Mühlenbein, H. (1999). The bivariate marginal distribution algorithm. In R. Roy, T. Furuhashi, and P. K. Chawdhry (Eds.), *Advances in soft computing*, pp. 521–535. New York: Springer.

Pelikan, M., Sastry, K., and Cantú-Paz, E. (2007). *Scalable optimization via probabilistic modeling: From algorithms to applications, volume 33 of Studies in Computational Intelligence*. New York: Springer.

Pelikan, M., Sastry, K., and Goldberg, D. E. (2005). Multiobjective hBOA, clustering, and scalability. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 663–670.

Potter, M. A., and Jong, K. A. D. (2000). Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation*, 8(1):1–29.

Rastrigin, L. A. (1974). *Systems of extremal control*. Moscow: Nauka.

Ros, R., and Hansen, N. (2008). A simple modification in CMA-ES achieving linear time and space complexity. In *International Conference on Parallel Problem Solving from Nature*, pp. 296–305.

Rosenbrock, H. (1960). An automatic method for finding the greatest or least value of a function. *The Computer Journal*, 3(3):175–184.

Thierens, D. (2010). The linkage tree genetic algorithm. In *International Conference on Parallel Problem Solving from Nature*, pp. 264–273.

Thierens, D., and Bosman, P. A. N. (2011). Optimal mixing evolutionary algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 617–624.

Tinós, R., Whitley, D., and Chicano, F. (2015). Partition crossover for pseudo-Boolean optimization. In *Proceedings of the 2015 ACM Conference on Foundations of Genetic Algorithms XIII*, pp. 137–149.

Varelas, K., Auger, A., Brockhoff, D., Hansen, N., ElHara, O. A., Semet, Y., Kassab, R., and Barbaresco, F. (2018). A comparative study of large-scale variants of CMA-ES. In *International Conference on Parallel Problem Solving from Nature*, pp. 3–15.

Virgolin, M., Alderliesten, T., Witteveen, C., and Bosman, P. A. N. (2017). Scalable genetic programming by gene-pool optimal mixing and input-space entropy-based building-block learning. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 1041–1048.

Whitley, D., Hains, D., and Howe, A. (2009). Tunneling between optima: Partition crossover for the traveling salesman problem. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 915–922.

Wierstra, D., Schaul, T., Glasmachers, T., Sun, Y., Peters, J., and Schmidhuber, J. (2014). Natural evolution strategies. *Journal of Machine Learning Research*, 15(1):949–980.

Yang, Z., Tang, K., and Yao, X. (2008). Large scale evolutionary optimization using cooperative coevolution. *Information Sciences*, 178(15):2985–2999.