Exploiting Linkage Information in Real-Valued Optimization with the Real-Valued Gene-Pool Optimal Mixing Evolutionary Algorithm

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

Tanja Alderliesten
Academic Medical Center
Amsterdam, The Netherlands
T.Alderliesten@amc.uva.nl

Cees Witteveen
Delft University of Technology
Delft, The Netherlands
C.Witteveen@tudelft.nl

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

ABSTRACT

The recently introduced Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA) has been shown to be among the state-of-the-art for solving discrete optimization problems. Key to the success of GOMEA is its ability to efficiently exploit the linkage structure of a problem. Here, we introduce the Real-Valued GOMEA (RV-GOMEA), which incorporates several aspects of the real-valued EDA known as AMaLGaM into GOMEA in order to make GOMEA well-suited for real-valued optimization. The key strength of GOMEA to competently exploit linkage structure is effectively preserved in RV-GOMEA, enabling excellent performance on problems that exhibit a linkage structure that is to some degree decomposable. Moreover, the main variation operator of GOMEA enables substantial improvements in performance if the problem allows for partial evaluations, which may be very well possible in many real-world applications. Comparisons of performance with state-of-the-art algorithms such as CMA-ES and AMaLGaM on a set of well-known benchmark problems show that RV-GOMEA achieves comparable, excellent scalability in case of black-box optimization. Moreover, RV-GOMEA achieves unprecedented scalability on problems that allow for partial evaluations, reaching near-optimal solutions for problems with up to millions of real-valued variables within one hour on a normal desktop computer.

CCS CONCEPTS

•Mathematics of computing → Evolutionary algorithms;

KEYWORDS

GOMEA, linkage learning, black-box optimization, gray-box optimization, scalability, real-valued optimization

GECCO '17, Berlin, Germany

DOI: http://dx.doi.org/10.1145/3071178.3071272

ACM Reference format:
Anton Bouter, Tanja Alderliesten, Cees Witteveen, and Peter A.N. Bosman. 2017. Exploiting Linkage Information in Real-Valued Optimization with the Real-Valued Gene-Pool Optimal Mixing Evolutionary Algorithm. In *Proceedings of GECCO '17, Berlin, Germany, July 15-19, 2017,* 8 pages.
DOI: http://dx.doi.org/10.1145/3071178.3071272

1 INTRODUCTION

Key to the success of any optimization algorithm is the efficient exploitation of problem structure. Model-based Evolutionary Algorithms (EAs) do this by maintaining an explicit model that should be aligned with the problem structure to obtain the best performance. In real-valued optimization, arguably the most state-of-the-art model-based EA is CMA-ES [6], where the model is a multivariate Gaussian probability distribution that determines the part of the search space where new solutions should be sampled. This model is very competent, but has an unnecessarily high capacity for many optimization problems, because, for instance, the rotational invariance of CMA-ES is not required to solve a wide range of optimization problems. Instead, to improve performance, model complexity can be reduced by better aligning required model capacity with problem structure. In particular, the model can be aligned with the structure of the problem by exploiting the so-called linkage structure of the optimization problem, which describes dependencies between problem variables. Recently, quite some attention has been given to the exploitation of linkage structure by EAs. One recent approach is the projection-based restricted CMA-ES [1], which reduces the size of the covariance matrix by parameterizing it with a smaller number of parameters that accurately describe the most important sources of variance. Although in itself a valid and successful approach, it is not directly clear how these reduced models correspond to dependent variables. A different approach involves explicitly describing whether or not interdependencies between specific subsets of variables must be considered. Such an approach previously proved successful in the domain of discrete optimization in the Linkage Tree Genetic Algorithm (LTGA) [12] and in its generalized form, the Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA) [3, 13], which is considered to be a state-of-the-art EA for discrete optimization problems. Despite the success of GOMEA in the domain of discrete optimization, it has

never been applied to real-valued optimization. In this paper, we study the explicit exploitability of linkage structure for real-valued optimization problems by adapting GOMEA into the Real-Valued GOMEA (RV-GOMEA). RV-GOMEA combines the Gene-pool Optimal Mixing (GOM) procedure of GOMEA with aspects of the Adapted Maximum-Likelihood Gaussian Model Iterated Density-Estimation Evolutionary Algorithm (AMaLGaM-IDEA or AMaLGaM) [2] in order to exploit the linkage structure of real-valued optimization problems, while still using continuous distributions to cover the real-valued search space.

Most work in model-based EAs considers Black-Box Optimization (BBO), meaning that virtually no knowledge about the structure of the optimization problem is supplied to the algorithm. Linkage structure can still be learned through statistical analysis of the population. However, for many real-world problems, some problem structure may be inferred directly from the problem definition. We speak of a Gray-Box Optimization (GBO) setting when the structure of dependencies between variables is roughly known. In this paper, we further consider a setting to be GBO when the problem structure is known to a degree that allows for efficient partial evaluations, meaning that the objective value of partially modified solutions can be updated more efficiently than through a full re-evaluation. Studying the design of EAs for such GBO settings is less well established. Yet, for many practical situations these assumptions are easily met, allowing important performance improvements of EAs applied to real-world problems, potentially opening up new possibilities based on already existing powerful model-based EA paradigms. It is mainly this situation for which we study the design of RV-GOMEA and aim to achieve improvements. GOMEA and also its real-valued variant RV-GOMEA that we introduce here are especially well-equipped to exploit additional problem-specific knowledge. The reason for this is that the main variation operator of GOMEA applies partial modifications to existing solutions. Partial evaluations then allow efficient update calculations of objective values. Note that the possibility of performing efficient partial evaluations is certainly not restricted to trivial or even decomposable problems. The well-known discrete optimization problem Max-Cut [8] is an example of an NP-complete problem where efficient partial evaluations are possible. Efficient partial evaluations are also possible in the real-world problem known as deformable image registration [4], which deals with real-valued variables.

To observe the performance of RV-GOMEA, RV-GOMEA is tested in GBO and BBO settings on a set of well-known benchmark problems and compared to that of the state-of-the-art EAs CMA-ES [6, 11] and AMaLGaM [2]. Excessive manual tuning of the population size parameter is avoided by applying an interleaved multistart scheme to all tested algorithms.

The remainder of this paper is structured as follows. Firstly, the way in which linkage structure is modeled by (RV-)GOMEA is described in Section 2. This is followed by a detailed description of RV-GOMEA in Section 3, and in Section 4 a description of the interleaved multistart scheme is given. All experiments and results are then discussed in Section 5, finally leading us to draw conclusions in Section 6.

2 MODELING LINKAGE INFORMATION

In GOMEA, linkage information is modeled using a so-called Family Of Subsets (FOS) that describes supposed dependencies between the variables of the problem. A FOS, denoted $\mathcal{F}$, is a subset of the powerset $\mathcal{P}(\mathcal{S})$ of $\mathcal{S}$, where $\mathcal{S} = \{0, 1, \ldots, \ell - 1\}$ denotes the set of indices of problem variables, and ℓ is the total number of problem variables. Each element $\mathcal{F}_j \in \mathcal{F}$, with $\mathcal{F}_j \subseteq \mathcal{S}$, describes the indices of a set of variables that are deemed dependent by the model. Any FOS can be used to model linkage structure, but in this section we highlight the marginal product FOS and the linkage tree FOS, which will be used throughout this paper.

2.1 Marginal Product FOS

A marginal product FOS is defined as a set $\mathcal{F}$ s.t. $\mathcal{F}_i \cap \mathcal{F}_j = \emptyset$ for any two $\mathcal{F}_i, \mathcal{F}_j \in \mathcal{F}$. The simplest marginal product FOS is the univariate FOS, which models each problem variable to be independent, i.e., $\mathcal{F} = \{\{0\}, \{1\}, \ldots, \{\ell - 1\}\}$. We define the k-block FOS as a marginal product model that models dependencies between non-overlapping blocks of k consecutive variables, i.e., $\mathcal{F} = \{\{0, 1, \ldots, k-1\}, \{k, k+1, \ldots, 2k-1\}, \ldots, \{\ell - k, \ldots, \ell - 1\}\}$.

2.2 Linkage Tree FOS

A linkage tree model [12] models multiple levels of dependencies at the same time. Partly because of this hierarchy of dependencies that is modeled, the linkage tree model was found to be the most reliable and overall most efficient FOS in discrete GOMEA for BBO [13]. The main property of a linkage tree FOS $\mathcal{F}$ is that each element $\mathcal{F}_i$ of size larger than one is the union of two other sets $\mathcal{F}_j, \mathcal{F}_k \in \mathcal{F}$ with $i \neq j \neq k$. Formally stated, for any subset $\mathcal{F}_i$ having more than one element, there exist subsets $\mathcal{F}_j$ and $\mathcal{F}_k$ such that $\mathcal{F}_j \cap \mathcal{F}_k = \emptyset$, $|\mathcal{F}_j| < |\mathcal{F}_i|$, $|\mathcal{F}_k| < |\mathcal{F}_i|$, and $\mathcal{F}_j \cup \mathcal{F}_k = \mathcal{F}_i$.

Learning a linkage tree structure is done through the Unweighted Pair Grouping Method with Arithmetic-mean (UPGMA) clustering method, of which we use the implementation with time complexity $O(\ell^2)$ discussed in [5]. The tree is initialized with all univariate elements as leaves. New elements are then iteratively created by merging the two nodes of the tree that are regarded as the most dependent. Dependence can be estimated based on the population or quantified by a problem-specific distance metric. Each node of the tree can only be merged once and the merging procedure stops when the root of the tree has been reached, which is an element that naturally contains the indices of all problem variables. The linkage tree FOS then consists of all internal nodes and leaves of the linkage tree, and has $2\ell - 1$ elements. This model therefore includes multiple levels of dependencies, ranging from the univariate level up to a level with complete dependency between all variables.

3 REAL-VALUED GOMEA

RV-GOMEA maintains a population $\boldsymbol{P}$ of n promising solutions, and during each generation applies variation to improve existing solutions. In the following, we describe the key mechanisms of RV-GOMEA, including the Anticipated Mean Shift (AMS), Adaptive Variance Scaling (AVS), and Standard Deviation Ratio (SDR) mechanisms [2], which are inherited from AMaLGaM.

Function RV-GOMEA :
  $\boldsymbol{P} \leftarrow$ Initial population, evaluated
  pop-NIS $\leftarrow t \leftarrow 0$
  for $\boldsymbol{P}_i \in \boldsymbol{P}$ **do** $\mathrm{NIS}(\boldsymbol{P}_i) \leftarrow 0$
  while not terminated **do**
    $\boldsymbol{S} \leftarrow$ The best $\lfloor \tau n \rfloor$ solutions in $\boldsymbol{P}$
    $\mathcal{F} \leftarrow$ Learn FOS from $\boldsymbol{S}$ /* Unless FOS is fixed. */
    for $\mathcal{F}_j \in \mathcal{F}$ **do**
      Estimate $\mathcal{N}(\hat{\boldsymbol{\mu}}_{\mathcal{F}_j}(t), \hat{\Sigma}_{\mathcal{F}_j}(t))$ with maximum-likelihood based on $\boldsymbol{S}$
      $\hat{\boldsymbol{\mu}}^{\mathrm{Shift}}_{\mathcal{F}_j}(t) \leftarrow \hat{\boldsymbol{\mu}}_{\mathcal{F}_j}(t) - \hat{\boldsymbol{\mu}}_{\mathcal{F}_j}(t-1)$
    for $i \in \{0, \ldots, n^{\mathrm{elitist}} - 1\}$ **do** $\boldsymbol{P}_i \leftarrow i^{\mathrm{th}}$ best solution in $\boldsymbol{P}$
    generateNewSolutions
    $t \leftarrow t + 1$

Algorithm 1: The basic structure of RV-GOMEA.

3.1 Gene-pool Optimal Mixing

The key to success of the discrete GOMEA is its main variation operator, known as Gene-pool Optimal Mixing (GOM), combined with a proper FOS linkage model. The GOM procedure is applied to each solution in the population, mixing this so-called parent solution with a random donor solution. Specifically, each FOS element describes a set of indices of problem variables for which the values are copied from the donor solution and inserted into the parent solution. If the modified parent solution has a better objective value than the parent solution had in its previous state, the modification is accepted. Otherwise, the parent solution is returned to its previous state. Linkage structure is exploited by GOM, because problem variables are mixed independently if they are deemed independent by the linkage model. Moreover, selection is directly integrated into the variation operator.

To adapt GOM to the domain of real-valued variables, certain mechanisms of AMaLGaM [2] are employed. For each FOS element a Gaussian model is learned that resembles the model used by AMaLGaM, but then restricted to the variables in the FOS element.

The basic structure of RV-GOMEA is described in Algorithm 1. Each generation of RV-GOMEA starts by the selection procedure, selecting $\boldsymbol{S}$ to be the fraction $\tau = 0.35$ (following the guidelines for AMaLGaM, see [2]) best solutions in the population $\boldsymbol{P}$. If no fixed FOS is set a priori, the selection procedure is followed by the learning of the linkage tree, which is described in Section 3.5. For each FOS element $\mathcal{F}_j$ of size k, a k-variate normal distribution is estimated with maximum likelihood based on $\boldsymbol{S}$. This normal distribution is described by its covariance matrix $\hat{\Sigma}_{\mathcal{F}_j}$ and its mean vector $\hat{\boldsymbol{\mu}}_{\mathcal{F}_j}$, whose components are estimated as follows:

$$\hat{\boldsymbol{\mu}}_{\mathcal{F}_j}[u] = \frac{1}{|S|} \sum_{\boldsymbol{s} \in \boldsymbol{S}} \boldsymbol{s}[u]$$

$$\hat{\Sigma}_{\mathcal{F}_j}[u, v] = \frac{1}{|S|} \sum_{\boldsymbol{s} \in \boldsymbol{S}} \left(\boldsymbol{s}[u] - \hat{\boldsymbol{\mu}}_{\mathcal{F}_j}[u]\right) \left(\boldsymbol{s}[v] - \hat{\boldsymbol{\mu}}_{\mathcal{F}_j}[v]\right)$$

A number of n^{elitist} elitist solutions are then copied into the population, overwriting existing solutions, and the untouched solutions are subject to the GOM procedure, which is described in Algorithm 2. For each FOS element $\mathcal{F}_j$, in a random order, a new partial solution is generated for each solution in the population by sampling from the normal distribution $\mathcal{N}\left(\hat{\boldsymbol{\mu}}_{\mathcal{F}_j}, \hat{\Sigma}_{\mathcal{F}_j}\right)$. All values of a newly generated partial solution $\boldsymbol{y}_{\mathcal{F}_j}$ are inserted into a solution

Function generateNewSolutions :
  for $i \in \{n^{\mathrm{elitist}}, \ldots, n-1\}$ **do** $\mathrm{improved}(\boldsymbol{P}_i) \leftarrow$ False
  for $\mathcal{F}_j \in \mathcal{F}$ **do** /* In a random order. */
    for $i \in \{n^{\mathrm{elitist}}, \ldots, n-1\}$ **do**
      for $u \in \mathcal{F}_j$ **do** $\boldsymbol{b}[u] \leftarrow \boldsymbol{P}_i[u]$
      $f_{\boldsymbol{b}} \leftarrow f_{\boldsymbol{P}_i}$
      $\boldsymbol{y} \leftarrow \mathcal{N}\left(\hat{\boldsymbol{\mu}}_{\mathcal{F}_j}, c^{\mathrm{Multiplier}}_{\mathcal{F}_j} \hat{\Sigma}_{\mathcal{F}_j}\right)$
      if $i < n^{\mathrm{elitist}} + \lfloor \frac{1}{2} \tau n \rfloor$ **then**
        $\boldsymbol{y} \leftarrow \boldsymbol{y} + c^{\mathrm{Multiplier}}_{\mathcal{F}_j} \delta^{\mathrm{AMS}} \hat{\boldsymbol{\mu}}^{\mathrm{Shift}}_{\mathcal{F}_j}$
      for $u \in \mathcal{F}_j$ **do** $\boldsymbol{P}_i[u] \leftarrow \boldsymbol{y}[u]$
      $f_{\boldsymbol{P}_i} \leftarrow f(\boldsymbol{P}_i)$
      if $f_{\boldsymbol{P}_i} < f_{\boldsymbol{b}}$ **then** $\mathrm{improved}(\boldsymbol{P}_i) \leftarrow$ True
      else if $U(0,1) > p^{\mathrm{accept}}$ **then**
        for $u \in \mathcal{F}_j$ **do** $\boldsymbol{P}_i[u] \leftarrow \boldsymbol{b}[u]$
        $f_{\boldsymbol{P}_i} \leftarrow f_{\boldsymbol{b}}$
    AdaptMultiplier $\left(c^{\mathrm{Multiplier}}_{\mathcal{F}_j}\right)$
  for $i \in \{n^{\mathrm{elitist}}, \ldots, n^{\mathrm{elitist}} + \lfloor \frac{1}{2} \tau n \rfloor - 1\}$ **do**
    $\boldsymbol{b} \leftarrow \boldsymbol{P}_i; f_{\boldsymbol{b}} \leftarrow f_{\boldsymbol{P}_i}$
    $\boldsymbol{P}_i \leftarrow \boldsymbol{P}_i + \delta^{\mathrm{AMS}} \hat{\boldsymbol{\mu}}^{\mathrm{Shift}}$
    $f_{\boldsymbol{P}_i} \leftarrow f(\boldsymbol{P}_i)$
    if $f_{\boldsymbol{P}_i} < f_{\boldsymbol{b}}$ **then** $\mathrm{improved}(\boldsymbol{P}_i) \leftarrow$ True
    else if $U(0,1) > p^{\mathrm{accept}}$ **then**
      $\boldsymbol{P}_i \leftarrow \boldsymbol{b}$
      $f_{\boldsymbol{P}_i} \leftarrow f_{\boldsymbol{b}}$
  for $i \in \{n^{\mathrm{elitist}}, \ldots, n-1\}$ **do**
    if $\mathrm{improved}(\boldsymbol{P}_i)$ **then** $\mathrm{NIS}(\boldsymbol{P}_i) = 0$
    else $\mathrm{NIS}(\boldsymbol{P}_i) \leftarrow \mathrm{NIS}(\boldsymbol{P}_i) + 1$
    if $\mathrm{NIS}(\boldsymbol{P}_i) > \mathrm{NIS}^{\mathrm{MAX}}$ **then**
      ForcedImprovements$(\boldsymbol{P}_i)$

Algorithm 2: Generating new solutions in RV-GOMEA.

$\boldsymbol{P}_i$. If the modification of the solution leads to an improvement, this modification is maintained. Otherwise, the modification is maintained with a probability of $p^{\mathrm{accept}} = 0.05$, and reset to the pre-existing state with a probability of $1 - p^{\mathrm{accept}}$. Allowing modifications that do not lead to improvements can help steering the population out of local optima, especially when the FOS does not contain high-dimensional elements, as was observed in preliminary experiments on the Rosenbrock function. Good results were observed for $p^{\mathrm{accept}} = 0.05$.

3.2 Adaptive Variance Scaling according to the Standard Deviation Ratio

The AVS mechanism multiplies $\hat{\Sigma}_{\mathcal{F}_j}$ by the distribution multiplier $c^{\mathrm{Multiplier}}_{\mathcal{F}_j}$ to counteract the variance-diminishing effect of selection. The distribution multiplier is initialized to 1.0 and is dynamically adapted by the SDR [2] approach, which scales it based on the average improvement vector $\boldsymbol{x}^{\mathrm{Avg\text{-}imp}}_{\mathcal{F}_j}$. The SDR procedure is described in Algorithm 3. The average improvement vector describes the average parameters of all solutions in $\boldsymbol{X}^{\mathrm{Improved}}_{\mathcal{F}_j}$, which includes all solutions for which an improvement over the best solution in the selection was observed during the mixing procedure of $\mathcal{F}_j$. If no solution was improved during the mixing procedure of $\mathcal{F}_j$, $c^{\mathrm{Multiplier}}_{\mathcal{F}_j}$ is decreased by a factor $\eta^{\mathrm{DEC}} = 0.9$, according to the guideline of AMaLGaM in [2]. The value of $c^{\mathrm{Multiplier}}_{\mathcal{F}_j}$ can only drop below 1 by this operation if not a single solution in the population has improved for at least a predefined number, the so-called maximum no improvement stretch $\mathrm{NIS}^{\mathrm{MAX}}$, of generations. The

vector $\boldsymbol{z}_{\mathcal{F}_j}^{\text{Avg-imp}} = \boldsymbol{L}_{\mathcal{F}_j}^{-1}\left(\boldsymbol{x}_{\mathcal{F}_j}^{\text{Avg-imp}} - \hat{\boldsymbol{\mu}}_{\mathcal{F}_j}\right)$ describes the average improvements in terms in standard deviations along all principal axes of $\mathcal{N}\left(\hat{\boldsymbol{\mu}}_{\mathcal{F}_j}, \hat{\Sigma}_{\mathcal{F}_j}\right)$, where $\boldsymbol{L}_{\mathcal{F}_j}^{-1}$ is obtained from a Cholesky decomposition $\hat{\Sigma}_{\mathcal{F}_j} = \boldsymbol{L}_{\mathcal{F}_j}\boldsymbol{L}_{\mathcal{F}_j}^{-1}$. If the average improvement along any of the principal axes of the normal distribution $\mathcal{N}\left(\hat{\boldsymbol{\mu}}_{\mathcal{F}_j}, \hat{\Sigma}_{\mathcal{F}_j}\right)$ is larger than $\theta^{\text{SDR}} = 1$ standard deviations, the search space is most likely slope-like and $c_{\mathcal{F}_j}^{\text{Multiplier}}$ is increased by a factor $\eta^{\text{INC}} = 1/\eta^{\text{DEC}}$.

Function AdaptMultiplier$\left(c_{\mathcal{F}_j}^{\text{Multiplier}}\right)$:
  if $\boldsymbol{X}_{\mathcal{F}_j}^{\text{Improved}} \neq \emptyset$ **then**
    pop-NIS $\leftarrow 0$
    if $c_{\mathcal{F}_j}^{\text{Multiplier}} < 1$ **then** $c_{\mathcal{F}_j}^{\text{Multiplier}} \leftarrow 1$
    SDR $\leftarrow \max_{u\in\mathcal{F}_j}\left\{\left|\left(\boldsymbol{L}_{\mathcal{F}_j}^{-1}(\boldsymbol{x}_{\mathcal{F}_j}^{\text{Avg-imp}} - \hat{\boldsymbol{\mu}}_{\mathcal{F}_j})\right)[u]\right|\right\}$
    if SDR $> \theta^{\text{SDR}}$ **then** $c_{\mathcal{F}_j}^{\text{Multiplier}} \leftarrow \eta^{\text{INC}} c_{\mathcal{F}_j}^{\text{Multiplier}}$
  else
    if $c_{\mathcal{F}_j}^{\text{Multiplier}} \leq 1$ **then** pop-NIS $\leftarrow$ pop-NIS + 1
    if $\left(c_{\mathcal{F}_j}^{\text{Multiplier}} > 1\right)$ **or** $\left(\text{pop-NIS} \geq \text{NIS}^{\text{MAX}}\right)$ **then**
      $c_{\mathcal{F}_j}^{\text{Multiplier}} \leftarrow \eta^{\text{DEC}} c^{\text{Multiplier}}$
    if $\left(c_{\mathcal{F}_j}^{\text{Multiplier}} < 1\right)$ **and** $\left(\text{pop-NIS} < \text{NIS}^{\text{MAX}}\right)$ **then**
      $c_{\mathcal{F}_j}^{\text{Multiplier}} \leftarrow 1$

Algorithm 3: Adapting $c_{\mathcal{F}_j}^{\text{Multiplier}}$ of FOS element $\mathcal{F}_j$ depending on the improvements found for this FOS element.

3.3 Anticipated Mean Shift

Applying AMS shifts a fraction $\frac{1}{2}\tau$ of the generated solutions by $\delta^{\text{AMS}} = 2$ times the difference between the means of subsequent generations $\hat{\boldsymbol{\mu}}_{\mathcal{F}_j}^{\text{Shift}}$, scaled by $c_{\mathcal{F}_j}^{\text{Multiplier}}$. This procedure is aimed at accelerating the search, mostly when the population is in a slope-like region of the search space. AMS is applied directly after the sampling of a partial solution, before the evaluation of the solution. After the complete GOM procedure, an additional round of AMS is applied to the same fraction $\frac{1}{2}\tau$ of the population. With this round of AMS, all parameters of a solution $\boldsymbol{P}_i$ are shifted by $\delta^{\text{AMS}}\hat{\boldsymbol{\mu}}^{\text{Shift}}$, with $\delta^{\text{AMS}} = 2$. If this shift leads to an improvement of the objective value of $\boldsymbol{P}_i$, the modification of the variables is accepted. Otherwise the modification is accepted with a probability of $p^{\text{accept}} = 0.05$, as in Section 3.1, and returned to the previous state with a probability of $1 - p^{\text{accept}}$. The second round of AMS allows for the shift of a fraction of the population in all dimensions at once, which may be prohibited by the mixing procedure if the linkage model does not include an element that includes the full set of problem variables, e.g., in case of the univariate model.

3.4 Forced Improvements

To move solutions out of local minima and ensure efficient convergence of the population to a single point in the search space, we adapt the Forced Improvements (FI) procedure that was introduced in [3] to the domain of real-valued variables. Pseudocode for this is shown in Algorithm 4. At the end of each generation, FI is applied

Function ForcedImprovements($\boldsymbol{P}_i$):
  $\boldsymbol{b} \leftarrow \boldsymbol{P}_i; f_{\boldsymbol{b}} \leftarrow f_{\boldsymbol{P}_i}; \alpha \leftarrow 0.5$
  while $\alpha \geq 0.01$ **do**
    for $\mathcal{F}_j \in \mathcal{F}$ **do**
      for $u \in \mathcal{F}_j$ **do** $\boldsymbol{P}_i[u] \leftarrow \left(\alpha\boldsymbol{P}_i[u] + (1-\alpha)\boldsymbol{x}^{\text{elitist}}[u]\right)$
      $f_{\boldsymbol{P}_i} \leftarrow f(\boldsymbol{P}_i)$
      if $f_{\boldsymbol{P}_i} < f_{\boldsymbol{b}}$ **then return**
      else
        for $u \in \mathcal{F}_j$ **do** $\boldsymbol{P}_i[u] \leftarrow \boldsymbol{b}[u]$
        $f_{\boldsymbol{P}_i} \leftarrow f_{\boldsymbol{b}}$
    $\alpha \leftarrow 0.5\alpha$
  $\boldsymbol{P}_i \leftarrow \boldsymbol{x}^{\text{elitist}}$

Algorithm 4: The FI procedure.

to any solution $\boldsymbol{P}_i$ that has not been improved for the last NIS^{MAX} generations. For each FOS element $\mathcal{F}_j$, this procedure then changes the relevant variables of $\boldsymbol{P}_i$ into a linear combination of $\boldsymbol{P}_i$ with weight α and the elitist solution $\boldsymbol{x}^{\text{elitist}}$ with weight $1-\alpha$.

The initial value of α is set to 0.5. If the adoption of a linear combination of variables leads to an improvement, this modification is accepted and the FI procedure is terminated. Otherwise, α is multiplied by a factor of 0.5. This process is repeated until the objective value of $\boldsymbol{P}_i$ is improved or α has reached a value below 0.01, at which point all variables of $\boldsymbol{P}_i$ are overwritten by all the values of the variables of $\boldsymbol{x}^{\text{elitist}}$.

3.5 Linkage Tree

A linkage tree FOS, discussed in Section 2.2, can be used by RV-GOMEA in order to model linkage. This FOS model was previously found to be the most reliable and overall efficient FOS model in discrete GOMEA for BBO [13].

3.5.1 Dynamic Linkage Tree. A new linkage tree can be learned at the start of each generation based on the population. Specifically, RV-GOMEA uses the mutual information (MI) [9] metric to estimate distances between problem variables to use when building a linkage tree. Considering the use of Gaussian distributions, the joint MI between two real-valued variables with indices i and j is defined as

$$\text{MI}_{ij} = \log\left(\sqrt{\frac{1}{1-\left(\frac{\hat{\Sigma}_{ij}}{\hat{\sigma}_i\hat{\sigma}_j}\right)^2}}\right),$$

where $\hat{\sigma}_i$ is the estimated standard deviation of i and $\hat{\Sigma}_{ij}$ is the estimated covariance of the variables with indices i and j. The factor $r = \hat{\Sigma}_{ij}/(\hat{\sigma}_i\hat{\sigma}_j) \in [-1,1]$ is known as the Pearson product-moment correlation coefficient, of which the absolute value here describes the estimated degree of linear correlation between problem variables i and j. The MI between a pair of problem variables will be high when a high absolute correlation coefficient is estimated. For this reason, the distance metric used to build the tree is actually the negative MI so that high correlation corresponds to being more likely to be merged first when building the linkage tree. For any merged set of variables $\boldsymbol{X} = \boldsymbol{A}\cup\boldsymbol{B}$ the MI between $\boldsymbol{X}$ and any other set of variables $\boldsymbol{Y}$ is calculated as done in [5], i.e.,

$$\text{MI}_{\boldsymbol{X}\boldsymbol{Y}} = \frac{|\boldsymbol{A}|}{|\boldsymbol{X}|}\text{MI}_{\boldsymbol{A}\boldsymbol{Y}} + \frac{|\boldsymbol{B}|}{|\boldsymbol{X}|}\text{MI}_{\boldsymbol{B}\boldsymbol{Y}}.$$

Learning a linkage tree from the population causes it to potentially be different at the start of each generation. The distribution multiplier $c_{\mathcal{F}_j}^{\text{Multiplier}}$ of some FOS element $\mathcal{F}_j$ can therefore not simply be copied from one generation to the next. The distribution multipliers can substantially accelerate convergence, so passing an existing distribution multiplier to a very similar FOS element in the following generation is likely beneficial. We pass the distribution multiplier of a FOS element to the most similar FOS element of the following generation by performing a matching algorithm. The similarity between two FOS elements is defined as the number of elements they have in common, divided by the average size of these two FOS elements. With this similarity metric the well-known Hungarian algorithm is performed to find a one-to-one matching of each element of the FOS of generation t to one element of the FOS in generation $t-1$. Each element of the newly learned FOS then directly inherits the distribution multiplier of the previous generation's FOS element it was matched with. The Hungarian algorithm has a time complexity of $O(|\mathcal{F}|^3) = O(\ell^3)$, which does not increase the overall complexity of RV-GOMEA, because the complexity of the Hungarian algorithm equals that of the Cholesky decomposition of a FOS element of size ℓ. Note that FOS elements of size 1 or size ℓ are present in all dynamic linkage trees, and should not be considered by the Hungarian algorithm, leading to a decrease in time complexity by a constant but substantial factor.

3.5.2 Fixed Linkage Tree. Alternative to learning a linkage tree online, a fixed linkage tree can be learned offline based on a predefined distance matrix. This matrix should describe a distance between problem variables that is meant to correspond to an inverted or negated definition of linkage strength, i.e., the stronger the suspected linkage, the smaller the distance, for which an estimate can likely be inferred from the problem definition if a GBO setting is assumed.

3.5.3 Bounded Linkage Tree. In order to prevent some of the largest FOS elements from dominating the required computational effort in high-dimensional problems, the maximum size of each linkage tree element can be bounded by a non-trivially sized constant assuming that the maximum order or (strong) interactions between variables is bounded by some constant. Such a so-called bounded linkage tree can be constructed by learning a linkage tree based on a distance matrix, while preventing merging operations that would create FOS elements larger than the upper bound. The upper bound on the FOS element size can be set to a small number to make the linkage tree more similar to the univariate model, or to a large number to make it more similar to the non-bounded linkage tree. The need for either a small or a large upper bound is problem-dependent, and could be tuned accordingly.

4 INTERLEAVED MULTISTART SCHEME

Correctly setting the population size parameter of an EA is often crucial to obtain good performance. The optimal population size is however problem-specific, so guidelines for the population size may be relatively large in order to be useful on a wide range of optimization problems. To avoid the need for excessive manual tuning of the population size parameter, we use an interleaved multistart scheme, to which we shall refer as IMS, inspired by one introduced in [7, 10] and apply it to all tested algorithms. In the IMS, generations of multiple independent instances of an EA with varying population sizes are interleaved, such that after c^{IMS} generations of the instance with population size n, one generation of the instance with population size $2n$ is performed. An instance can be terminated if a different instance with a larger population size is deemed to be better. In our implementations, if some instance with a population size n has a better average objective value than the instance with population size $n/2$, each instance with a population size smaller than n is terminated. The first instance of a run with IMS is initialized with a population size of n^{base}, which is typically chosen to be small.

5 EXPERIMENTS

5.1 Optimization Problems

We consider a set of five optimization functions that are decomposable to some degree, all of which need to be minimized. This set includes the sphere, Rosenbrock, Rastrigin, Michalewicz, and Sum of Rotated Ellipsoid Blocks (SoREB) functions.

Firstly, we consider the well-known sphere function, which is a trivial, completely decomposable problem, defined as:

$$f_{\text{sphere}}(\boldsymbol{x}) = \sum_{i=0}^{\ell-1} \boldsymbol{x}_i^2$$

Secondly, the Rosenbrock function is considered, which has overlapping dependencies because each pair of consecutive problem variables is dependent on each other. This function is known to be relatively difficult, mostly when a univariate variation operator is used. The Rosenbrock function is defined as follows:

$$f_{\text{Rosenbrock}}(\boldsymbol{x}) = \sum_{i=0}^{\ell-2} \left[100 \left(\boldsymbol{x}_{i+1} - \boldsymbol{x}_i^2 \right)^2 + (1 - \boldsymbol{x}_i)^2 \right]$$

The Rastrigin function, like the sphere function, is completely decomposable in each of its problem dimensions. It is however more difficult due to regular "noise" that is added to the function, causing the landscape to have many local minima. The definition of the Rastrigin function is as follows:

$$f_{\text{Rastrigin}}(\boldsymbol{x}) = 10\ell + \sum_{i=0}^{\ell-1} \left[\boldsymbol{x}_i^2 - 10 \cos\left(2\pi \boldsymbol{x}_i\right) \right]$$

The Michalewicz function is also completely decomposable in each of its problem dimensions, but the number of local optima in the i^{th} dimension is larger than that in the $i\text{-}1^{\text{th}}$ dimension. The definition of the Michalewicz function is as follows:

$$f_{\text{Michalewicz}}(\boldsymbol{x}) = \sum_{i=0}^{\ell-1} \left[-\sin\left(\boldsymbol{x}_i\right) \cdot \sin\left(\left(i+1\right) \boldsymbol{x}_i^2 / \pi \right)^{20} \right]$$

Finally, we use the SoREB function, which is defined in terms of the ellipsoid function and a rotation function R_θ that defines the counterclockwise rotation of a vector around the origin by an angle of θ. For a sufficiently large rotation angle, this problem has very tight interdependencies between blocks of k consecutive variables, but no dependencies between variables in different blocks. In particular, we use the SoREB function with block size $k = 5$ and

$\theta = 45°$. The ellipsoid function and the SoREB function are defined as follows:

$$f_{\text{ellipsoid}}(\boldsymbol{x}) = \sum_{i=0}^{\ell-1} \left[10^{\frac{6i}{\ell-1}} \boldsymbol{x}_i^2 \right]$$

$$f_{\text{SoREB}}(\boldsymbol{x}, k) = \sum_{i=0}^{\ell/k-1} \left[f_{\text{ellipsoid}} \left(R_\theta \left(\left[\boldsymbol{x}_{ki}, \ldots, \boldsymbol{x}_{k(i+1)-1} \right] \right) \right) \right]$$

5.2 Setup

The main goal of our experiments is to study scalability, because this gives a very broad, informative, and general impression of an algorithm's performance, including prediction of performance for higher-dimensional problems. We compare the scalability of RV-GOMEA with different linkage structures to that of CMA-ES, sep-CMA-ES, and the univariate AMaLGaM. We use the CMA-ES with the parameters described in [6]. The sep-CMA-ES differs from CMA-ES in that the covariance matrix is constrained to the diagonal and the learning rate is scaled accordingly [11]. RV-GOMEA-Uni uses a univariate FOS, RV-GOMEA-5B uses a 5-block FOS dependency structure, introduced in Section 2, and RV-GOMEA-LT uses a dynamic linkage tree. A random bounded fixed linkage tree used by RV-GOMEA-BFLT is learned for the sphere, Rosenbrock, Michalewicz, and Rastrigin functions, based on a random distance between each pair of variables. In order to avoid excessive memory requirements, no explicit distance matrix is used for these problems. For the SoREB function, a bounded fixed linkage tree is learned based on a distance matrix for which a very small random distance is assigned between variables in the same block, and a very large random distance is assigned to variables in different blocks. Learning a linkage tree from this distance matrix results in a linkage tree where each element is either a subset of all variables of a single block, or the union of a number of complete blocks of variables. All bounded linkage trees use an upper bound of 100 for the size of any FOS element. For the SoREB function, a version of AMaLGaM that factorizes the covariance matrix into blocks of 5 consecutive variables, called AMaLGaM-5B, is also considered, because this factorization substantially reduces the time required to compute the Cholesky decomposition of the covariance matrix. All parameters for AMaLGaM are set identical to RV-GOMEA. All algorithms use the IMS described in Section 4 with $c^{\text{IMS}} = 8$ and $n^{\text{base}} = 10$.

Although the key motivation for this paper is the GBO setting, all optimization problems are considered in a GBO and in a BBO setting. In the GBO setting RV-GOMEA is able to use efficient partial evaluations. For the sphere, Rosenbrock, Rastrigin, and Michalewicz functions, a partial evaluation after a modification of q variables is counted as a fraction q/ℓ of an evaluation, because the computational effort to perform a partial evaluation for any considered benchmark problem scales with $O(q)$. For the SoREB problem, the modification of any variable requires the re-evaluation of the entire block that contains this variable. A partial evaluation of q blocks of the SoREB function is therefore counted as a fraction qk/ℓ of an evaluation, where k is the block size of the SoREB function.

For each problem dimensionality and each algorithm, 30 independent runs are performed. Each run has a time limit of one hour. For the Michalewicz function a value-to-reach (VTR) of 95% of the optimum, and an initialization range of $[0, \pi]$ are used. All other problems are initialized within the range $[-115, -100]$ to prevent biased initialization around the optimum, and had a VTR of 10^{-10}.

All experiments are performed on desktop computers, either using the Intel Core i7-2600 CPU @ 3.40GHz or the Intel Core i7-3770S CPU @ 3.10GHz, and sufficient memory. Computation times of the fastest CPU, the former, are scaled up by 4.4% to match that of the slower CPU. This factor is determined by using RV-GOMEA to perform 10^4 runs on the 1000-dimensional sphere problem in a GBO setting. All algorithms are implemented in C. Source code of RV-GOMEA is available on the homepage of the last author.

5.3 Results

Scalability graphs showing the medians and interdecile ranges are displayed in Figure 1. Only results for which all 30 runs were successful are reported. A run is considered successful when VTR is reached within the time limit.

5.3.1 GBO. We observe that, especially in a GBO setting, RV-GOMEA has the capacity to exploit linkage structure, achieving superior scalability compared to all other tested algorithms when a suitable dependency structure is used, i.e., a univariate structure on the sphere, Rosenbrock, Rastrigin, and Michalewicz functions, and a 5-block structure on the SoREB function. Unprecedented scalability for an EA is achieved on the sphere problem, where the $5 \cdot 2^{20} = 5{,}242{,}880$-dimensional problem was solved to near-optimality within one hour. With the bounded linkage tree, RV-GOMEA is a constant factor slower than with the univariate model, but the performance gap between the two can be decreased by tightening the upper bound of the FOS element size.

The SoREB problem is the only problem where a non-univariate structure is necessary to solve the problem efficiently, and here we see that the capability of using a predefined dependency structure leads to a clear advantage over the restriction of having to use either a univariate factorization or no factorization at all. Moreover, RV-GOMEA with a bounded linkage tree achieves better scalability than CMA-ES, although it still performs worse for most tested problem sizes. This can be attributed to small FOS elements barely contributing to the optimization, and to relatively expensive partial evaluations due to the rotated block structure of the problem. However, the biggest contribution to the observed performance decrease is the fact that, before FOS elements of a certain size can be effective, a minimal population size is required that is governed by the internal mechanisms of AMaLGaM. The standard guideline for this is $17 + 3\lambda\sqrt{\lambda}$, where λ is the size of the largest FOS element. Consequently, this is quite a bit bigger than the $n^{\text{base}} = 10$ that is started from, resulting in substantial overhead of the IMS in these cases. Indeed, on problems that admit a univariate linkage structure, the performance of the linkage tree in both BBO and GBO and the BFLT in GBO is much better, scaling often better than CMA-ES both in time and evaluations, because the smallest FOS elements in the linkage tree (i.e., the univariate ones) can already contribute much even for small population sizes. Additional experiments indeed show substantial improvements if n^{base} is increased to 50, according to the guideline for AMaLGaM with $k = 5$.

A remarkable pattern can be seen in the scalability graphs of RV-GOMEA for the Michalewicz function, where the time and number of evaluations required to solve the problem decrease substantially

for problems of higher dimensionality. Considering the fact that the number of local optima per dimension of the Michalewicz problem is correlated with the index of this dimension, approaching the global optimum up to a constant ε becomes very difficult for high-dimensional problems. Using a VTR that differs from the global optimum by ε would therefore, depending on the size of ε, make the problem trivial in small dimensions and virtually unsolvable in high dimensions, leading to a very small number of dimensions for which reliable results could be achieved. In contrast, using a VTR that is a fraction of the global optimum, as we did here, will result in very high-dimensional problems becoming trivial, because the modularity of these dimensions will cause even initial samples to be relatively close to the global optimum of this dimension. The apparent pattern in the scalability of RV-GOMEA is on one hand caused by the naturally increasing difficulty of the problem, and on the other hand by the decreasing difficulty of reaching a fixed fraction of the global optimum, as problem dimensionality increases. We expect the same pattern to appear for the other algorithms given enough time to solve the very high-dimensional problems.

5.3.2 BBO. For the sphere function, RV-GOMEA using the univariate model is a constant factor worse than CMA-ES and sep-CMA-ES in terms of the required number of evaluations. However, RV-GOMEA is still a constant factor better than sep-CMA-ES in terms of required computation time. This also applies to the Rosenbrock, Rastrigin, Michalewicz, and SoREB functions, where RV-GOMEA not only performs better in terms of computation time, but also in terms of the number of evaluations. The difference in performance is caused by the fact that RV-GOMEA using the univariate model performs $O(\ell n)$ evaluations per generation, whereas sep-CMA-ES performs only $O(n)$ evaluations. Therefore, a relatively small amount of time is spent on computational effort other than function evaluations compared to sep-CMA-ES. Note that sep-CMA-ES can still perform better in terms of computation time if the evaluation function requires (much) more computation time.

For BBO problems with strong dependencies, we compare the performance of RV-GOMEA-LT with that of CMA-ES, because these are the only models that use a non-factorized Gaussian distribution. The comparative performance of the two algorithms is very problem dependent, seeing as RV-GOMEA performs better than CMA-ES on the Rastrigin and Michalewicz functions, but it performs worse on the sphere and SoREB functions. This is partly due to the larger required population size by the AMaLGaM mechanisms as pointed out in Section 5.3.1. Moreover, in terms of actual computation time however, apart from SoREB, RV-GOMEA has the upper hand.

6 CONCLUSIONS

We have introduced RV-GOMEA by combining strengths of the discrete GOMEA and the real-valued EDA AMaLGaM, aiming to exploit linkage structure in real-valued optimization problems, especially in a gray-box setting where efficient partial problem evaluations may be possible. The performance of RV-GOMEA was tested on a set of well-known benchmark problems and compared to the state-of-the-art algorithms CMA-ES and AMaLGaM. Our experiments show that RV-GOMEA achieves excellent scalability, sometimes performing better and sometimes performing worse than CMA-ES in case of black-box optimization in number of evaluations. In case of problems that are computationally fast to evaluate, RV-GOMEA does perform better than CMA-ES in black-box optimization when considering the required computation time, because GOMEA spends less time on computational effort other than function evaluations. Moreover, when efficient partial evaluations are possible, RV-GOMEA achieves unprecedented scalability, achieving near-optimal solutions for the sphere problem with over 5 million variables in less than one hour on a normal desktop computer. It also achieves superior scalability compared to any other tested algorithm on any of the tested benchmark problems when an appropriate linkage model is used. This shows that RV-GOMEA is very much capable of exploiting the linkage structure of problems with real-valued variables, preserving a key strength of the discrete GOMEA. This is especially interesting when considering the use of RV-GOMEA for solving real-world optimization problems where oftentimes it may be possible to perform partial evaluations, even when the problem is not fully understood, nor decomposable, nor easy from a computational complexity perspective.

Moreover, the combination of GOMEA with AMaLGaM is not a mandatory choice. Rather, the linkage learning capabilities offered by GOMEA may be seen as a general idea or framework that may also be combined with other real-valued EAs or EDAs, including CMA-ES, which may potentially lead to further improvements and interesting results. Generally speaking, given the observations of how RV-GOMEA performs, we conclude that regardless of the choice of real-valued EA to combine GOMEA with, in contrast to CMA-ES, which is considered to be state-of-the-art for real-valued BBO, RV-GOMEA excels in the setting of GBO.

REFERENCES

[1] Y. Akimoto and N. Hansen. 2016. Projection-Based Restricted Covariance Matrix Adaptation for High Dimension. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2016)*. ACM, 197–204.

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

[3] P.A.N. Bosman and D. Thierens. 2012. Linkage neighbors, optimal mixing and forced improvements in genetic algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2012)*. ACM, 585–592.

[4] A. Bouter, T. Alderliesten, and P.A.N. Bosman. 2017. A novel model-based evolutionary algorithm for multi-objective deformable image registration with content mismatch and large deformations: benchmarking efficiency and quality. In *Proc. SPIE*, Vol. 10133. 1013312. DOI: http://dx.doi.org/10.1117/12.2254144

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

[6] N. Hansen, S.D. Müller, and P. Koumoutsakos. 2003. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evolutionary Computation* 11, 1 (2003), 1–18.

[7] G.R. Harik and F.G. Lobo. 1999. A parameter-less genetic algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 1999)*. Morgan Kaufmann Publishers Inc., 258–265.

[8] R.M. Karp. 1972. Reducibility among combinatorial problems. Complexity of Computer Computations, (R.E. Miller and J.M. Thatcher, eds.), 85–103. (1972).

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

[10] J.C. Pereira and F.G. Lobo. 2015. A Java Implementation of Parameter-less Evolutionary Algorithms. *arXiv preprint arXiv:1506.08694* (2015).

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

[12] D. Thierens. 2010. The linkage tree genetic algorithm. In *International Conference on Parallel Problem Solving from Nature*. Springer, 264–273.

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

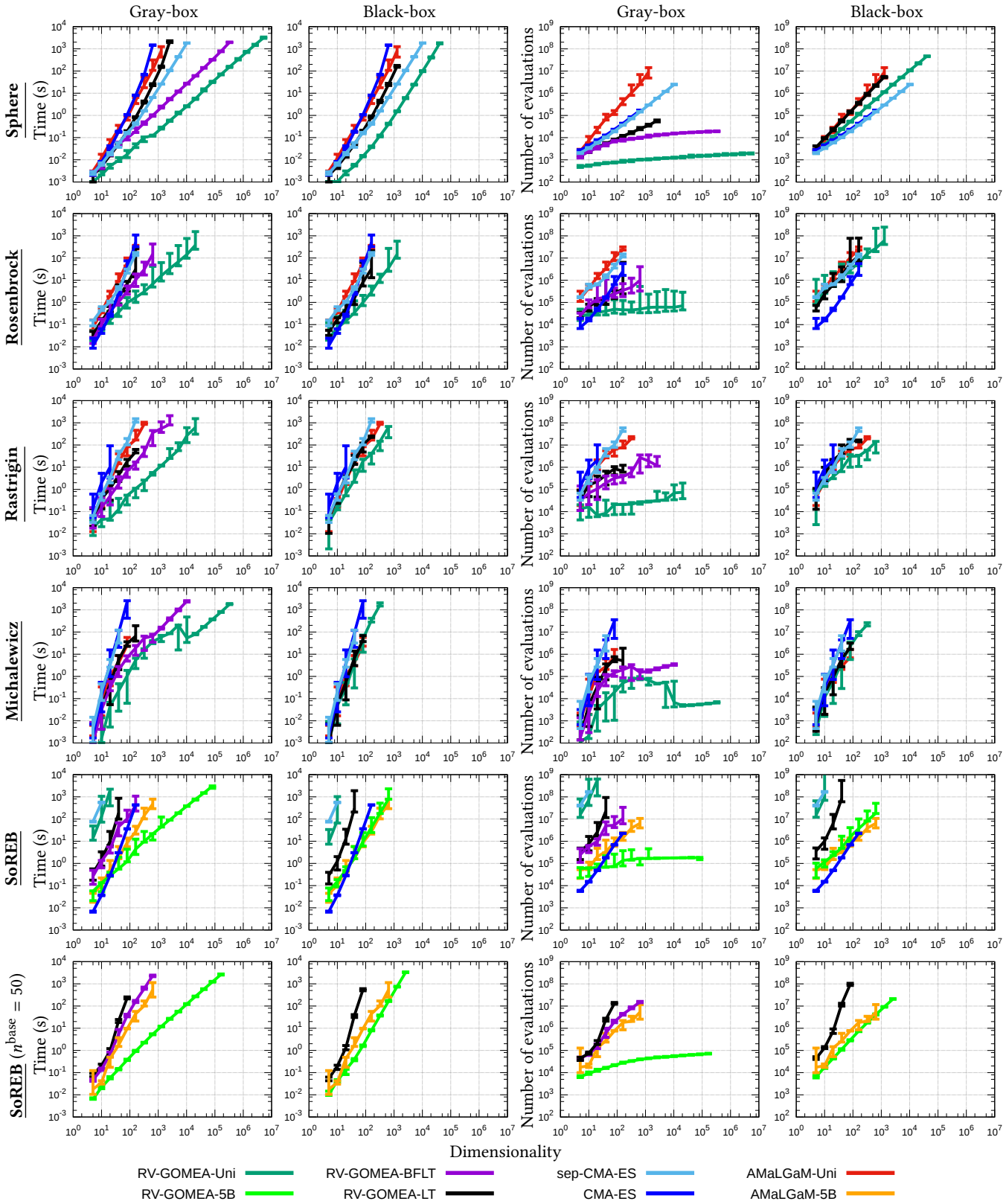

Figure 1: Medians and interdecile ranges of all experiments with each data point being the median of 30 successful runs.