On Gradients and Hybrid Evolutionary Algorithms for Real-Valued Multi-Objective Optimization

Peter A.N. Bosman

***Abstract*—Algorithms that make use of the gradient, i.e. the direction of maximum improvement, to search for the optimum of a single objective function have been around for decades. They are commonly accepted to be important and have been applied to tackle single-objective optimization problems in many fields. For multi-objective optimization problems, much less is known about the gradient and its algorithmic use. In this article we aim to contribute to the understanding of gradients for numerical, i.e. real-valued, multi-objective optimization. Specifically, we provide an analytical parametric description of the set of all non-dominated, i.e. most promising, directions in which a solution can be moved such that the objective values either improve or remain the same. This result completes previous work where this set is described only for one particular case, namely when some of the non-dominated directions have positive, i.e. non-improving, components and the final set can be computed by taking the subset of directions that are all non-positive. In addition we use our result to assess the utility of using gradient information for multi-objective optimization where the goal is to obtain a Pareto set of solutions that approximates the optimal Pareto set. To this end, we design and consider various multi-objective gradient-based optimization algorithms. One of these algorithms uses the description of the multi-objective gradient provided here. Also, we hybridize an existing multi-objective evolutionary algorithm (MOEA) with the various multi-objective gradient-based optimization algorithms. During optimization, the performance of the gradient-based optimization algorithms is monitored and the available computational resources are redistributed to allow the (currently) most effective algorithm to spend the most resources. We perform an experimental analysis using a few well-known benchmark problems to compare the performance of different optimization methods. The results underline that the use of a population of solutions that is characteristic of MOEAs is a powerful concept if the goal is to obtain a good Pareto set, i.e. instead of only a single solution. This makes it hard to increase convergence speed in the initial phase using gradient information to improve any single solution. However, in the longer run, the use of gradient information does ultimately allow for better fine-tuning of the results and thus better overall convergence.**

***Index Terms*—Gradient methods, multi-objective optimization, evolutionary algorithms, memetic algorithms, numerical optimization**

I. Introduction

In many fields of society, people are called upon to solve complex optimization problems. Because of the importance of optimization, it has been a prominent research topic for several decades. For a given optimization problem, it is a challenging task to design algorithms that can find the optimal solution or, alternatively, to find a solution of acceptable quality as fast as possible. One of the most commonly used and well-studied concepts in numerical optimization (i.e. optimization in real-valued search spaces) is the gradient. The gradient at any point in the design (or decision) space, i.e. a solution, indicates the direction in the design space along which the function to be optimized, improves the most. Hence, this direction can be used in an algorithm to find local optima of the function by iteratively moving a solution in a direction that is derived from gradient information. For a single objective function, many such algorithms exist, ranging from straightforward ones such as gradient descent (see, e.g. [1]) to more advanced ones such as conjugate gradients [2]. Use of methods like these is widely accepted. For instance, a substantial part of the methods studied in the field of machine learning is based on the principle of following the gradient of a performance function [3], [4]. Also, gradients continue to be an important topic in current and ongoing research in novel areas of optimization as is for instance the case in optimization under uncertainty, i.e. with stochastic objective functions; which is a setting that is typically encountered when dealing with complex simulations [5].

Multi-objective optimization [6], [7] is a particular type of optimization that naturally arises in many real-world situations. Multi-objective optimization differs from single-objective optimization in that a multiple of *objectives* are available that should be optimized simultaneously such that no expression of weights is available that allows the objectives to be combined in a single objective to be optimized. Typically, these multiple objectives are conflicting, which gives rise to a key characteristic of multi-objective optimization problems: the existence of sets of solutions, called Pareto sets, that cannot be ordered in terms of quality when only considering their objective function values. The goal is to find a diverse and representative subset of all optimal solutions instead of only a single one. For single-objective optimization, it is commonly accepted that the additional use of gradient information can be beneficial. For multi-objective optimization, this is less well established. One reason for this is that for a single objective, gradients are well understood, both in theory and in practice. The body of literature on theory and practice of gradients in multi-objective optimization is relatively limited, both in size and in scope. The majority of the literature focuses on finding a single direction of improvement. Here, we broaden the scope and take an in-depth look at the definition of the gradient for multi-objective optimization problems. Our results provide a more general picture through a fully analytical description of the gradient for multi-objective optimization. This is also the first and foremost goal of this article: to provide insight

Manuscript received October 1, 2009; revised February 4, 2010 and April 29, 2010. Date of publications October 17, 2011; date of current version January 31, 2012.

The author is with Centrum Wiskunde & Informatica (CWI), Amsterdam, The Netherlands (e-mail: Peter.Bosman@cwi.nl).

Digital Object Identifier 10.1109/TEVC.2010.2051445

into the structure of the multi-objective gradient. As will become clear, computing the multi-objective gradient is in itself a multi-objective optimization problem. An important contribution that we make in this article is that we provide an analytical description of all Pareto-optimal solutions to the latter problem, i.e. all Pareto-optimal improving directions. This result as well as the insights obtained from deriving the result may serve as a basis for better understanding the use of gradients in multi-objective optimization, like it is now understood for single-objective optimization.

Next to studying the theory and definition of gradients, their algorithmic use is equally important for optimization. Secondary goals in this article are therefore to design an algorithm to compute the multi-objective gradient and to design algorithms that use the the multi-objective gradient for optimization. A contribution toward this secondary goal is also provided in this article. The analytical derivation of the multi-objective gradient naturally transforms into an algorithm for its computation. We present this algorithm and analyze its computational complexity. Then, we design an optimization algorithm that uses the gradient by employing line search (i.e. find a local optimum along the gradient) repeatedly. To measure the quality of the performance of this optimization algorithm, a comparison should be made with existing state-of-the-art optimization algorithms. Evolutionary Algorithms (EAs) belong to the state-of-the-art in multi-objective optimization; its research field has seen explosive growth in recent years [6], [7]. The main reason for this is that the most commonly studied goal in multi-objective optimization is to find an approximation set of multiple Pareto-optimal solutions. The notion of searching a search space through maintaining a set of solutions is a key characteristic of EAs, which makes them natural candidates for multi-objective optimization. EAs for multi-objective optimization are commonly called MOEAs (multi-objective Evolutionary Algorithms).

Although EAs are a powerful optimization methodology and an active area of research, hybridization of EAs is commonly accepted to be important when tackling real-world problems with EAs [8], [9]. An EA is called hybrid if a (usually single-solution based) local optimization algorithm is integrated in its procedure. Hybrid EAs are even often referred to under a specific name: Memetic Algorithms [9], [10], [11]. Given this premise, the design of a hybrid EA that integrates gradient-based optimization algorithms for multi-objective optimization is an interesting topic. However, while some problems may benefit from the use of gradient information, in other problems the additional cost required to calculate gradients is expected not to weigh up to the benefits. This issue can be taken into account simultaneously with the determination of the utility of gradient-based optimization algorithms by varying the probability of applying the algorithm during optimization. Adaptively choosing the probability of using local search algorithms has several advantages in addition to the potential improvements in efficiency. First, a practitioner wishing to apply an algorithm that uses multiple local optimization algorithms is relieved of the need to select and tune the different probabilities. Furthermore, by leaving the choice of the probabilities to the overall algorithm, a large part of the optimization task is automated. Also, automatically adapting the probabilities during optimization can render optimization algorithms more robust, as unfavorable choices of parameters may still be corrected during the course of optimization. Finally, in our case, it provides insight into the utility of the gradient-based optimization algorithms because if the probability of applying local optimization is consistently reduced to 0 there is no indication that the local optimization algorithm contributes significantly to the overall optimization process. We therefore employ such an adaptive hybridization scheme in this article in addition to the different non-hybrid optimization algorithms.

The remainder of this article is organized as follows. In Section II we first discuss related literature to position the work presented in this article. Then, in Section III we provide the basic definitions and notation used throughout the article. Gradients for the single-objective case are briefly recalled in IV after which we present our derivations of the gradient and an algorithm for its computation in the multi-objective case in Section V. In Section VI we then run experiments. Specifically, random-restart optimization algorithms that employ gradient information to optimize a single point are presented in Section VI-B. An adaptive hybridization of a MOEA with these techniques is experimentally investigated in Section VI-C. We close this article with a discussion of our results in Section VII and a presentation of our conclusions in Section VIII.

II. Related work

A. Gradient information indirectly based on the objectives

Gradient information for multi-objective optimization is used explicitly by Lahanas, Baltas and Giannouli [12]. However, they use weighted aggregation to construct a single objective function which is subsequently optimized. Hence, there is no guarantee that all objectives are optimized simultaneously. Instead of analytically deriving improving directions, Goh, Ong and Tan recently tried evolving such directions instead [13]. A third approach to exploiting gradient information without directly considering the objective functions is by Emmerich, Deutz and Beume [14]. In their approach they follow the gradient of a one-dimensional metric of the Pareto-front. Specifically, the well-known S-metric by Zitzler and Thiele [15] is used. Although the both of the latter methods were reported to be beneficial when used to hybridize MOEAs, they do not provide any additional insight into the structure of the multi-objective gradient as they do not consider the actual objective functions.

B. Gradient information directly based on the objectives

Results on gradient information that is derived by considering the actual objective functions while ensuring all objectives are improved upon simultaneously, also exist in the literature. Fliege and Svaiter provide an analytic description of a direction that has the specific property that it is the largest direction of simultaneous improvement [16]. This direction is referred to by the authors as the multi-objective gradient. A similar derivation of a single direction of descent is originally given by Mukai [17] and additionally by Schäffler, Schultz and Weinzierl [18]. If the objectives have different ranges, the largest direction of simultaneous descent will be biased towards the objective with the largest range. Harada, Sakuma and Kobayashi define the multi-objective gradient as a single direction in a similar way [19]. Their derivations however additional allow to consider constraints on the problem variables at the same time. Using linear programming techniques they are able to find a Pareto descent direction for solutions inside feasible regions.

It should be pointed out that, especially if the number of objectives isn't large, computing a descent direction following the definitions of e.g. Fliege and Svaiter, isn't very difficult. For two objectives, it boils down to solving a quadratic equation in a single variable. However, even if the objectives are first similarly scaled, there are, as we shall show, still multiple (typically infinitely many) directions of improvement that do not dominate each other (e.g. improving objective 0 and leaving objective 1 unchanged versus improving objective 1 and leaving objective 0 unchanged). Our results analytically describe all of these directions and are therefore more general than the results listed above. It should however be noted that, for the eventual use of our analytical results, a single, suitable, direction still needs to be chosen from this set of directions. In particular it should be noted that when choosing a random direction, convergence toward the optimal Pareto front can not be guaranteed, in contrast to the above listed methods.

Another way of using gradient information is to find all solutions of similar quality, i.e. search for solutions along the Pareto front. This can for instance be done using continuation methods such as predictor-corrector methods, see e.g. [20]. In the case of multi-objective optimization, given a point on the Pareto front, a prediction is made using gradient information of the individual objectives as to where another solution lies on the Pareto front. Typically, that point is then off the manifold that is the Pareto front and it is pushed back onto the Pareto front by solving a single-objective optimization problem. Such an approach has been taken by Hillermeier [21] and by Schütze, Dell'Aere and Dellnitz [22]. Although this type of gradient exploitation can be highly useful, it does not focus on the actual definition of the multi-objective gradient, which is what we are interested in studying more closely here.

C. Hybrid MOEAs

One of the best known publications regarding real-valued multi-objective memetic algorithms (or hybrid MOEAs) is the M-PAES [23]. However, the M-PAES does not explicitly make use of gradient information. The analytic description by Fliege and Svaiter mentioned earlier is used by Brown and Smith in a hybrid EA [24]. The result of Schäffler, Schultz and Weinzierl as mentioned earlier was used in a multilevel subdivision technique that subdivides the search space to perform local search in each subspace based by Dellnitz, Schütze and Hestermeyer [25]. The result of Schäffler, Schultz and Weinzierl was also later used to hybridize MOEAs by Shukla [26]. Continuation methods are studied as a hybridization of MOEAs by Harada *et al.* [27] and by Schütze *et al.* [28]. Finally, a hill climbing method named HCS (Hill Climber with Sidestep) was proposed by Lara *et al.* [29] that uses gradient information and can be used to realize movement both toward and along the Pareto set. In the same work, the HCS was combined with the well-known MOEA SPEA2.

All of the memetic approaches mentioned above hybridize a MOEA in a non-adaptive way. In other words, if the gradient method is not useful for a specific optimization problem, resources will still be spent on trying out the method. In this article, we therefore use an adaptive method that is specialized for use in multi-objective optimization [30]. We further point out that most hybrid MOEAs are only tested on test problems that have nice gradient properties, which are not expected to be good practical test-cases. Here we use both test problems that have nice gradient properties as well as a well-known set of benchmark problems that have a higher dimensionality and vary in difficulty [30], [31]. This will allow us to better asses the true added value of exploiting gradient information compared to the use of MOEAs.

D. New contributions

The most important difference between the existing literature and the work presented here is that we analytically describe the *complete set* of non-dominated simultaneously improving directions and thereby obtain insight into the geometric structure of the multi-objective gradient. Hence, we consider the multi-objective gradient to be a set of directions (specifically a $m-1$ dimensional manifold in a m-dimensional space where m is the number of objectives).

In previous work [32], we gave a description of this set under the assumption that the set of all Pareto-optimal directions that improve at least one objective is larger and the final result can therefore be computed by taking the subset of all Pareto-optimal directions that improve all objectives. This article completes this previous work by considering the case in which the set of all Pareto-optimal directions that improve at least one objective consists of only directions that improve all objectives, i.e. it equals the desired result. Moreover, in this article we provide the full picture by unifying the two cases.

III. Definitions

A. Notation

In the case of single-objective optimization, we write the function to be optimized as f. Function f returns, given a vector $\boldsymbol{x}$ of l real values, a single real value i.e. $\boldsymbol{x} = (x_0, x_1, \ldots, x_{l-1})$, $\boldsymbol{x} \in \mathbb{R}^l$, $f(\boldsymbol{x}) \in \mathbb{R}$.

In the case of multi-objective optimization, we assume to have m real-valued objective functions. We denote these objective functions by $f_i(\boldsymbol{x})$ where $i \in \{0, 1, \ldots, m-1\}$. We write the function to be optimized as $\boldsymbol{f}$. Vector function $\boldsymbol{f}$ returns, given a vector $\boldsymbol{x}$ of l real values, a vector of real values i.e. $\boldsymbol{f}(\boldsymbol{x}) = (f_0(\boldsymbol{x}), f_1(\boldsymbol{x}), \ldots, f_{m-1}(\boldsymbol{x})) \in \mathbb{R}^m$.

A solution $\boldsymbol{x}^0$ is said to (Pareto) *dominate* a solution $\boldsymbol{x}^1$ (denoted $\boldsymbol{x}^0 \succ \boldsymbol{x}^1$) if and only if $f_i(\boldsymbol{x}^0) \leq f_i(\boldsymbol{x}^1)$ holds for all $i \in \{0, 1, \ldots, m-1\}$ and $f_i(\boldsymbol{x}^0) < f_i(\boldsymbol{x}^1)$ holds for at least one $i \in \{0, 1, \ldots, m-1\}$. A *Pareto set* of size n then is a set of solutions $\boldsymbol{x}^j$, $j \in \{0, 1, \ldots, n-1\}$ for which no solution dominates any other solution, i.e. there are no $j, k \in \{0, 1, \ldots, n-1\}$ such that $\boldsymbol{x}^j \succ \boldsymbol{x}^k$ holds. A *Pareto front* corresponding to a Pareto set is the set of all m-dimensional objective function values corresponding to the solutions, i.e. the set of all $\boldsymbol{f}(\boldsymbol{x}^j)$, $j \in \{0, 1, \ldots, n-1\}$.

A solution $\boldsymbol{x}^0$ is said to be *Pareto optimal* if and only if there is no other $\boldsymbol{x}^1$ such that $\boldsymbol{x}^1 \succ \boldsymbol{x}^0$ holds. Further, the *optimal Pareto set* is the set of all Pareto-optimal solutions and the *optimal Pareto front* is the Pareto front that corresponds to the optimal Pareto set. We denote the optimal Pareto set by $\boldsymbol{\mathcal{P}_S}$ and the optimal Pareto front by $\boldsymbol{\mathcal{P}_F}$.

B. Goal

Without loss of generality, we assume that the goal is to *minimize* the objectives. Ultimately then, we are interested in finding a direction $\boldsymbol{u}$ along which to move a given point $\boldsymbol{x}$, starting from that point, i.e. $\boldsymbol{x} \leftarrow \boldsymbol{x} + \delta\boldsymbol{u}$. The specific direction that we want to find is the one in which the rate of change in the objective function(s), starting from a given point $\boldsymbol{x}$, is as negative as possible. We call the direction that we are looking for the Direction Of Interest (DOI) and denote it $\hat{\boldsymbol{u}}^{\text{DOI}}(\boldsymbol{x})$. Directions do not have lengths, so we will use unit vectors, i.e. vectors of unit lengths: $\hat{\boldsymbol{u}} \in \mathbb{R}^l$, $\|\hat{\boldsymbol{u}}\| = 1$.

IV. Single-objective case

A. Derivation

The rate of change in f in a direction $\hat{\boldsymbol{u}}$ is commonly called the directional derivative and is denoted $\nabla_{\hat{\boldsymbol{u}}}$:

$$\nabla_{\hat{\boldsymbol{u}}} f(\boldsymbol{x}) = \lim_{h \to 0} \left\{ \frac{f(\boldsymbol{x} + h\hat{\boldsymbol{u}}) - f(\boldsymbol{x})}{h} \right\} \quad (1)$$

To find the DOI, an optimization problem over unit vectors $\hat{\boldsymbol{u}} \in \mathbb{R}^l$ must be solved:

$$\hat{\boldsymbol{u}}^{\text{DOI}}(\boldsymbol{x}) = \arg\min_{\hat{\boldsymbol{u}}} \{\nabla_{\hat{\boldsymbol{u}}} f(\boldsymbol{x})\} \quad (2)$$

It can be shown [33] that the directional derivative is

$$\nabla_{\hat{\boldsymbol{u}}} f(\boldsymbol{x}) = (\nabla f(\boldsymbol{x}))^T \hat{\boldsymbol{u}} \quad (3)$$

where $\nabla f(\boldsymbol{x})$ is the gradient of f at point $\boldsymbol{x}$. Recall that the gradient of f at any point $\boldsymbol{x}$ is a vector of all l partial derivatives $\partial f(\boldsymbol{x})/\partial x_i$, $i \in \{0, 1, \ldots, l-1\}$ of f at point $\boldsymbol{x}$:

$$\nabla f(\boldsymbol{x}) = \left(\frac{\partial f(\boldsymbol{x})}{\partial x_0}, \frac{\partial f(\boldsymbol{x})}{\partial x_1}, \ldots, \frac{\partial f(\boldsymbol{x})}{\partial x_{l-1}} \right) \quad (4)$$

Recall further that the partial derivative $\partial f(\boldsymbol{x})/\partial x_i$ of f with respect to a single variable x_i is the rate of change in f, starting from point $\boldsymbol{x}$, when only x_i is varied, i.e. :

$$\frac{\partial f(\boldsymbol{x})}{\partial x_i} = \lim_{h \to 0} \left\{ \frac{f(\ldots, x_i + h, \ldots) - f(\ldots, x_i, \ldots)}{h} \right\} \quad (5)$$

Using the equation for computing the angle θ between two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ (i.e. $\cos(\theta) = (\boldsymbol{a}^T\boldsymbol{b})/(\|\boldsymbol{a}\|\|\boldsymbol{b}\|)$) the DOI can be found from Equation 3:

$$\hat{\boldsymbol{u}}^{\text{DOI}}(\boldsymbol{x}) = -\frac{\nabla f(\boldsymbol{x})}{\|\nabla f(\boldsymbol{x})\|} \quad (6)$$

The result in Equation 6, i.e. the fact that the direction of maximum increase in a function is given by the gradient of that function and, due to symmetry, the direction of maximum decrease is given by the negative gradient, is quite commonly known.

B. Computation

Computing $\hat{\boldsymbol{u}}^{\text{DOI}}(\boldsymbol{x})$ is rather straightforward in the single-objective case. It requires an elementary normalization operation of the gradient to compute the final result in Equation 6. To compute the gradient itself, the partial derivatives need to be computed. To this end, a fixed value for h is often taken for which Equation 5 is evaluated. This is commonly known as the finite difference approximation. Computing the DOI this way requires $l + 1$ evaluations and has a computational complexity of $\Theta(lF)$ where F is the complexity of a single evaluation. Choosing the best value for h is not trivial. If h is too large, the approximation can be bad because the actual partial derivatives of f may change a lot inside the interval indicated by h, causing $f(\ldots, x_i + h, \ldots)$ to be very different from $f(\ldots, x_i, \ldots) + h\frac{\partial f(\boldsymbol{x})}{\partial x_i}$, i.e. the value that is obtained if the partial derivatives remain constant. If h is too small, numerical instabilities can occur.

Evaluating the gradient using Equation 5 directly with a small value for h is also known as the forward difference method. Alternatively, backward differences can be used in which the approximation is computed by subtracting h rather than adding it to each parameter. Also, central differences can be used by looking at the interval of length h with $\boldsymbol{x}$ as the central point, i.e. using $f(\ldots, x_i + \frac{1}{2}h, \ldots) - f(\ldots, x_i - \frac{1}{2}h, \ldots)$ as the numerator. Depending on the size of h and the smoothness of the objective function, these methods can give different results, both in terms of the actual gradient computed and in terms of numerical stability.

In some domains gradients can be obtained with little extra cost compared to performing a function evaluation. Such is the case for instance using so-called adjoint methods [34] in the case that objective functions are computed using partial differential equation (PDE) solvers (e.g. in computational fluid dynamics). Finally, if formulations of the objectives are known, it may be feasible to compute the gradient analytically. This is preferable because it may greatly reduce the computational burden as no additional evaluations are required. It further typically reduces the risk of numerical instability substantially.

C. Gradient-based optimization algorithms

Many classical optimization algorithms exist that use the gradient to find a local minimum of a single-objective problem [35]. The most commonly known ones are variations of gradient descent. Gradient descent is an iterative approach that alters a point by moving it a short distance in the direction of the gradient. Using line minimization, the distance that is moved in the direction of the steepest descent, takes the search to a point at which the gradient in that direction is 0. A commonly adopted approach to perform line minimization is Brent's method [36]. In Brent's method a bracket is used of three points where the outer points have a larger function value than the interior point. A one-dimensional parabola is fit to the bracket. A minimum along the search direction is found by iteratively finding the minimum of the parabola and using it to update the bracket accordingly.

Following the direction of steepest descent in each subsequent line minimization is in general not optimal. The reason for this is that each subsequent search direction is orthogonal to the previous one. This can cause the search to oscillate around the optimal direction towards the optimum. The conjugate-gradients algorithm [2] overcomes this problem. In this algorithm, each subsequent search direction is conjugate with the previous one. This means that the new direction is chosen so that the component of the gradient in the direction of the previous line minimization remains zero along the new direction, resulting in much more efficient local optimization. Specifically, any quadratic optimization function of dimensionality l can be minimized using at most l line minimizations.

V. Multi-objective case

In this Section we will illustrate the derivations and algorithms using a generalization of the MED (Multiple Euclidean Distances) problems [19]. We refer to this adaptation as GenMED. GenMED is a class of problems parameterized by a parameter d. The i-th objective of GenMED is

$$f_i(\boldsymbol{x}, d) = \left(\frac{\|\boldsymbol{x} - \boldsymbol{c}^i\|}{\sqrt{2}}\right)^d = \left\|\frac{1}{2}\left(\boldsymbol{x} - \boldsymbol{c}^i\right)\right\|^d \tag{7}$$

where $\boldsymbol{c}^i$ is a l-dimensional unit vector for which $(\boldsymbol{c}^i)_i = 1$ and $(\boldsymbol{c}^i)_j = 0$, $j \neq i$. For $d > 1$ the optimal Pareto front is convex whereas for $d < 1$ the optimal Pareto front is concave. Typical values are $d = 2$ for the convex case and $d = 1/2$ for the concave case. In two dimensions, for any value of d, the optimal Pareto front is given by $(t^d, (1-t)^d)$ for $t \in [0, 1]$. GenMED is a smooth function with equally scaled objectives and without any locally optimal Pareto fronts.

A. Derivation

In the multi-objective case, we define the directional derivative in direction $\hat{\boldsymbol{u}}$ at point $\boldsymbol{x}$ as the vector of real values that indicates the change in each objective separately. In other words, $\nabla_{\hat{\boldsymbol{u}}}\boldsymbol{f}(\boldsymbol{x})$ is a m-dimensional vector of the directional derivatives of the individual objectives:

$$\nabla_{\hat{\boldsymbol{u}}}\boldsymbol{f}(\boldsymbol{x}) = (\nabla_{\hat{\boldsymbol{u}}}f_0(\boldsymbol{x}), \nabla_{\hat{\boldsymbol{u}}}f_1(\boldsymbol{x}), \ldots, \nabla_{\hat{\boldsymbol{u}}}f_{l-1}(\boldsymbol{x})) \tag{8}$$

The definition of the DOI in the multi-objective case is quite similar to the definition in the single-objective case. The only difference is that in Equation 2 f is replaced with $\boldsymbol{f}$, i.e. :

$$\hat{\boldsymbol{u}}^{\text{DOI}}(\boldsymbol{x}) = \arg\min_{\hat{\boldsymbol{u}}} \{\nabla_{\hat{\boldsymbol{u}}}\boldsymbol{f}(\boldsymbol{x})\} \tag{9}$$

Equation 9 now however is a multi-objective optimization problem. This means that in general, there will be more than one Pareto-optimal DOI. In the remainder of this Subsection, we derive equations that allow us to describe the complete set of DOI. In the next Subsection we summarize how these equations can be used to actually compute the DOI.

The derivations below are structured as follows. First, we list the assumptions that we make for the derivations to hold. We then start by showing that the set of all multi-objective directional derivatives (i.e. Equation 8 for all possible unit directions $\hat{\boldsymbol{u}}$) form the surface and interior of an m-dimensional hyperellipsoid. The optimal solution to Equation 9 is given by all directions that map to a multi-objective directional derivative that is all negative and non-dominated. This implies that the optimal solution is part of the surface of the hyperellipsoid. To obtain an analytical description of these directions, we first use the non-domination criterion to find the non-dominated part of the surface of the hyperellipsoid. We then intersect this set with the negative hypercube to keep only the directions that correspond to improvement (i.e. all-negative directions). The main derivations are illustrated in Figure 3.

1) Assumptions: We assume to have at least as many problem variables as objectives, i.e. $l \geq m$. In practice, this is only a minor restriction as in most cases the number of problem variables is typically large.

We further assume that the individual objectives are differentiable and that their gradients at $\boldsymbol{x}$ are linearly independent. If they are linearly dependent, this means, following Equation 9, that some of the objectives in the particular multi-objective optimization problem that we have to solve are linearly dependent. In that case, at least one of the objectives is redundant [37] and the computations can be done with a subset of the objectives, or rather the gradients, that *are* linearly independent without affecting the optimality of the final outcome.

2) Pareto-optimal hyperellipsoid surface: The set of all directions $\hat{\boldsymbol{u}}$ is the set of all l-dimensional unit vectors. Therefore, they form the surface of a unit hypersphere in l dimensions, centered at $(0, 0, \ldots, 0)$. The directional derivative $\nabla_{\hat{\boldsymbol{u}}}\boldsymbol{f}(\boldsymbol{x})$ in Equation 8 maps this l-dimensional unit hypersphere into an m-dimensional space by means of a linear transformation. This transformation can be written in matrix notation by defining an $m \times l$ matrix $\boldsymbol{G}$. This matrix contains the gradients of the objective functions in its rows, i.e. :

$$\boldsymbol{G} = \begin{bmatrix} (\nabla f_0(\boldsymbol{x}))^T \\ (\nabla f_1(\boldsymbol{x}))^T \\ \vdots \\ (\nabla f_{m-1}(\boldsymbol{x}))^T \end{bmatrix} = \begin{bmatrix} \frac{\partial f_0(\boldsymbol{x})}{\partial x_0} & \frac{\partial f_0(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial f_0(\boldsymbol{x})}{\partial x_{l-1}} \\ \frac{\partial f_1(\boldsymbol{x})}{\partial x_0} & \frac{\partial f_1(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial f_1(\boldsymbol{x})}{\partial x_{l-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_{m-1}(\boldsymbol{x})}{\partial x_0} & \frac{\partial f_{m-1}(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial f_{m-1}(\boldsymbol{x})}{\partial x_{l-1}} \end{bmatrix} \tag{10}$$

Equation 8 can now be written as:

$$\nabla_{\hat{\boldsymbol{u}}}\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{G}\hat{\boldsymbol{u}} \tag{11}$$

In the following, by the exterior, or the surface, of a convex shape we mean the convex shape itself. By the interior we mean all points that lie inside this convex shape, i.e. all points that can be obtained by taking an arbitrary convex combination of exterior points. For the hypersphere in l dimensions for instance this means that the exterior is defined by all $\boldsymbol{u}$ such that $\|\boldsymbol{u}\|= 1$, i.e. by all $\hat{\boldsymbol{u}}$, and the interior is defined by all $\boldsymbol{u}$ such that $\|\boldsymbol{u}\|< 1$.

The linear transformation $\boldsymbol{G}\hat{\boldsymbol{u}}$ transforms the surface of the l-dimensional hypersphere into the surface and interior of an m-dimensional hyperellipsoid, centered at $(0, 0, \ldots, 0)$. To see why this is the case, consider the Singular Value Decomposition (SVD) of matrix $\boldsymbol{G}$. The SVD exists for any matrix [38] and decomposes the matrix into three matrices. Using SVD we may write $\boldsymbol{G} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T$ where $\boldsymbol{U}$ has dimensions $m\times m$, $\boldsymbol{D}$ has dimensions $m \times l$ and matrix $\boldsymbol{V}^T$ has dimensions $l \times l$. Moreover, matrices $\boldsymbol{U}$ and $\boldsymbol{V}^T$ are orthonormal and matrix $\boldsymbol{D}$ is diagonal. Now, any orthonormal matrix is a combination of rotations and reflections [39]. Hence, $\boldsymbol{V}^T\hat{\boldsymbol{u}}$ is still the surface of an l-dimensional hypersphere. Since $\boldsymbol{D}$ is diagonal, it can be written as $\boldsymbol{D} = \boldsymbol{S}\boldsymbol{P}$ where $\boldsymbol{S}$ has dimensions $m \times m$ and $\boldsymbol{P}$ has dimensions $m \times l$ and both are diagonal, specifically $\boldsymbol{S}_{ii} = \boldsymbol{D}_{ii}$ and $\boldsymbol{P}_{ii} = 1$. Matrix $\boldsymbol{P}$ is a projection matrix that drops all components $j \geq m$ from a l-dimensional vector. It therefore collapses the l-dimensional hypersphere onto the exterior, and interior of an m-dimensional hypersphere. Multiplication with matrix $\boldsymbol{S}$ scales the axes independently. Hence, $\boldsymbol{D}\boldsymbol{V}^T\hat{\boldsymbol{u}} = \boldsymbol{S}\boldsymbol{P}\boldsymbol{V}^T\hat{\boldsymbol{u}}$ is the surface and interior of a hyperellipsoid that is aligned with all major axes. The final multiplication with matrix $\boldsymbol{U}$, i.e. $\boldsymbol{G}\hat{\boldsymbol{u}} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^T\hat{\boldsymbol{u}}$, finally unalignes the hyperellipsoid with the main axes through rotations and reflections.

We are interested in the non-dominated part of the hyperellipsoid that is made up of the directional derivatives. There are m extreme directional derivatives of interest to this non-dominated part. These extreme points are minimal in one of the objectives, i.e. one of the m one-dimensional directional derivatives is minimal. To find the unit directions that correspond to these extreme directional derivatives, the following minimization problem must be solved for each $i \in \{0, 1, \ldots, m-1\}$:

$$\arg\min_{\hat{\boldsymbol{u}}} \{\nabla_{\hat{\boldsymbol{u}}} f_i(\boldsymbol{x})\} \tag{12}$$

Because Equation 12 is in fact similar to Equation 2, we can use the result in Equation 6 to find that the set of unit vectors $\hat{\boldsymbol{u}}^{\text{Extr-elli},i}, i \in \{0, 1, \ldots, m-1\}$ that, using the directional derivative, map to the negative extrema of the hyperellipsoid are given by:

$$\hat{\boldsymbol{u}}^{\text{Extr-elli},i} = -\frac{\nabla f_i(\boldsymbol{x})}{\|\nabla f_i(\boldsymbol{x})\|} \tag{13}$$

To find the set of directions that map to entire surface of the hyperellipsoid (i.e. not the interior), take another look at the decomposition $\boldsymbol{G} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{P}\boldsymbol{V}^T$. Multiplication with $\boldsymbol{U}\boldsymbol{S}$ only scales and subsequently rotates and reflects the m-dimensional hypersphere. Hence, directions $\hat{\boldsymbol{u}}$ that map to the surface of the m-dimensional hyperellipsoid must already map to the surface of the m-dimensional hypersphere after multiplication with $\boldsymbol{P}\boldsymbol{V}^T$. In other words, $\hat{\boldsymbol{u}}$ maps to the hyperellipsoid surface if and only if $\|\boldsymbol{P}\boldsymbol{V}^T\hat{\boldsymbol{u}}\|= 1$ holds. Because $\boldsymbol{V}^T$ is orthonormal, $\|\boldsymbol{V}^T\hat{\boldsymbol{u}}\|= 1$ automatically holds for any $\hat{\boldsymbol{u}}$. Now, since $\boldsymbol{P}$ is a simple projection from the l-dimensional space to the m-dimensional space that drops all components $j \geq m$ from a vector, we have:

$$\|\boldsymbol{P}\boldsymbol{V}^T\hat{\boldsymbol{u}}\|= 1 \Leftrightarrow \left(\boldsymbol{V}^T\hat{\boldsymbol{u}}\right)_{(m,m+1,\ldots,l-1)} = (0, 0, \ldots, 0) \tag{14}$$

Now we take linear combinations of the negative extrema $\hat{\boldsymbol{u}}^{\text{Extr-elli},i}$. To ensure they are again unit vectors, we normalize them. We denote the resulting set $\mathbb{U}^{\text{Elli}}$:

$$\mathbb{U}^{\text{Elli}} = \left\{ \frac{\sum_{i=0}^{m-1} a_i\hat{\boldsymbol{u}}^{\text{Extr-elli},i}}{\|\sum_{i=0}^{m-1} a_i\hat{\boldsymbol{u}}^{\text{Extr-elli},i}\|} \;\middle|\; a_i \in \mathbb{R} \right\} \tag{15}$$

From the definition of $\boldsymbol{P}$ and Equation 15 it follows for any $\hat{\boldsymbol{u}}^{\text{Elli}} \in \mathbb{U}^{\text{Elli}}$ that:

$$\|\boldsymbol{P}\boldsymbol{V}^T\hat{\boldsymbol{u}}^{\text{Elli}}\|=\|\left(\boldsymbol{V}^T\hat{\boldsymbol{u}}^{\text{Elli}}\right)_{(0,1,\ldots,m-1)}\|= \tag{16}$$
$$\left\|\frac{\sum_{i=0}^{m-1} a_i\left(\boldsymbol{V}^T\hat{\boldsymbol{u}}^{\text{Extr-elli},i}\right)_{(0,1,\ldots,m-1)}}{\|\sum_{i=0}^{m-1} a_i\hat{\boldsymbol{u}}^{\text{Extr-elli},i}\|}\right\|$$

We know for certain that $\|\boldsymbol{P}\boldsymbol{V}^T\hat{\boldsymbol{u}}^{\text{Extr-elli},i}\| = 1$ for all $i \in \{0, 1, \ldots, m-1\}$ because those are the directions that map to the extreme points of the hyperellipsoid and thus lie on the surface. Equation 14 tells us that for these directions, the components $j > m$ are all zero after multiplication with $\boldsymbol{V}^T$. Hence, for the computation of the length of the vector in Equation 16 we can drop the vector cropping and replace $(\boldsymbol{V}^T\hat{\boldsymbol{u}}^{\text{Extr-elli},i})_{(0,1,\ldots,m-1)}$ with $\boldsymbol{V}^T\hat{\boldsymbol{u}}^{\text{Extr-elli},i}$, i.e. :

$$\|\boldsymbol{P}\boldsymbol{V}^T\hat{\boldsymbol{u}}^{\text{Elli}}\|= \left\|\boldsymbol{V}^T\frac{\sum_{i=0}^{m-1} a_i\hat{\boldsymbol{u}}^{\text{Extr-elli},i}}{\|\sum_{i=0}^{m-1} a_i\hat{\boldsymbol{u}}^{\text{Extr-elli},i}\|}\right\| \tag{17}$$

Since $\boldsymbol{V}^T$ is an orthonormal matrix, it doesn't change vector lengths and hence it can be dropped from the righthandside of Equation 17. Clearly, the righthandside then evaluates to 1, giving $\|\boldsymbol{P}\boldsymbol{V}^T\hat{\boldsymbol{u}}^{\text{Elli}}\|= 1$. And thus, as noted earlier, this means that $\hat{\boldsymbol{u}}^{\text{Elli}}$ maps to the hyperellipsoid surface.

Each direction in $\mathbb{U}^{\text{Elli}}$ thus maps to the hyperellipsoid surface. To be sure that the entire hyperellipsoid surface is reached, for any m-dimensional unit vector $\hat{\boldsymbol{v}}$ there must be at least one direction $\hat{\boldsymbol{u}}$ in $\mathbb{U}^{\text{Elli}}$ for which $\boldsymbol{G}\hat{\boldsymbol{u}}$ points in the same direction as $\hat{\boldsymbol{v}}$. To see that this is indeed the case, we first define an $m \times l$ matrix $\boldsymbol{U}^{\text{Extr-elli}}$. This matrix contains the $\hat{\boldsymbol{u}}^{\text{Extr-elli},i}$ in its rows, i.e. :

$$\boldsymbol{U}^{\text{Extr-elli}} = \begin{bmatrix} \hat{\boldsymbol{u}}^{\text{Extr-elli},0,T} \\ \hat{\boldsymbol{u}}^{\text{Extr-elli},1,T} \\ \vdots \\ \hat{\boldsymbol{u}}^{\text{Extr-elli},m-1,T} \end{bmatrix} \tag{18}$$

We can then rewrite Equation 15 as:

$$\mathbb{U}^{\text{Elli}} = \left\{ \frac{\boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{a}}{\|\boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{a}\|} \;\middle|\; a_i \in \mathbb{R} \right\} \tag{19}$$

It suffices to show that a vector $\boldsymbol{a} \in \mathbb{R}^m$ exists for which

$$\boldsymbol{G}\frac{\boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{a}}{\|\boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{a}\|} = c\hat{\boldsymbol{v}} \tag{20}$$

holds for some $c \in \mathbb{R}$. Since $\boldsymbol{G}\boldsymbol{U}^{\text{Extr-elli},T}$ is a square matrix with dimensions $m \times m$, we find that $\boldsymbol{a} = (\boldsymbol{G}\boldsymbol{U}^{\text{Extr-elli},T})^{-1}\hat{\boldsymbol{v}}$ is a solution. For this solution the equality in Equation 20 holds with $c = 1/\|\boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{a}\|$. This solution cannot be computed if $\boldsymbol{G}\boldsymbol{U}^{\text{Extr-elli},T}$ is not invertible. However, linear dependence of the rows or columns in this product implies linear dependence of the gradients, which we specifically assumed not to be the case.

Ultimately, we are only interested in the non-dominated part of the hyperellipsoid surface. The particular subset of $\mathbb{U}^{\text{Elli}}$ that we are interested in is obtained by taking only convex combinations instead of linear combinations. The reason for taking only convex combinations of the extremal directional derivatives is that these extrema are boundary points of the non-dominated part of the hyperellipsoid surface. At the directional-derivative transformation into the objective space of the normalized convex hull defined by the boundary directions, the hyperellipsoid curves away from (or into) the non-dominated region. The interior of the convex hull (i.e. the entirety of the convex combination) therefore maps to the non-dominated area that we are interested in. We denote this subset by $\mathbb{U}^{\text{Extr-elli}}$:

$$\mathbb{U}^{\text{Extr-elli}} = \left\{ \frac{\boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{\alpha}}{\|\boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{\alpha}\|} \,\middle|\, \begin{array}{l} \alpha_i \geq 0, i \in \{0,1,\ldots,m-1\}, \\ \sum_{i=0}^{m-1}\alpha_i = 1 \end{array} \right\} \tag{21}$$

The main derivations above are illustrated in Figure 3.

3) Intersection with negative hypercube: Although the non-dominated part of the hyperellipsoid is important and interesting, it may contain directions that map to directional derivatives that are not all-negative. An example of such a case for 2 objectives can seen in Figure 3 (first column, bottom row). To find those directions, we need to intersect $\mathbb{U}^{\text{Extr-elli}}$ with the negative hypercube in the m-dimensional objective space. The negative hypercube clearly is also a convex combination. To ensure that the negative hypercube is large enough to contain the entire hyperellipsoid, these vertices need to be placed sufficiently far. An illustration of the negative hypercube and its intersection with the hyperellipsoid is given in Figure 1.

To find the unit directions that map to the intersection of the negative hypercube in m dimensions and $\mathbb{U}^{\text{Extr-elli}}$ it suffices to compute the intersection of the negative hypercube and the linear mapping of the non-normalized convex combination of the negative extrema $\boldsymbol{G}\hat{\boldsymbol{u}}^{\text{Extr-elli},i}$, $i \in \{0,1,\ldots,m-1\}$ (i.e. $\mathbb{U}^{\text{Elli}}$ (Equation 21) without the normalization factor $\|\boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{\alpha}\|$), and then normalize the intersection points. The reason is that the convex combination $\sum_{i=0}^{m-1}\alpha_i\hat{\boldsymbol{u}}^{\text{Extr-elli},i}$ describes orientations. Normalization doesn't change the orientation. Neither does it change the orientation of $\boldsymbol{G}(\sum_{i=0}^{m-1}\alpha_i\hat{\boldsymbol{u}}^{\text{Extr-elli},i})$. So, a normalized vector maps to the negative hypercube if and only if the non-normalized version of that vector does. The non-normalized vector is a convex combination. After multiplication with matrix $\boldsymbol{G}$ it is still a convex combination, but it is a convex combination of the mapped vertices because matrix multiplication is a linear transformation, i.e. $\boldsymbol{G}(\sum_{i=0}^{m-1}\alpha_i\hat{\boldsymbol{u}}^{\text{Extr-elli},i}) = \sum_{i=0}^{m-1}\alpha_i(\boldsymbol{G}\hat{\boldsymbol{u}}^{\text{Extr-elli},i}) = \boldsymbol{G}\boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{\alpha}$. Now, the intersection of convex combinations is again a convex combination [40]. So, we can compute the intersection of the non-normalized convex combination of the mapped vertices, i.e. $\sum_{i=0}^{m-1}\alpha_i(\boldsymbol{G}\hat{\boldsymbol{u}}^{\text{Extr-elli},i})$, and the negative hypercube. The directions $\hat{\boldsymbol{u}}$ that map, i.e. $\boldsymbol{G}\hat{\boldsymbol{u}}$, to the vertices of this intersection define the convex combination that we ultimately want.

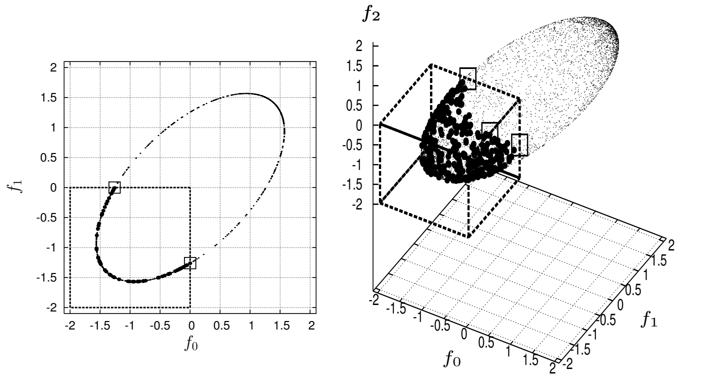

Fig. 1. Intersection of the hyperellipsoid of directional derivatives and the negative hypercube in objective space for the convex GenMED with two objectives (*left*) and three objectives (*right*) for $\boldsymbol{x} = (\frac{1}{2}, \frac{1}{2}, \ldots, \frac{1}{2})$ and $l = 10$. The surface of the hyperellipsoid and its negative subset are depicted using randomly drawn samples.

The vertices of the intersection of two convex combinations A and B consist of all vertices of A that are in B, all vertices of B that are in A, the intersection points of all boundary line-segments of A with B and the intersection points of all boundary line-segments of B with A.

a) Vertices of the negative hypercube: The vertices of the negative hypercube can never be in the convex combination of the $\boldsymbol{G}\hat{\boldsymbol{u}}^{\text{Extr-elli},i}$. The points on the main axes in the m-dimensional space can be chosen sufficiently far so that the negative part of the hyperellipsoid is completely contained in the negative hypercube. The only vertex of the negative hypercube that is then not outside the negative part of the hyperellipsoid is the origin. The origin is also never contained in the convex combination of the $\boldsymbol{G}\hat{\boldsymbol{u}}^{\text{Extr-elli},i}$ because the hyperellipsoid is centered there.

b) Vertices of the negative-extrema convex combination: The vertices of the linearly transformed convex combination of the negative-extrema directions, i.e. the $\boldsymbol{G}\hat{\boldsymbol{u}}^{\text{Extr-elli},i}$ for $i \in \{0,1,\ldots,m-1\}$, may be contained in the negative hypercube. Testing for this is simple, just test whether $\boldsymbol{G}\hat{\boldsymbol{u}}^{\text{Extr-elli},i}$ is all non-positive.

c) Line-segments of the negative hypercube: The only line-segments of the negative-hypercube that can intersect with the convex combination of the $\boldsymbol{G}\hat{\boldsymbol{u}}^{\text{Extr-elli},i}$ are the m negative axes. To see this, again the negative hypercube can be made as big as desired, completely containing the entire negative part of the hyperellipsoid. The only vertex of the negative hypercube that is then inside the hyperellipsoid is the origin.

All other vertices are outside. The lines connecting these vertices either move completely outside the hyperellipsoid or connect to the origin, intersecting the hyperellipsoid. Possibly this intersection lies in the subset $\mathbb{U}^{\text{Extr-elli}}$.

To compute the m intersection points with the hyperellipsoid, let $\hat{\boldsymbol{e}}^i$ be an m-dimensional unit vector such that $\hat{\boldsymbol{e}}^i_i = 1, \hat{\boldsymbol{e}}^i_j = 0, i, j \in \{0, 1, \ldots, m-1\}, i \neq j$. Let $\lambda_i \geq 0$. We know that the hyperellipsoid surface is defined by Equation 19. The intersection of the hyperellipsoid surface with the negative part of i-th coordinate axis is therefore given by an $\boldsymbol{a}^i \in \mathbb{R}^m$ in Equation 19 for which the following holds:

$$\frac{\boldsymbol{G}\boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{a}^i}{\| \boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{a}^i \|} = -\lambda_i \hat{\boldsymbol{e}}^i \tag{22}$$

Such an $\boldsymbol{a}^i$ is given by $-(\boldsymbol{G}\boldsymbol{U}^{\text{Extr-elli},T})^{-1}\hat{\boldsymbol{e}}^i$, for which we have $\lambda_i = 1/ \| \boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{a}^i \|$. We now define a matrix $\boldsymbol{V}$ of dimensions $l \times m$ as follows:

$$\boldsymbol{V} = \boldsymbol{U}^{\text{Extr-elli},T} \left(\boldsymbol{G}\boldsymbol{U}^{\text{Extr-elli},T}\right)^{-1} \tag{23}$$

The unit directions $\hat{\boldsymbol{u}}^{\text{Cube-elli},i}, i \in \{0, 1, \ldots, m-1\}$ that map to the desired intersection points, i.e. $\boldsymbol{G}\hat{\boldsymbol{u}}^{\text{Cube-elli},i} = -\lambda_i \hat{\boldsymbol{e}}^i$, can now be written as:

$$\hat{\boldsymbol{u}}^{\text{Cube-elli},i} = -\frac{\boldsymbol{V}\hat{\boldsymbol{e}}^i}{\|\boldsymbol{V}\hat{\boldsymbol{e}}^i\|} \tag{24}$$

We must still construct a test to see whether $\hat{\boldsymbol{u}}^{\text{Cube-elli},j}$ is in the final intersection. For $\hat{\boldsymbol{u}}^{\text{Cube-elli},j}$ to be in $\mathbb{U}^{\text{Extr-elli}}$, $\hat{\boldsymbol{u}}^{\text{Cube-elli},j}$ must be a normalized version of some convex combination of the $\hat{\boldsymbol{u}}^{\text{Extr-elli},i}, i \in \{0, 1, \ldots, m-1\}$. In other words, for $\alpha_i \geq 0, i \in \{0, 1, \ldots, m-1\}$, $\sum_{i=0}^{m-1} \alpha_i = 1$ and some $\lambda > 0$, we require:

$$\boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{\alpha} = \lambda \hat{\boldsymbol{u}}^{\text{Cube-elli},j} = -\lambda \frac{\boldsymbol{V}\hat{\boldsymbol{e}}^j}{\|\boldsymbol{V}\hat{\boldsymbol{e}}^j\|} \tag{25}$$

We now define a matrix $\boldsymbol{W}$ of dimensionality $m \times m$ such that $\boldsymbol{V} = \boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{W}$, i.e. :

$$\boldsymbol{W} = \left(\boldsymbol{G}\boldsymbol{U}^{\text{Extr-elli},T}\right)^{-1} \tag{26}$$

Combining Equations 25 and Equation 26 and multiplying both sides in Equation 25 from the left by $\boldsymbol{G}$ now gives:

$$\boldsymbol{G}\boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{\alpha} = \boldsymbol{G}\boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{W}\hat{\boldsymbol{e}}^j \left(-\frac{\lambda}{\|\boldsymbol{V}\hat{\boldsymbol{e}}^j\|}\right) \tag{27}$$

By letting $\gamma = -\lambda / \|\boldsymbol{V}\hat{\boldsymbol{e}}^j\|$ we find:

$$\boldsymbol{\alpha} = \gamma \boldsymbol{W}\hat{\boldsymbol{e}}^j \tag{28}$$

Because of the convexity constraint $\sum_{i=0}^{m-1} \alpha_i = 1$ we have $\gamma = 1/\sum_{i=0}^{m-1}(\boldsymbol{W}\hat{\boldsymbol{e}}^j)_i$. Now $\boldsymbol{\alpha}$ is uniquely determined. Since $\lambda = -\gamma \| \boldsymbol{V}\hat{\boldsymbol{e}}^j \|$ and $\| \boldsymbol{V}\hat{\boldsymbol{e}}^j \| \geq 0$, requiring that $\lambda > 0$ is equivalent to requiring $\gamma < 0$. Hence, to see whether $\hat{\boldsymbol{u}}^{\text{Cube-elli},j}$ is in the intersection it suffices to check that $\gamma < 0$ and $\alpha_i \geq 0, i \in \{0, 1, \ldots, m-1\}$.

d) Line-segments of negative-extrema convex combination: All line-segments between combinations of all vertices of the convex combination $\boldsymbol{G}\boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{\alpha}$ are boundary line-segments of that convex combination. In other words, there are no line-segments between vertices that pass through the interior of the convex polytope $\boldsymbol{G}\boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{\alpha}$. The reason for this is that the convex polytope has m vertices in m dimensions. Hence, in two dimensions it is a line, in three dimensions it is a triangle, in four dimensions it is a prism, and so on.

To compute the intersection of the line-segment between $\boldsymbol{G}\hat{\boldsymbol{u}}^{\text{Extr-elli},i}$ and $\boldsymbol{G}\hat{\boldsymbol{u}}^{\text{Extr-elli},j}$, $i, j \in \{0, 1, \ldots, m-1\}, j > i$ and the boundary of the negative hypercube, note that the only parts of the boundary of the negative-hypercube that can be intersected are the m subspaces obtained by forcing one of the coordinates to be 0. The other subspaces that define boundaries of the negative hypercube only bound the hyperellipsoid, assuming that we make the hypercube large enough.

Let $\lambda^k_k = 0$, $k \in \{0, 1, \ldots, m-1\}$ and $\lambda^k_q \in \mathbb{R}$, $k, q \in \{0, 1, \ldots, m-1\}, k \neq q$. Moreover, let $\boldsymbol{\lambda}^k = (\lambda^k_0, \lambda^k_1, \ldots, \lambda^k_{m-1})$ and $b^k_{ij} \in \mathbb{R}, i, j, k \in \{0, 1, \ldots, m-1\}$. The intersection point of the line between $\boldsymbol{G}\hat{\boldsymbol{u}}^{\text{Extr-elli},i}$ and $\boldsymbol{G}\hat{\boldsymbol{u}}^{\text{Extr-elli},j}$ and the subspace of the negative hypercube that excludes axis k is given by solving:

$$\boldsymbol{G}\left(\hat{\boldsymbol{u}}^{\text{Extr-elli},i} + b^k_{ij}\left(\hat{\boldsymbol{u}}^{\text{Extr-elli},j} - \hat{\boldsymbol{u}}^{\text{Extr-elli},i}\right)\right) = \boldsymbol{\lambda}^k \tag{29}$$

Because of the special form of $\boldsymbol{\lambda}^k$, this boils down to only a single equality:

$$(\nabla f_k(\boldsymbol{x}))^T\left(\hat{\boldsymbol{u}}^{\text{Extr-elli},i} + b^k_{ij}\left(\hat{\boldsymbol{u}}^{\text{Extr-elli},j} - \hat{\boldsymbol{u}}^{\text{Extr-elli},i}\right)\right) = 0 \tag{30}$$

The solution to Equation 30 is given by:

$$b^k_{ij} = -\frac{(\nabla f_k(\boldsymbol{x}))^T \hat{\boldsymbol{u}}^{\text{Extr-elli},i}}{(\nabla f_k(\boldsymbol{x}))^T\left(\hat{\boldsymbol{u}}^{\text{Extr-elli},j} - \hat{\boldsymbol{u}}^{\text{Extr-elli},i}\right)} \tag{31}$$

Hence, the $\frac{1}{2}(m^3 - m^2)$ candidate vertices $(i, j, k \in \{0, 1, \ldots, m-1\}, j > i)$ are:

$$\hat{\boldsymbol{u}}^{\text{can}} = \frac{\hat{\boldsymbol{u}}^{\text{Extr-elli},i} + b^k_{ij}\left(\hat{\boldsymbol{u}}^{\text{Extr-elli},j} - \hat{\boldsymbol{u}}^{\text{Extr-elli},i}\right)}{\left\|\hat{\boldsymbol{u}}^{\text{Extr-elli},i} + b^k_{ij}\left(\hat{\boldsymbol{u}}^{\text{Extr-elli},j} - \hat{\boldsymbol{u}}^{\text{Extr-elli},i}\right)\right\|} \tag{32}$$

Moreover, a candidate vertex is on the line-segment between $\boldsymbol{G}\hat{\boldsymbol{u}}^{\text{Extr-elli},i}$ and $\boldsymbol{G}\hat{\boldsymbol{u}}^{\text{Extr-elli},j}$ (and thus in $\mathbb{U}^{\text{Extr-elli}}$) if and only if $0 \leq b^k_{ij} \leq 1$. Finally, a candidate vertex maps to the negative-hypercube if and only if all components of $\boldsymbol{G}\hat{\boldsymbol{u}}^{\text{can}}$ are non-positive.

The main derivations above are illustrated in Figure 3. In the following Section, we describe, in the form of an algorithm, how the derivations can be used to actually compute the final result, i.e. the set of all non-dominated simultaneously improving directions. To this end, as we will see, the most important equations are Equation 13, Equation 24 and Equation 32. These equations, as well as the algorithm in which these equations are used, can be seen as a summary of the most important results so far.

B. Computation

The results in the previous Subsection now allow us to formulate an algorithm that computes the complete set of unit directions that we are ultimately interested in. This set of unit directions is described by the normalized convex combination of the unit directions that map to the intersection points as described above. The unit directions that are positively tested to be in the intersection of the non-dominated part of the hyperellipsoid and the negative hypercube make up a matrix $\boldsymbol{U}^\star$ of dimensions $n \times l$ where n is the number of vertices in the convex intersection[1]:

$$\mathbb{U}^\star = \left\{ \frac{\boldsymbol{U}^{\star,T}\boldsymbol{\beta}}{\|\boldsymbol{U}^{\star,T}\boldsymbol{\beta}\|} \,\middle|\, \begin{array}{l} \beta_i \geq 0, i \in \{0,1,\ldots,m-1\}, \\ \sum_{i=0}^{m-1} \beta_i = 1 \end{array} \right\} \tag{33}$$

A direction vector for which the directional derivative is non-dominated and all-negative can now be sampled by sampling a vector $\boldsymbol{\beta}$ such that $\sum_{i=0}^{m-1} \beta_i = 1$, computing $\boldsymbol{U}^{\star,T}\boldsymbol{\beta}$ and normalizing the resulting vector. To sample $\boldsymbol{\beta}$ uniformly, the following approach can be used [41]. First, draw m uniformly distributed samples x_i, $i \in \{0,1,\ldots,m-1\}$ from the uniform distribution over $]0,1]$. Then, set $y_i = -\ln(x_i)$. Finally, the βi are given by $\beta_i = y_i / \sum_{j=0}^{m-1} y_j$.

Pseudo-code for the algorithm that determines $\boldsymbol{U}^\star$ is given in Figure 2. A set T is maintained to which unit directions are added that are found to be in the intersection of the non-dominated set and the negative-hypercube set. First, the vertices of the non-dominated set are tested (i.e. Equation 13). Then the intersections of the line-segments of the negative hypercube with the non-dominated set are checked (i.e. Equation 24). Finally, the intersections of the line-segments of the negative-extrema convex combination with the negative hypercube are checked (i.e. Equation 32).

Line 1 costs $\mathcal{O}(1)$. Line 2 costs $\mathcal{O}(mlF + m^2l + m^2 + ml)$. Line 3 costs $\mathcal{O}(ml)$. Line 4 costs $\mathcal{O}(m^2l + m^3)$. Line 5 costs $\mathcal{O}(m^2l)$. Line 6 costs $\mathcal{O}(ml + m^2 + m^2 + m^2 + m^2 + ml)$. Line 7 costs $\mathcal{O}(m^3l + m^3l + m^4l + m^4 + m^3l)$. Line 8 costs $\mathcal{O}(m^3l)$ (there are $\mathcal{O}(m^3)$ candidate directions). Hence, the overall computational complexity of the algorithm is $\mathcal{O}(mlF + m^4l)$ and it requires $m(l+1)$ function evaluations.

Illustrations of the application of the algorithm in Figure 2 and the main involved derivations are given in Figure 3 on the convex variant of the GenMED problem with two objectives and three objectives and different starting points. For the two-objective case, the different starting points correspond to different orientations of the hyperellipsoid. Previous work [32] considered only the bottom case where the DOI is equal to the negative surface of the hyperellipsoid which, in turn, is a subset of the Pareto-optimal search directions in this case. The top illustration for two objectives shows that if the hyperellipsoid is oriented differently, the negative part of the hyperellipsoid surface contains also dominated search directions. The DOI in that case is equal to the Pareto-optimal search directions which, in turn, now is a subset of the negative part of the hyperellipsoid surface. The algorithm presented in this article succeeds in finding the correct subset in both cases.

[1]The only reason for using β for the mixing components instead of α is to avoid confusion with Equation 21.

```
1 $T \leftarrow \emptyset$
2 for $i \in \{0,1,\ldots,m-1\}$ do
  2.1 $\hat{\boldsymbol{u}}^{\text{Extr-elli},i} \leftarrow -\frac{\nabla f_i(\boldsymbol{x})}{\|\nabla f_i(\boldsymbol{x})\|}$
  2.2 $\boldsymbol{v} \leftarrow \boldsymbol{G}\hat{\boldsymbol{u}}^{\text{Extr-elli},i}$
  2.3 if $\forall q \in \{0,1,\ldots,m-1\} : v_q \leq 0$ then
      2.3.1 $T \leftarrow T \cup \hat{\boldsymbol{u}}^{\text{Extr-elli},i}$
3 $\boldsymbol{U}^{\text{Extr-elli},T} \leftarrow \left[\hat{\boldsymbol{u}}^{\text{Extr-elli},0} \;\; \hat{\boldsymbol{u}}^{\text{Extr-elli},1} \;\; \ldots \;\; \hat{\boldsymbol{u}}^{\text{Extr-elli},m-1}\right]$
4 $\boldsymbol{W} \leftarrow \left(\boldsymbol{G}\boldsymbol{U}^{\text{Extr-elli},T}\right)^{-1}$
5 $\boldsymbol{V} \leftarrow \boldsymbol{U}^{\text{Extr-elli},T}\boldsymbol{W}$
6 for $i \in \{0,1,\ldots,m-1\}$ do
  6.1 $\hat{\boldsymbol{u}}^{\text{Cube-elli},i} \leftarrow -\frac{\boldsymbol{V}\hat{\mathbf{e}}^i}{\|\boldsymbol{V}\hat{\mathbf{e}}^i\|}$
  6.2 $\boldsymbol{v} \leftarrow \boldsymbol{W}\hat{\mathbf{e}}^i$
  6.3 $\gamma \leftarrow \frac{1}{\sum_{j=0}^{m-1} v_j}$
  6.4 $\boldsymbol{\alpha} \leftarrow \gamma\boldsymbol{v}$
  6.5 if $\gamma < 0$ and $\forall q \in \{0,1,\ldots,m-1\} : \alpha_q \geq 0$ then
      6.5.1 $T \leftarrow T \cup \hat{\boldsymbol{u}}^{\text{Cube-elli},i}$
7 for $i \in \{0,1,\ldots,m-1\}$ do
  7.1 for $j \in \{i+1,i+2,\ldots,m-1\}$ do
      7.1.1 for $k \in \{0,1,\ldots,m-1\}$ do
            7.1.1.1 $b_{ij}^k \leftarrow -\frac{(\nabla f_k(\boldsymbol{x}))^T\hat{\boldsymbol{u}}^{\text{Extr-elli},i}}{(\nabla f_k(\boldsymbol{x}))^T\left(\hat{\boldsymbol{u}}^{\text{Extr-elli},j}-\hat{\boldsymbol{u}}^{\text{Extr-elli},i}\right)}$
            7.1.1.2 $\hat{\boldsymbol{u}}^{\text{can}} \leftarrow \frac{\hat{\boldsymbol{u}}^{\text{Extr-elli},i}+b_{ij}^k\left(\hat{\boldsymbol{u}}^{\text{Extr-elli},j}-\hat{\boldsymbol{u}}^{\text{Extr-elli},i}\right)}{\|\hat{\boldsymbol{u}}^{\text{Extr-elli},i}+b_{ij}^k\left(\hat{\boldsymbol{u}}^{\text{Extr-elli},j}-\hat{\boldsymbol{u}}^{\text{Extr-elli},i}\right)\|}$
            7.1.1.3 $\boldsymbol{v} \leftarrow \boldsymbol{G}\hat{\boldsymbol{u}}^{\text{can}}$
            7.1.1.4 if $0 \leq b_{ij}^k \leq 1$ and
                        $\forall q \in \{0,1,\ldots,m-1\} : v_q \leq 0$ then
                    7.1.1.4.1 $T \leftarrow T \cup \hat{\boldsymbol{u}}^{\text{can}}$
8 Construct matrix $\boldsymbol{U}^\star$ by using the vectors in $T$ as rows.
```

Fig. 2. Algorithm for computing matrix $\boldsymbol{U}^\star$ containing the direction vectors that constitute the convex combination of direction vectors for which the multi-objective directional derivative is non-dominated and all-negative.

In three dimensions there is a third possibility for the DOI. It can then be the case that neither the negative part of the hyperellipsoid surface is a subset of the Pareto-optimal part nor the other way around. The first two illustrations for the three-objective case in Figure 3 show cases where the two sets are, similar to the two-objective cases, subsets of each other. The final illustration shows a case when these sets overlap. In this case, the DOI contains only a single point from the original two sets. The algorithm in this article finds the required five intersection points that define the DOI.

C. Gradient-based optimization algorithms

Although in the above we have extensively investigated the structure of the gradient in the multi-objective case, single-objective optimization algorithms can still be used to perform multi-objective optimization. In the following we define two of such algorithms (ROCG and AORL) as well as a method that uses the DOI for the multi-objective case as computed by the algorithm in Figure 2.

1) Random-Objective Conjugate Gradients (ROCG): In this straightforward approach the conjugate-gradients algorithm is applied to a randomly chosen objective. It depends on the correlation between the objectives whether the best local improvement in a single objective also leads to an improvement in the other objectives.

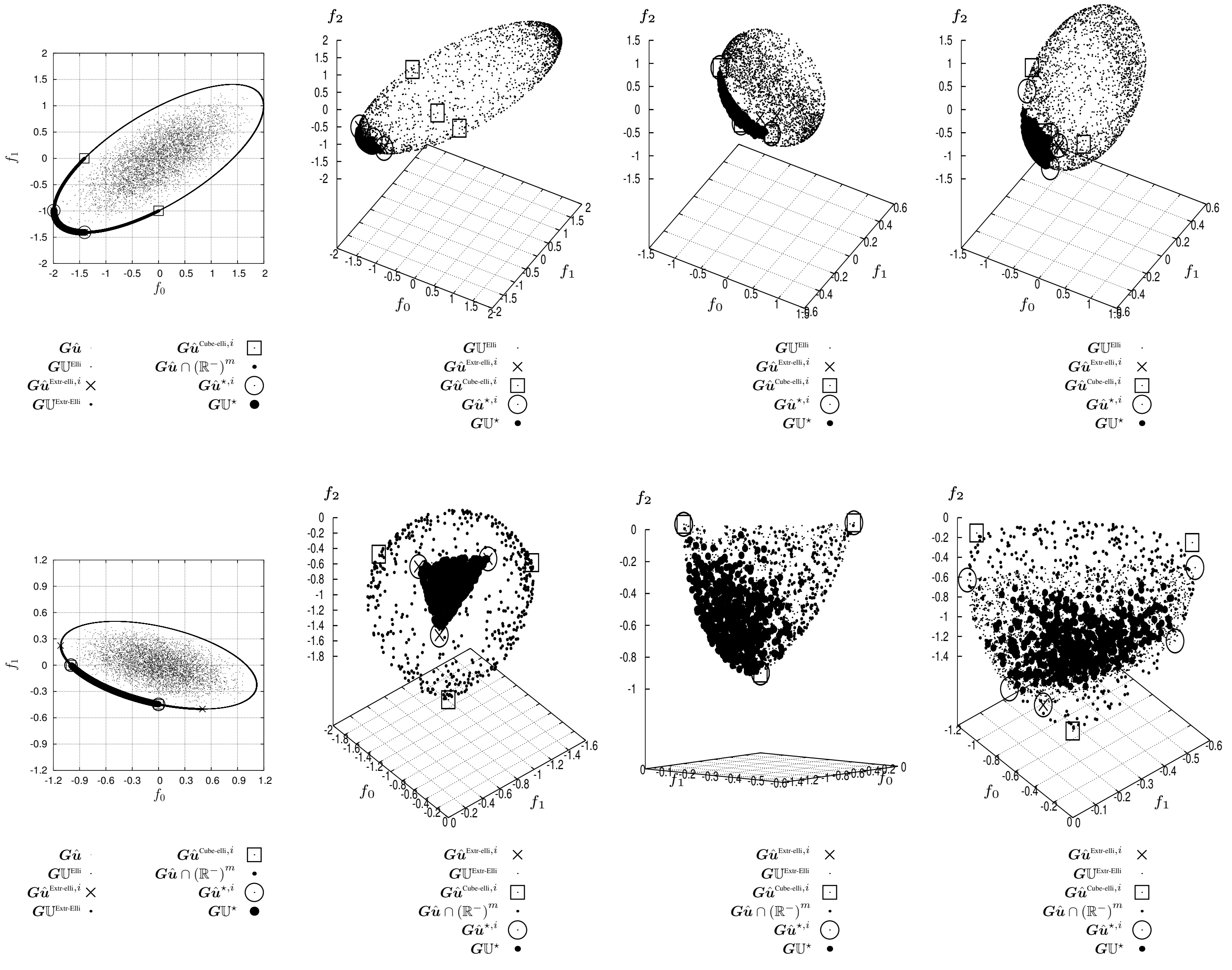

Fig. 3. Illustration of the most important equations that contribute to the DOI for the multi-objective case. Shown are samplings of the complete hyperellipsoid of directional derivatives ($\boldsymbol{G\hat{u}}$, only for the case of 2 objectives), the hyperellipsoid surface ($\boldsymbol{G}\mathbb{U}^{\text{Elli}}$), the boundary points of the Pareto-optimal region ($\boldsymbol{G\hat{u}}^{\text{Extr-elli},i}$), their convex combinations ($\boldsymbol{G}\mathbb{U}^{\text{Extr-elli}}$), the intersection points with the negative hypercube ($\boldsymbol{G\hat{u}}^{\text{Cube-elli},i}$), the negative part of the hyperellipsoid ($\boldsymbol{G\hat{u}} \cap \left(\mathbb{R}^-\right)^m$), the points contributing to the DOI ($\boldsymbol{G\hat{u}}^{\star,i}$) and the entire set of DOI ($\boldsymbol{G}\mathbb{U}^\star$). The first column shows the convex GenMED problem with two objectives for points $(0, 1, \frac{1}{2}, \frac{1}{2}, \ldots, \frac{1}{2})$ (*top*) and $(0, \frac{1}{2}, 0, 0, \ldots, 0)$ (*bottom*). The other columns show the GenMED problem with three objectives where the bottom plot is a zoomed version showing only the negative subspace, rotated for the best view. The corresponding points are $(0, 1, \frac{1}{2}, \frac{1}{2}, \ldots, \frac{1}{2})$ (*column 2*), $(0, \frac{1}{2}, 0, 0, \ldots, 0)$ (*column 3*) and $(0, \frac{1}{2}, -\frac{1}{4}, 0, 0, \ldots, 0)$ (*column 4*). For all problems, $l = 10$.

2) Alternating-Objective Repeated Line search (AORL): To reduce the probability of improving a single objective while making the objective value in the other objective worse, the objective that is searched locally can be altered during optimization. In AORL, a single line search in the direction of the negative gradient of that objective is performed in a single, alternatingly chosen objective. This process is repeated until a multi-objective local minimum is found.

3) Combined-Objectives Repeated Line search (CORL): To use the DOI and follow the gradient in the multi-objective case, the single-objective and single-dimensional Brent's minimization method mentioned earlier in Section IV-C can be used on a specially designed function to perform a line search. After a line search terminates, a new line search can be executed after computing the DOI at the new location. This can be repeated until a maximum of iterations is reached or until no further improvements are found.

To perform a single line search, we use the negative scaled Euclidean distance in the objective space to the point $\boldsymbol{x}$ from where Brent's method is to start. The scaling is required to obtain invariance to different scales of objective functions. Scaling is done by dividing the distance in each objective by the observed range of objective values for that objective. This is a quite natural generalization of the single-objective case where the distance is just the difference in function value between the starting point and the local minimum. The further we travel in objective space while improving upon the point where we started from, the better. If a point $\boldsymbol{x} + a\hat{\boldsymbol{u}}^{\text{DOI}}(\boldsymbol{x})$ along

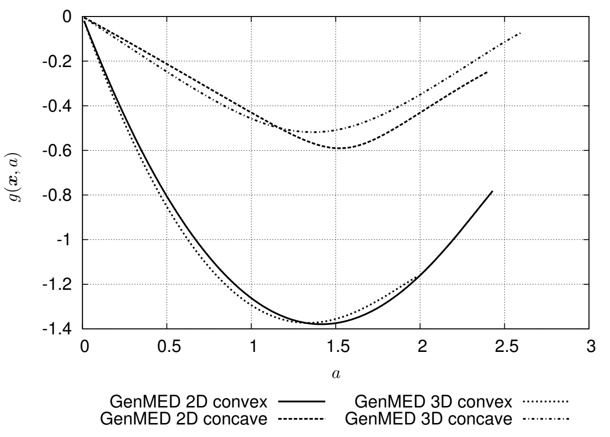

Fig. 4. Shape of the $g(\boldsymbol{x}, a)$ function to be optimized with Brent's method in the multi-objective case. A random search direction is chosen from the DOI for the two-objective and three-objective GenMED problem (both the convex and concave variant) for starting point $\boldsymbol{x} = (\frac{1}{2}, \frac{1}{2}, \ldots, \frac{1}{2})$ and $l = 10$.

a Pareto-optimal search direction $\hat{\boldsymbol{u}}^{\text{DOI}}(\boldsymbol{x})$ does not dominate $\boldsymbol{x}$ in objective space, then we define the distance to be infinity. In other words, let r_i be the range observed for objective i, then we use Brent's method to minimize ($a \geq 0$):

$$g(\boldsymbol{x}, a) = \begin{cases} -d(\boldsymbol{f}(\boldsymbol{x}), \boldsymbol{f}(\boldsymbol{x} + a\hat{\boldsymbol{u}}^{\text{DOI}}(\boldsymbol{x}))) & \text{if } \boldsymbol{f}(\boldsymbol{x} + a\hat{\boldsymbol{u}}^{\text{DOI}}(\boldsymbol{x})) \\ & \quad \prec \boldsymbol{f}(\boldsymbol{x}) \\ \infty & \quad otherwise \end{cases} \tag{34}$$

where

$$d(\boldsymbol{y}, \boldsymbol{z}) = \sqrt{\sum_{i=0}^{m-1} \left(\frac{y_i - z_i}{r_i} \right)^2} \tag{35}$$

Function $g(\boldsymbol{x}, a)$ is illustrated in Figure 4 for the GenMED problem with two objectives and three objectives, a randomly chosen DOI and the same starting point. The functions clearly have a well-defined minimum. Moving too far away from the starting point leads to solutions that don't dominate the starting point, which is where the curves in Figure 4 disappear.

VI. Experiments

In this section we experimentally investigate the performance of various gradient-based optimization algorithms on a set of well-known multi-objective optimization problems. Because MOEAs are known to be good at finding a representative subset of the Pareto front, we study the performance of gradient-based optimization algorithms both in a standalone manner as well as in conjunction with the use of a MOEA to see whether gradient-based optimization algorithms are of additional value. Specifically, a random-restart optimization algorithms that restarts the algorithms in Section V-C to optimize a single point are presented in Section VI-B. An adaptive hybridization of a MOEA with the same algorithms is presented in Section VI-C. In the next Section we first describe the optimization problems we consider and how we will measure an algorithm's performance.

A. Optimization problems and measuring performance

1) Multi-objective optimization problem test suite: The definitions of the problems in our multi-objective optimization problem test suite are presented in Table I.

The first two problems we use are the most simple ones. They are the convex and concave GenMED problems from Section V. Each objective of GenMED is similarly scaled. There are furthermore no constraints and no local Pareto fronts, making the problem relatively simple in a way that is comparable to the simplicity of the sphere function in single-objective real-valued optimization. The domain of $[-1; 1]$ for each variable is only used for initialization, it is not a hard constraint. In the following we refer to the convex version of GenMED as GM_1 and the concave version as GM_2. The optimal Pareto front for GM_1 is described parametrically by $(t^2, (1-t)^2)$ with $t \in [0; 1]$. For GM_2 a parametric description of the optimal Pareto front is given by $(\sqrt{t}, \sqrt{1-t})$ with $t \in [0; 1]$

We also used the well-known problems[2] EC_i, $i \in \{1, 2, 3, 4, 6\}$. As these problems are well-known in MOEA literature, we only very briefly discuss these problems here and refer the interested reader for more details about these functions to the literature [42], [31]. The box-boundary constraints on all EC_i problems are to be considered rigid. The reason for this is that otherwise some objectives can not always be evaluated. Such rigid constraints can be hard for a numerical optimizer. EC_1 and EC_2 are continuous and do not have any local fronts. EC_1 has a convex Pareto front whereas EC_2 has a concave Pareto front. The problems differ from the GenMED problems in that the objectives are not similarly defined and not similarly scaled. EC_3 is similar to EC_1 but has a discontinuous Pareto front. EC_4 has many locally optimal Pareto fronts. Finally, the Pareto front of EC_6 is non-uniformly distributed. The optimal Pareto front for EC_1 is described parametrically by $(t, 1 - \sqrt{t})$ with $t \in [0; 1]$. For EC_2 it is defined by $(t, 1 - t^2)$ with $t \in [0; 1]$. The description for EC_3 is $(t, 1.0 - \sqrt{t} - t\sin(10\pi t))$ with $t \in [0; 1]$. However, it is only the non-dominated parts of this parametric curve that make up the optimal Pareto front for EC_3. The optimal Pareto front for EC_4 is described parametrically by $(t, 1 - \sqrt{t})$ with $t \in [0; 1]$. For EC_6 it is defined by $(t, 1 - t^2)$ with $t \in [1 - e^{-\frac{1}{3}}; 1]$.

We have taken two more problems from more recent literature on numerical multi-objective optimization [30]. These problems are labeled BD_i, $i \in \{1, 2\}$. These problems were introduced to remedy a shortcoming in the range of problem-difficulties presented by the EC_i problems. Both problems make use of Rosenbrock's function. Premature convergence on this function is likely without proper induction of the structure of the search space. Function BD_2 is harder than BD_1 in the sense that the objective functions overlap in all variables instead of only in the first one. Further, the domain of x_0 in function BD_1 is rigid. Finally, we have scaled the objectives of BD_2 to ensure that the optimum of all problems is in approximately the same range. By doing so, using the same value-to-reach for the $\boldsymbol{D}_{\boldsymbol{\mathcal{P}_F} \rightarrow \boldsymbol{\mathcal{S}}}$ indicator (which is explained in the next Section) on all problems corresponds to a similar

[2]These problems are also known as ZDT_i.

front-quality on all problems. The optimal Pareto front for BD_1 is described parametrically by $(t, 1-t)$ with $t \in [0;1]$. For BD_2 we do not have a parametric description available.

Name	Objectives	Domain
GM_1	$f_0 = \left\Vert \frac{1}{2}(\boldsymbol{x}-\boldsymbol{c}^0)\right\Vert^d, \quad f_1 = \left\Vert \frac{1}{2}(\boldsymbol{x}-\boldsymbol{c}^1)\right\Vert^d$ $\boldsymbol{c}^0 = (1,0,0,\ldots), \quad \boldsymbol{c}^1 = (0,1,0,0,\ldots), \quad d = 2$	$[-1;1]^{10}$ $(l = 10)$
GM_2	$f_0 = \left\Vert \frac{1}{2}(\boldsymbol{x}-\boldsymbol{c}^0)\right\Vert^d, \quad f_1 = \left\Vert \frac{1}{2}(\boldsymbol{x}-\boldsymbol{c}^1)\right\Vert^d$ $\boldsymbol{c}^0 = (1,0,0,\ldots), \quad \boldsymbol{c}^1 = (0,1,0,0,\ldots), \quad d = \frac{1}{2}$	$[-1;1]^{10}$ $(l = 10)$
EC_1	$f_0 = x_0, \quad f_1 = \gamma\left(1-\sqrt{f_0/\gamma}\right)$ $\gamma = 1+9\left(\sum_{i=1}^{l-1} x_i/(l-1)\right)$	$[0;1]^{30}$ $(l = 30)$
EC_2	$f_0 = x_0, \quad f_1 = \gamma\left(1-(f_0/\gamma)^2\right)$ $\gamma = 1+9\left(\sum_{i=1}^{l-1} x_i/(l-1)\right)$	$[0;1]^{30}$ $(l = 30)$
EC_3	$f_0 = x_0$ $f_1 = \gamma\left(1-\sqrt{f_0/\gamma}-(f_0/\gamma)\sin(10\pi f_0)\right)$ $\gamma = 1+9\left(\sum_{i=1}^{l-1} x_i/(l-1)\right)$	$[0;1]^{30}$ $(l = 30)$
EC_4	$f_0 = x_0, \quad f_1 = \gamma\left(1-\sqrt{f_0/\gamma}\right)$ $\gamma = 1+10(l-1)+\sum_{i=1}^{l-1}\left(x_i^2-10\cos(4\pi x_i)\right)$	$[-1;1]\times$ $[-5;5]^9$ $(l = 10)$
EC_6	$f_0 = 1-e^{-4x_0}\sin^6(6\pi x_0)$ $f_1 = \gamma\left(1-(f_0/\gamma)^2\right)$ $\gamma = 1+9\left(\sum_{i=1}^{l-1} x_i/(l-1)\right)^{0.25}$	$[0;1]^{10}$ $(l = 10)$
BD_1	$f_0 = x_0$ $f_1 = 1-x_0+\gamma$ $\gamma = \sum_{i=1}^{l-2}\left(100(x_{i+1}-x_i^2)^2+(1-x_i)^2\right)$	$[0;1]\times$ $[-5.12;5.12]^9$ $(l = 10)$
BD_2^{s}	$f_0 = \frac{1}{l}\sum_{i=0}^{l-1} x_i^2$ $f_1 = \frac{1}{l-1}\sum_{i=0}^{l-2}\left(100(x_{i+1}-x_i^2)^2+(1-x_i)^2\right)$	$[-5.12;5.12]^{10}$ $(l = 10)$

TABLE I
THE MULTI-OBJECTIVE OPTIMIZATION PROBLEM TEST SUITE.

It is important to note that all variables have a bounded domain. For the EC_i problems and for x_0 in function BD_1, this domain is rigid, i.e. a constraint. If these variables move outside of their domain, some objective values can become non-existent. It is therefore important to keep these variables within their domains. However, a simple repair mechanism that changes a variable to its boundary value if it has exceeded this boundary value gives artifacts that may lead us to draw false conclusions about the performance of the tested algorithms. If for instance the search on problem EC_6 probes a solution that has a negative value for each of the variables x_i with $i \geq 1$, then the repair mechanism sets all these variables to 0. This is especially well possible during a gradient-search procedure because the gradient with respect to the second objective points in the direction of all negative values for variables x_i with $i \geq 1$. It is not hard to see that the solution that results after boundary repair lies on the Pareto front. We have therefore adapted the gradient-based optimization algorithms such that a solution is never changed into one outside of the problem range. Similarly, the variation procedure of the MOEA is changed to prevent generating solutions that are out of bounds. Finally, we note that it was shown by Mukai [17] that it is possible to derive directions that improve all objectives and preserve the feasibility of the solution by taking into account constraint information. This result has however not been extended to the description of all Pareto-optimal improving directions that we use here.

2) Measuring performance: To measure performance we only consider the subset of all non-dominated solutions in the population upon termination. We call such a subset an approximation set and denote it by $\mathcal{S}$. A performance indicator is a function of approximation sets $\mathcal{S}$ and returns a real value that indicates the quality of $\mathcal{S}$ in some aspect. More detailed information regarding the importance of using good performance indicators may be found in literature [43], [44], [45].

Here we use a performance indicator that uses knowledge of the optimum, i.e. the optimal Pareto front. We define the distance $d(\boldsymbol{x}^0, \boldsymbol{x}^1)$ between two multi-objective solutions $\boldsymbol{x}^0$ and $\boldsymbol{x}^1$ to be the Euclidean distance between their objective values $f(\boldsymbol{x}^0)$ and $f(\boldsymbol{x}^1)$. The performance indicator we use computes the average distance to the closest solution in an approximation set $\mathcal{S}$ over all solutions in the optimal Pareto set $\boldsymbol{\mathcal{P}_S}$. We denote this indicator by $\boldsymbol{D}_{\boldsymbol{\mathcal{P}_F}\rightarrow\mathcal{S}}$ and refer to it as the distance from the optimal Pareto front to an approximation set. This indicator was first used by Van Veldhuizen and was called the inverted generational distance [46]. A smaller value for this performance indicator is preferable and a value of 0 is obtained if and only if the approximation set and the optimal Pareto front are identical. This indicator is useful for evaluating performance if the optimum is known because it describes how well the optimal Pareto front is covered and thereby represents an intuitive trade-off between the diversity of the approximation set and its proximity (i.e. closeness to the optimal Pareto front). Even if all points in the approximation set are on the optimal Pareto front the indicator is not minimized unless the solutions in the approximation set are spread out perfectly.

Because the optimal Pareto front may be continuous, there are infinitely many solutions possible on the optimal Pareto front. Therefore, a uniformly sampled set of many solutions along the optimal Pareto front can be computed to use in the discretized version of $\boldsymbol{\mathcal{P}_F}$ instead as an approximation of the continuous version. We have used this approach with 5000 uniformly sampled points. The performance indicator now is defined as follows:

$$\boldsymbol{D}_{\boldsymbol{\mathcal{P}_F}\rightarrow\mathcal{S}}(\mathcal{S}) = \frac{1}{|\boldsymbol{\mathcal{P}_S}|}\sum_{\boldsymbol{x}^1\in\boldsymbol{\mathcal{P}_S}} \min_{\boldsymbol{x}^0\in\mathcal{S}}\{d(\boldsymbol{x}^0,\boldsymbol{x}^1)\} \tag{36}$$

To obtain the 5000 points on the optimal Pareto fronts, we used the parametric descriptions that we provided when we described the test suite. A uniform sampling in the parametric parameter t was performed and the corresponding parametric description was computed. This process is repeated until 5000 non-dominated solutions have been found. Note that on problem EC_3 the parametric description also includes dominated solutions, i.e. parts that do not belong to the optimal Pareto front. For this reason, we need to continue sampling until 5000 non-dominated solutions are found instead of merely sampling 5000 times. Finally, because we do not have a parametric description for the optimal Pareto front for BD_2^{s}, we used a different approach here. Values for f_0 were uniformly sampled, i.e. t^2 for $t \in [0;1]$, after which a single-objective optimization algorithm was used to minimize $f_1(\boldsymbol{x}) + p(\boldsymbol{x})$ where $p(\boldsymbol{x})$ is the difference of the sampled value for f_0 and $f_0(\boldsymbol{x})$ if that difference is positive, and 100 otherwise, i.e. if $t^2 > f_0(\boldsymbol{x})$ then $p(\boldsymbol{x}) = t^2 - f_0(\boldsymbol{x})$, otherwise $p(\boldsymbol{x}) = 100$. The single-objective optimization algorithm used is AMaLGaM [47] and is capable of reliably solving this problem with a very high precision. On problems such as the sphere function (i.e. f_0 in $\mathrm{BD}_2^{\mathrm{s}}$) and Rosenbrock's function (i.e. f_1 in $\mathrm{BD}_2^{\mathrm{s}}$), approximations of the optimum are easily achieved within a precision of 10^{-30}.

For the problems in our test-suite, given the ranges of the objectives for the optimal Pareto front configurations, a value of 0.01 for the $\boldsymbol{D}_{\mathcal{P}_F \to \mathcal{S}}$ indicator corresponds to fronts that are quite close to the optimal Pareto front. Fronts that have a $\boldsymbol{D}_{\mathcal{P}_F \to \mathcal{S}}$ value of 0.01 are presented in Figure 7 for all EC_i and BD_i problems.

B. Random restart gradient-based optimization

1) Approach: Traditional single-point gradient-based optimization algorithms use a random-restart scheme. That is, a random point is generated and gradient-based optimization is used to improve the point. When the search terminates, a new point is randomly generated and the search is restarted from that point. We used this traditional scheme in combination with each of the three different ways of exploiting gradient information described in Section IV-C, i.e. ROCG, AORL and CORL.

We assume a black-box setting in which we do not know the objective functions and therefore compute gradient information using finite differences, i.e. we approximated Equation 5 using a small value for h. Specifically, we used $h = 10^{-13}$. Note that using finite differences has numerical drawbacks and requires $l + 1$ evaluations to approximate the gradient at a single point.

We allowed the conjugate-gradients algorithm in ROCG to run for at most 10 iterations each time it was called, i.e. for each starting point. Furthermore, we have used the Polak-Ribiere variant of the conjugate-gradients algorithm [1]. One iteration terminates only if 1) the conjugate-gradient computations return a zero gradient, 2) no improvements could be found anymore or 3) if the overall maximum number of evaluations happens to have been reached. There are no further bounds on precision. Similarly, we allowed for at most 10 consecutive line-searches in AORL and CORL. For AORL we additionally terminated the algorithm if after a single line search the solution has worsened, i.e. become dominated. The changes in the solution resulting from the last line-search are then rejected.

2) Results: For a first impression of the performance of the three different gradient-based optimization algorithms, for each problem and each algorithm, we sample solutions randomly in the domain and then start the algorithm from there. The results on both the convex and the concave GenMED problems with 2 and 3 objectives are shown in Figure 5. Because the GenMED problems are relatively simple, all algorithms are capable of finding points on the Pareto front. However, only CORL is able to find points other than

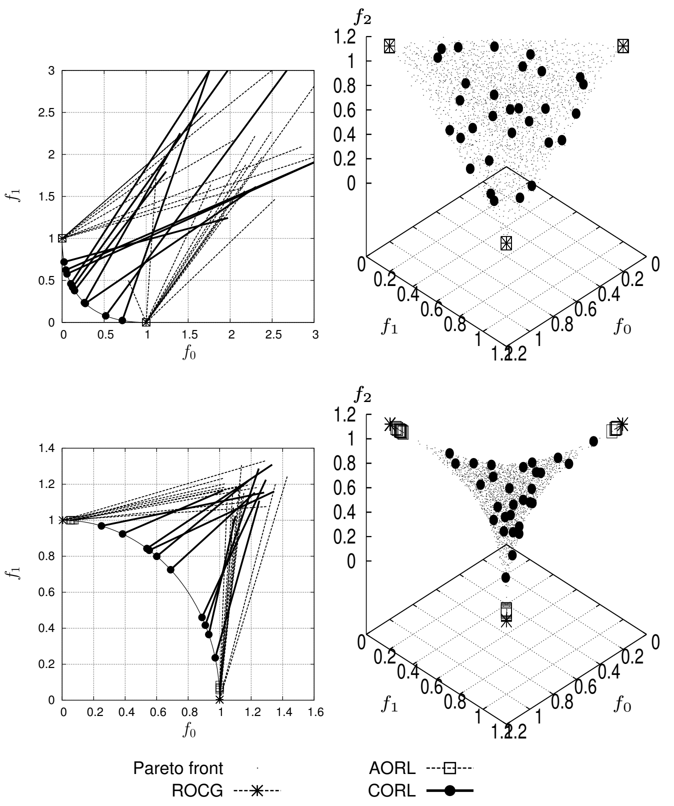

Fig. 5. Application of the ROCG, AORL and CORL gradient-based optimization algorithms to the convex (*top row*) and concave (*bottom row*) GenMED problems with two objectives (*left column*) and three objectives (*right column*). Starting points (10 for two objectives and 30 for three objectives) are randomly chosen in $[-1, 1]^l$, $l = 10$. Improvement lines are only shown for the two-objective case.

the extremes. This holds both for the 2-dimensional and 3-dimensional case. Because CORL considers all possible directions of improvement and randomly picks one, restarting CORL can find points across the entire optimal Pareto front of GenMED, regardless of whether it is convex or concave. In Figure 8 the convergence of the method in terms of the $\boldsymbol{D}_{\mathcal{P}_F \to \mathcal{S}}$ metric is shown, averaged over 100 independent runs. Indeed, of all gradient-based optimization algorithms tested, only CORL is able to find a satisfactory approximation of the Pareto front.

Although the benefits of CORL compared to the other gradient-based optimization algorithms are overly clear for the GenMED problems, the nice and smooth properties of GenMED are not likely to hold in general. The EC_4 problem for instance has local Pareto fronts. This translates to local optima also in the function to be optimized when performing line search in CORL, i.e. using Brent's method, as shown in Figure 6. These local optima correspond to points where the Kuhn-Tucker conditions (i.e. conditions under which a point is a (Pareto-)local optimum) hold. It is known that these conditions hold exclusively for the globally Pareto-optimal points only if the problem is convex or concave [48], [49]. One of the reasons local optima are typically difficult to cope with for gradient-based optimization algorithms is because line-search algorithms such as Brent's method are not well-equipped to deal with local optima. An important question now

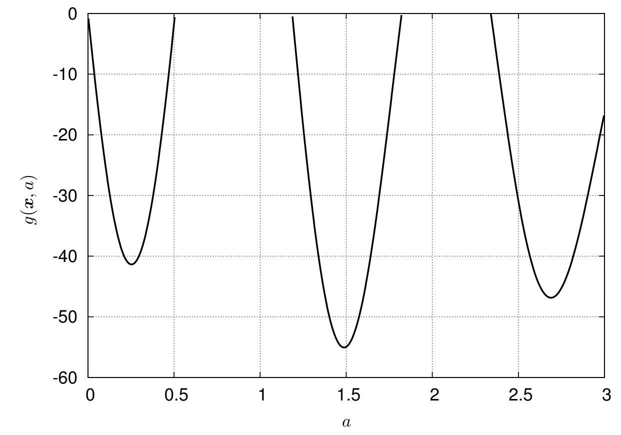

Fig. 6. Local optima in the $g(\boldsymbol{x}, a)$ function on the EC$_4$ problem for a single search direction.

is whether under increased problem difficulty, such as local fronts, the application of CORL remains equally beneficial compared to ROCG and AORL and whether gradient-based optimization algorithms are still valuable to use at all.

Figure 7 shows results for the application of all gradient-based optimization algorithms to random starting points on all other optimization problems. From these results it becomes immediately clear that the performance of all gradient-based optimization algorithms on the remaining problems is far inferior compared to the performance on the GenMED problems. None of the algorithms is able to find points on the Pareto Front except on BD$_2^{\text{s}}$, but there the distribution across the Pareto front is nowhere near uniform. It is clear that none of the algorithms in a random-restart fashion is capable of finding a satisfactory approximation to the Pareto front with the maximum number of evaluations that was allowed.

Regarding relative performance of the various gradient-based optimization algorithms, CORL is clearly still capable of finding improvements in various directions in the objective space whereas the diversity in directions found by ROCG and AORL is limited. However, the overall performance in terms of uniform convergence onto the Pareto front is virtually indistinguishable on EC$_i$, $i \in \{1, 2, 3, 4, 6\}$ as can be seen in Figure 8 where convergence of the $\boldsymbol{D}_{\mathcal{P}_F \to \mathcal{S}}$ metric is shown. Moreover, because of the major difference in difficulty and scale between the two objectives in BD$_1$, CORL is outperformed by the ROCG and AORL methods that have the ability to explicitly focus on a single objective. This can also clearly be seen in Figure 8. Note that for the convergence on the GenMED problems, we chose to only focus on 2 objectives. The reason for this is that from Figure 5 it is clear that the methods work similarly for 2 and 3 objectives and that for 2 objectives the $\boldsymbol{D}_{\mathcal{P}_F \to \mathcal{S}}$ is easier to compute.

From our results, we conclude that the use of gradient-based search algorithms alone is not likely to provide a satisfactory approximation of the optimal Pareto front. Also, even though the various algorithms find improvements along different directions in the objective space, their overall performance in terms of convergence to the Pareto front is similar unless the problem at hand has smoothness properties the likes of GenMED problems. Still, all algorithms are capable of finding improvements in a local manner, i.e. moving a single solution towards the nearest suboptimum. Therefore, the combination of such local-search properties with the more global-search properties of evolutionary algorithms may still result in an application of gradient-based optimization algorithms that is beneficial. For this reason we look at such a hybridization next.

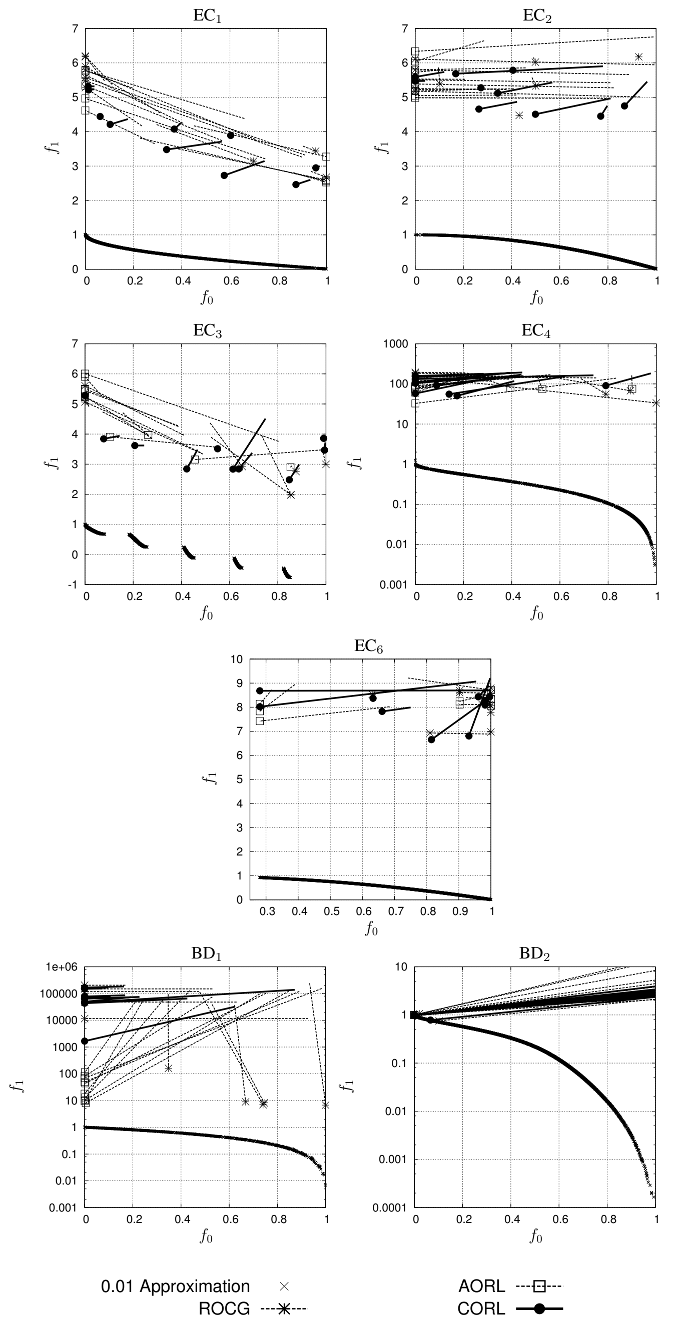

Fig. 7. Application of the ROCG, AORL and CORL gradient-based optimization algorithms to the EC$_i$, $i \in \{1, 2, 3, 4, 6\}$ and BD$_i$, $i \in \{1, 2\}$ problems. Starting points are randomly chosen. Also shown are approximation sets with a $\boldsymbol{D}_{\mathcal{P}_F \to \mathcal{S}}$ value of 0.01.

C. Hybrid evolutionary optimization

1) Approach: While optimization of some problems may benefit from the use of gradient information, for other problems the additional cost required to calculate gradients may not weigh up to the benefits. Given a black-box scenario, it is impossible to determine beforehand whether the additional use of gradient information will be advantageous. Even if we did have a notion of such a possible advantage, it is not clear whether the use of gradient-based optimization algorithms will be advantageous during the course of running the MOEA. In other words, a fixed ratio of the number of evaluations used by the MOEA compared to the number of evaluations used by the gradient-based optimization algorithms is generally not optimal. If one gradient-based optimization algorithm is at a certain point during optimization clearly superior to another one for a particular problem, it is more efficient to allow the superior algorithm to spend more search effort. An intuitively more favorable integration of gradient-based search algorithms with a MOEA is thus one in which the most effective gradient-based optimization algorithm is assigned the largest probability. Or, if no gradient-based optimization algorithm is efficient compared to the base MOEA, to reduce the use of the gradient-based optimization algorithms to a minimum. This amounts to the adaptive allocation of resources [50].

To perform resource allocation adaptively, the utility of using gradient-based search algorithms will have to be determined online, i.e. during the course of a run of the MOEA. If the true utilities of the gradient-based optimization algorithms can be determined properly, this information can be used to vary the probabilities of applying the gradient-based optimization algorithms in the course of running the MOEA.

Adaptive resource allocation has several advantages in addition to the potential improvements in efficiency. First, a practitioner is relieved of the need to select and tune the different probabilities. Second, by leaving the choice of the probabilities to the optimization algorithm, a large part of the optimization task is automated. Third, adapting the probabilities of the operators can render optimization methods more robust, as unfavorable choices of parameters can be corrected during the course of the run.

Here we use an adaptive resource-allocation scheme from the literature that was previously shown to be very effective [30]. To get a better idea of the added performance of gradient-based optimization algorithms, we combine the scheme with a recently introduced MOEA [51] that is more efficient than the one used in the work that introduced the scheme. For an in-depth description of the scheme and the base MOEA we refer the interested reader to the indicated literature. In the remainder of this section we shall provide only brief descriptions.

a) Adaptive resource allocation scheme: In every generation of the MOEA, the utility of each algorithm is estimated anew. The algorithms considered are the base MOEA itself and the three different gradient-based search algorithms, i.e. ROCG, AORL and CORL. Because the three gradient-based search algorithms all exploit gradient information differently and the adaptive resource allocation scheme will determine for itself which operator is the most beneficial to use at which point, we consider all three gradient-based optimization algorithms at the same time.

To determine the utilities, the number of improvements is counted that each algorithm was able to obtain in the last generation. For the base MOEA, i.e. the variation operator, improvements are offspring solutions that are not dominated by any solution in the set of selected solutions. For the gradient-based optimization algorithms improvements are solutions that after running the algorithm resulted in a solution that is not dominated by any solution in the population. This notion of improvement is not strict in the sense that we it is not required that new solutions must also dominate the solution(s) they were created from. This allows the search to perform "sideway" steps in addition to "forward" or "domination" steps, thereby stimulating the search *along* the Pareto front to obtain a diverse front during the search and also to ensure that the front may be expanded sideways once the search gets near the optimal Pareto front. It should be noted however that such sideway steps cannot be obtained by CORL. The proposed use of Brent's method only considers points that dominate the starting point. Non-dominating points *can* be achieved by ROCG and AORL. Moreover, given that the latter two methods are not explicitly designed to find dominating solutions, allowing them to also generate non-dominated solutions greatly increases their rate of successful application.

The utility of an algorithm is obtained by dividing the number of improvements by the number of evaluations that were required by each algorithm to obtain those improvements. The total number of evaluations is then proportionally redistributed among the algorithms for use in the next generation.

b) Base MOEA: The base MOEA we use is the SDR-AVS-$\mathbb{MIDEA}$ [51]. This MOEA is an Estimation-of-Distribution Algorithm (EDA [52]) specifically designed for multi-objective optimization.

In SDR-AVS-$\mathbb{MIDEA}$, a population of size n is maintained. In each generation, a subset of this population of size $\lfloor \tau n \rfloor$, $\tau \in [\frac{1}{n}; 1[$, is selected to perform variation with. By means of variation $n - \lfloor \tau n \rfloor$ new solutions are generated which replace the solutions in the population that were not selected.

Selection is performed using a diversity-preserving selection operator. Since the goal in multi-objective optimization is both to get close to the optimal Pareto front and to get a good diverse representation of that front, a good selection operator must exert selection pressure with respect to both aspects. The selection operator in the SDR-AVS-$\mathbb{MIDEA}$ does this by using truncation selection on the basis of domination count (i.e. the number of times a solution is dominated). If the number of non-dominated solutions exceeds the targeted selection size $\lfloor \tau n \rfloor$, a nearest-neighbour heuristic in the objective space is used to ensure that a well-spread, representative subset of all non-dominated solutions is chosen.

The variation operator is geometrical in nature and is specifically designed to provide an advantage in multi-objective optimization compared to traditional variation operators. The selected solutions are first clustered in the objective space. Subsequently, the actual variation takes place only between individuals in the same cluster, i.e. a mating restriction is employed. The rationale is that variation inside each cluster can process specific information about the different regions along the Pareto front. Such a parallel exploration automatically gives a better probability of obtaining a well-spread set of offspring solutions. To further stimulate diversity along the Pareto front each new offspring solution is constructed from a randomly chosen cluster. New solutions are generated according to the EDA principle, i.e. the estimation of a probability distribution and the subsequent re-sampling of new solutions from this estimated distribution. In each separate cluster a Bayesian-factorized normal distribution is estimated. The estimated covariance matrix of each normal distribution is subsequently separately adaptively scaled to prevent premature convergence. Specifically, this means that if improvements are found more than one standard deviation away from the estimated mean of the distribution, then the covariance matrix of the estimated distribution is scaled up to increase the area of exploration. If, however, the improvements are obtained near the mean of the estimated distribution, then the covariance matrix is scaled down. This mechanism of preventing premature convergence is the main difference with the base MOEA used in the experiments of the work that introduced the adaptive resource allocation scheme [30].

As in [51], an elitist archive is maintained that is updated in a fashion similar to ε-dominance [53]. Without a technique such as ε-dominance archiving, true convergence to the Pareto-optimal front may not occur. As soon as selection based on diversity is required to prune non-dominated solutions because there are too many of them in the population, it is possible that, over multiple generations, solutions end up in the population that are dominated by solutions that were pruned earlier. Hence, only maintaining the best solutions of the current generation doesn't lead to true elitism. The same is true for the use of the gradient-based optimization algorithms because we allow sideway steps to be performed by ROCG and AORL. As a result of sideway steps, a solution $\boldsymbol{x}^0$ may be changed into another solution $\boldsymbol{x}^1$ that doesn't dominate $\boldsymbol{x}^0$. Starting a new gradient-based algorithm from that solution however may then result in a solution $\boldsymbol{x}^2$ that doesn't dominate $\boldsymbol{x}^1$, but because no direct comparison is made to $\boldsymbol{x}^0$ it could be the case that $\boldsymbol{x}^0$ dominates $\boldsymbol{x}^2$, meaning we end up with a worse solution than the one from which we started. Without the use of an elitist archive, $\boldsymbol{x}^0$ would have been lost and the outcome of the algorithm could indeed be worse if it is run longer.

The settings of the parameters in SDR-AVS-$\mathbb{MIDEA}$ are based on the guidelines reported in [47] and the best results reported in [51]. The percentile for truncation selection set to $\tau = 0.3$ and $k = 10$ clusters. The cluster size is set to 50, making the overall population size $n = 500$. The variance multiplier decreaser of the adaptive variance scaling mechanism equals 0.9 and the standard-deviation ratio threshold is set to 1.0. For the elitist archive, the objective space is discretized in each objective with a discretization length of 10^{-3}. This provides sufficient granularity for the final Pareto-front approximations. In our experiments, we set the maximum number of evaluations to $1 \cdot 10^6$, where one evaluation involves computing the values of both objectives.

2) Results: Convergence graphs of the $\boldsymbol{D}_{\boldsymbol{\mathcal{P}_F} \rightarrow \mathcal{S}}$ metric for all tested algorithms and all problems are presented in Figure 8. From this Figure, the power of MOEAs is immediately clear when comparing the convergence results of the MOEA and the hybrid MOEA with the convergence results of the individual random-restart gradient-based optimization algorithms. All gradient-based optimization algorithms have a probability of 0 of reaching the target value of 0.01 for the $\boldsymbol{D}_{\boldsymbol{\mathcal{P}_F} \rightarrow \mathcal{S}}$ metric, with the exception of CORL on the GenMED problems, for which the probability is 1. Both the MOEA and the hybrid MOEA on the other hand reach the target vale of 0.01 for the $\boldsymbol{D}_{\boldsymbol{\mathcal{P}_F} \rightarrow \mathcal{S}}$ metric on all problems with probability 1, with the exception of problem EC_4, for which both the base MOEA and the hybrid MOEA have a probability of 0 of reaching the target value within the predefined budget of evaluations. Although the target value of 0.01 was never obtained within the maximum number of evaluations, the convergence graph indicates the hybrid MOEA however continues to improve the Pareto front at the function-evaluation limit, whereas the pure MOEA clearly converges prematurely in some runs. Overall, in the long run the performance of the hybrid MOEA is never worse than the performance of the pure MOEA on the problems in our test suite. However, it may take many evaluations before this added advantage becomes clear.

In most cases, the adaptive resource allocation scheme is capable of detecting when it is not fruitful to use gradient-based optimization algorithms. Also, the adaptive resource allocation scheme exploits the gradient-search optimization algorithms to obtain faster convergence on two other problems, i.e. on EC_4 and BD_2^{s}. However, on EC_6 and BD_1, a slower convergence is obtained compared to when only the MOEA is used. In the work in which the adaptive resource scheme was proposed [30], this situation was not encountered. One reason for this is that the base MOEA used here is more efficient. Another reason is that the *length* of the improvements is not taken into account. Although the gradient-based operators may be able to obtain improvements, they may be small, leading to an over-estimate of the number of times a gradient-based operator should be applied.

A more detailed image of the resource division among the different gradient-based optimization algorithms is depicted in Figure 9. The number of evaluations that the adaptive resource allocation scheme allows each of the gradient-based optimization algorithms to use, is different for each problem. Overall, CORL can be said to be the most useful as it is always among the most frequently used algorithms. On the GenMED problems and BD_2^{s} it is even used substantially more often. With the exception of the GenMED problems and AORL on the BD_2^{s} problem and ROCG on the BD_1 problem, *all* gradient-based optimization algorithms are found to be quite equally beneficial to use by the adaptive resource allocation scheme. This supports the use of a portfolio of local search algorithms, in this case gradient-based optimization algorithms, together with an adaptive resource allocation scheme for multi-objective optimization. It also underlines the difficulty in multi-objective gradient exploitation for optimization when searching for a good approximation of the optimal Pareto front.

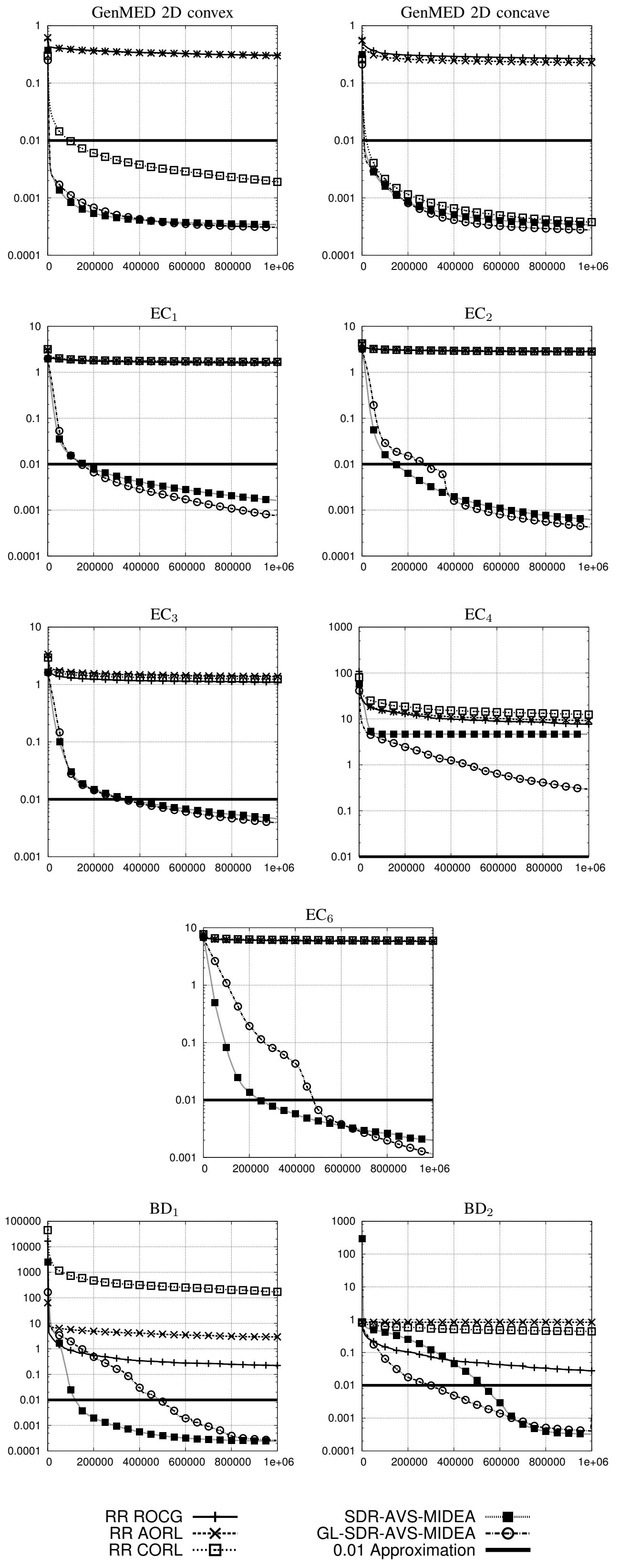

Fig. 8. Convergence of the $D_{\mathcal{P}_F \rightarrow \mathcal{S}}$ metric for all tested algorithms and all problems, averaged over 100 independent runs. Horizontal axis: number of evaluations (both objectives per evaluation). Vertical axis: $D_{\mathcal{P}_F \rightarrow \mathcal{S}}$.

There are multiple ways in which gradient information can be exploited and our experiments indicate that it is a priori not clear which way is going to be the most rewarding.

VII. Discussion

In single-objective optimization, direct use of the gradient is representative of only the simplest approaches, e.g. gradient descent. More efficient and more advanced algorithms that use second order gradient information or combinations of gradients have long been known. Gradients for multi-objective optimization have a much shorter history. It is for this reason that only recently first steps have been reported in the literature of going beyond direct use of gradient information only. For instance, an algorithm that uses second-order derivative information in multi-objective optimization was recently studied by Shukla [54], both separately and in combination with EAs. Similarly, Fliege, Graña Drummond and Svaiter [55] proposed an algorithm to exploit second-order information in multi-objective optimization. This algorithm was subsequently used to hybridize MOEAs with by Koch *et al* [56]. Promising results are reported, showing improvements over the use of the MOEA alone on all tested problems. However, in that study, gradient information was determined analytically instead of using finite differences as is done in the work presented here. The use of analytical gradients substantially reduces the number of required function evaluations, making the use of gradient-based optimization algorithms much more efficient if only the number of function evaluations is counted.

The work presented here contributes to the understanding of gradient information in multi-objective optimization by considering not a single optimal direction as the gradient, but by considering *all* optimal improving directions simultaneously. This paves the way for different interpretations of advanced uses of gradient information such as second-order gradients. This, in turn, may lead to new algorithms and an even more in-depth understanding of numerical multi-objective optimization problems. One interesting and important direction of research along this line is to study the notion of conjugated gradients for the multi-objective case, either taking a single optimal improving direction to be the gradient, or considering all optimal improving directions simultaneously, i.e. based on the equations provided here.

VIII. Conclusion

We have presented a parameterized, analytical description of the set of all non-dominated improving directions for any point in the parameter space of a multi-objective optimization problem. This description and its derivation provides insights into the structure of the multi-objective gradient as well as a solid basis for exploiting gradient information in numerical multi-objective optimization. We have used this description in a gradient-based optimization algorithm that we named CORL (Combined-Objectives Repeated Line search). We have investigated the use of CORL and two other gradient-based optimization algorithms for numerical multi-objective optimization separately and in combination with a MOEA.

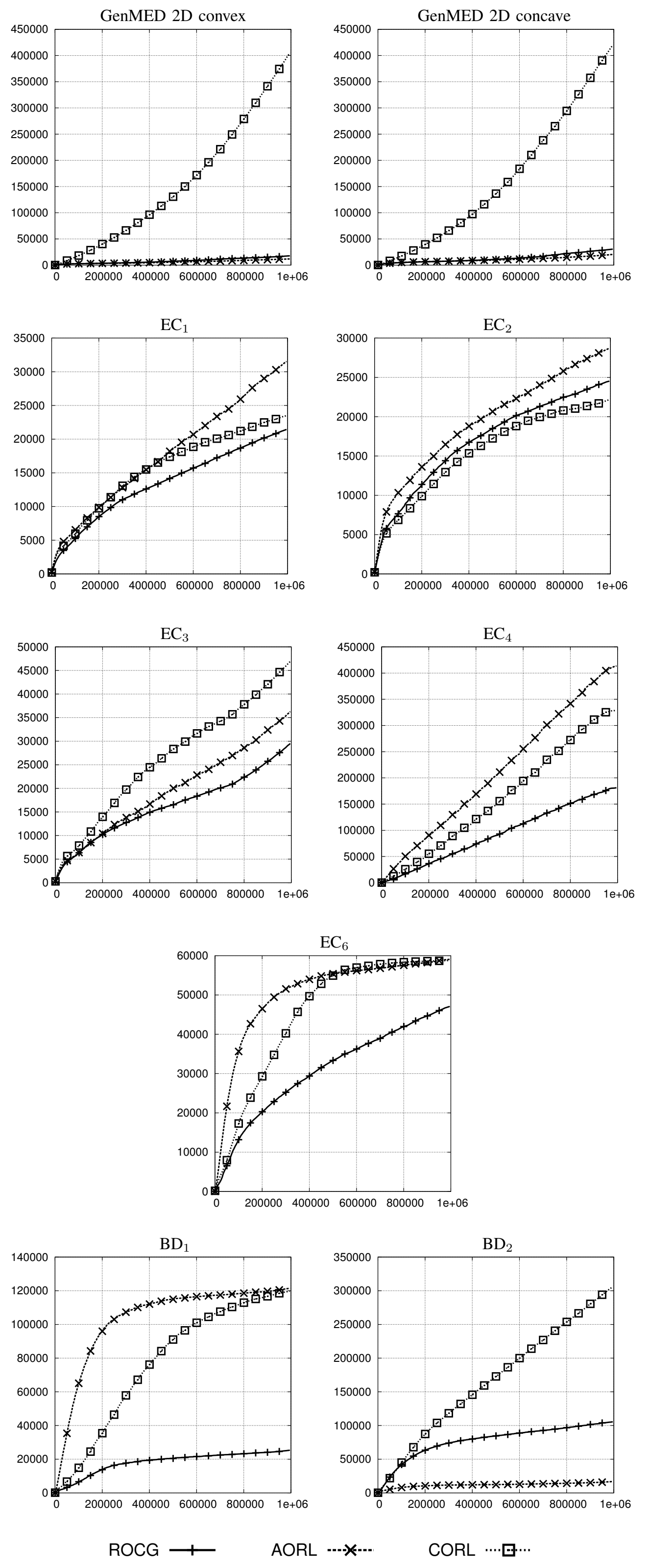

Fig. 9. Number of evaluations used during a run by each gradient-based optimization algorithm within the hybrid MOEA, averaged over 100 independent runs. Horizontal axis: total number of evaluations (both objectives per evaluation). Vertical axis: number of evaluations used by a gradient-based optimization algorithm.

Although CORL considers all optimal improving directions, the direct application of CORL has upon application not always been found to be superior. Superior performance has only been observed on smooth functions, analogous to the case of unimodal smooth functions in single-objective optimization. In that case, upon repeated application, CORL was able to find a spread-out approximation of the optimal Pareto front, making the true practical value of CORL problem-specific.

The added use of gradient-based optimization algorithms in a MOEA is only useful if the relative contributions made to the overall improvement are at least as big as the relative amount of resources it is allowed to spend. For multi-objective optimization this criterion is harder to achieve because MOEAs have the ability to advance multiple solutions simultaneously towards different regions of the optimal Pareto front through variation, giving it a bigger relative advantage than in the single-objective case. For this reason, we believe a good adaptive resource allocation mechanism for hybridization is very important in multi-objective optimization. Given such a mechanism, we found that gradient-based optimization algorithms can indeed provide improvements compared to using a non-hybrid MOEA, even if the gradients are estimated using costly finite-difference approximations.

Acknowledgment

The author would like to thank Edwin D. de Jong for discussion of an early draft of this article and prior collaboration on two conference papers that led up to this work.

References

[1] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipies in C: The Art of Scientific Computing*. Cambridge, Massachusetts: Cambridge University Press, 1992.

[2] M. R. Hestenes and E. Stiefel, "Methods of conjugate gradients for solving linear systems," *Journal of Research of the National Bureau of Standards*, vol. 6, no. 49, pp. 409–436, 1952.

[3] T. M. Mitchell, *Machine Learning*. New York, New York: McGraw-Hill, 1997.

[4] C. M. Bishop, *Pattern Recognition and Machine Learning*. Berlin: Springer–Verlag, 2006.

[5] M. C. Fu, "Gradient estimation," in *Handbooks on Operations Research and Management Science: Simulation*, S. Henderson and B. Nelson, Eds. Berlin: Elsevier, 2006, pp. 575–616.

[6] J. Knowles, D. Corne, and K. Deb, *Multiobjective Problem Solving from Nature. From Concepts to Applications*. Berlin: Springer–Verlag, 2008.

[7] C. A. C. Coello, G. B. Lamont, and D. A. V. Veldhuizen, *Evolutionary Algorithms for Solving Multi–Objective Problems*. Berlin: Springer–Verlag, 2007.

[8] C. Grosan, A. Abraham, and H. Ishibuchi, *Hybrid Evolutionary Algorithms*. Berlin: Springer-Verlag, 2007.

[9] N. Krasnogor, W. Hart, and J. Smith, *Recent Advances in Memetic Algorithms and Related Search Technologies*. Berlin: Springer-Verlag, 2004.

[10] N. Krasnogor and J. Smith, "A tutorial for competent memetic algorithms: Model, taxonomy, and design issues," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 5, pp. 474–488, 2005.

[11] P. Merz and B. Freisleben, "Fitness landscape analysis and memetic algorithms for the quadratic assignment problem," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 4, pp. 337–352, 2000.

[12] M. Lahanas, D. Baltas, and S. Giannouli, "Global convergence analysis of fast multiobjective gradient based dose optimization algorithms for high–dose–rate brachytherapy," *Physics in Medicine and Biology*, vol. 48, no. 5, pp. 599–617, 2003.

[13] C. K. Goh, Y. S. Ong, and K. C. Tan, "An investigation on evolutionary gradient search for multi-objective optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation — CEC–2008*. Piscataway, New Jersey: IEEE Press, 2008, pp. 3741–3746.

[14] M. Emmerich, A. Deutz, and N. Beume, "Gradient–based/evolutionary relay hybrid for computing Pareto front approximations maximizing the S–metric," in *Hybrid Metaheuristics — HB–2007*, T. Bartz-Beielstein *et al.*, Eds. Berlin: Springer–Verlag, 2007, pp. 140–156.

[15] E. Zitzler and L. Thiele, "Multiobjective optimization using evolutionary algorithms — a comparative case study," in *Parallel Problem Solving from Nature — PPSN V*, A. E. Eiben *et al.*, Eds. Berlin: Springer–Verlag, 1998, pp. 292–301.

[16] J. Fliege and B. F. Svaiter, "Steepest descent methods for multicriteria optimization," *Mathematical Methods of Operations Research*, vol. 51, no. 3, pp. 479–494, 2000.

[17] H. Mukai, "Algorithms for multicriterion optimization," *IEEE Transactions on Automatic Control*, vol. 25, no. 2, pp. 177–186, 1980.

[18] S. Schäffler, R. Schultz, and K. Weinzierl, "Stochastic method for the solution of unconstrained vector optimization problems," *J. of Optim. Theory and Applications*, vol. 114, no. 1, pp. 209–222, 2002.

[19] K. Harada, J. Sakuma, and S. Kobayashi, "Local search for multiobjective function optimization: Pareto descend method," in *Proc. of the Genetic and Evolutionary Comp. Conf. — GECCO–2006*, M. Keijzer *et al.*, Eds. New York, New York: ACM Press, 2006, pp. 659–666.

[20] E. L. Allgower and K. Georg, *Numerical Continuation Methods: An Introduction*. Berlin: Springer–Verlag, 1990.

[21] C. Hillermeier, "Generalized homotopy approach to multiobjective optimization," *Journal of Optimization Theory and Applications*, vol. 110, no. 3, pp. 557–583, 2001.

[22] O. Schütze, A. Dell'Aere, and M. Dellnitz, "On continuation methods for the numerical treatment of multi-objective optimization problems," in *Practical Approaches to Multi-Objective Optimization*, J. Branke *et al.*, Eds. Schloss Dagstuhl, Germany: Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), 2005.

[23] J. D. Knowles and D. Corne, "M-PAES: a memetic algorithm for multiobjective optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation — CEC–2000*. Piscataway, New Jersey: IEEE Press, 2002, pp. 325–332.

[24] M. Brown and R. E. Smith, "Directed multi-objective optimization," *The International Journal of Computers, Systems and Signals*, vol. 6, no. 1, pp. 3–17, 2005.

[25] M. Dellnitz, O. Schütze, and T. Hestermeyer, "Covering Pareto sets by multilevel subdivision techniques," *Journal of Optimization Theory and Applications*, vol. 124, no. 1, pp. 113–136, 2005.

[26] P. K. Shukla, "Gradient based stochastic mutation operators in evolutionary multi–objective optimization," in *Proc. of the international conf. on Adaptive and Natural Computing Algorithms — ICANNGA–2007*, B. Beliczynski *et al.*, Eds. Berlin: Springer–Verlag, 2007, pp. 58–66.

[27] K. Harada, J. Sakuma, S. Kobayashi, and I. Ono, "Uniform sampling of local Pareto-optimal solution curves by Pareto path following and its applications in multi-objective GA," in *Proceedings of the Genetic and Evolutionary Computation Conference — GECCO–2007*, D. Thierens *et al.*, Eds. New York, New York: ACM Press, 2007, pp. 813–820.

[28] O. Schütze, C. A. C. Coello, S. Mostaghim, E.-G. Talbi, and M. Dellnitz, "Hybridizing evolutionary strategies with continuation methods for solving multi-objective problems," *Engineering Optimization*, vol. 40, no. 5, pp. 383–402, 2008.

[29] A. Lara, G. Sanchez, C. A. C. Coello, and O. Schütze, "HCS: A new local search strategy for memetic multiobjective evolutionary algorithms," *IEEE Trans. on Evol. Comp.*, vol. 14, no. 1, pp. 112–132, 2010.

[30] P. A. N. Bosman and E. D. de Jong, "Combining gradient techniques for numerical multi–objective evolutionary optimization," in *Proc. of the Genetic and Evolutionary Comp. Conf. — GECCO–2006*, M. Keijzer *et al.*, Eds. New York, New York: ACM Press, 2006, pp. 627–634.

[31] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000.

[32] P. A. N. Bosman and E. D. de Jong, "Exploiting gradient information in numerical multi–objective evolutionary optimization," in *Proc. of the Genetic and Evolutionary Comp. Conf. — GECCO–2005*, H.-G. Beyer *et al.*, Eds. New York, New York: ACM Press, 2005, pp. 755–762.

[33] W. Kaplan, *Advanced Calculus, 4th ed.* Reading, Massachusetts: Addison–Wesley, 1991.

[34] I. C. Kampolis, D. I. Papadimitriou, and K. C. Giannakoglou, "Evolutionary optimization using a new radial basis function network and the adjoint formulation," *Inverse Problems in Science and Engineering*, vol. 14, no. 4, pp. 397–410, 2006.

[35] P. E. Gill, W. Murray, and M. H. Wright, *Practical Optimization*. New York, New York: Academic Press, 1981.

[36] R. P. Brent, *Algorithms for Minimization without Derivatives*. Englewood Cliffs, New Jersey: Prentice–Hall, 1973.

[37] T. Gal and H. Leberling, "Redundant objective functions in linear vector maximum problems and their determination," *European Journal of Operational Research*, vol. 1, no. 3, pp. 176–184, 1977.

[38] J. C. Nash, *Compact Numerical Methods for Computers: Linear Algebra and Function Minimisation*. Bristol: Adam Hilger Ltd, 1990.

[39] G. Strang, *Introduction to Linear Algebra*. Wellesley, Massachusetts: Wellesley–Cambridge Press, 1993.

[40] S. R. Lay, *Convex Sets and their Applications*. New York, New York: John Wiley & Sons Inc., 1979.

[41] L. Devroye, *Non–Uniform Random Variate Generation*. Berlin: Springer–Verlag, 1986.

[42] K. Deb, "Multi-objective genetic algorithms: Problem difficulties and construction of test problems," *Evolutionary Computation*, vol. 7, no. 3, pp. 205–230, 1999.

[43] P. A. N. Bosman and D. Thierens, "The balance between proximity and diversity in multiobjective evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 174–188, 2003.

[44] J. D. Knowles and D. Corne, "On metrics for comparing non–dominated sets," in *Proceedings of the IEEE Congress on Evolutionary Computation — CEC–2002*, D. B. Fogel *et al.*, Eds. Piscataway, New Jersey: IEEE Press, 2002, pp. 666–674.

[45] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, 2003.

[46] D. A. V. Veldhuizen, "Multiobjective evolutionary algorithms: Classifications, analyses, and new innovations," 1999, phD thesis, Graduate School of Engineering of the Air Force Institute of Technology.

[47] P. A. N. Bosman, J. Grahl, and D. Thierens, "Enhancing the performance of maximum–likelihood Gaussian edas using anticipated mean shift," in *Parallel Problem Solving from Nature — PPSN X*, G. Rudolph *et al.*, Eds. Berlin: Springer–Verlag, 2008, pp. 133–134.

[48] H. W. Kuhn and A. W. Tucker, "Nonlinear programming," in *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, J. Neyman, Ed. Berkeley, California: University of California Press, 1951, pp. 481–492.

[49] K. Miettinen, *Nonlinear Multiobjective Optimization*. Boston, Massachusetts: Kluwer Academic, 1999.

[50] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, Massachusetts: MIT Press, 1998.

[51] P. A. N. Bosman and D. Thierens, "Adaptive variance scaling in continuous multi–objective estimation–of–distribution algorithms," in *Proceedings of the Genetic and Evolutionary Computation Conference — GECCO–2007*, D. Thierens *et al.*, Eds. New York, New York: ACM Press, 2007, pp. 500–507.

[52] M. Pelikan, K. Sastry, and E. Cantú-Paz, *Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications*. Berlin: Springer–Verlag, 2006.

[53] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, "Combining convergence and diversity in evolutionary multi-objective optimization," *Evolutionary Computation*, vol. 10, no. 3, pp. 263–282, 2002.

[54] P. K. Shukla, "Exploiting second order information in computational multi-objective evolutionary optimization," in *Progress in Artificial Intelligence - Proceedings of the 13th Portuguese Conference on Aritficial Intelligence — EPIA–2007*, J. Neves *et al.*, Eds. Berlin: Springer–Verlag, 2007, pp. 271–282.

[55] J. Fliege, J. M. Graña Drummond, and B. F. Svaiter, "Newton's method for multiobjective optimization," *SIAM Journal on Optimization*, vol. 20, no. 2, pp. 602–626, 2009.

[56] P. Koch, O. Kramer, G. Rudolph, and N. Beume, "On the hybridization of SMS-EMOA and local search for continuous multiobjective optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference — GECCO–2009*, G. Raidl *et al.*, Eds. New York, New York: ACM Press, 2009, pp. 603–610.

Peter A.N. Bosman received his Ph.D. in computer science on the topic of estimation-of-distribution algorithms (EDAs) in 2003 from the Utrecht University in the Netherlands. He is currently a scientific staff member at Centrum Wiskunde & Informatica (CWI, Centre for Mathematics and Computer Science) in Amsterdam, the Netherlands. His main research interests are in the field of computational intelligence and specifically include evolutionary computation (particularly EDAs) and statistical learning.