# Fitness-based Linkage Learning in the Real-Valued Gene-pool Optimal Mixing Evolutionary Algorithm

Chantal Olieman

# Fitness-based Linkage Learning in the Real-Valued Gene-pool Optimal Mixing Evolutionary Algorithm

by

Chantal Olieman

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Friday the 28th of June, 2019 at 10:00 AM.

| | | |
|---|---|---|
| Student number: | 4284712 | |
| Faculty: | EEMCS | |
| Master Program: | Computer Science, Data Science and Technology Track | |
| Thesis committee: | Prof. Dr. P. A. N. Bosman, | TU Delft, CWI |
| | Dr. N. Yorke-Smith, | TU Delft |
| | Dr. C. C. S. Liem, | TU Delft |

# Abstract

The recently introduced Real-Valued Gene-pool Optimal Mixing Evolutionary Algorithm (RV-GOMEA) has been shown to be among the state-of-the-art for solving grey-box optimization problems where partial evaluations can be leveraged. A core strength is its ability to effectively exploit the linkage structure of a problem. For many real-world optimization problems, the linkage structure is unknown a priori and has to be learned online. Previously published work on RV-GOMEA however demonstrated excellent scalability only when the linkage structure is pre-specified appropriately. The commonly used mutual-information-based metric that is used to a learn linkage structure online in the discrete version of GOMEA did not show as effective in the real-valued domain and did not result in similarly excellent results, especially in a black-box setting. In this thesis, the strengths of RV-GOMEA are combined with a new fitness-based linkage learning approach that is inspired by differential grouping but reduces its computational overhead by an order of magnitude for problems with fewer interactions. The resulting new version of RV-GOMEA achieves scalability similar to when a predefined linkage model is used. Additionally, for the first time, the EDA AMaLGaM, that served as a foundation for RV-GOMEA is outperformed in a black-box setting, where partial evaluations cannot be leveraged.

# Preface

Before you lies the thesis that concludes my master in Computer Science and simultaneously my time as a student at Delft University of Technology. I am looking back with gratefulness and fulfillment whilst looking ahead with eagerness and excitement, ready to face the new challenges that life might bring.

From the day I started my bachelor in Computer Science almost 6 years ago I have been fascinated by the abstract reasoning and logic that lies in the core of all theoretical Computer Science. Finding proofs for complex algorithmic problems felt like solving a puzzle, something that I did with great pleasure. After many enthusiastic attempts to explain some of my favorite concepts of theoretical computer science to friends and family, it should have come as no surprise to them that I decided to conduct my thesis on the same topics that I have been so captivated by for the past years. It is for this reason that whilst the process of writing this thesis has been extremely demanding, it has most of all been an incredibly exciting journey and I am both proud and relieved that it has finally been concluded.

*Chantal Olieman*
*Delft, June 2019*

# Acknowledgments

First and foremost I would like to thank Peter Bosman for his supervision during the course of this project. Most of all I want to point out his excellent ability to balance critical feedback with a tremendous amount of support and passion for his subject. He has been one of the core sources of my motivation over the past months and there has never been a doubt in my mind over his dedication and commitment towards this project. His spot-on comments, constructive feedback and red remarks all over our article have helped me make this thesis into something that has exceeded my own expectations on all levels.

For answering all my questions and guiding me through the sometime confusing process of implementing my algorithms I would like to thank Anton Bouter. His critical questions and quick responses, for example when implementing AMaLGaM, have been of great help to me through the course of this project and I am grateful to have him as a co-author on the article that forms the core of this thesis.

I would like to thank the other members of my thesis committee, Neil Yorke-Smith and Cynthia Liem, for their feedback and guidance throughout the final stages of this project.

Last, but certainly not least I would like to thank my friends and family for their tremendous support and ever lasting tenderness during the course of this thesis. I would like to thank my parents specifically for their unconditional faith in me during the past years, letting me choose my own path, even at times that I, myself, did not know where it would lead. To my friends I want to say: thank you for being there every step along the way, every moment that I needed distraction, motivation or just a hug! Thank you for listening, even when I had nothing to say and most of all, thank you for always creating an environment where I could be myself and feel loved.

# Contents

# 1

# Introduction

What do creating a radiation therapy treatment plan for treating cancer and finding the best placement of windmills in a sea area have in common? At first sight, these problems seem totally unrelated, but when approaching them in a more mathematical way without making all too many assumptions or simplifications, they can both be modeled as complex (multi-objective) optimization problems with real-valued variables.

A commonly used approach to tackle these types of optimization problems is to use Evolutionary Algorithms (EAs). An EA is a generic population-based meta-heuristic optimization algorithm that uses methods inspired by biological evolution. Population-based search algorithms have been shown to perform particularity well on un-smooth problem landscapes where no gradient can be calculated and are more robust against noisy problem landscapes than point-based search methods [20]. Because population-based search algorithms maintain a population of solutions that can be extended into multiple directions simultaneously, EAs are less prone to get stuck in local optima and are more likely to search past plateau-like regions of the problem landscape. In practice, many real-world problems have un-smooth problem landscapes which contain local optima, therefore EAs have been a widely used approach to solve many real-world problems

Generally speaking, an EA maintains a population $\boldsymbol{P}$, that consists of $n$ solutions, also called an individual, to the optimization problem. At the start of the algorithm, the solutions are (often) randomly sampled and the value of the objective function is calculated for every solution, this so-called fitness value is used to compare the individuals of a population against one another. Every iteration (also called generation) the best solutions are selected and combined (called variation) to create a new generation of $n$ solutions. This process is repeated until the computational budget is exhausted or an optimal solution is found.

## 1.1. Genetic Algorithms

### 1.1.1. Discrete optimizations

One of the most widely used EAs for discrete variables is the simple Genetic Algorithm (GA) [12]. Typically, every generation of the simple GA consists of two steps: selection and combination. In every generation, the best $\tau \cdot n$ solutions are selected based on certain selection criteria that use the objective value of a solution to rank solutions against each-other. Here $\tau$ is the selection size, specifying which proportion of the population will be selected and used for recombination. After selection, $n$ new solutions are created by taking $\frac{n}{2}$ pairs of solutions from the set of selected solutions and combining them to create two new solutions, this is the combination step. There are various ways to recombine two parent solutions into new solutions. A classic example is given by uniform crossover: for every variable in the solution, with an equal probability the variables is taken from either of the parent solutions. A second solution is created by taking every variable from the opposite parents' solution. An example of uniform crossover can be seen below.

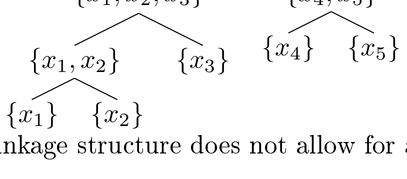
$$\text{parents}\quad \begin{array}{|c|c|c|c|c|c|c|c|c|c|}\hline 1&1&1&1&1&1&1&1&1&1\\\hline \multicolumn{10}{c}{}\\\hline 0&0&0&0&0&0&0&0&0&0\\\hline\end{array} \quad \xRightarrow[crossover]{uniform} \quad \text{offspring}\quad \begin{array}{|c|c|c|c|c|c|c|c|c|c|}\hline 1&1&0&1&0&1&1&1&0&0\\\hline \multicolumn{10}{c}{}\\\hline 0&0&1&0&1&0&0&0&1&1\\\hline\end{array}$$

A key issue with this simple genetic algorithm, that uses crossover as its variation operator, is that when a problem has strong interactions between variables, the fitness of a solution is no longer dependent only on changes made to individual variables that can be exchanged independently to combine into better solutions. In this case, variables that are strongly dependent will contribute to fitness in non-decomposable combination with each other. These variables should not be exclusively crossed over independently. This is because when the probability of picking a value from either parent is equal, the probability that these variables will be crossed over together becomes $\frac{1}{2^b}$ where $b$ is the number of dependent variables. This causes the simple GA to only scale well when such prior knowledge about the problem structure is known and can be acted upon i.e., different types of crossover operators can be designed that respect the dependencies. When no prior knowledge about the optimization problem is known, which is called black-box optimization, simple GAs, have a high risk of not scaling well, as can be seen in [27], showing that the required population size scales exponentially with the problem size for problems with high interdependence when using uniform crossover. The process of identifying these dependent problem variables (or building blocks), that should be conserved during crossed-over is called linkage learning. Due to the scalability problems of the simple GA, theoretical studies have shown that if an effective linkage learning GA could be designed, this GA would have significant advantages over the simple GA. [30]

To deal with this, new techniques have been introduced to allow for detecting dependence between problem variables and exploit them automatically. The first truly successful approach was to build probabilistic models on the whole group of selected parent solutions and sample the new population from the distribution of these parents. This group of algorithms is called Estimation of Distribution Algorithms [19, 23] and the most simple version of an EDA is known as the Univariate Marginal Distribution Algorithm [23]. For every variable, the probability distribution is based on the frequency of that variable in the selection parent set. For an $\ell-$dimensional problem, there will be $\ell$ independent probability distributions i.e., univariate distributions, that are used to sample the variables of new solutions. Whilst this algorithm was shown to work significantly better on linear problems, it still uses a univariate distribution that samples every variable independently and thus suffers from the same difficulties as the simple GA with uniform crossover when it comes to linkage.

To better employ the underlying structure of a problem, EDAs were introduced that model probabilistic dependencies. Among other EAs, a notable algorithm is the Extended compact GA (ECGA) [11] that aligns the problem of linkage learning with learning the probability distributions of the selected individuals of a population in multi-variate spaces. Following a greedy algorithm, a marginal product model (MPM) is learned. A MPM is a model that contains sets of variables such that every problem variable $x_i$ is present in exactly one of the subsets of a MPM. An example of an MPM for a problem with $l = 4$ is $\{\{x_0, x_2\}, \{x_1\}, \{x_3\}\}$. Using this MPM, a univariate distribution will be learned for $x_1$ and $x_3$, while for $x_0$ and $x_2$ a joint distribution will be created from the selected individuals. The ECGA was one of the first EA's to automatically model higher-order dependencies between decision variables. Whilst this approach overcame some of the problems encountered with the simple GA regarding linkage learning, estimating the marginal product model is computationally expensive. The time complexity of $O(n\ell^3)$ results in a substantial overhead for higher dimensional problems, which leads to the poor scalability of the ECGA.

Next to its computational complexity, another drawback of using a MPM is that every variable is represented exactly once in the linkage structure, which requires a non-overlapping partitioning to be learned. The minimal population size required to accurately learn this partitioning increases as the size of dependent components increases [32]. To better model complex problem structures and represent different orders of dependencies into one model, a hierarchical linkage model is proposed that models dependencies in a tree-shaped model called a linkage tree (LT) [28]. An example of a linkage tree for a 5-dimensional problem can be seen below.



The hierarchical nature of this linkage structure does not allow for a probabilistic model to be built on every subset of the LT, since that would mean that sampled variables would be overwritten by samples drawn form higher order subsets e.g., when a sample is drawn from $\{x_1, x_2\}$ and later a new sample is drawn for the subset $\{x_1, x_2, x_3\}$, the values sampled for $x_1$ and $x_2$ will be overwritten until eventually a sample is drawn for the complete set of variables, overwriting all previous samples.

To overcome this, the proposed linkage tree model is combined with a new variation operator called optimal mixing, introduced in [29]. To effectively exploit the dependencies modeled at every level of the linkage tree, optimal mixing uses intermediate function evaluations to determine whether a certain permutation of problem variables was beneficial. For every solution, a donor solution is taken and for every linkage set present in the linkage tree, the variables represented by the linkage set from the original solutions are replaced by those of the donor solution. The newly created solution is evaluated and if the fitness has increased, the changes are accepted and the original solution is replaced by the new solution. If not, the changes are reverted. This process is repeated for every linkage set in the linkage tree and for every solution in the set of selected solutions. Employing this approach to variation, the gene-pool optimal mixing evolutionary algorithm GOMEA [4] is derived. This state-of-the-art evolutionary algorithm for discrete optimization uses multiple different donor solutions for every linkage set in the linkage model. Below, a partial example is shown, showing the optimal mixing steps for the three linkage sets $\{x_1, x_2\}, \{x_3\}$ and $\{x_5\}$. Here the fitness function is defined as $f(x) = \sum_{i=0}^{l} x_i$. At first, the fitness of the newly created solution is improved and thus the changes are accepted. As inserting the values of the new donor solution for the second linkage set into the new solution does not yield an improvement, these changes are not accepted. Lastly changing $x_5$ to the value of the last donor solution results in an improvement, thus, the changes are accepted, leading to the optimal solution for this problem.

original solutions $\boxed{0}\boxed{0}\boxed{1}\boxed{1}\boxed{1}$ $\xRightarrow[\{x_1,x_2\}]{\text{insert values}}$ new solution $\boxed{1}\boxed{1}\boxed{1}\boxed{1}\boxed{0}$ $\xRightarrow{f(x)=4}$ accepted

donor solution $\boxed{1}\boxed{1}\boxed{0}\boxed{0}\boxed{0}$

accepted solutions $\boxed{1}\boxed{1}\boxed{1}\boxed{1}\boxed{0}$ $\xRightarrow[\{x_1,x_2,x_3\}]{\text{insert values}}$ new solution $\boxed{1}\boxed{1}\boxed{0}\boxed{1}\boxed{0}$ $\xRightarrow{f(x)=3}$ not accepted

donor solution $\boxed{1}\boxed{1}\boxed{0}\boxed{0}\boxed{1}$

previous solutions $\boxed{1}\boxed{1}\boxed{1}\boxed{1}\boxed{0}$ $\xRightarrow[\{x_5\}]{\text{insert values}}$ new solution $\boxed{1}\boxed{1}\boxed{1}\boxed{1}\boxed{1}$ $\xRightarrow{f(x)=5}$ accepted

donor solution $\boxed{0}\boxed{1}\boxed{0}\boxed{1}\boxed{1}$

Inserting the variables from random donor solutions as employed in gene-pool optimal mixing (GOM) has an effect similar to that of sampling variables from a probabilistic model based on the frequency of the combination of variables in a population as used in model-based EDAs. However, two main improvements on the ECGA like EAs should be noted. Firstly, as a hierarchical linkage structure is used, variables can be present in multiple solutions and overlapping thus more complex structures can be used to model the linkage of a problem. Secondly, when solutions contain partially optimal combinations of variables, e.g., the first two solutions in our previous example, instead of using the fitness-value of the complete solution, a part of that solution is used and the fitness-values is recalculated while keeping all other variables as they were. The new fitness-value is thus based on only the new variables which decreases the probability of discarding partially optimal solutions because the rest of the solution is sub-optimal.

### 1.1.2. Grey-box optimization

As described earlier, when no prior information about the problem structure and objective function of an optimization problem is known, we use black-box optimization. For many problems however, some information about the problem structure is known beforehand or can be derived form the problem definition, which is called grey-box optimization. Furthermore, some grey-box optimization problems allow for partial evaluations, meaning that when a subset of the optimization variables are modified, the new objective value can be calculated by only recalculating the modified variables. In the grey-box domain, GOMEA becomes even more efficient because partial evaluations can be leveraged during optimal mixing. When exploiting the problem structure of a problem, only modified variables have to be re-evaluated which can be done in $\frac{g}{\ell}$ of the original costs of an evaluation, where $g$ is the size of the linkage set. Since almost all evaluations made by GOMEA are partial evaluations, this algorithm works particularity well in the grey-box domain, outperforming model based EDAs that often exclusively evaluate a completely new solutions to the optimization problem.

### 1.1.3. Real-valued optimization

In the real-valued domain i.e., problems with real-valued variables, crossover alone does not suffice to successfully explore the search space. Instead of recombining existing solutions, EDAs, therefore, use a probability distribution over the solution space to sample new solutions. In earlier EDAs applied to the real-valued domain, the normal distribution is estimated based on the Maximum Likelihood estimate from the selected solutions. This approach, however, has been shown to be inefficient, as the variance decreases too fast and the EA converges prematurely, even on slope-like regions of problem landscape [9]. To overcome the problem of premature convergence due to variance shrinking, multiple ideas have been introduced to reshape the estimated distribution, one of which is the Adaptive Variance Scaling (AVS) in combination with the Anticipated Mean Shift (AMS) that demonstrate excellent performance when used together to form the Adapted Maximum-Likelihood Gaussian Model (AMaLGaM) [1]. AVS is designed to counteract the variance-diminishing effect of selection by multiplying the variance stored in the covariance matrix $\Sigma$ with a multiplier that is based on the rate of improvement of the population. AMS uses the direction in which solutions were moving between the previous and current generation and extrapolates this into the probability distribution used to sample the next generation.

Similar to discrete problems, the linkage of an optimization problem remains a key element in solving large-scale (complex) optimization problems efficiently. An example of a real-valued function containing linkage between problem variables is the rotated hyper ellipsoid function defined by:

$$f(x) = \sum_{i=0}^{l} \Big( \sum_{j=0}^{i} x_j \Big)^2$$

The two dimensional version of this problem is defined by $f(x_0, x_1) = x_0^2 + (x_0 + x_1)^2$ and a contour plot of this function can be seen in Figure 1.1. Consider a solution at $\boldsymbol{x} = [5, -7]$ with $f(\boldsymbol{x}) = 29$. If we would insert the optimal value for $x_0$ into this solutions, we would get $\boldsymbol{x} = [0, -7]$ with $f(\boldsymbol{x}) = 49$, showing that moving to the optimum in one direction does not necessarily result in a better objective value. However, a small step towards the optimum does lead to an improvement in this case, consider $\boldsymbol{x} = [3, -7]$ with $f(\boldsymbol{x}) = 26$, whilst getting too close to the optimum does result in a slightly worse objective value $\boldsymbol{x} = [1, -7]$ with $f(\boldsymbol{x}) = 37$. For this function $x_0$ and $x_1$ are dependent, when variables are changed in one single dimension, only small steps towards the optimum lead to an improvement in overall fitness, whilst a big step towards the optimum leads to a decreased fitness. As a result, when updated independently, only small improvements can be made to one dimension successfully. To solve this problem efficiently, $x_0$ and $x_1$ should thus be considered dependent and updated jointly.

To encapsulate the linkage information of an optimization problem, the sampling model used by AMaLGaM can be set to sample from the full $\ell \times \ell$ covariance matrix of a problem. However, this approach has consequences when it comes to scalability. Firstly, to sample a distribution based on the full covariance matrix, a Cholesky decomposition is computed in $O(\ell^3)$ time. For high-dimensional search spaces, computing this decomposition becomes very expensive. Secondly, it is necessary to maintain a big population size to ensure enough data points can be used to build an accurate multi-dimensional sampling model when the dimensionality of the problem increases. Whilst these disadvantages might seem inescapable for strongly dependent problems, the extra computation time and large population sizes are superfluous for partially-decomposable problems which can be solved faster using other types of AEs. A univariate version of AMaLGaM was introduced that is efficient on problems with few or weak dependencies but does not scale well on problems with non-decomposable sub-components. Given the positive results produced by linkage learning methods employed in the discrete domain, a mechanism is needed that allows for accurate linkage learning in the real-valued domain as well. Since many real-valued problems allow for partial evaluations as well, introducing a method to effectively leverage partial evaluations in the real-valued domain could also lead to significant improvement in the

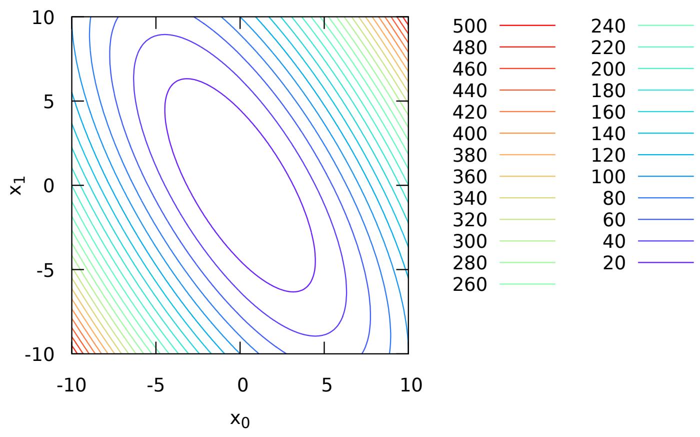


Figure 1.1: contour plot of the 2-dimensional rotated hyper hyper ellipsoid function.

real-valued domain.

Whilst such a model does exist in the discrete domain it has only recently been tried to extend to the real-valued domain. The strengths of AMaLGaM in sampling new solutions from an estimated Gaussian distribution are thereby combined with the optimal mixing operator that allows GOMEA to exploit the linkage of a problem very effectively which resulted in RV-GOMEA [5]. RV-GOMEA has been shown to indeed perform exceptionally well if the linkage structure is modeled adequately, especially in the grey-box scenario. However, difficulties arise in learning the correct linkage model online. Whilst a linkage tree based on mutual information is capable of capturing the correct linkage of a problem in the discrete domain, applying the use of mutual information of the established probability distribution to the real-valued domain does not necessarily reveal the true nature of dependencies as imposed by the fitness function. This is because selected solutions tend to align with the joint direction of improvement due to mechanisms such as the anticipated mean shift from AMaLGaM [3]. The resulting inability to correctly recognize fully decomposable sub-components can result in an over-complicated model for decomposable problems.

An alternative approach to identifying linkage is to use the changes in fitness values when changing certain variables. The most prominent method that takes such an approach is Differential Grouping [21]. Whilst this method has been shown to correctly identify the decomposability of variables, it does not allow for overlapping linkage models like the linkage tree and is computationally expensive in checking all $\ell(\ell-1)$ pairwise dependencies.

## 1.2. Research Objectives

Solving the problems stated above form the core objectives of this research project. To better display the research goals of this project, three research questions have been defined that will be answered in this thesis.

### 1.2.1. Online linkage learning

Whilst optimal mixing has provable added value in the real-world and turned out to be an effective method to exploit the linkage structure of a given optimization problem, it has never been successfully combined with an online linkage learning method in the real-valued domain. The first research question focuses on how we can leverage the strengths of the state-of-the-art for optimal-mixing RV-GOMEA with an online linkage learning method in a way that RV-GOMEA can be effectively applied to problems of which the linkage structure is not known a priori.

**Research question 1** How can we combine RV-GOMEA with an online linkage learning method that is able to correctly detect decomposability of (sub) components?

### 1.2.2. Scalability

From literature we can conclude that RV-GOMEA has shown excellent scalability in combination with predefined linkage models. Whilst multiple algorithms have been proposed to learn a linkage structure online based on fitness values. These methods are computationally expensive and do not scale well for problems that are highly decomposable.

**Research question 2** How can we minimize the overhead of our online linkage learning method on the scalability of the optimization algorithm?

### 1.2.3. Linkage structure

Linkage Trees have been widely used to model the linkage structure of a problem. It has already been suggested that pruning the linkage tree might yield improvements in the discrete domain [8]. Therefore it makes sense to re-evaluate this linkage structure and see if improvements can be made by decreasing the size of the linkage tree, which in turn also results in a more specific optimal mixing phase where only linkage sets containing actual dependencies are mixed.

**Research question 3** How can we translate pairwise dependencies to a linkage structure that can capture the higher order dependency structure of an optimization problem?

## 1.3. Outline

This thesis contains a full article in which RV-GOMEA is extended with a fitness-based linkage learning method that aims to overcome the earlier described problems in linkage and scalability. The resulting EA is compared with an existing state-of-the-art AE for black-box real-valued optimization, AMaLGaM, as well as with the original RV-GOMEA and RV-GOMEA with predefined linkage structures. To validate the performance of our method the dependencies found will be evaluated and the resulting linkage structures will be compared with the expected linkage structures for known benchmark problems. Finally, this thesis is concluded with a discussion and future work section.

# 2

# Article to be submitted

## 2.1. Introduction

A key strength of many state-of-the-art-model based evolutionary algorithms (EA's) lies in the effective exploitation of a problem's linkage structure [5, 21, 24, 26]. When the linkage structure of a problem is known, this information can be used to solve the optimization more effectively. If a problem is fully decomposable into sub-problems, these lower-dimensional sub-problems can be solved independently to achieve better efficiency. Conversely, if a problem is (partially) inseparable and its variables are strongly dependent, trying to solve the problem with a model that wrongly assumes decomposability is very inefficient. This is known to hold for problems with discrete (binary) variables, e.g., deceptive trap function [27], as well as real-valued variables, e.g., rotated ellipsoid function [16].

A well-known approach that effectively exploits the linkage model of a problem in the discrete domain is Optimal Mixing (OM) [29]. In Optimal Mixing, variables modeled in the same linkage set will be affected by recombination together, ensuring that no valuable information captured in the specific combination of variables is lost. The recombination operator applies recombination to partial solutions by iterating over all linkage sets in a linkage model. For every linkage set, recombination is executed only on the variables represented in the current linkage set exchanging partial solutions between individuals. If this recombination leads to an improved fitness of the individual solution, the changes are accepted. The Gene-pool Optimal Mixing Evolutionary Algorithm (GOMEA) [4] randomly chooses a donor solution for different linkage sets, hereby exploiting the entire gene-pool in search for optimal sub-solutions. In black-box optimization (BBO) this approach has shown excellent scalability, effectively exploiting the problem structure of the problem at hand. When partial solutions can be leveraged, GOM has proven to be even more efficient, allowing for the inexpensive re-evaluation of partially changed solutions. The recently introduced RV-GOMEA [5] leverages the strengths of GOMEA for the real-valued domain. The linkage learned method employed by GOMEA has shown excellent performance and scalability in the discrete domain, but some issues have been encounter when applying the same approach to the continuous search spaces of the real-valued domain.

A fundamental drawback of the current approach lies in its inability to correctly recognize fully decomposable sub-components. At the root of this lies that this method has problems in identifying independent variables in the case of real-valued variables because selection causes the solutions to align with the density contours of the search space. Additional mechanisms such as the anticipated mean shift in AMaLGaM [3] and the evolution path in CMA-ES [7] cause the solution and Gaussian model of variation to align with the joint direction of improvement (i.e., the gradient in smooth problems). In either case, the mutual information of the Gaussian model will indicate that dependencies exist, even if this is not the case, e.g., on the sphere function, especially when the population is initialized far away and not bordering the optimum. Moreover, the method is based on the spread of the population and often many generations are needed for the linkage structure to be properly exposed by a population.

An alternative approach to identifying the linkage structure of a problem is based on measuring the changes in fitness values by perturbing certain variables. This method was first introduced in combination with the greedy linkage learning approach known as Differential Grouping [21]. Whilst this method is able to correctly identify independent variables, it does not allow for overlapping linkage sets nor does it define a comparable measure on the dependence of variables. Lastly, when the problem consists mainly of decomposable sub-components, learning the linkage model is unnecessarily computationally expensive, as all of the $\ell(\ell-1)$ possible pairs have to be checked, even for completely decomposable problems.

In this paper, we try to overcome the earlier stated drawbacks of existing linkage learning methods by using the fitness-based dependency strengths to build an adapted linkage model based on the Linkage Tree as used in RV-GOMEA. Two different linkage model building methods are proposed. Both methods separate decomposable sub-problems as much as possible without separating non-decomposable variables that are strongly dependent. The resulting linkage models are integrated into RV-GOMEA, which has been proven to perform excellently on real-valued benchmarks when correct linkage models are provided [5]. The introduced methods will be compared with existing linkage learning methods in combination with RV-GOMEA for a variety of benchmark problems. The hypothesis is that the proposed method is able to scale almost identically to offline learned linkage models but without the need of problem-specific knowledge. In the black-box domain, we expect a performance similar to that of AMaLGaM, something that has not been achieved before.

As for many real-world applications of (RV-)GOMEA like brachytherapy treatment planning [14, 17], deformable image registration [18] and more [2, 15], the optimal linkage model is not known and strong dependencies are imposed through geometry, e.g., transformation vector field nodes or potential windmill locations that are near each-other are strongly dependent, but those far apart are weakly dependent. This results in problem structures that are only non-decomposable to some extent. The provable added value of RV-GOMEA for these real-world problems could be increased even further if correct linkage models could be learned efficiently online.

The remainder of this article is structured as follows. In Section 2.2 we elaborate on the existing RV-GOMEA. Existing methods to model the dependencies of an optimization problem are discussed in Section 2.3. In Section 2.4 our newly proposed incremental approach for learning the linkage structure of a problem is introduced. The benchmark problems used to validate the performance of our method are introduced in Section 2.5, and Section 2.5.3 shows scalability results on these problems. The implications of our work and further challenges ahead are discussed in Section 2.7. Lastly, this article is concluded in Section 2.8 with a conclusion.

## 2.2. RV-GOMEA

One of the key elements of GOMEA is its variation operator: the gene-pool optimal mixing method. This method uses a so-called Family of Subsets (FOS) to exploit the linkage structure of a problem. The current version of RV-GOMEA [5] is a combination of the existing GOMEA [4], which performs excellently in the discrete domain, extended with a continuous sampling model as employed in the state-of-the-art for numerical optimization, the Adapted Maximum-Likelihood Gaussian Model Iterated Density-Estimation Evolutionary Algorithm (AMaLGaM) [1].

### 2.2.1. Family of subsets

The linkage structure of a problem is modeled in GOMEA using a FOS, denoted as $\mathcal{F}$. The set $S = \{0, 1, ..., \ell-1\}$ contains all problem variables and every set $\mathcal{F}_i \in \mathcal{F}$ is a subsets of $S$. The complete FOS $\mathcal{F}$ is a subset of the power-set $P(S)$ of $S$. Lastly, the FOS $\mathcal{F}$ is complete, meaning that every problem variable is represented in at least one subset of $\mathcal{F}$. There are various methods for constructing a FOS. Different FOS structures have proven to work best on different problems [29]. Here we partially focus on two types:

Marginal product FOS
A marginal product FOS is defined as a set $\mathcal{F}$ where for every $\mathcal{F}_i$ and $\mathcal{F}_j \in \mathcal{F}$ it holds that $\mathcal{F}_i \cap \mathcal{F}_j = \emptyset$. The univariate FOS is a special case of a marginal product FOS with $|\mathcal{F}| = \ell$ and thus $|\mathcal{F}_i| = 1$ for all $\mathcal{F}_i \in \mathcal{F}$.

Linkage tree FOS
The linkage tree FOS is most commonly used and shown to be the most universal in discrete optimization [28, 29]. The defining property of a linkage tree FOS is that every set $\mathcal{F}_i \in \mathcal{F}$ that contains more than one set is the union of two other sets in $\mathcal{F}$. Conceptually the linkage tree is built by iteratively merging the two FOS elements with the highest dependence $\mathcal{F}_i$ and $\mathcal{F}_j$ to form a new FOS element $\mathcal{F}_k$ that is

added to the FOS, thus $\mathcal{F}_i \cup \mathcal{F}_j = \mathcal{F}_k$. When only pairwise-dependencies are used an implementation exists that allows for an LT FOS to be built in $O(n\ell^2)$ time [10]. The process of iteratively merging linkage sets is repeated until no more merges are possible, i.e., the full set of variables is added or the maximum linkage set size $|\mathcal{F}_k| = 100$ is reached.

### 2.2.2. Gene-Pool Optimal Mixing

The core principle of GOMEA, Gene-pool Optimal Mixing (GOM), mixes the population following the FOS subsets represented in the linkage model. Variables that are represented in the same FOS subset will be recombined together. In the real-valued domain, recombination alone does not suffice as one needs to sample values not currently present in the population and thus a continuous model is needed. In the recently introduced RV-GOMEA [5] to sample solutions in the space represented by a FOS subset, a Gaussian sampling model is used as found in AMaLGaM [1]. During one generation of RV-GOMEA, for every $\mathcal{F}_i \in \mathcal{F}$ of size $|\mathcal{F}_i| = k$ a $k-$dimensional AMaLGaM model is estimated from the $n \cdot \tau$ best individuals in the population $\boldsymbol{P}$ ($\tau$ being the fraction of solutions selected from the population, in this case 0.35 as used in original AMaLGaM [1]). To create new offspring, all linkage sets $\mathcal{F}_i$ are considered in random order. For every solution $|\mathcal{F}_i|$ new values are sampled from the AMaLGaM model and inserted into the existing solution. If this results in an increase in fitness, the changes are accepted. If not, the change is accepted with probability $p^{accept} = 0.05$. The next linkage set is considered. If an individual does not improve for a certain number of generations, a method called forced improvement is applied to alter the individual following a convex linear combination of the parent solution and the elitist solution of the population, i.e. by moving it closer to the elitist solution. To obtain good performance in an EA it is often important to correctly set the population size parameter. However, the best-suited population size is problem dependent and can thus not be set without any problem specific knowledge. To avoid needing to tune the population size parameter, RV-GOMEA uses an interleaved multi-start scheme (IMS) that runs multiple independent EA instances with growing population sizes. As smaller population sizes converge quicker but get stuck in local optima, an instance will be terminated once it is outperformed by another instance that has a larger population size.

### 2.2.3. Grey-Box domain

Most of the research done one EAs is aimed at black-box optimization problems where no knowledge about the problem or its underlying structure is known. In this article, we also consider a domain of grey-box optimization problems where partial problem evaluations can be performed. Whilst partial evaluations can be applied, that does not impose that the optimal problem structure is known and thus in both domains, effective linkage learning plays an important part in optimization. In a grey-box setting, partial evaluations allow for the recalculation of fitness when there are only few, e.g. k, modified variables in $O(g(k))$ time, rather than incurring the $O(g(\ell))$ overhead of full evaluations since these are only needed when all variables are changed. With $g()$ typically being a polynomial function e.g. $g(\ell) = l$ or $g(\ell) = \ell^2$. The cost of one partial function evaluation is therefore counted as $\frac{k}{\ell}$ with $k$ as the number of changed variables. Since the optimal mixing phase of RV-GOMEA makes almost exclusively partial modifications to existing solutions, RV-GOMEA can very effectively leverage partial evaluations which makes it an effective algorithm for grey-box problems.

## 2.3. Existing linkage learning methods

Multiple methods have been proposed to identify the structure of an unknown optimization function. In this section two of these methods will be introduced. Firstly, a population-based method that uses the spread of a population in the search-space to model dependencies is introduced. Secondly, a fitness-based method where the fitness values are directly used to measure linkage between variables is introduced. Both of these methods focus on pairwise dependencies. Once these dependencies are known, dependencies between subsets of variables are extrapolated from them.

### 2.3.1. Distribution-based methods

One of the methods used to extract dependencies based on the distribution of a population is the Mutual Information method (MI) [13]. The mutual information $MI_{ij}$ of variables $x_i$ and $x_j$ defines how much information about $x_i$ we can derive by knowing $x_j$ and vice versa. The MI is computed based on the probability distribution associated with the variables. In the case of real-valued variables, a parametric distribution is often used. In our case, as the AMaLGaM model is essentially a normal distribution, to calculate the MI between two variables $x_i$ and $x_j$, the Pearson product-moment correlation coefficient $r_{ij}$ can be used with $r_{ij} \in [-1, 1]$. A high absolute value of $r$ corresponds to a high linear correlation between $x_i$ and $x_j$. The mutual information between $x_j$ and $x_j$ is defined as follows:

$$\mathrm{MI}_{ij} = \log\left(\sqrt{\frac{1}{1-(r_{ij})^2}}\right) \tag{2.1}$$

$$\text{where } r_{ij} = \hat{\Sigma}_{ij}/\left(\hat{\sigma}_i\hat{\sigma}_j\right) \in [-1, 1] \tag{2.2}$$

### 2.3.2. Fitness-based methods

The second method we consider to define whether two variables interact is the Differential Grouping method [21]. This method is directly based on fitness values and classifies a pair of variables as either separable or non-separable specifically by comparing the difference in fitness whilst making the exact same perturbation for $x_i$ for different values of $x_j$. Variables $x_i$ and $x_j$ are said to interact when $|\Delta_i - \Delta_{i,j}| \geq \epsilon$ for some user-defined small $\epsilon$, where $\Delta_i$ and $\Delta_{i,j}$ are defined as follows:

$$\Delta_i = (f(\boldsymbol{x})|x_i = a_i, x_j = a_j) - (f(\boldsymbol{x})|x_i = a_i + b_i, x_j = a_j) \tag{2.3}$$

$$\Delta_{i,j} = (f(\boldsymbol{x})|x_i = a_i, x_j = a_j + b_j) - (f(\boldsymbol{x})|x_i = a_i + b_i, x_j = a_j + b_j) \tag{2.4}$$

where $a_i$ and $b_i$ can be any real value as long as $x_j$ and $x_j$ remain within the function bounds. In this method $a_i$ and $b_i$ are selected randomly such that for every $x_i$, $a_i$ and $a_i + b_i$ fall within the bound for $x_i$ inside the current population.

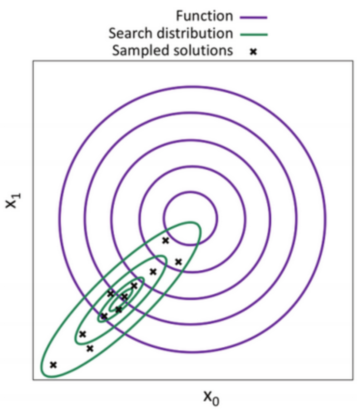


Figure 2.1: Non-decomposable Gaussian search distribution (green iso-lines) learned from a population of sampled solutions on a 2D version of the decomposable sphere problem (purple iso-lines).

## 2.4. Scaled fitness-based linkage learning

To learn a linkage tree FOS, it is necessary to define a notion of linkage, or dependency, strength between pairs of variables. We denote $d_{i,j}$ as the pairwise dependency strength between $x_i$ and $x_j$ where:

$$d_{i,j} = \begin{cases} 1 - \Delta_{i,j}/\Delta_i & \text{if } \Delta_i \geq \Delta_{i,j} \\ 1 - \Delta_i/\Delta_{i,j} & \text{otherwise} \end{cases} \tag{2.5}$$

From $d_{i,j}$ a matrix $D$ of size $\ell \times \ell$ can be constructed, storing all pairwise dependency strengths of an $\ell$-dimensional problem. By definition of equation 2.5 the values of this matrix will lie within $[0, 1)$ with 0 for independent variables and $d_{i,j} > 0$ indicating some interaction between $x_i$ and $x_j$. It is worth noting that even though $d_{i,j}$ does not represent a absolute dependency strength between variables, it can be used to compare the relative pairwise dependency strength by comparing $d_{i,j}$ and $d_{i,k}$. This property makes it possible to learn a linkage tree FOS based on the information stored in $D$ as described in [5].

### 2.4.1. Analysis of overhead

Filling matrix $D$ requires $\frac{\ell(\ell-1)}{2}$ dependency checks. For each of these dependency checks, four evaluations are needed, which results in a total number of $2\ell(\ell-1)$ evaluations. It is shown in [22] that it is possible to decrease the number of evaluations to $1+\ell+\frac{\ell(\ell-1)}{2}$ by using the same $a_i$ and $b_i$ for every check and storing $f(x)|_{x_i=a_i+b_i}$ for every $x_i \in x$. When partial evaluations van be leveraged, the overhead can be decreased even further. Since only one variable is changed for the evaluations done to compute $d_{i,j}$, the number of changed variables $k=1$ and therefore it is possible to decrease the total number of evaluations even further to $1+\frac{1}{\ell}(\ell+\frac{\ell(\ell-1)}{2}) = 2+\frac{\ell-1}{2}$.

### 2.4.2. Incremental dependency updating

It is plausible to assume that real-world high-dimensional problems tend not to be fully dependent with strong dependencies (e.g., a 1000-dimensional rotated ellipsoid), but to be non-decomposable only to some extent. We, therefore, aim to slightly bias our method toward the assumption of less-than-fully dependent problems by decreasing the number of evaluations needed on sparsely dependent problems. To this end, to spread the computational load of filling the dependency matrix, every $D_{ij}$ is initially set to the default value of 0 (independent). In each generation, $\ell$ random pairs are evaluated and a new FOS is learned. Since the right linkage model is not dependent on the population size, the same linkage structure is used for every population that is maintained with respect to the IMS explained in Section 2.2.2. We call this process incremental dependency updating. The pseudo-code for this algorithm can be found in algorithm 1. The process of checking all $\frac{\ell(\ell-1)}{2}$ pairs will be called a dependency cycle. When all pairs have been checked, no checks will happen for $2^k$ generations where $k$ is the number of cycles that have taken place already. To better allocate the computational budget during optimization of a problem that is suspected to be independent, the dependency cycle is stopped prematurely and started again after $2^k$ generations with new random pairs. Specifically, the dependency cycle is stopped whenever no dependencies are found in one iteration ($\ell$ checks) and if the average number of found dependencies over all dependence cycles so far is smaller than some minimum value. We used $\frac{2}{\ell}$ as this was empirically found to work well on a variety of problems. Since the matrix is initialized with 0's the computational resources are geared more toward univariate optimization which results in a slight bias to decomposable problems. Yet, by restarting the dependency cycle every $2^k$ generations, changes in function landscapes can be captured and the dependencies can be updated accordingly. As a result of the initialization of the dependency matrix, all variables are assumed to be independent and will only be considered to be dependent once an actual dependency is detected. Stopping the dependency detection when no dependencies are found is therefore not expected to significantly change the outcome of the dependency checks.

### 2.4.3. Pruning

Based on the information obtained from the scaled fitness-based dependency detection, a pruning method for the Linkage Tree FOS described in Section 2.2.1 can be used. The goal of pruning is to eliminate unnecessary linkage sets. A smaller FOS reduces the number of function evaluations and time spent on GOM per generation. If only the linkage sets that best capture the dependency between variables are correctly maintained, the efficiency of GOMEA may very well improve.

Consider the moment during the learning of the FOS, that two linkage sets $\mathcal{F}_i$ and $\mathcal{F}_j$ are to be merged to create $\mathcal{F}_k = \mathcal{F}_i \cup \mathcal{F}_j$. If all variables in $\mathcal{F}_k$ are pairwise dependent, $\mathcal{F}_i$ and $\mathcal{F}_j$ are removed from the FOS. Since all variables in $\mathcal{F}_k$ are dependent, mixing these variables together (which in RV-GOMEA entails sampling form a joint Gaussian distribution) will likely yield better results than separately mixing the variables from $\mathcal{F}_i$ or $\mathcal{F}_j$. Similarly, if there is no pairwise dependence between any variable in $\mathcal{F}_i$ and any other variable in $\mathcal{F}_j$, $\mathcal{F}_k$ is removed from the FOS.

In case that there are some pairwise dependencies between subsets, but not every variable in $\mathcal{F}_i$ is dependent on every variable in $\mathcal{F}_j$, the problem consists of non-decomposable overlapping sub-components. For this case we present two different pruning approaches, resulting in different FOS structures:

(partial) Linkage Tree

In this case, the two subsets $\mathcal{F}_i$ and $\mathcal{F}_j$ are merged together into $\mathcal{F}_k$ and all linkage FOS sets are kept in the FOS. This approach will result in a (partial) linkage tree where the biggest linkage sets are the size of the biggest non-decomposable sub-components.

**Algorithm 1** Incremental dependency updating

```
1: $pairs \leftarrow shuffelePairs()$                ▷ all possible pairs, randomly ordered
2: $w \leftarrow 0$                                  ▷ The number of waiting cycles
3: $k \leftarrow 0$                                  ▷ The number of cycles that have take place
4: $g \leftarrow 0$                                  ▷ The generations that have take place
5: $t \leftarrow 0$                                  ▷ The total dependencies found
6: while not terminated do
7:     if $w = 0$ then
8:         $d =\leftarrow evaluateLpairs(pairs)$      ▷ Dependencies found
9:         $t = t + d$
10:        $g = g + 1$
11:        if ($d = 0$ and $t \leq 2 \cdot g$) or all pairs are evaluated then
12:            $w \leftarrow 2^k$
13:            $k \leftarrow k + 1$
14:            $pairs \leftarrow shuffelePairs()$
15:            $t, d \leftarrow 0$
16:    else
17:        $w - 1$
18:    continue RV-GOMEA
19:    ...
```

Marginal product
The second approach ignores the subset of dependencies between $\mathcal{F}_i$ and $\mathcal{F}_j$ and keeps only the fully dependent linkage sets $\mathcal{F}_i$ and $\mathcal{F}_j$ in the FOS without merging any more sets. Combined with the other pruning steps this will always create a marginal product FOS, containing every variable exactly once.

### 2.4.4. FOS-based population size

With the problem-specific knowledge obtained by our linkage learning method, we can project a minimally required population size needed for RV-GOMEA to work well. If $\gamma$ is the size of the biggest linkage set in $\mathcal{F}$ then following [3] the minimal population size $n_{base}$ needed can be calculated as $n_{base} = 17 + 3\gamma\sqrt{\gamma}$. We combine this baseline with the IMS described in Section 2.2.2. Across all populations in the IMS, one FOS is maintained since the linkage structure of a problem is not dependent on the population size. The incremental dependency updating as described in Section 2.4.2 thus happens in every generation of the IMS. Every time a new FOS is built and the size of the biggest fully dependent linkage set has increased, $n_{base}$ is recalculated and the populations with a population size smaller than $n_{base}$ are stopped.

A second use of population sizing is if the linkage tree FOS is built, and not all variables in one linkage set are pairwise dependent, e.g., if $\mathcal{F}_i = \{1, 2, 3\}$ and $d_{1,2} = 0.5, d_{1,3} = 0, d_{2,3} = 0.5$. The population size is then not updated as described earlier, but the FOS set $\mathcal{F}_i$ is only added to $\mathcal{F}$ if $|\mathcal{F}_i| \leq \gamma_{max}$, with $\gamma_{max} = (\frac{n-17}{3})^{\frac{2}{3}}$ the maximal acceptable linkage set size for a population in the IMS of size $n$.

## 2.5. Experiments

### 2.5.1. Benchmark algorithms

To conduct our experiments we compare the performance of RV-GOMEA in combination with our two proposed methods for fitness-based linkage learning, described in Section 2.4, to existing versions of RV-GOMEA and AMaLGaM. Table 2.1 gives an overview of all versions of RV-GOMEA and AMaLGaM that are used for our experiments, differing only in how the linkage model is defined or learned. For our newly proposed methods, we make a distinction between a fitness-based linkage tree (RV-GOMEA-FBLT) and a fitness-based marginal product linkage structure (RV-GOMEA-FBMP) of which the differences between the resulting models are described in Section 2.4.3. Because the second pruning approach of the fitness-based linkage learning method will always create a marginal product linkage structure, this method can also be used to create a linkage structure for AMaLGaM, equipping it with a linkage learning method for the first time. The MP linkage model is used to restrict the

| Linkage model | Description |
|---|---|
| RV-GOMEA-FBLT | RV-GOMEA with a learned fitness-based linkage tree |
| RV-GOMEA-FBMP | RV-GOMEA with a learned fitness-based marginal product model |
| RV-GOMEA-UNI | RV-GOMEA with a predefined univariate linkage model |
| RV-GOMEA-UNI5 | RV-GOMEA with a predefined linkage model using blocks of 5 consecutive variables. |
| RV-GOMEA-FULL | RV-GOMEA with a predefined full linkage model |
| RV-GOMEA-LT | RV-GOMEA with a learned linkage tree model, based on Mutual Information as described in [5] |
| RV-GOMEA-DG | RV-GOMEA with a learned linkage structure based on differential grouping as proposed in [21] |
| AMaLGaM-UNI | AMaLGaM with a predefined univariate linkage model |
| AMaLGaM-FB | AMaLGaM with a learned fitness based linkage model |

Table 2.1: All algorithms used for our experiments.

covariance matrix of AMaLGaM, i.e., the covariance is assumed to be 0 for variables in different FOS elements. AMaLGaM-FB will be compared to the previously described versions of RV-GOMEA.

### 2.5.2. Benchmark problems

To study the impact of different types of linkage learning on the performance of RV-GOMEA, we first consider a set of six optimization problems. Whilst some of these problems are not decomposable, none of the used benchmark problems are fully dependent, aligned with the idea that real-world high-dimensional optimization problems are highly unlikely to have a linkage structure where each variable is dependent on every other variable. The problems we will consider are Sphere, Michalewicz, Rastrigin, Rosenbrock, Sum of Rotated Ellipsoid Blocks (SoREB) and an overlapping version of SoREB.

The first three benchmark functions exhibit no dependencies. First, we consider the sphere function which is a widely-used benchmark for real-valued optimization. The sphere function has a smooth landscape and no local minima.

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

Second, we consider the Michalewicz function. In comparison with the smooth sphere function, it contains $l!$ local optima that are unevenly distributed throughout the search space. The definition of the Michalewicz function is as follows, with $x_i \in [0, \pi]$:

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

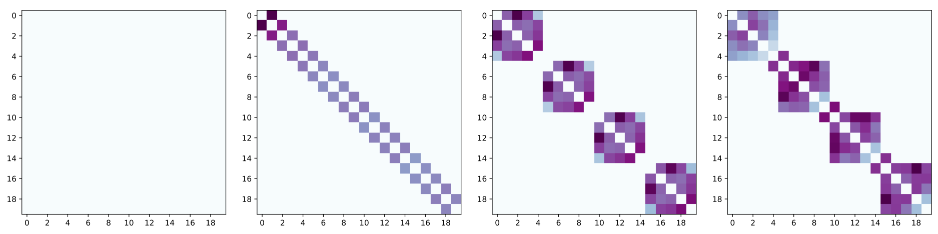
Next, the Rastrigin function is also a non-linear and multi-modal function but its local minima are evenly spread and superimposed on the sphere function.

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

The fourth benchmark function we consider is the Rosenbrock function that contains a parabolic valley with one global optimum and one local optimum for $4 \leq l \leq 100$ [25]. Finding the global optimum in this valley is considered relatively hard. Finding the valley is trivial, but converging to the global minimum requires a search through this parabolic valley that requires differently oriented covariance matrices at different points during the search. By design, every consecutive pair of optimization variables in this function is dependent which results in $l-1$ overlapping dependent components. The definition of the Rosenbrock function is 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 SoREB function uses a rotation function $R_\theta$ that defines the counterclockwise rotation of a vector around the origin by an angle of $\theta$ and an ellipsoid function $f_{\text{Ellipsoid}}$ . Due to the construction of



(a) Sphere, Rastrigin and Michalewicz (b) Rosenbrock (c) SoREB (d) OSoREB

Figure 2.2: Heatmaps over 30 runs for $\ell = 50$ with the Fitness-Based measure

the SoREB function, the variables in every block of $k$ consecutive optimization variables have strong dependencies, but are independent from any other optimization variables outside of their block. For our benchmark we use a block size of $k = 5$ and a rotation of $\theta = 45°$. The ellipsoid function and the SoREB function are defined as follows:

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

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

The SoREB function is a problem containing only non-overlapping non-decomposable sub-components of size $k$. We define an overlapping version of this problem as OSoREB. In addition to the original SoREB problem, a second set of SoREB blocks is used with blocks of length 2 for every pair of consecutive parameters in successive blocks of SoREB with $k = 5$ (e.g., for $x_4, x_5$ and $x_9, x_{10}$). The definition of OSoREB is as follows:

$$f_{\text{OSoREB}}(x,k) = f_{\text{SoREB}}(x,k) + \sum_{i=1}^{\ell/k-1} \left[ f_{\text{Ellipsoid}} \left( R_\theta \left( [x_{ki-1}, x_{ki}] \right) \right) \right]$$

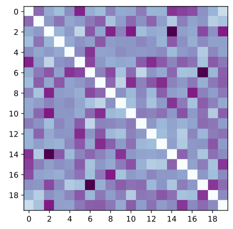
## 2.5.3. Setup

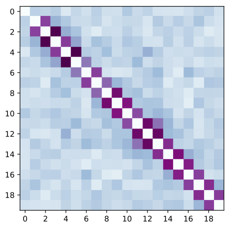
Evaluating linkage learning

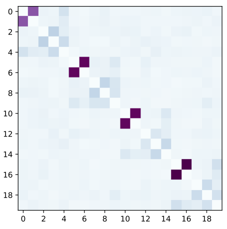
We employ different means to verify the validity and impact of different linkage learning algorithms. The dependency matrices produced by our fitness-based method is compared to the matrices produced by the existing mutual information method. The dependency matrices give us valuable insight into the pairwise dependencies found during optimization, which is used to create the FOS structures used for GOM. For all benchmark problems, the pairwise dependencies are known and can thus be easily compared to the learned dependency matrices. Average dependency matrices are be computed over 30 independent runs with $\ell = 50$ for all benchmark problems. As our linkage learning approach builds a model on the relative dependencies between variables, the heatmaps shown are normalized according to min-max feature scaling, such that all values range between $[0, 1]$ without loss of information. For all non-overlapping benchmark problems, we also verify whether the learned linkage sets corresponds to the combination of optimal linkage sets that capture all existing dependencies but do not combine independent variables.

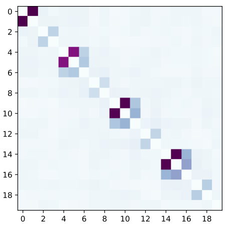
Evaluating scalability

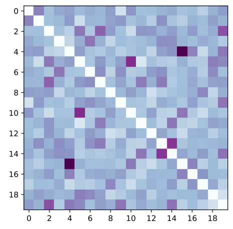
Another important aspect of our evaluation is the scalability analysis of RV-GOMEA on a subset of the benchmark functions. Scalability graphs are commonly used to benchmark the performance of optimization algorithms because they summarize the most important aspects of the algorithm's performance as well as provide a prediction regarding the performance on higher-dimensional problems. We compare the scalability of RV-GOMEA when using different linkage learning models as described in Section 2.5.1. For a broader comparison of the previously existing versions of RV-GOMEA with different state-of-the-art EAs, we refer to [5]. For visibility we have only plotted RV-GOMEA-FBMP

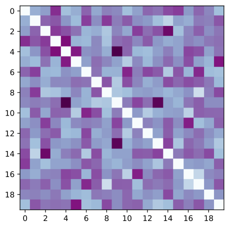

(a) Sphere, 1e4 evaluations

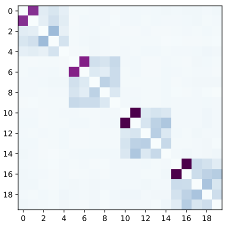

(b) Rosenbrock, 1e4 evaluations

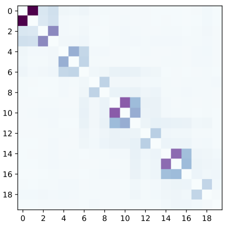

(c) SoREB, 1e4 evaluations


(d) OSoREB, 1e4 evaluations


(e) Rastrigin, 1e4 evaluations


(f) Michalewicz, 1e4 evaluations


(g) SoREB, 1e5 evaluations


(h) OSoREB, 1e5 evaluations

0.0
0.2
0.4
0.6
0.8
1.0

Figure 2.3: Heatmaps over 30 runs for $\ell = 50$ with the Mutual Information measure

on non-overlapping problems since RV-GOMEA-FBLT both produce the same FOS structure and thus have the same scalability.

For every benchmark problem, 30 independent runs are performed with a time limit of $10^4$ seconds (roughly 2 hours and 45 minutes). All experiments are performed on a 64-core (4 x 16-core AMD Opteron(tm) Processor 6386 SE) server running Fedora 28 where each run is performed on a single core. In every run the population is randomly initialized between $[-115, -100]$ for every variable and $[0, \pi]$ for $f_{\text{Michalewicz}}$ i.e, definitely not bracketing the optimum. A problem is considered to be sufficiently minimized if the elitist solution reaches a value to reach (VTR) of $1e{-}10$ if the optimum value is 0 (which is the case for all problems except $f_{\text{Michalewicz}}$) and 95% of the optimum for $f_{\text{Michalewicz}}$).

If all runs are solved within the time limit, the problem size is doubled, until the maximum dimensionality of $10^4$ is reached. Since a large range of real-world optimization problems, as well as all our benchmark problems, allow for partial evaluations, we have decided to focus mainly on grey-box optimization in order to obtain a realistic view of the performance of our algorithm on most real-world optimization problems.

## 2.6. Results

### 2.6.1. Dependency matrices

Figure 2.2 shows heatmaps of the dependency matrices for all benchmark problems as calculated by RV-GOMEA-FBLT. For the mutual information measure shown in Figure 2.3, the dependency matrices for SoREB and OSoREB become more specific and less noisy after more evaluations, this becomes apparent in Figures 2.3h and 2.3g. All heatmaps shown are zoomed in to the first 20 dimensions, making it easier to inspect the dependencies, whilst still optimizing a 50-dimensional problem. On the fully decomposable problems: sphere, Rastrigin and Michalewicz the mutual information measure is not able to correctly identify the independence of the variables. Even if variables are independent, a correlation is measured. In combination with the normalization of the dependency matrices used to create a FOS we can conclude that this measure encounters high amounts of noise for decomposable problems. The fitness-based method is able to correctly identify two independent variables, which results in a matrix containing only 0's for all three decomposable benchmark problems. By definition of the SoREB and OSoREB problems, the first two variables of a block exercise the highest influence on the total sum and thus show the strongest dependencies. This strong dependency can be seen in all Figures 2.2c, 2.2d, 2.3c, 2.3d, 2.3g and 2.3h, but only the fitness-based method (2.2c, 2.2d) and the mutual information method on SoREB after 1e5 evaluations have extracted the correct block structure without displaying noise between decomposable variables. One of the main drawbacks of the mutual information method now, becomes immediately clear, because even though the dependencies will eventually be found, RV-GOMEA is able to already solve this instance of SoREB within 1e5 evaluations if a proper FOS is provided [5].

### 2.6.2. FOS structures

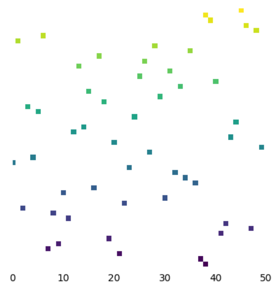
We can divide the benchmark problems into non-overlapping and overlapping optimization problems. Sphere, SoREB, Michalewicz, and Rastrigin are non-overlapping. The Rosenbrock problem and the OSoREB problem contain overlapping components. To better visualize the FOS structures produced by our methods, Figures 2.4 and 2.5 show the elements captured in a single FOS structure. In these figures, the horizontal axis represents the index of the optimization variables. Every linkage set is represented in one row of the figure, the highlighted $x$ values mark the presence of that optimization variable in that one linkage set. Colors are used to improve visibility but do not contain any additional information.

#### Non-overlapping benchmarks

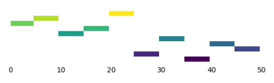
For the non-overlapping problems, the optimal FOS structures are known and can be compared with the FOS structures generated by RV-GOMEA-FB. We will look at the algorithm's FOS structures created for sphere and SoREB. Rastrigin and Michalewicz are not considered here since for these problems the dependency matrix and thus FOS structure is equal to that of sphere. The FOS structures found and used by RV-GOMEA-FB for sphere and SoREB that can be seen in Figures 2.4a and 2.4b are as expected, considering the dependency matrices discussed in Section 2.6.1. For sphere, it holds that a fully decomposable problem can best be represented by a univariate FOS consisting of exactly $l$ subsets, each containing a single optimization variable. As stated in Section 2.5.2 the SoREB function is rotated in blocks of $k$ consecutive optimization variables with $k = 5$ in this case. Thus the dependencies of SoREB should be represented by a marginal product FOS containing blocks of size $k$ as is the case in Figure 2.4b.

#### Overlapping benchmarks

The optimal FOS structures for overlapping benchmarks are unknown because no marginal product FOS can describe all dependencies without combining independent variables or leaving out dependencies. For these overlapping problems, a distinction is made between RV-GOMEA-FBLT and RV-GOMEA-FBMP. The latter linkage learning method creates a marginal product FOS, whereas the former continues to build a linkage tree, eventually containing all non-decomposable linkage sets. Figure 2.5 shows the FOS structures created for Rosenbrock and OSoREB by RV-GOMEA-FBLT and RV-GOMEA-FBMP where $l = 20$.



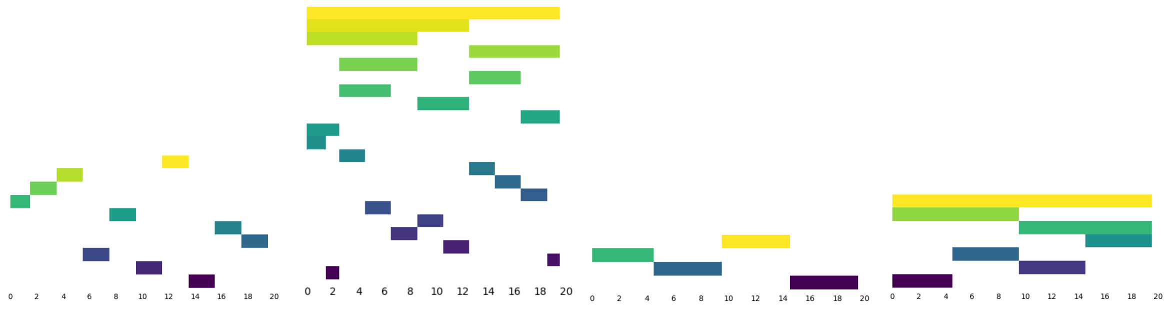
(a) Sphere



(b) SoREB

Figure 2.4: FOS structures for $\ell = 50$ with every block representing a single linkage set

#### Scalablility



(a) Rosenbrock MP (b) Rosenbrock LT (c) OSoREB MP (d) OSoREB LT

Figure 2.5: Linkage structures for $\ell = 20$ with every block representing a single linkage set

### 2.6.3. Scalability analysis

Figure 2.6 shows the performance of different linkage learning methods in combination with RV-GOMEA on all six benchmark problems.

(Partially) Decomposable problems

For the fully decomposable problems Sphere, Rastrigin and Michalewicz, we can observe that RV-GOMEA-FBLT and RV-GOMEA-FBMP scale as well as RV-GOMEA-UNI and better than AMaLGaM-FB. Whilst RV-GOMEA is partially based on AMaLGaM, the optimal mixing employed in RV-GOMEA has not been shown to outperform the model based EDA approach used by AMaLGaM on all benchmark problems before. In this paper suitable comparison has been made between RV-GOMEA and AMaLGaM as both algorithms have been provided with the same linkage learning method and better results have been obtained by RV-GOMEA, implying that the optimal mixing of (RV-)GOMEA has significant added value in the real-valued domain.

The incremental dependency updates have minimal overhead on the overall scalability as opposed to the original differential grouping (RV-GOMEA-ODG) where the number of dependency checks needed to build a linkage model scales quadratically with the problem size.

On SoREB, a non-univariate linkage model is used as RV-GOMEA baseline, containing blocks of 5 consecutive optimization variables (RV-GOMEA-UNI5). RV-GOMEA-FBLT and RV-GOMEA-FBMP find the same structure (fig. 2.4b), but in GBO a small overhead is noticeable as the problem size increases. This overhead is caused by a decrease in the ratio of dependent to independent pairs as the problem size increases. By the definition of SoREB, every variable is dependent on the other $k-1$ variables in its block and has no dependency on all other $\ell-k$ problem variables. If the dimensionality of SoREB increases, the number of blocks increases, but the size of the blocks will always be equal to $k$. In other words, as $\ell$ grows larger, the number of independent variables $\ell-k$ becomes larger. Eventually, every possible pair has to be checked to ensure all existing dependencies are found, which ultimately still causes a quadratic overhead compared to the baseline RV-GOMEA-UNI5, which uses a predefined structure, but still scales better than the original fitness-based linkage learning used in RV-GOMEA-DG.

Non-decomposable problems

As the Rosenbrock and OSoREB problem contain non-decomposable sub-components, RV-GOMEA-FBLT and RV-GOMEA-FBMP generate different linkage models and their scalability should be evaluated independently. As shown in Section 2.6.2, a full linkage tree FOS is built in RV-GOMEA-FBLT to capture the dependence between single parameters in different sub-components, whereas a marginal product structure is used by RV-GOMEA-FBMP. The former results in slightly better scalability on OSoREB implying that there is indeed added value in bigger FOS elements that can capture the linkage over partially dependent (sub)-components. On the Rosenbrock problem, RV-GOMEA-FBLT and RF-GOMEA-FBMP show similar scalability but are outperformed by RV-GOMEA-UNI, implying that even though this problem contains dependencies, it can still be efficiently solved by a univariate linkage model.

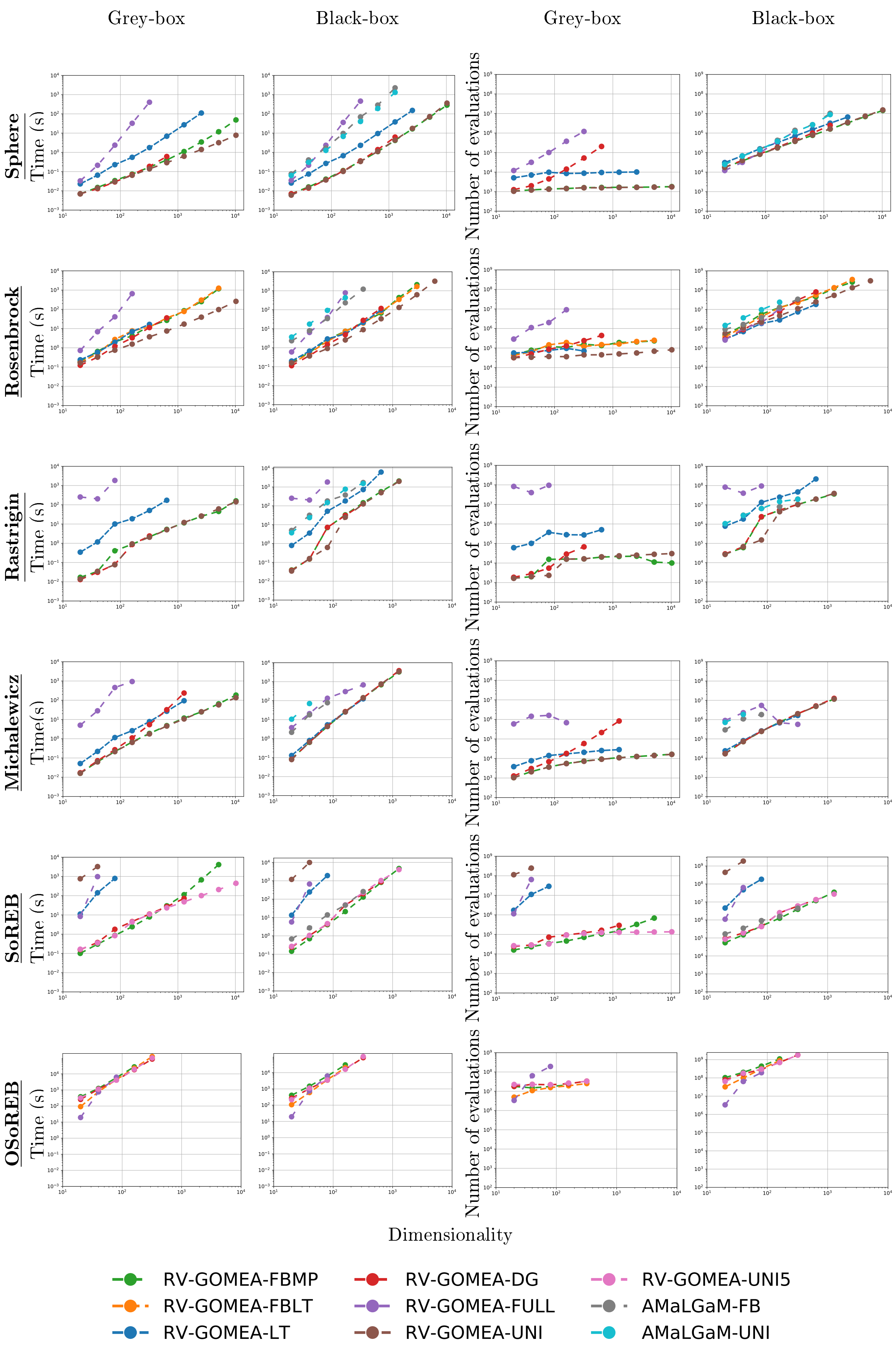


Figure 2.6:
Medians of scalability experiments with each data point being the median of 30 successful runs.

## 2.7. Discussion

The method introduces in this paper is able to learn pairwise dependencies between variables online. There are, however, certain issues left unaddressed in finding the optimal linkage structure to any optimization problem. One of the key questions left unanswered is how to deal with problems containing overlapping sub-components. The optimal linkage structure to solve these benchmark problems is unknown, nor do we know whether a universally optimal linkage structure for these kinds of problems exists for RV-GOMEA. Whilst two of our benchmark problems contain overlapping components, the results between our two proposed linkage models did not vary much and we did not manage to find a definitive optimal structure for these benchmark problems.

While our proposed approach defines relative dependencies on pairs of variables, it does not provide an absolute or relative minimal value to define non-separability of sub-components. More research is required to determine the possibility of finding a measurement that allows for the further decomposition of weakly dependent sub-components.

When focusing on the pairwise dependency checks done on the Rosenbrock problem, it occurs that whilst the results obtained from these checks are as expected, pairwise dependencies might not be suitable to model the higher order dependence of this non-decomposable problem. Even though two variables $x_i$ and $x_{i+2}$ are not pairwise dependent, their optimal values both depend on the value of $x_{i+1}$ and vice versa, making them dependent to some extent. These higher order dependencies cannot be captured by the pairwise dependency checks done by the approach introduced in this article.

Lastly, the order in which pairs are checked is completely random, a smarter approach might be to evaluate the pairs that are more likely to be dependent first, which is likely to improve RV-GOMEA-FBLT's performance even more on specific problems.

## 2.8. Conclusion

We have introduced a fitness-based linkage learning approach that can find pairwise dependencies between variables and build a linkage structure online without the need for any problem specific knowledge. Using the found dependencies this approach also sets a lower bound on the required population size for the learned problem structure. The proposed method has been evaluated on different well-known benchmark problems and has proven to be efficient in determining the correct pairwise dependencies between variables. Two different methods have been proposed based on estimated pairwise dependencies to model the linkage structure of a problem. These methods have been integrated into RV-GOMEA and AMaLGaM and resulted in both algorithms being able to learn well-performing dependencies online. RV-GOMEA-FBMP and RV-GOMEA-FBLT have shown to outperform the current state-of-the-art (for black-box scenarios) an EA know as AMaLGaM upon which RV-GOMEA was based, whilst leveraging the same linkage model. Because a comparison between RV-GOMEA and AMaLGaM has never before resulted in better performance of one algorithm on all benchmark problems, this is the first time we can conclude that RV-GOMEA has outperformed the model based EA that it was partially based on. Parts of RV-GOMEA and AMaLGaM are equivalent, but RV-GOMEA distinguishes itself from AMaLGaM by its optimal mixing, the better performance of RV-GOMEA on our benchmark problems thus clearly shows the added value of the optimal mixing employed by RV-GOMEA.

Whilst the two methods used for RV-GOMEA have equal performance on (partially) decomposable benchmarks, the generated linkage models differ for benchmark problems with overlapping sub-components. The difference lies in whether a (partial) linkage tree is built that captures all possible dependencies, or a marginal product linkage model is used. Whilst both methods have their strengths, RV-GOMEA-FBLT scales better on problems with strong dependencies and overlapping sub-components, thus proving to be a more robust method for large-scale real-world optimization problems with unknown problem structures.

Given the overall scalability of RV-GOMEA-FBLT, we conclude that the proposed algorithm is able to learn a linkage structure online and scale as well as the current state-of-the-art for grey-box optimization, RV-GOMEA when provided with the optimal structure.

# 3 Conclusion

## 3.1. Research questions

To conclude this thesis, the research objectives introduced in Section 1.2 will be revisited.

### 3.1.1. Online linkage learning

**Research question 1** How can we combine RV-GOMEA with an online linkage learning method that is able to correctly detect decomposability of (sub) components?

The fitness based linkage learning approach proposed in Section 2.4 has been shown to successfully detect decomposability as well as the linkage between pairwise problem variables. The heatmaps presented in Figure 2.2 clearly support this claim and show that for all benchmark problems the correct pairwise dependencies are found by this method. To combine this pairwise dependency learning approach with RV-GOMEA, a linkage model is created online that leverages the pairwise dependencies to create a linkage tree as was previously used by RV-GOMEA. More about this linkage model will be discussed in Section 3.1.3

### 3.1.2. Scalability

**Research question 2** How can we minimize the overhead of our online linkage learning method on the scalability of the optimization algorithm?

The overhead of an online linkage learning method based on fitness evaluations has shown to be a problem for the original differential grouping method when integrated into RV-GOMEA. The incremental dependency updating introduced in Section 2.4.2 aims to overcome this by initially assuming a completely decomposable linkage structure and ceasing to learn the rest of the linkage structure when the problem is suspected to be decomposable. The dependency checks are iteratively resumed after a certain number of generations but will be stopped again when no linkage is found. Whilst this approach reduces some of the overhead created by redundant dependency checks, it also introduces a slight bias towards (partially) decomposable problems.

### 3.1.3. Linkage structure

**Research question 3** How can we translate pairwise dependencies to a linkage structure that can capture the higher order dependency structure of an optimization problem?

The proposed fitness-based linkage learning method is able to correctly identify decomposability between (sub-)components. Section 2.4.3 shows two methods for pruning the linkage tree to remove redundant (sub-)components resulting in a more effective linkage structure and better mixing of variables during optimal mixing. The pruning of linkage model works two ways as small linkage sets are removed if bigger linkage sets are required to express the linkage of a problem and similarly, big linkage sets are removed if the smaller linkage present set in the linkage model would suffice to model the linkage of a problem. Two different pruning method have been compared against each other, whilst both models create the same linkage model for partially decomposable problems, they produce different linkage models for problems that contain overlapping sub-components. The marginal product model linkage structure produces better results on the Rosenbrock, but the partial linkage tree works better on the overlapping SoREB problem. Whilst both models have their strengths, their results are fairly similar and they are both able to create well-scaling linkage models that can be effectively used by RV-GOMEA.

## 3.2. Discussion

The research presented in this paper provides a baseline for fitness-based linkage learning in optimal mixing algorithms for the real-valued domain. For the non-overlapping benchmark problems, the optimal structure could be determined, the models produced by our algorithm have been verified using the known optimal models. For non-overlapping benchmark problems, however, the optimal structures are not known, nor do we know if a generalized approach exists to determine these structures. When comparing our two linkage learning approaches, the linkage tree performed better on OSoREB, whilst the marginal product model works slightly better on the Rosenbrock problem. Since these results do not give us a definitive answer as to which linkage structure works best on these problems, this thesis does not provide an optimal approach to learn the structure of a problem with overlapping sub-components.

Looking more closely at the pairwise dependencies found on the Rosenbrock problem, it can be noted that all pairs of variables that are not consecutive have a dependence of 0, meaning that when all variables are unmodified and the dependency check specified in Equation 2.5 is performed, there is no dependence found between $x_i$ and $x_{i+2}$. Whilst this is what we would expect for $x_i$ and $x_{i+2}$, they are both dependent on $x_{i+1}$ and optimizing either one will require optimization of $x_{i+1}$ as well. Some higher order of dependence does exist for $x_i$ and $x_{i+2}$ that can not be measured with our pairwise dependency checks. Finding such a measure would be a great addition to many real-world problems such as deformable image registration [18] but also to the circles in a square problem which is related to a wide variety of real-world applications [6] because many of these problems contain strong dependencies imposed through geometry, e.g., transformation vector field nodes that are near each-other have strong dependence, but those far apart are independent. Resulting in problem structures that are only non-decomposable to some extent. When using our fitness-based dependency check on these kinds of problems, nodes that do not have any direct connection will be marked as independent, even though they can be dependent on the same intermediate node, making them linked even though the dependency check suggests otherwise.

### 3.2.1. Future work

Our proposed approach defines relative dependencies on pairs of variables but does not provide an absolute or relative minimal value to define non-separability of sub-components. Interesting open issues, therefore, lie in researching a measure of minimal (absolute or relative) dependency strength to decompose weakly dependent (sub-)components. An interesting research question is thus how high the measured dependency between two variables should be to consider these variables dependent on one-another and when can we better model two variables as independent even when they have measurable dependence? The importance of this issue is highlighted in RV-GOMEA-FBLT's performance on the Rosebrock problem, compared to the univariate baseline of RV-GOMEA-UNI. Even though the Rosenbrock problem contains decomposable sub-components, a model capturing these dependencies is still outperformed by a univariate model. This raises the priority of the question of how to define the non-separability of sub-components even further.

A second, related, open issue is that of the overlapping components in optimization problems. Even when the pairwise dependencies between variables are known, no method exists to create an optimal linkage model for every kind of overlapping problem. When looking at the overlapping SoREB function defined in Section 2.6.3, we can verify whether the pairwise dependencies are correct, but it is impossible to verify the correctness of the linkage model created based on these dependencies, since it is not known what this model is supposed to look like. The question here is whether there exists a definitive approach to correctly model overlapping dependencies and if it does, what does it look like? More research into linkage structures is thus needed to determine whether such an optimal structure exists.

This fitness-based linkage learning method has shown to work well on the real-valued single objective problems. It has however not yet been applied to the multi-objective domain, where a lot of real-world applications of the current version of RV-GOMEA lie [17], [15]. To learn a linkage model based on fitness as has been done in this thesis is not as trivial in the multi-objective domain as it is for single objective optimization. Multiple objectives can often be calculated by multiple fitness functions and variables can be dependent on each other in one objective, but independent in a second objective. An open issue is thus how to correctly apply this fitness-based linkage learning in the multi-objective domain. In many real-world problems, dependencies are not fully dependent, neither fully independent. The introduced approach checks pairs of variables in random order. An interesting improvement to this randomized approach might be to make soft assumptions about where interesting dependencies might lie. Variables that have been marked as independent in the last $n$ checks will be more likely to be independent again than a pair of variables that have been dependent on all previous checks. Also, if variable $x_i$ is dependent with $x_{i+1}$ and $x_{i+2}$, it might be more interesting to check the pair $\{x_{i+1}, x_{i+2}\}$ than $\{x_{i+1}, x_{i+100}\}$. A possible improvement to the introduced approach is thus to find a smart ordering of pairs that can be beneficial to finding dependencies early on and will reduce the total number of evaluations needed to build the linkage model.

Lastly, one of the key aspects of RV-GOMEA-FBLT is its focus on high-dimensional real-valued problems. The improvements made in this thesis have been shown to work on well-established benchmark problems, but not on real-world problems. A clear next step would, therefore, be to also apply the new version of RV-GOMEA to real-world optimization problems.

### 3.2.2. No free lunch

As this thesis introduces an optimization algorithm that is aimed at solving a wide variety of optimization problems, it is important to discuss how this aim relates to the no-free-lunch-theorem [31]. This theorem states that the computational cost of finding the optimal solution averaged over all optimization problems, is the same for any optimization method, hence claiming that no algorithm offers a computational "short cut". That is, improvement of performance in problem-solving hinges on using prior information to match the method applied to problems at hand.

As our optimization method is specifically designed to be efficient in a certain class of problems, the no free lunch theorem does not apply to this research. However, in more specifically defining which class of problems our method is biased towards, we can get a better understanding of the strengths and limitations of our method. In this research, we have focused mainly on problems that are (partially) decomposable and problems where partial evaluations can be leveraged. One of the key strengths of the introduced method is the ability to exploit and extract dependency-based problem structures. Our method is less suitable for fully non-decomposable problems especially when the problem structure is known a priori, as no decomposition and partial evaluations will then be possible, disabling two of the big advantages of our method. Lastly, when the objective function of a problem is very computationally expensive, certain drawbacks to a fitness-based linkage learning approach become evident. For every pair of variables, at least one function evaluation is needed, and in our approach in every generation at most $l$ pairs of variables will be evaluated. This linear upper-bound on the number of evaluations required by our method makes it thus less suitable for high-dimensional problems with costly function evaluations.

## 3.3. Conclusion

In this thesis, a fitness-based linkage learning approach is introduced that is able to effectively exploit a linkage structure online, based on pairwise dependencies between problem variables. The proposed method has been evaluated on different well-known benchmark problems and has proven to be efficient in determining the correct pairwise dependencies between variables. The resulting linkage structures together with a found lower bound on the population size are combined with a state-of-the-art optimization algorithm RV-GOMEA and have proven excellent scalability on real-valued benchmark problems.

Whilst RV-GOMEA is partially based upon AMaLGaM, a strength of RV-GOMEA lies in the optimal mixing employed by RV-GOMEA that allows for efficient mixing of partially optimal solutions whilst leveraging partial evaluations. One fundamental difference is that AMaLGaM (and other model-based EDAs) does not allow for partial evaluations or overlapping linkage structures because the entire solution is sampled at once and only evaluated as a whole. The optimal mixing of RV-GOMEA has not been objectively compared with the model based EDA approach employed by AMaLGaM before, as both models advance under different linkage models. The linkage learning approach introduced in this thesis, however, suggested a marginal product model for RV-GOMEA that could be employed by

AMaLGaM as well, allowing, for the first time, a fair comparison between these two types of algorithms. From the scalability results, it could be concluded that RV-GOMEA outperforms AMaLGaM whilst using the exact same linkage learning techniques. These results imply that the optimal mixing employed by RV-GOMEA has measurable added value on the EA as a whole, resulting in a better performance than the EDA that forms the basis for the sampling of new solutions in RV-GOMEA but does not employ optimal mixing of variables.

Given the overall scalability of RV-GOMEA-FBLT, we conclude that the proposed algorithm is able to learn a linkage structure online and scale as well as the current state-of-the-art for GBO, RV-GOMEA when provided with the optimal structure.

# Bibliography

[1] Peter A. N. Bosman, Jörn Grahl, and Dirk Thierens. Enhancing the performance of maximum–likelihood gaussian EDAs using anticipated mean shift. *International Conference on Parallel Problem Solving from Nature*, pages 133–143, 01 2008.

[2] Peter A. N. Bosman Anton Bouter, Tanja Alderliesten. A novel model-based evolutionary algorithm for multi-objective deformable image registration with content mismatch and large deformations: benchmarking efficiency and quality, 2017.

[3] Peter A. N. Bosman, Jörn Grahl, and Dirk Thierens. Benchmarking parameter-free AMaLGaM on functions with and without noise. *Evolutionary Computation*, 21(3):445–469, 2013.

[4] Peter A.N. Bosman and Dirk Thierens. Linkage neighbors, optimal mixing and forced improvements in genetic algorithms. In *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation*, GECCO '12, pages 585–592, New York, NY, USA, 2012. ACM.

[5] Anton Bouter, Tanja Alderliesten, Cees Witteveen, and Peter A. N. Bosman. 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 '17, pages 705–712, New York, NY, USA, 2017. ACM.

[6] Ignacio Castillo, Frank Kampas, and Janos Pinter. Solving circle packing problems by global optimization: Numerical results and industrial applications. *European Journal of Operational Research*, 191:786–802, 12 2008.

[7] Charles Consel. Program adaptation based on program transformation. *ACM Computing Surveys*, 28, 02 2001.

[8] Roy de Bokx, Dirk Thierens, and Peter Bosman. In search of optimal linkage trees. *Proceedings of Genetic and Evolutionary Computation Conference*, 14, 01 2015.

[9] J Grahl, Stefan Minner, and Franz Rothlauf. Behaviour of umdac with truncation selection on monotonous functions. pages 2553 – 2559 Vol. 3, 10 2005.

[10] Ilan Gronau and Shlomo Moran. Optimal implementations of UPGMA and other common clustering algorithms. *Inf. Process. Lett.*, 104:205–210, 12 2007.

[11] Georges Harik, Fernando Lobo, and Kumara Sastry. *Linkage Learning via Probabilistic Modeling in the Extended Compact Genetic Algorithm (ECGA)*, volume 33, pages 39–61. 01 2007.

[12] John H. Holland. Genetic algorithms. *Scientific American*, 267(1):66–73, 1992.

[13] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 69:066138, 07 2004.

[14] Hoang Luong, Tanja Alderliesten, Arjan Bel, Yury Niatsetski, and Peter Bosman. Application and benchmarking of multi-objective evolutionary algorithms on high-dose-rate brachytherapy planning for prostate cancer treatment. *Swarm and Evolutionary Computation*, June 2018.

[15] Hoang Luong, Han La Poutré, and Peter Bosman. Exploiting linkage information and problem-specific knowledge in evolutionary distribution network expansion planning. *Evolutionary Computation*, 26(3):471–505, September 2018.

[16] Czesław Smutnicki Marcin Molga. Test functions for optimization needs. *Comput. Inform. Sci.*, pages 1–43, 01 2005.

[17] Stef Maree, Hoang Luong, Ernst Kooreman, Niek van Wieringen, Arjan Bel, Karel Hinnen, Henrike Westerveld, Bradley Pieters, Peter Bosman, and Tanja Alderliesten. Evaluation of bi-objective treatment planning for high-dose-rate prostate brachytherapy—a retrospective observer study. *Brachytherapy*, February 2019.

[18] Hemant K. Abhyankar Medha V. Wyawahare, Pradeep M. Patil. Image registration techniques: An overview. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, pages 11–39, 2009.

[19] Heinz Mühlenbein and Gerhard Paass. From recombination of genes to the estimation of distributions i. binary parameters. volume 1141, pages 178–187, 01 1996.

[20] Volker Nissen and Jorn Propach. On the robustness of population-based versus point-based optimization in the presence of noise. *Evolutionary Computation, IEEE Transactions on*, 2:107 – 119, 10 1998.

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

[22] Mohammmad Nabi Omidvar, Ming Yang, Yi Mei, Xiaodong Li, and Xin Yao. Dg2: A faster and more accurate differential grouping for large-scale black-box optimization. *IEEE Transactions on Evolutionary Computation*, 21(6):929–942, Dec 2017.

[23] Martin Pelikan and Heinz Mühlenbein. Marginal distributions in evolutionary algorithms. In *In Proceedings of the International Conference on Genetic Algorithms Mendel '98*, pages 90–95, 1999.

[24] Martin Pelikan, Kumara Sastry, and Erick Cantu-Paz. *Scalable optimization via probabilistic modeling. From algorithms to applications*, volume 33. 01 2006.

[25] Yun-Wei Shang and Yu-Huang Qiu. A note on the extended rosenbrock function. *Evolutionary Computation*, 14:119–126, 03 2006.

[26] Yi-En Su and Tian-Li Yu. Use model building on discretization algorithms for discrete EDAs to work on real-valued problems. pages 2491–2498, 07 2014.

[27] Dirk Thierens. Scalability problems of simple genetic algorithms. *Evolutionary computation*, 7: 331–52, 02 1999.

[28] Dirk Thierens. Linkage tree genetic algorithm: First results. In *Proceedings of the 12th Annual Conference Companion on Genetic and Evolutionary Computation*, GECCO '10, pages 1953–1958, New York, NY, USA, 2010. ACM.

[29] Dirk Thierens and Peter A.N. Bosman. Optimal mixing evolutionary algorithms. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, GECCO '11, pages 617–624, New York, NY, USA, 2011. ACM.

[30] Dirk Thierens and David Edward Goldberg. Mixing in genetic algorithms. *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 38–45, 1993.

[31] D. H. Wolpert and W. G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, April 1997.

[32] Tian-Li Yu, Kumara Sastry, David E. Goldberg, and Martin Pelikan. Population sizing for entropy-based model building in discrete estimation of distribution algorithms. In *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation*, GECCO '07, pages 601–608, New York, NY, USA, 2007. ACM.