The Directed Search Method for Multi-Objective Memetic Algorithms

Oliver Schütze[1], Adanay Martín[1], Adriana Lara[2], Sergio Alvarado[1], Eduardo Salinas[1], and Carlos A. Coello Coello[1]

[1]Cinvestav-IPN, Computer Science Department
Av. IPN 2508, C. P. 07360, Col. San Pedro Zacatenco
Mexico City, Mexico
{schuetze,ccoello}@cs.cinvestav.mx,
{amartin,salvarado,esalinas}@computacion.cs.cinvestav.mx

[2]ESFM-IPN, Mathematics Department
Edificio 9, UPALM, Mexico City, Mexico
adriana@esfm.ipn.mx

Abstract. We propose a new iterative search procedure for the numerical treatment of unconstrained multi-objective optimization problems (MOPs) which steers the search along a predefined direction given in objective space. Based on this idea we will present two methods: Directed Search (DS) Descent which seeks for improvements of the given model, and a novel continuation method (DS Continuation) which allows to search along the Pareto set of a given MOP. One advantage of both methods is that they can be realized with and without gradient information, and if neighborhood information is available the computation of the search direction comes even for free. The latter makes our algorithms interesting candidates for local search engines within memetic strategies. Further, the approach can be used to gain some interesting insights into the nature of multi-objective stochastic local search which may explain one facet of the success of multi-objective evolutionary algorithms (MOEAs). Finally, we demonstrate the strength of the method both as standalone algorithm and as local search engine within a MOEA.

Keywords: multi-objective optimization, continuation, memetic algorithm, stochastic local search

1 Introduction

In many applications one is faced with the problem that several objectives have to be optimized concurrently leading to a *multi-objective optimization problem* (MOP). Multi-objective evolutionary algorithms (MOEAs) have proven to be very effective on the treatment of such problems. Reasons for this include that algorithms of this kind are very robust, do not require hard assumptions on

the model, and allow to compute a finite size representation of the solution set, the Pareto set, in one single run ([1–3]). On the other hand, it is known that MOEAs need quite a few function evaluations in order to evolve to a suitable approximation of the set of interest. As one way out, researchers have proposed *memetic algorithms* that hybridize local search techniques with MOEAs in order to obtain fast and reliable global search procedures. (e.g., [4–8]).
In this paper, we propose a new point-wise iterative search procedure that allows to steer the search into any direction d given in objective space. Since there is no restriction on d, the *Directed Search (DS) method* can be used to steer the search both toward and along the Pareto set of a given MOP. Further, the method can be used with or without gradient information. To realize the gradient free DS, the neighborhood information of the point designated for local search can be exploited so that the related search direction can ideally be computed for free in terms of additional function evaluations. The latter makes the DS an interesting candidate for integration into set based heuristics such as MOEAs since in that case the neighbors of a given individual may be utilized. In this paper, we make a first attempt to demonstrate the strength of DS within a memetic algorithm.
As a by-product, we gain some insights into the behavior of multi-objective stochastic local search (MOSLS) by using the DS approach which yields some interesting insights and may explain one facet of the huge success of evolutionary algorithms for the treatment of MOPs.
The idea to steer the search by a direction given in objective space is not new but appears in the literature in several variations (e.g., [9–13]). Most remarkably, the DS Descent Method shares some characteristics with NBI ([11]) and the method presented in [13]. The DS, however, has a broader applicability. In particular the possibility to move along the Pareto set and the gradient free realization present novelties that are beneficial for hybridization with set based heuristics. Preliminary studies of the DS can be found in [14–16]. In [14] the general idea of the DS is presented, in [15] some details of the DS Descent method are provided, and in [16] the gradient free DS Descent method is proposed together with a first memetic algorithm.

The remainder of this paper is organized as follows: in Sec. 2, we will present the background required for the understanding of the sequel. In Sec. 3, we present the basic idea of the DS as well as the gradient based descent and continuation methods. In Sec. 4, we explain some behavior of MOSLS using DS. In Sec. 5, we present the idea of the gradient free DS and present further on a gradient free continuation method. In Sec. 6, we present some numerical results of the DS both as standalone algorithm and as local searcher within a MOEA. Finally, we draw our conclusions and give paths for future research in Sec. 7.

2 Notations and Background

In the following we consider unconstrained continuous MOPs

$$\min_{x \in \mathbb{R}^n} F(x), \tag{MOP}$$

where F is defined as the vector of the objective functions $F : \mathbb{R}^n \to \mathbb{R}^k$, $F(x) = (f_1(x), \ldots, f_k(x))^T$, and where each objective $f_i : \mathbb{R}^n \to \mathbb{R}$ is (for simplicity) smooth. The optimality of a MOP is defined by the concept of *dominance* ([17]): a vector $y \in \mathbb{R}^n$ is *dominated* by a vector $x \in \mathbb{R}^n$ ($x \prec y$) with respect to (MOP) if $f_i(x) \leq f_i(y)$, $i = 1, \ldots, k$, and there exists an index j such that $f_j(x) < f_j(y)$, else y is non-dominated by x. A point $x \in \mathbb{R}^n$ is called *(Pareto) optimal* or a *Pareto point* if there is no $y \in \mathbb{R}^n$ which dominates x. The set of all Pareto optimal solutions is called the *Pareto set*, and is denoted by $\mathcal{P}$. The image $F(\mathcal{P})$ of the Pareto set is called the *Pareto front*. Both sets typically form a $(k-1)$-dimensional object ([18]).
The Jacobian of F at a point x is given by

$$J(x) = \begin{pmatrix} \nabla f_1(x)^T \\ \vdots \\ \nabla f_k(x)^T \end{pmatrix} \in \mathbb{R}^{k \times n}, \tag{1}$$

where $\nabla f_i(x)$ denotes the gradient of objective f_i. If all the objectives of the MOP are differentiable, the following famous theorem of Kuhn and Tucker [19] states a necessary condition for the Pareto optimality of unconstrained MOPs.

Theorem 1. *Let x^* be a Pareto point of (MOP), then there exists a vector $\alpha \in \mathbb{R}^k$ with $\alpha_i \geq 0, i = 1, \ldots, k$, and $\sum_{i=1}^k \alpha_i = 1$ such that*

$$\sum\nolimits_{i=1}^k \alpha_i \nabla f_i(x^*) = J(x)^T \alpha = 0. \tag{2}$$

Points satisfying (2) are called *Karush–Kuhn–Tucker (KKT) points.*

3 The Directed Search Method

Here we first describe the idea of the Directed Search method and will further on derive a descent and a continuation method out of it.

3.1 Central Idea

Assume a point $x_0 \in \mathbb{R}^n$ with $\operatorname{rank}(J(x_0)) = k$ is given and a vector $d \in \mathbb{R}^k$ representing a desired search direction in objective space. Then, a search direction $\nu \in \mathbb{R}^n$ in decision space is sought such that for $y_0 := x_0 + h\nu$, where $h \in \mathbb{R}_+$ is the step size (i.e., y_0 represents a movement from x_0 in direction ν), it holds:

$$\lim_{h \searrow 0} \frac{f_i(y_0) - f_i(x_0)}{h} = \langle \nabla f_i(x_0), \nu \rangle = d_i, \quad i = 1, \ldots, k, \tag{3}$$

if $\|\nu\| = 1$. Throughout this paper, $\|\cdot\|$ denotes the 2-norm unless specified otherwise. Using the Jacobian of F, Eq. (3) can be stated in matrix vector notation as

$$J(x_0)\nu = d. \tag{4}$$

Hence, such a search direction ν can be computed by solving a system of linear equations. Since typically the number of decision variables is (much) higher than the number of objectives for a given MOP, i.e., $n \gg k$, the system (4) is (probably highly) underdetermined which implies that its solution is not unique. One possible choice is to take

$$\nu_+ := J(x_0)^+ d, \tag{5}$$

where $J(x_0)^+ \in \mathbb{R}^{n\times k}$ denotes the pseudo inverse[1] of $J(x_0)$ as the following discussion shows (see Sec. 3.2 for further insights): given a candidate solution x_0, a new solution is obtained via $x_1 = x_0 + h\nu$, where $\nu \in \mathbb{R}^n$ is a vector that satisfies (4). Among the solutions of (4), ν_+ is the one with the smallest Euclidean norm. Hence, given h, one expects for a step in direction ν_+ (in decision space) the largest progress in direction d (in objective space).

3.2 The DS Descent Method

Assume we are given a direction $d \in \mathbb{R}^k \backslash \{0\}$ with $d_i \leq 0$, $i = 1, \ldots, k$. Further, we assume that we are given a point $x_0 \in \mathbb{R}^n$ with $\operatorname{rank}(J(x_0)) = k$ and that the image of F is bounded from below. A greedy search in d-direction using Eq. (5) leads to the (numerical) solution of the following initial value problem:

$$\begin{aligned} x(0) &= x_0 \in \mathbb{R}^n \\ \dot{x}(t) &= J(x(t))^+ d, \quad t > 0, \end{aligned} \tag{DS(x_0, d)}$$

where t denotes the time. In the following we investigate solutions of (DS(x_0, d)) qualitatively. Let $\gamma : [0, t_f] \to \mathbb{R}^n$ be such a solution, where t_f is the final value, and let t_c be the smallest value of $t \geq 0$ such that

$$\nexists \nu \in \mathbb{R}^n : \quad J(x(t))\nu = d. \tag{6}$$

We will call t_c the critical value and $\gamma(t_c)$ the critical point of (DS(x_0, d)). γ can be divided into two parts (compare to Fig. 1): $\gamma([0, t_c])$ and $\gamma([t_c, t_f])$. In the first part, $F(\gamma(t))$ yields the desired decay in d-direction. From the critical point $\gamma(t_c)$ on a 'best fit' is computed (which follows directly by the properties of the pseudo inverse [20]), i.e.,

$$\nu_+(x(t)) = J(x(t))^+ d = \arg\min_{\nu \in \mathbb{R}^n} \|J(x(t))\nu - d\|. \tag{7}$$

For the end point $\gamma(t_f)$ it holds $J(\gamma(t_f))^+ d = 0$. On the one hand, such end points are of particular interest since they are KKT points with associated convex weight $\alpha = -d/\|d\|_1$. To see this, let $J(\gamma(t_f)) = U\Sigma V^T$ be a singular value decomposition of $J(\gamma(t_f))$. Since $J(\gamma(t_f))^+ d = 0$, where $J(\gamma(t_f))^+ = V\Sigma^+ U^T$, it is also $J(\gamma(t_f))^T d = V\Sigma U^T d = 0$, i.e., it holds

[1] If the rank of $J := J(x_0)$ is k (i.e., maximal) the pseudo inverse is given by $J^+ = J^T(JJ^T)^{-1}$.

$\sum_{i=1}^{k} \alpha_i \nabla f_i(\gamma(t_f)) = 0$.
On the other hand, the computation of γ in $[t_c, t_f]$ might get computationally expensive since (DS(x_0, d)) is stiff in the second part. Further, the computation of $\gamma([t_c, t_f])$ does not fit to the original idea of the directed search. Hence, we will restrict ourselves here to the detection of $\gamma(t_c)$.

As seen above, one cannot expect to get KKT points with associated weight

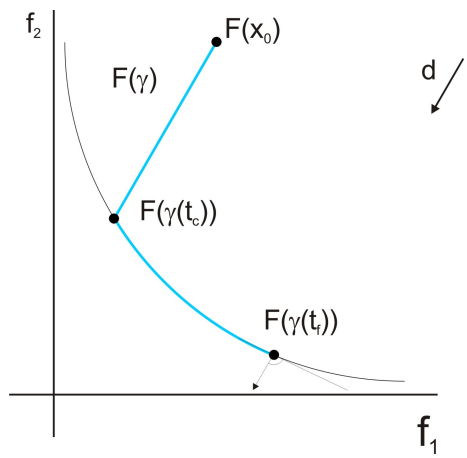


Fig. 1. One possible solution curve $F(\gamma)$ of (DS(x_0, d)).

α when computing $\gamma(t_c)$. However, the following result shows a relation to the well-known NBI method. The NBI subproblem can be stated as follows[2]

$$\begin{aligned} \max_{x,l} \quad & l \\ \text{s.t.} \quad & F(x_0) + ld = F(x). \end{aligned} \tag{NBI(x_0, d)}$$

Proposition 1. *Let x^* be the critical point of (DS(x_0, d)), then it is a local solution of (NBI(x_0, d)).*

Proof. Let $g(x, l) := l$ and $h_i(x, l) := f_i(x_0) + ld_i - f_i(x)$. Assume x^* is not a local solution of (NBI(x_0, d)). Then there exist $\nu = (\tilde{\nu}, \nu_{n+1}) \in \mathbb{R}^{n+1}$, $\tilde{\nu} \in \mathbb{R}^n$, and $l^* \in \mathbb{R}$ such that

$$\langle \nabla g(x^*, l^*), \nu \rangle = \left\langle \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} \tilde{\nu} \\ \nu_{n+1} \end{pmatrix} \right\rangle > 0, \quad \text{and} \tag{8}$$

$$\langle \nabla h_i(x^*, l^*), \nu \rangle = \left\langle \begin{pmatrix} -\nabla f_i(x^*) \\ d_i \end{pmatrix}, \begin{pmatrix} \tilde{\nu} \\ \nu_{n+1} \end{pmatrix} \right\rangle = 0, \quad i = 1, \ldots, k. \tag{9}$$

[2] We note that the original idea of NBI is not to maximize the distance from $F(x_0)$ for a given point x_0, but this is a straightforward adaption to the current context.

By (8) it follows that $\nu_{n+1} \neq 0$, and by (9) that

$$\langle \nabla f_i(x^*), \tilde{\nu} \rangle = \nu_{n+1} d_i, \quad i = 1, \ldots, k. \tag{10}$$

Hence, it is $\frac{1}{\nu_{n+1}} J(x^*)\tilde{\nu} = d$ which contradicts that x^* is a critical point. □

In turn, local solutions of (NBI(x_0, d)) are also potential critical points of (DS(x_0, d)): let x^{**} be a solution of (NBI(x_0, d)). Assume that there exists a $\nu \in \mathbb{R}^n$ such that $J(x^{**})\nu = d$. Then, $\tilde{\nu} = (\nu, 1) \in \mathbb{R}^{n+1}$ satisfies (8) and (9) which is in contradiction to the assumption of x^{**}.

Numerical treatment of (DS(x_0, d)) Since we are only interested in the detection of critical points the numerical treatment of (DS(x_0, d)) gets significantly simplified. In the following we describe the steps for a particular realization.
To trace the solution curve of (DS(x_0, d)), one can e.g. choose well-established numerical discretization methods (e.g., [21]). As these methods do not allow to perform a correction back to the solution curve, computed errors can not be corrected. In the following, we present one possible way to compute the solution curve numerically by a specialized predictor corrector (PC) method (e.g., [22]), i.e., with a procedure which allows for such a correction: recall that for every point x on the solution curve it holds

$$F(x) = F(x_0) + \lambda_y d, \tag{11}$$

where $\lambda_y \in \mathbb{R}$. Hence, the curve is contained in the zero set of

$$H : \mathbb{R}^{n+1} \to \mathbb{R}^k, \quad H(x, \lambda_y) = F(x) - F(x_0) - \lambda_y d. \tag{12}$$

To comply with the needs of PC methods we introduce an additional parameter λ_x into the solution curve (which is defined in decision space):

$$\begin{aligned} x(0) &= (x_0, \lambda_{x,0} = 0) \in \mathbb{R}^{n+1} \\ \dot{x}(t) &= \begin{pmatrix} J(x(t))^+ d \\ 1 \end{pmatrix}, \quad t > 0. \end{aligned} \tag{13}$$

So far, we are not able to apply PC methods since the parameters λ_x and λ_y parametrize different curves. The following little consideration, however, argues that it is reasonable to match the two parameters: let x_0 be given and $\lambda_{x,0} = 0$, and let $(x_1, \lambda_{x,1})$ be an Euler step of (13) with a small step size $\Delta\lambda_x$, i.e., $x_1 = x_0 + \Delta\lambda_x \nu_+(x_0)$ and $\lambda_{x,1} = \Delta\lambda_x$. By construction of $\nu_+(x_0)$ and since $\Delta\lambda_x$ is small we have

$$d_i \approx \frac{f_i(x_1) - f_i(x_0)}{\Delta\lambda_x \|\nu_+(x_0)\|}, \quad i = 1, \ldots, k. \tag{14}$$

This implies that $F(x_1) - F(x_0) \approx \Delta\lambda_x d \|\nu_+(x_0)\|$, and hence,
$H(x_1, \lambda_{x,1})/\|\nu_+(x_0)\| \approx 0$. Using this, (13) and (12) can now be used to perform classical PC methods in order to trace the solution curve: starting with the point

$(x_0, \lambda_{x,0})$ one can integrate (13) numerically for a small time step, e.g. via the Euler method as described above leading to a predictor solution $(\tilde{x}_1, \tilde{\lambda}_{x,1})$. In a next step, this solution can be corrected to the desired curve. That is, starting with $(\tilde{x}_1, \tilde{\lambda}_{x,1})$ and using a root finding method applied on (12) one can seek for a solution $(x_1, \lambda_{x,1})$ with $H(x_1, \lambda_{x,1})/\|J(x_0)^+ d\| \approx 0$, and so on. For details such as step size control we refer to a preliminary study of the DS in [15].
To define a stopping criterion we can utilize the facts that $\text{rank}(J(x_0)) = k$ (by assumption) and that $\text{rank}(J(x^*)) < k$ (by definition of the critical point x^*). The rank of a matrix can of course not be used to detect the critical point numerically, but instead the condition number κ_2 of $J(x)$ can be used: one can e.g. compute

$$\kappa_2(J(x)) = \|J(x)\|\|J(x)^+\| = \sigma_1/\sigma_k, \tag{15}$$

where σ_1 and σ_k are the largest and smallest singular values of $J(x)$, respectively, and stop the process if $\kappa_2(J(x_i)) \geq tol$, where $tol \in \mathbb{R}_+$ is a given (large) threshold. This can be done since by the above discussion $\kappa_2(J(x(t))) \to \infty$ for $x(t) \to x^*$. Alternatively, one can stop the search if the difference between two consecutive solutions is below a given threshold.
This discussion shows one potential drawback of the approach, namely that the determination of the search direction by solving (4) gets inaccurate for points near the Pareto set due to the high condition number of $J(x)$. However, our experience has shown that state-of-the-art numerical tools allow to come 'near enough' to the Pareto set even for larger problems (see also the results in Sec. 6). However, large condition numbers (say, $tol > 1000$) should be avoided as they might lead to inaccuracies and higher computational times when solving (4) numerically.

Crucial for the above descent method is of course the proper choice of d to steer the process. This is in general not an easy task since this is highly problem dependent. In the next subsection, one particular choice of d is discussed for the realization of the continuation method. Some other choices are used in Sec. 6 where the DS is integrated into a MOEA.

3.3 The DS Continuation Method

In this section we propose a new PC method for the continuation along (local) Pareto sets of a given MOP. The central difference to a classical method is that we suggest a new predictor direction which is based on the geometry of the Pareto front and is realized by DS. Interesting is the fact that this new method does not require to compute the Hessians of the objectives. In the following we present the alternative predictor direction and propose then a 'complete' Hessian free PC method (and the variant in Sec. 5 is even gradient free).

Predictor Direction Assume we are given a (local) Pareto point x and the associated convex weight α, i.e., such that Eq. (2) is satisfied. Further, we assume that $\text{rank}(J(x)) = k - 1$. It is known (e.g., [18]) that in this case α is orthogonal to the linearized Pareto front at $F(x)$. Thus, a search orthogonal to α (in

objective space) could be promising to obtain new predictor points. To use DS, for instance a QR-factorization of α can be computed, i.e.,

$$\alpha = QR, \tag{16}$$

where $Q = (q_1, \dots, q_k) \in \mathbb{R}^{k \times k}$ is orthogonal and $q_i,\ i = 1, \dots, k$, are its column vectors, and $R = (r_{11}, 0, \dots, 0)^T \in \mathbb{R}^{k \times 1}$ with $r_{11} \in \mathbb{R} \backslash \{0\}$. Since by (16) $\alpha = r_{11} q_1$, i.e., $\alpha \in span\{q_1\}$, and Q orthogonal, it follows that the column vectors $q_2, \dots, q_k$ build an orthonormal basis of the hyperplane which is orthogonal to α. Thus, a promising well-spread set of search directions ν_i may be the ones which satisfy

$$J(x)\nu_i = q_i, \quad i = 2, \dots, k. \tag{17}$$

Since α is not in the image of $J(x)$ (otherwise x would not be a Pareto point) and by assumption on the rank of $J(x)$ it follows that the vectors $q_2, \dots, q_k$ are in the image of $J(x)$, i.e., Eq. (17) can be solved for each $i \in \{2, \dots, k\}$.
We stress that for the case $k = 2$ the predictor direction (17) coincides—apart from its length—with one of the gradients $\nabla f_i(x)$, $i = 1, 2$, which has already been successfully used in [23, 24] for bi-objective continuation. Further, we note that such predictor directions $\nu_p = J(x)^+ q$ do not have to be tangent to the Pareto *set*. Instead, $J(x)\nu_p$ points along the linearized Pareto *front*. Since as for (5) ν_p is the most greedy solution (compare also to Sec. 4) we can expect that the image $F(p)$, where p is the chosen predictor, is also close to the Pareto front, and thus, that only few iteration steps are required to correct back to this set.

PC Method Based on the above discussion we derive in the following a new PC method.
Predictor Assume we are given a Pareto point x_0 with associated weight α_0. The predictor direction can—except for its signum—be chosen as described above, i.e., one of the normalized vectors $\nu := \pm\nu_2/\|\nu_2\|$, where ν_2 satisfies (4) as described above for $d = q_2$. To orientate the curve (i.e., to determine the signum of ν) one can simply use the change of one of the objective values. For this, the signum of the according entry of the direction vector q_2 can be taken. If, for instance, an improvement according to f_2 is sought, then

$$p := x_0 - sgn(q_{2,2}) h \nu_2 / \|\nu_2\| \tag{18}$$

can be chosen as predictor, where $q_{2,2}$ denotes the 2nd entry of q_2, and h is the desired step size.
To get the value of h we proceed as follows: assume we are given x_0 and the direction ν, $\|\nu\| = 1$, associated to the direction q in objective space for the predictor $p = x_0 + h\nu$. To obtain an adequate spread of the solutions the function values $f_j(x)$ and $f_j(p)$ of at least one objective differ ideally by a (problem dependent) value ϵ while the difference for all other objectives does not exceed this threshold. Since this value can differ for each objective we obtain for the demand on the spread

$$d_w(F(p), F(x)) \approx \epsilon, \tag{19}$$

where $d_w(x,y) = \max_{i=1,\dots,k}(w_i|x_i - y_i|)$ is the weighted maximum norm distance (used to weight the objective space). Assuming that all f_i's are Lipschitz continuous and that the step size h_i for the i-th objective is sufficiently small we obtain

$$\underbrace{|f_i(p) - f_i(x)|}_{\overset{!}{=}\epsilon/w_i} \approx L_{i,x} \underbrace{\|p - x\|}_{=h_i}, \quad i = 1,\dots,k. \tag{20}$$

Since $L_{i,x}$ can be approximated by the norm of the directional derivative we obtain for each objective the control

$$h_i = \frac{\epsilon}{w_i|\langle \nabla f_i(x), \nu\rangle|}, \quad i = 1,\dots,k, \tag{21}$$

and hence for the entire MOP

$$h := \min_{i=1,\dots,k} h_i. \tag{22}$$

Corrector Given p, the subsequent solution along the curve can be computed by solving numerically (DS(x_0, d)), using p as initial value and choosing $d := -\alpha_0$, i.e., the negative of the weight from the previous solution x_0 leading to a new solution x_1. The new associated weight α_1 can be updated as follows ([25]):

$$\alpha_1 \in \arg\min_{\lambda} \left\{ \| \sum\nolimits_{i=1}^{k} \lambda_i \nabla f_i(x)\|^2, \text{ s.t. } \lambda_i \geq 0,\ i = 1,\dots,k,\ \sum\nolimits_{i=1}^{k} \lambda_i = 1 \right\}. \tag{23}$$

Alg. 1 shows a possible realization of the continuation method for bi-objective problems (BOPs). There are only two choices possible for a continuation along the Pareto front: 'right down' or 'left up'. Alg. 1 shows the 'right down' movement starting with one initial solution, the respective realization for a 'left up' movement is analog. The consideration of MOPs with $k > 2$ requires an additional data structure for the efficient representation of the approximation. See e.g. [26–28] for further details.

Algorithm 1 BOP Continuation for a 'right down' movement in objective space

Require: Initial solution (x_0, α_0), threshold $\epsilon \in \mathbb{R}_+$, tolerance $tol \in \mathbb{R}_+$.
Ensure: Set of candidate solutions x_i aligned 'right down' in objective space

```
i := 0
repeat
    compute q_2, ν, and h as in (16), (17), and (22)
    p_i := x_i − sgn(q_{2,2}) h ν
    compute x_{i+1} by solving DS(p_i, −α_i)
    compute α_{i+1} as in (23)
    set i := i + 1
until α_{i+1,2} ≥ 1 − tol or no improvement in f_2 direction could be achieved
```

4 On Multi-Objective Stochastic Local Search

In the following we discuss the behavior of multi-objective stochastic local search (MOSLS) which can to a certain extent be explained using the approach of the DS. In particular, it may explain one facet of the huge success of global multi-objective stochastic search algorithms such as MOEAs.
To examine the behavior of MOSLS we investigate the relation of search directions $\nu \in \mathbb{R}^n$ in decision space and the movement performed in objective space at a given point $x \in \mathbb{R}^n$. The latter can be expressed by $J(x)\nu$: if a line search along ν is performed, then the new iterate is given by $x_{new} = x + t\nu$, where t is a (small) step size. If on the other hand x_{new} is chosen from a small neighborhood $N(x)$ of x (e.g., via mutation), then we are in the same setting. To see this, define $\nu := (x_{new} - x)/\|x_{new} - x\|$, and we have $x_{new} = x + \|x_{new} - x\|\nu$.
In the following we will first consider the extreme cases—x is either far away or very close to the Pareto set—and based on this we will derive conclusions of MOSLS within global search heuristics.

x far away from the Pareto set In [29], it has been observed that the objectives' gradients typically point nearly in the same direction if x is far from the Pareto set. For MOPs with this property, the gradients point in the extreme case into the same direction, and we have for $g := \nabla f_1(x)/\|\nabla f_1(x)\|$

$$\nabla f_i(x) = \mu_i g, \quad i = 1, \ldots, k, \tag{24}$$

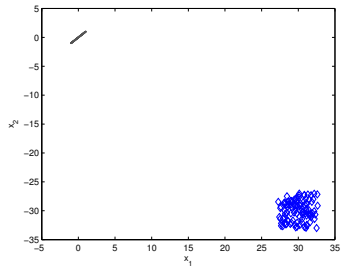
where each $\mu_i > 0$. Then, we have for a search direction $\nu \in \mathbb{R}^n$ that

$$J(x)\nu = \begin{pmatrix} \mu_1 g^T \nu \\ \vdots \\ \mu_k g^T \nu \end{pmatrix} = g^T \nu \begin{pmatrix} \mu_1 \\ \vdots \\ \mu_k \end{pmatrix}. \tag{25}$$

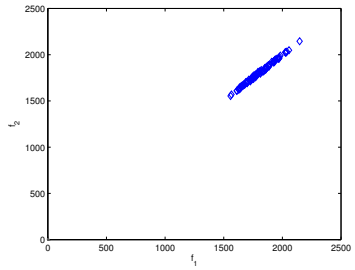
That is, it is either (a) $J(x)\nu = 0$ or (b) a search in direction $\nu \in \mathbb{R}^n$ leads to a movement in direction $d = \pm(\mu_1, \ldots, \mu_k)^T$ in objective space. Since the dimension of the kernel of $J(x)$ is $n-1$, this means that for a randomly chosen search direction $\nu \in \mathbb{R}^n$ the probability is zero that $J(x)\nu = 0$. Hence, if we choose a point x_{new} randomly from a small neighborhood $N(x)$ we can expect that the difference $F(y) - F(x)$ is basically a multiple of d. If $F(x_{new}) - F(x) > 0$, where '$>$' is considered component-wise, (i.e., $x \prec x_{new}$), one can simply flip the search and use $\tilde{x}_{new} := x - \nu$ where $\nu = x_{new} - x$ since by the above considerations one can expect that $F(x) - F(x - \nu) \approx -(F(x) - F(x + \nu)) < 0$ (i.e., $\tilde{x}_{new} \prec x$). Note, however, that the largest improvement along direction d can be obtained for $\nu = \pm g$ since in that case $\|J(x)\nu\|$ is maximized.
As an example we consider the bi-objective problem ([30])

$$f_1, f_2 : \mathbb{R}^n \to \mathbb{R}^2, \quad f_i(x) = \|x - a_i\|^2, \; i = 1, 2. \tag{26}$$

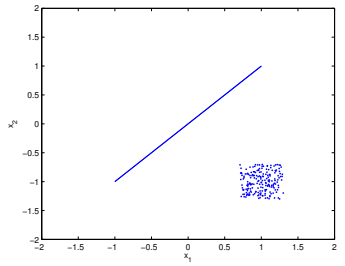
Fig. 2 (a) shows an example for $n = 2$ where 100 points were randomly sampled around $x_0 = (30, -30)^T$. In objective space, a clear movement along $d = \pm(1, 1)^T$ can be observed which is (by construction) not the case in decision space.



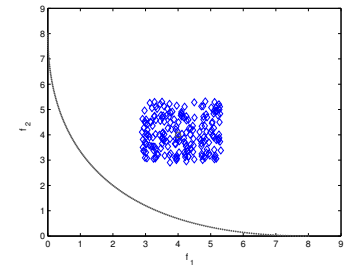
(a) x_0 far away, decision space



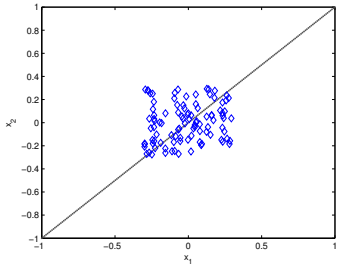
(b) x_0 far away, objective space



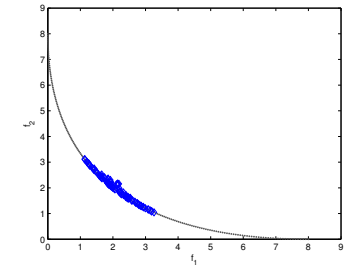
(c) x_0 in between, decision space



(d) x_0 in between, objective space



(e) x_0 near, decision space



(f) x_0 near, objective space

Fig. 2. Behavior of MOSLS in different stages of the search process.

x near to the Pareto set We again consider the extreme situation, namely now that x is a Pareto point. That is, there exists a convex weight $\alpha \in \mathbb{R}^k$ such that $J(x)^T\alpha = 0$. Further, let $\text{rank}(J(x)) = k-1$. Then it holds for any $\nu \in \mathbb{R}^n$:

$$\langle J(x)\nu, \alpha\rangle = \langle \nu, J(x)^T\alpha\rangle = 0. \tag{27}$$

Hence, we have that either (a) $J(x)\nu = 0$ or (b) that a movement along ν leads to a movement along the Pareto front since $J(x)\nu \neq 0$ is orthogonal to α (compare to Sec. 3.3). Since the dimension of the kernel of $J(x)$ is $n-k+1$, the probability for event (a) is zero for a chosen direction ν (or a randomly chosen point $x_{new} \in N(x)$, where $N(x)$ is a small neighborbood of x). Note that the movement along the Pareto front is regardless of the choice of ν, i.e., regardless if the search in decision space is performed along the Pareto *set* or not.
Let us consider the important special case $k = 2$. Without loss of generality assume that $\nabla f_1(x) \neq 0$ and $\nabla f_2(x) \neq 0$. Then there exist $\alpha_1 \neq 0$ and $\alpha_2 \neq 0$ such that $\alpha_1\nabla f_1(x) + \alpha_2\nabla f_2(x) = 0$. Defining $w = -\alpha_1/\alpha_2$ we can write

$$J(x)\nu = \begin{pmatrix} \nabla f_1(x)^T\nu \\ w\nabla f_1(x)^T\nu \end{pmatrix}. \tag{28}$$

We see that $J(x)\nu = 0$ iff ν is orthogonal to $\nabla f_1(x)$. Further, $\nu \in \{\pm\nabla f_1(x)\}$ is the greedy direction since in that case $\|J(x)\nu\|$ is maximized.
Fig. 2 shows 100 randomly sampled points in a neighborhood of $x_0 = (0,0)^T$. Again, by construction, no pattern is visible in decision space, but in objective space a clear movement along the Pareto front can be observed.

Contribution of MOSLS within global heuristics Based on the observations made above we make a first attempt to explain the contribution of MOSLS within global search heuristic such as MOEAs. The above discussion shows mathematically and empirically that two important features are inherent in MOSLS, namely a pressure both toward and along the Pareto front. The above considerations hold for extreme cases. Apparently, candidate solutions do not have to be near nor far away from the Pareto set. For points x that are 'in between' the search can be steered into any direction. Fig. 2 (b) shows the result of 100 randomly chosen points around $x_0 = (1,-1)^T$. This 'opening' allows set based stochastic search methods such as MOEAs in principle to find all regions of the Pareto set/front. The latter of course with all the restrictions that hold for local seach methods.
To examine the behavior of MOSLS within the entire search process empirically we consider a simple set based neighborhood search (SNS, see Alg. 2). In this method, for entry a of a given archive A_i a solution $b \in N(a)$ is chosen in a small neighborhood leading to the new set of candidate solutions B_i. The next archive A_{i+1} consists of the non-dominated solutions of A_i and B_i.

Fig. 3 shows some numerical results for problem (26). Fig. 3 (a) shows a result of SNS where A_0 has been built by 10 randomly chosen points within the domain $D = [-10, 10]^2$. In this figure, every second generation is plotted. As neighborhood in step i we have chosen the maximum norm with radius $\delta_i = 5/i$.

Algorithm 2 Simple Neighborhood Search (SNS)

Require: Neighborhood $N_i(x)$ of a given point x in iteration i.
Ensure: Sequence A_i of candidate solutions

```
Generate A_0 ⊂ ℝ^n at random
for i = 1, 2, ... do
    B_i := ∅
    for all a ∈ A_i do
        choose b ∈ N_i(a) at random
        B_i := B_i ∪ {b}
    end for
    A_{i+1} = {a ∈ A_i ∪ B_i : ∄b ∈ A_i ∪ B_i s.t. b ≺ a}
end for
```

The result indicates that for this model SNS has no problem to converge. The final archive is indeed very close to the true Pareto front. Since a random search along the Pareto set leads to many non-dominated solutions leading to many entries in the archive we have for sake of a (very simple) comparison coupled SNS with the archiver ArchiveUpdateTight2 that allows to reduce the number of archive entries while preserving certain convergence properties ([31]). See Fig. 3 (b) for a result where the number of function evaluations were restricted to 500 (only the final archive is plotted). As comparison, Fig. 3 (c) contains the non-dominated solutions from a simple global searcher (all points are chosen randomly from D) with a budget of $5,000$ function calls. This observation gets confirmed in Tab. 1, where GS and SNS are compared on 14 different benchmark functions with different characteristics such as modality and connectedness of the Pareto fronts. SNS wins in 13 out of 14 cases (and significant in 12 cases due to the Wilcoxon rank-sum test) and is only inferior (significantly) compared to GS on ZDT4 which is highly multi-modal as it contains 21^9 local Pareto fronts.

Though SNS does not make use of any gradient information, explicit steering, or any kind of population/swarm based intelligence we think that the results are already reasonable. Certainly, state-of-the-art MOEAs will outperform SNS (in particular when n or k are increased, and/or more complicated models are beeing considered) which results from the great importance of the interplay of the different elements of a search heuristic such as the balance of local and global search (e.g., [32]). However, our focus in this section is the behavior of stochastic local search which is used within every state-of-the-art heuristic. As the above discussion shows and the numerical results indicate, the two key features, pressure toward and along the Pareto front (which are closely related to the terms *convergence* and *diversity*) are already inherent in MOSLS. Hence, one can even say that the classical EMO task (i.e., to find a finite size approximation of the Pareto front) is a well-conditioned problem. This might be one facet that explains the huge success of global search heuristics such as MOEAs. We stress that the above considerations are not restricted to differentiable problems. Note that many maps are only non differentiable on a zero set. In other words, for

such a map F and a randomly chosen feasible point x the probability is one that $J(x)$ exists, and hence, that the above considerations hold.

5 Gradient Free Directed Search

So far, DS requires the Jacobian if the method is realized via (4). Here, we present an alternative way to obtain such search directions ν without explicitly computing or approximating the Jacobian. Instead, the information of function values of points near a given point is utilized for the approximation of ν. This method can be viewed as a particular finite difference (FD) method, however, it has two crucial advantages over the classical Jacobian approximation via FD: (i) fewer additional function evaluations are required to obtain a direction ν such that Eq. (4) is satisfied, and (ii) existing neighborhood information can be utilized leading to a further reduction of the cost. The latter is in particular interesting in the context of set-based optimization strategies such as MOEAs.
The general idea behind the method is as follows: given a point x_0 that is designated for local search as well as another point x_i in the current population (i.e., the objective value is known) that is in the vicinity of x_0, then the given information can be used to approximate the directional derivative in direction

$$\nu_i := (x_i - x_0)/\|x_i - x_0\| \tag{29}$$

without any additional cost (in terms of function evaluations). That is, it holds

$$f'_{\nu_i}(x_0) = \langle \nabla f(x_0), \nu_i \rangle = \frac{f(x_i) - f(x_0)}{\|x_i - x_0\|} + O(\|x_i - x_0\|), \tag{30}$$

where O denotes the Landau symbol. This can be seen by considering the forward difference quotient on the line search function $f_{\nu_i}(h) = f(x_0 + h\nu_i)$.
Now assume a candidate solution $x_0 \in \mathbb{R}^n$ is designated for local search and further r search directions $\nu_i \in \mathbb{R}^n$, $i = 1, \ldots, r$, are given. Then, the matrix $\mathcal{F} := JV \in \mathbb{R}^{k \times r}$, where $V = (\nu_1, \ldots, \nu_r) \in \mathbb{R}^{n \times r}$, is as follows:

$$\mathcal{F} = JV = (\langle \nabla f_i(x), \nu_j \rangle)_{i=1,\ldots,k, j=1,\ldots,r}\,. \tag{31}$$

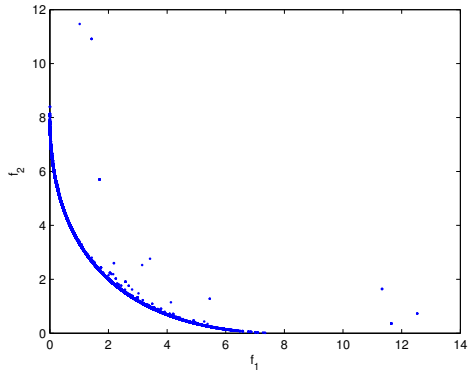
Hence, every element m_{ij} of JV is defined by the value of the directional derivative of objective f_i in direction ν_j, and can be approximated as in Eq. (30). Given JV and a direction d, one can thus obtain a search direction via solving

$$JV\lambda = d, \tag{32}$$

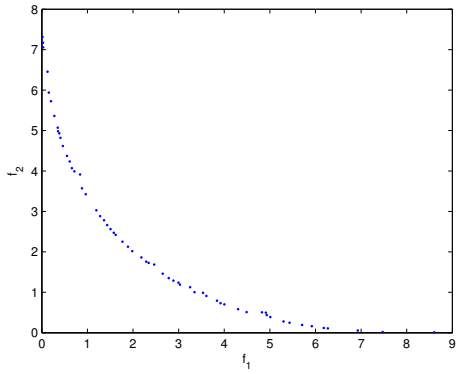
and then setting

$$\nu = V\lambda. \tag{33}$$

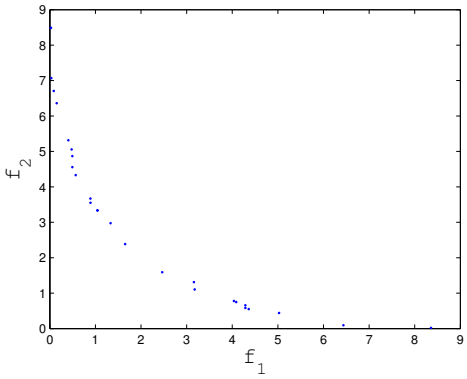
We stress that the above method is *not* a gradient approximation via sampling (as e.g. done in [33]) but instead a new way to get an approximation of ν without explicitly computing the gradient.



(a) SNS



(b) SNS + AU-Tight2



(c) Simple global search

Fig. 3. (a): application of SNS for 20 generations, (b): SNS coupled with AU-Tight2 and a budget of 500 function evaluations, (c): result of a simple global search with a budget of 5,000 function evaluations.

Next we investigate the solvability and the condition of Eq. (32) which depends both on the choice of the value of r and the position of the search directions ν_i to each other. In case $\text{rank}(J(x)) = k$ it is known that

$$\text{rank}(J(x)) = k \quad \Rightarrow \quad \text{rank}(JV) = \min(\text{rank}(V), \text{rank}(J)). \tag{34}$$

If on the other hand x is a critical point (and hence, $\text{rank}(J(x)) < k$), then it follows by the properties of matrix multiplication that also $\text{rank}(JV) < k$ regardless of the choice of V (i.e., regardless of the number r and the choice of the search directions ν_i). This indicates that the condition number of JV can be used to check numerically if a current iterate is already near to an end point of $(\text{DS}(x_0, d))$. It seems to make sense to choose the search directions orthogonal to each other. This is motivated by the fact that $\kappa_2(JV) \leq \kappa_2(J(x))$ for the case that V is orthogonal which means that the condition number $\kappa_2(JV)$ can indeed be used as a stopping criterion analog to the original DS method.
The above considerations show that already for $r = k$ search directions ν_i, $i = 1, \ldots, r$, one can find a descent direction $\tilde{\nu}$ by solving Eq. (32) regardless of n. However, by construction it is $\nu \in span\{\nu_1, \ldots, \nu_r\}$ which means that only a r-dimensional subspace of the $\mathbb{R}^n$ is explored in one step. One would expect that the more search directions ν_i are taken into account, the better the choice of $\tilde{\nu}$ is, which is indeed the case: for $r > k$, we suggest to choose analog to (5)

$$\nu_+^{(r)} := V(JV)^+ d. \tag{35}$$

In fact, $\nu_+^{(r)}$ comes closer to $\nu_+ = J(x)^+ d$ as r increases, and both vectors are equal for $r = n$ in case the v_i's are chosen orthogonal to each other. This is due to the fact that in this case VV^T (i.e., the orthogonal projection onto $span\{v_1, \ldots, v_r\}$) converges to the identity matrix $I_n \in \mathbb{R}^{n \times n}$.
In the following we investigate empirically the influence of r on the performance of the new gradient free DS descent method. For this reconsider MOP (26), this time for $n = 10$. Consider the point $x_0 = (1, -1, \ldots, 1, -1)^T \in \mathbb{R}^{10}$ and direction $d = (-1, -1)^T$. We obtain $\nabla f_1(x_0) = (0, -4, \ldots, 0, -4)^T$, $\nabla f_2(x_0) = (4, 0, \ldots, 4, 0)^T$, and $\nu_+ = (-1, 1, \ldots, -1, 1)^T$. Choose $\nu_i = e_i$ as the canonical unit vectors and consider the search directions $\nu_+^{(r)}$, $r \in \{2, 4, 6, 8, 10\}$, where the first r search directions are taken. It follows that $\nu_+^{(2)} = (-1, 1, 0, \ldots, 0)^T$, $\nu_+^{(4)} = (-1, 1, -1, 1, 0, \ldots, 0)^T$ etc., and $\nu_+^{(10)} = \nu_+$. It is $\|J(x_0)\nu_+\| = \sqrt{2} \cdot 20$ and $\|J(x_0)\nu_+^{(r)}\| = \sqrt{2} \cdot 2r$. Fig. 4 shows the Pareto front together with the images $y_r = F(x_0 + t\nu_+^{(r)})$, $r \in \{2, 4, \ldots, 10\}$, for the step size $t = 1$. Apparently, the larger the value of r, the more the improvement in d-direction for the same step size. This shows—as anticipated—the trade off between cost and performance of one iteration step depending on r. Crucial for the approximation of $\mathcal{F}$ is next to r the choice of the test points x_i. If the function values of points in a neighborhood $N(x_0)$ are already known, it seems to be wise to include them to build the matrix. Nevertheless, it might be that further test points have to be sampled to obtain a better search direction which is motivated by the above

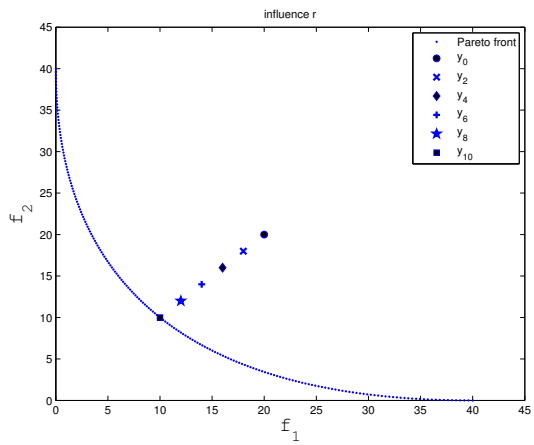


Fig. 4. Influence of the choice of r in the DDS method.

discussion. As a rule of thumb, we have observed in our computations that we present in the next section that $r \approx 0.4n$ leads to good results for the use within MOEAs.

Assume that we are given $x_0 \in \mathbb{R}^n$ as well as l neighboring solutions $x_1, \ldots, x_l \in N(x_0)$. According to the discussion above, it is desired that all further search directions are both orthogonal to each other and orthogonal to the previous ones. In order to compute the new search directions $\nu_{l+1}, \ldots, \nu_r$, $r > l$, one can proceed as follows: compute $V = QR = (q_1, \ldots, q_l, q_{l+1}, \ldots, q_n)R$. Then it is by construction $\nu_i \in span\{q_1, \ldots, q_i\}$ for $i = 1, \ldots, l$, and hence

$$\langle \nu_i, q_j \rangle = 0, \quad \forall i \in \{1, \ldots, l\},\ j \in \{l+1, \ldots, r\}. \tag{36}$$

One can thus e.g. set

$$\begin{aligned} \nu_{l+i} &= q_{l+i}, \quad i = 1, \ldots, r-l \\ x_{l+i} &= x_0 + \nu_{l+i}, \quad i = 1, \ldots, r-l. \end{aligned} \tag{37}$$

Since the cost for the QR-factorization is $O(n^3)$ in terms of flops, one may alternatively use the Gram-Schmidt procedure (e.g., [20]) to obtain the remaining sample points (e.g., if $r - l$ is small and n is large). This leads to a cost of $O((r-l)^2 n)$ flops. For the special case that $\nu_1 = x_1 - x_0$ and $\tilde{\nu}_2 = \tilde{x}_2 - x_0$ are given such that $\{\nu_1, \tilde{\nu}_2\}$ are linearly independent the second search vector ν_2 can be computed by

$$\nu_2 = \tilde{\nu}_2 - \langle \nu_1, \tilde{\nu}_2 \rangle \nu_1. \tag{38}$$

Having stated the basic idea of the gradient free DS (called DDS for *discrete* Directed Search) we are now in the position to perform a movement both toward and along the Pareto front as for the classical DS. For the DDS Descent Method

as standalone algorithm we refer to some studies made in [34]. More promising is the use of DDS within set based algorithms such as MOEAs since then the search direction can be computed for free which we will address in Sec. 6. In the following we will consider modifications required to make the above continuation method gradient free which makes it applicable to a broader class of problems.

5.1 DDS Continuation

A gradient free realization of the DS Continuation is mainly possible due to the idea in (32) and (33) and the observation made in Sec. 4, namely that a local search along x_0 leads with probability one to a movement from $F(x_0)$ along the Pareto front. In the following we describe the details about several tasks that have to be performed to realize the continuation and will then present the algorithm.

Task 1: steering the search (a) Predictor: in the DS method, a set of predictors is generated by computing a QR-factorization of the KKT weight α at the given point x_0. This approach can apparently not be chosen here since the computation of α via (23) requires gradient information. Instead, we utilize the observation that $J(x_0)\nu$ points along the Pareto front for almost all vectors $\nu \in \mathbb{R}^n$: let $\nu_1, \ldots, \nu_{k-1} \in \mathbb{R}^n$ such that $J(x_0)\nu_1, \ldots, J(x_0)\nu_{k-1}$ are linearly independent. Consider the QR-factorization

$$(J(x_0)\nu_1, \ldots, J(x_0)\nu_{k-1}) = QR = (q_1, \ldots, q_k)R. \tag{39}$$

Then, $\{q_1, \ldots, q_{k-1}\}$ is such a desired orthonormal basis, i.e., one can e.g. set $d_p = q_i$, $i \in \{1, \ldots, k-1\}$, as possible predictor direction. Also the KKT weight α can be gained by this factorization: since q_k is orthogonal to each $J(x_0)\nu_i$, $i = 1, \ldots, k-1$, and either $q_k < 0$ or $q_k > 0$, it is

$$\alpha = sign(q_{k,1})q_k/\|q_k\|_1. \tag{40}$$

Hence, by choosing test points x_i in the neighborhood $N(x_0)$ of x_0 and setting $\nu_i := x_i - x_0$, one obtains a gradient free way to get both tangent vectors and KKT weights if $J(x_0)\nu_i$ is approximated via (30). The cost for this is ideally given by k-1 evaluations of F, further evaluations may occur if $J(x_0)\nu_1, \ldots, J(x_0)\nu_{k-1}$ are not linearly independent (the probability for this event, however, is zero if the x_i's are chosen uniformly at random). For the special case $k = 2$ we assume w.l.o.g. that an improvement according to f_1 is sought for (i.e., a 'left up' movement along the Pareto front). Then, compute $x_1 \in N(x_0)$ and evaluate $F(x_1)$. The desired direction given in objective space is

$$d_p := \tilde{d} = F(x_1) - F(x_0) \tag{41}$$

if $f_1(x_1) < f_1(x_0)$, else set $d_p := -\tilde{d}$.
(b) Corrector: since the KKT weight α of the previous Pareto point is known (see Eq. (40)), one can proceed as in the gradient based DS continuation method,

i.e., to set $d_c = -\alpha$. For $k = 2$, this can be expressed analytically: if $d_p = (a, b)^T$, then the corrector direction is given by

$$d_c = -sign(a)(-b, a)^T. \tag{42}$$

Task 2: step size control (a) Predictor: the directional derivative used in the step size control (21) can be replaced by the finite difference leading to

$$\tilde{h}_i = \frac{\epsilon}{|f_i(x_1) - f_i(x_0)|}, \quad i = 1, \ldots, k, \tag{43}$$

where $x_1 \in N(x_0)$ and $\nu := x_1 - x_0$. Note that if $f_i(x_1)$ is close to $f_i(x_0)$, then the step size can get large. Hence, it is advisable to bound the overall step size by a maximal value t_{max}.
In case Eq. (35) is used to compute the search direction ν the above step size control cannot be used since for ν there does not exist a function value in that direction (since it is only known that $\nu \in span(\nu_1, \ldots, \nu_r)$). Instead, the step size in Eq. (21) can be chosen using the following estimation of the directional derivative: given x_0 and directions ν_i, $i = 1, \ldots, r$, and $\nu = V(JV)^+ d$, then

$$\langle \nabla f_j(x), \nu \rangle = \langle \nabla f_j(x), \sum\nolimits_{i=1}^{r} \lambda_i \nu_i \rangle = \sum\nolimits_{i=1}^{r} \lambda_i \langle \nabla f_j(x), \nu_i \rangle = \sum\nolimits_{i=1}^{r} \lambda_i m_{j,i}, \tag{44}$$

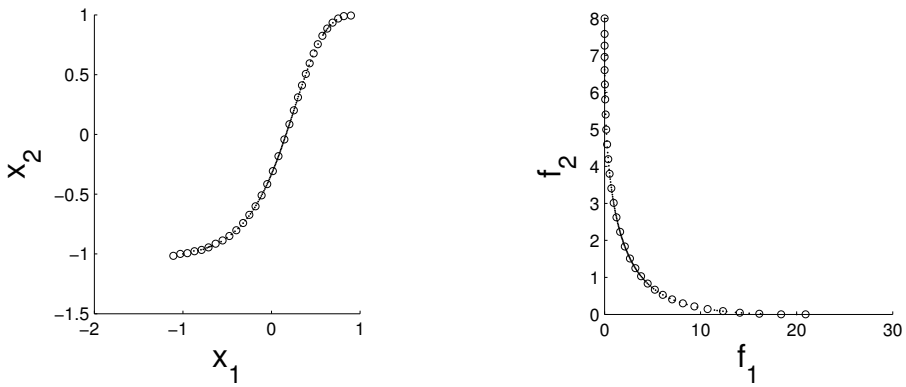
where the $m_{j,i}$'s are the entries of $\mathcal{F}$.
(b) Corrector: for the corrector steps, we follow the suggestions made in [15].
Now we are in the position to explain the gradient free PC method. Starting from an (approximate) local solution x_i, the next (approximate) solution is obtained as follows (compare to Alg. 3). First, a predictor p_i is selected by performing a step in direction d_p that points along the linearized Pareto front at $F(x_i)$ (lines 1-4). This point is only optimal if the Pareto set $\mathcal{P}$ is not bended around x_i but still near to $\mathcal{P}$ if the step size was chosen sufficiently small. Hence, in the second step, p_i is corrected back to the Pareto set (lines 5-8).
It is important to note that the continuation method has to be started with a (approximate) solution and is of local nature. That is, there is no guarantee that the method detects several connected components starting from one single solution, however, this is in certain cases possible since the movement is in fact performed along the boundary of the image which yields the possibility to reach other parts of the Pareto front. See Sec. 6.1 for such an example.
Note further that the overall cost of the search process can be reduced significantly by utilizing existing information. If, for instance, the predictor p_i is known and the corrector has to be performed via DDS, one can (among others) use the direction $v_i = (x_i - p_i)/\|x_i - p_i\|$. Since the value $F(x_i)$ of the previous candidate solution x_i is already known, the incorporation of ν_i comes for free in the context of DDS. Similar for all other points in the neighborhood of p_i those images are known.
As a first example, consider the problem ([27]):

$$f_1, f_2 : \mathbb{R}^2 \to \mathbb{R}, \quad f_i(x) = \sum\nolimits_{\substack{j=1 \\ j \neq i}}^{2} (x_j - a_j^i)^2 + (x_i - a_i^i)^4, \tag{45}$$

Algorithm 3 Gradient Free PC Step

Require: x_i: (approximate) Pareto point
Ensure: x_{i+1}: further candidate solution along Pareto front
1: compute d_p and $\mathcal{F}$ for x_i (Task 1 and Eq. (31))
2: compute ν_p as in Eq. (35)
3: compute t_p (Task 2)
4: $p_i := x_i + t_p \nu_p$
5: compute d_c and $\mathcal{F}$ for p_i (Task 1 and Eq. (31))
6: compute ν_c as in Eq. (35)
7: compute t_c (Task 2)
8: **return** $x_{i+1} = p_i + t_c \nu_c$

where $a^1 = (1,1)^T$ and $a^2 = -a^1$. Fig. 5 shows one result of the gradient free continuation method for $\epsilon = 0.08$. We started the search with $x_0 = a^1$ and used the maximum norm with radius $\delta = 0.3$ as neighborhood. A total of 32 solutions (i.e., correctors) were obtained where 95 function evaluations were needed. Here, a satisfying approximation of the Pareto front was obtained for a reasonable amount of function calls spent.



(a) Decision Space (b) Objective Space

Fig. 5. Application of the DDS continuation method on MOP (45) for $n = 2$ and $\epsilon = 0.08$.

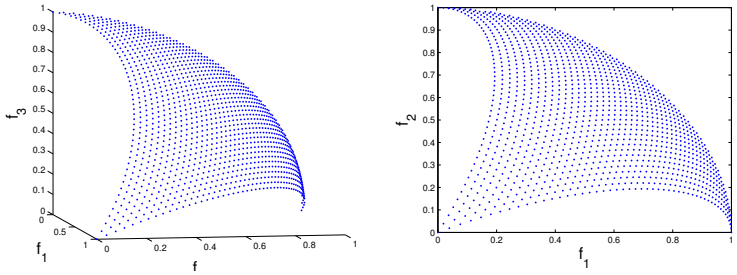
6 Numerical Results

Here we present some numerical results of the DS as standalone algorithm and as local searcher within the state-of-the-art algorithm MOEA/D ([3]).

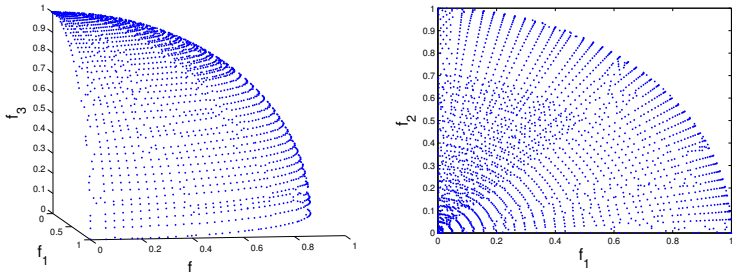
6.1 (D)DS Continuation

First we briefly investigate the potential of the DS Continuation Method (DS-C) as standalone algorithm. For this, we first consider the three-objective problem DTLZ2 [35] for $n = 12$ those Pareto front is equal to the first quadrant of the unit sphere. Fig. 6 shows the numerical results obtained by NBI as well as from a coupling of DS-C and the recover algorithm [27] in order to obtain a 'global' view on the part of the solution set that is already computed: first a partition of the domain is generated via (small) boxes. Then, for every newly generated candidate solution x we checked if the box that contains x is already associated to a solution found by DS-C. If so, we have discarded x. Else, we have added it to the archive and have associated this vector to the respective box. As stopping criterion for the corrector we have chosen $\|x_{i+1} - x_i\| \leq 1e-6$. For NBI, we have used the quasi-normal $\hat{n} = (-1, -1, -1)^T$ as proposed in [11]. Apparently, NBI misses some regions of the front (the region which is outside the '$\hat{n}$-shadow' of the convex hull of individual minima) which is not the case for the recover algorithm which captures all regions of the front. The crucial difference is that the recover algorithm determines the search direction from a given solution which allows in this case to find points at the boundary of the Pareto set. Given the strong relation of DS and NBI, the recover method can thus be seen as a possible remedy for the '$\hat{n}$-shadow problem'. Another possible remedy is e.g. proposed in [36]. We also compared the recover method that used DS (called R-DS) with its original variant that uses the continuation method proposed in [18] (R-H) on the same example. While the approximations qualities were the same—a Hausdorff distance of the image of the solution set to the Pareto front $d_H = 0.03423$ for R-DS and $d_H = 0.03545$ for R-H—this does not hold for the computational cost: R-DS used a budget of 3,696 function and 935 Jacobian evaluation while R-H spent 4,786 function, 4,479 Jacobian, and 13,437 Hessian evaluations.

Despite its local nature, DS-C is not necessarily restricted to the connected components of the Pareto sets: recall that the search is performed along the boundary of the image which yields the possibility that along it further connected components of the Pareto front can be found. Fig. 7 shows a numerical result of DDS-C on ZDT3 [37] for $n = 30$ starting with the minimizer of the first objective and using the radius $\delta = 0.1$ for the neighborhood search. By performing a movement along the boundary of the domain, all connected components of the Pareto front could be found. Shown is also the result of the Zigzag Method [24] (shifted in f_2-space for sake of a better comparison) which also incorporates a movement along the boundary of the domain. The approximation qualities obtained by both methods are very similar while the costs differ: DDS Continuation used 2,036 function evaluations while the Zigzag Method needed 36,290 function evaluations plus another 36,290 Jacobian calls. Note that a budget of 2,036 function evaluations is equivalent to about 70 Jacobian calls if the Jacobians are approximated by finite differences. Thus, better results by gradient based continuation-like methods can hardly be expected.



(a) NBI



(b) Recover

Fig. 6. Numerical result for DTLZ2 for NBI and the DS Continuation: left the obtained front and right a projection to the $(f_1 - f_2)$-plane.

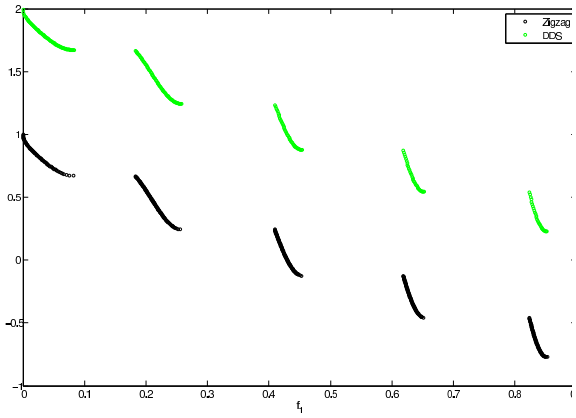


Fig. 7. Numerical result of DDS Continuation and Zigzag Method on ZDT3.

6.2 DS coupled with MOEA/D

MOEA/D employs a decomposition approach to convert the problem of approximating the Pareto front into a certain number of scalar optimization subproblems. There is a weight vector λ_i associated to each scalar subproblem i. From a practical point of view, this weight can be used to steer the local search over this direction. Alg. 4 describes the coupling of DS (or DDS) and MOEA/D. The notation regarding MOEA/D procedures and parameters is taken from [3]. There exist more sophisticated approaches to control the application of the local search for this (for example [38–40]); most of them have been proposed for discrete domains and are valuable to explore for future work.

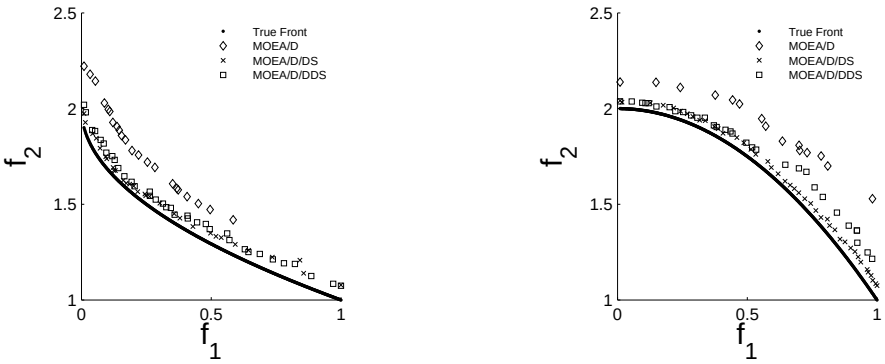
Algorithm 4 Hybrid MOEA/D (MOEA/D/DS and MOEA/D/DDS)

Set the weight vectors λ_i and the neighborhoods $B(i) = \{i_1, \ldots, i_T\}$ for each decomposed problem ($\lambda_{i_1}, \ldots, \lambda_{i_T}$ are the T closest weight vectors to λ_i).
Set up an initial population $P_0 = \{x_1, \ldots, x_N\}$.
Initialize the reference point z, $EP = \emptyset$, $gen = 1$.
repeat
    for $i = 1, \ldots, N$ **do**
        Select two indices k, l from $B(i)$ and generate, using genetic operators, a new solution y_i from x_k and x_l.
        Improving stage: use y_i to replace x_i and its corresponding x_j, $j \in B(i)$, regarding the corresponding scalar problem to λ_i.
        Apply the LS procedure to y_i. (Alg. 5)
        Update the reference point z.
        Remove from EP all the vectors dominated by y_i, and add it to EP if no vectors in EP dominate y_i.
    end for
    $gen = gen + 1$.
until Stopping criteria is satisfied
return EP.

Algorithm 5 LS Procedure (gen, i)

if $Start_{ls} \leq gen$ **then**
    if $mod(gen, k_{ls}) == 0$ and $mod(i, h_{ls}) == 0$ **then**
        if T_{ls} has not been reached **then**
            Apply, up to D_{ls} times, the LS (DS/DDS) procedure to y_i in order to get y_i'
            Set $y_i \leftarrow y_i'$.
        end if
    end if
end if
return y_i

For the sake of comparison we have used the ZDT problems [35] and tested over the functions ZDT1 to ZDT4 (using the modifications presented in [41] in order to make them unconstrained and differentiable over the entire domain). The parameter setting used for this benchmark was: population size = 50, neighborhood size $T = 10$, function evaluations for ZDT 1 to 3 = 5,000, function evaluations for ZDT 4= 10,000, initial step size = 1, $h_{ls} = 1$, $k_{ls} = 20$, $Start_{ls} = 20$, r for DDS = 7, D_{ls}=3, T_{ls} for ZDT 1 to 3 = 500, T_{ls} for ZDT 4 = 1,000, where T_{ls} denotes the number of function evaluations after which the local search is not applied any more. Further, we have used the UF functions of the CEC09-competition problem suite [42]. The code for this specific version of MOEA/D was taken from [43]. The parameters were: population size = 120, neighborhood size T =60, function evaluations = 50,000, replaced neighbors T_r =6, initial step size=1, D_{ls}=3, F (for Differential Evolution (DE)) = 0.5, CR (for DE) = 1, start of LS application = generation 10, r for DDS = 7, $h_{ls} = 1$, $k_{ls} = 10$, $Start_{ls} = 20$, T_{ls} =1,500. Different coding for the basis MOEA/D is justified from a practical point of view, since each particular algorithm is known to have the best performance for that corresponding benchmark. Since it is accepted that the Tchebycheff approach is the most efficient scalarization function for MOEA/D in terms of function evaluations, we have chosen this variant for comparison for both test suites. The performance indicators used were the averaged Hausdorff distance Δ_2 ([44]) and the Hypervolume indicator ([45]). Table 2 compiles the average over 30 independent runs. We can observe that MOEA/D/DS and MOEA/D/DDS have the best values in most of the cases. Some plots of the output population are shown in Fig. 8.



(a) ZDT 1 (b) ZDT 2

Fig. 8. Numerical results for MOEA/D (diamonds), MOEA/D/DS (crosses) and MOEA/D/DDS (squares) on the benchmark models ZDT 1 to ZDT 4. The true Pareto fronts are indicated by dotted lines.

7 Conclusions and Future Work

We have presented the Directed Search method for unconstrained MOPs which allows to steer the search into any given direction d in objective space. Based on this idea we have presented a class of descent methods and a novel continuation procedure. Both methods are applicable with and without gradient information. For the latter, neighborhood information can be expoited which makes the DS an interesting candidate as local searcher within memetic algorithms. We have further on illustrated both standalone algorithms on some numerical examples and have shown the potential of DS within MOEA/D.
As a by-product we were able to explain the behavior of multi-objective stochastic local search using the DS approach. We could show that both movement toward and along the Pareto front—which are closely related to the terms spread and convergence—are inherent features in MOSLS which explains a facet of the huge success of global multi-objective stochastic search algorithms such as specialized evolutionary algorithms. We conjecture that this new insight might be interesting for the design of future memetic strategies.

For future work, we intend to adapt DS to constrained models which is not done yet but needs careful considerations. Further, the improvement of the hybrids of DS and global search methods will be an important task. In this paper, we have considered a general purpose MOEA. In order to tap the full potential of the DS, however, we will have to specialize on particular performance indicators. The reason for this is that such indicators implicitly transform the MOP into a scalar optimization problem, and hence, a greedy search direction for every point is well-defined, and this direction is even defined in objective space in case of Pareto *front* approximations. Hence, the DS comes as a natural choice. Finally, an application to many objective problems (i.e., $k > 3$) by means of the hybrid MOEA would an interesting task which is, however, strongly related to the previous problem as the 'optimal' distribution of the solutions along the Pareto set/front is not accepted yet.

Acknowledgements

The first author acknowledges support from Conacyt project no. 128554.

References

1. K. Deb, Multi-Objective Optimization using Evolutionary Algorithms, John Wiley & Sons, Chichester, UK, 2001.
2. N. Beume, B. Naujoks, M. Emmerich, SMS-EMOA: Multiobjective selection based on dominated hypervolume, European Journal of Operational Research 181 (3) (2007) 1653–1669.
3. Q. Zhang, H. Li, MOEA/D: A multi-objective evolutionary algorithm based on decomposition, IEEE Transactions on Evolutionary Computation 11 (6) (2007) 712–731.

4. J. Knowles, D. Corne, M-PAES: a memetic algorithm for multiobjective optimization, in: Proceedings of the IEEE Congress on Evolutionary Computation. Piscataway, New Jersey, 2000, pp. 325 – 332.
5. H. Ishibuchi, T. Yoshida, T. Murata, Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling, IEEE Transactions on Evolutionary Computation 7 (2) (2003) 204–223.
6. A. Lara, G. Sanchez, C. A. C. Coello, O. Schütze, HCS: A new local search strategy for memetic multiobjective evolutionary algorithms, IEEE Transactions on Evolutionary Computation 14 (1) (2010) 112–132.
7. M. Vasile, F. Zuiani, Multi-agent collaborative search: an agent-based memetic multi-objective optimization algorithm applied to space trajectory design, Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering 225 (11) (2011) 1211–1227.
8. F. Zuiani, M. Vasile, Multi agent collaborative search based on tchebycheff decomposition, Computational Optimization and Applications 56 (1) (2013) 189–208.
9. B. Roy, Problems and methods with multiple objective functions, Mathematical Programming 1 (1971) 239–266.
10. F. W. Gembicki, Y. Y. Haimes, Approach to performance and multiobjective sensivity optimization: The goal attainment method, IEEE Transactions on Automatic Control 20 (1975) 769–771.
11. I. Das, J. Dennis, *Normal-boundary intersection*: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems., SIAM Journal of Optimization 8 (1998) 631–657.
12. A. Potschka, F. Logist, J. F. V. Impe, H. G. Bock, Tracing the Pareto frontier in bi-objective optimization problems by ODE techniques., Numer. Algorithms 57 (2) (2011) 217–233.
13. P. A. N. Bosman, On gradients and hybrid evolutionary algorithms for real-valued multiobjective optimization, IEEE Trans. Evolutionary Computation 16 (1) (2012) 51–69.
14. O. Schütze, A. Lara, C. A. C. Coello, The directed search method for unconstrained multi-objective optimization problems, in: Proceedings of the EVOLVE – A Bridge Between Probability, Set Oriented Numerics, and Evolutionary Computation, 2011, pp. 1–4.
15. E. Mejia, O. Schütze, A predictor corrector method for the computation of boundary points of a multi-objective optimization problem, in: CCE 2010, 2010, pp. 395–399.
16. A. Lara, S. Alvarado, S. Salomon, G. Avigad, C. A. C. Coello, O. Schütze, The gradient free directed search method as local search within multi-objective evolutionary algorithms, in: EVOLVE II, 2013, pp. 153–168.
17. V. Pareto, Manual of Political Economy, The MacMillan Press, 1971 (original edition in French in 1927).
18. C. Hillermeier, Nonlinear Multiobjective Optimization - A Generalized Homotopy Approach, Birkhäuser, 2001.
19. H. Kuhn, A. Tucker, Nonlinear programming, in: J. Neumann (Ed.), Proceeding of the 2nd Berkeley Symposium on Mathematical Statistics and Probability, 1951, pp. 481–492.
20. J. Nocedal, S. Wright, Numerical Optimization, Springer Series in Operations Research and Financial Engineering, Springer, 2006.
21. P. Deuflhard, F. Borneman, Scientific Computing with Ordinary Differential Equations, Texts in Applied Mathematics 42, Springer New York, 2002.

22. E. L. Allgower, K. Georg, Numerical Continuation Methods, Springer, 1990.
23. K. Harada, J. Sakuma, S. Kobayashi, I. Ono, Uniform sampling of local Pareto-optimal solution curves by Pareto path following and its applications in multi-objective ga, in: GECCO, 2007.
24. H. Wang, Zigzag search for continuous multiobjective optimization, Informs J Computing 25 (4).
25. S. Schäffler, R. Schultz, K. Weinzierl, A stochastic method for the solution of unconstrained vector optimization problems, Journal of Optimization Theory and Applications 114 (1) (2002) 209–222.
26. M. E. Henderson, Multiple parameter continuation: Computing implicitly defined k-manifolds, I J Bifurcation and Chaos 12 (3) (2003) 451 – 476.
27. O. Schütze, A. Dell'Aere, M. Dellnitz, On continuation methods for the numerical treatment of multi-objective optimization problems, in: J. Branke et. al (Ed.), Practical Approaches to Multi-Objective Optimization, no. 04461 in Dagstuhl Seminar Proceedings, 2005.
28. J.-D. Boissonnat, A. Ghosh, Triangulating smooth submanifolds with light scaffolding, Mathematics in Computer Science 4 (4) (2010) 431 – 461.
29. M. Brown, R. E. Smith, Directed multi-objective optimisation, International Journal of Computers, Systems and Signals 6 (1) (2005) 3–17.
30. M. Köppen, K. Yoshida, Many-objective particle swarm optimization by gradual leader selection, in: ICANNGA 2007, Springer-Verlag, Berlin, Heidelberg, 2007, pp. 323–331.
31. O. Schütze, M. Laumanns, E. Tantar, C. A. C. Coello, E.-G. Talbi, Convergence of stochastic search algorithms to gap-free Pareto front approximations, in: GECCO-2007, 2007, pp. 892–901.
32. H. Ishibuchi, T. Murata, Multi-objective genetic local search algorithm, in: Proc. of 3rd IEEE Int. Conf. on Evolutionary Computation, Nagoya, Japan, 1996, pp. 119–124.
33. I. S. Domínguez, A. H. Aguirre, S. I. Valdez, A new EDA by a gradient-driven density, in: Parallel Problem Solving from Nature - PPSN XIII - 13th International Conference, 2014, pp. 352–361.
34. S. Alvarado, El punto a punto de las técnicas de búsquedas local para algoritmos de optimización multiobjetivo, MSc thesis, Cinvestav-IPN (2012).
35. K. Deb, L. Thiele, M. Laumanns, E. Zitzler, Scalable test problems for evolutionary multiobjective optimization, in: A. Abraham, L. Jain, R. Goldberg (Eds.), Evolutionary Multiobjective Optimization. Theoretical Advances and Applications, Springer, USA, 2005, pp. 105–145.
36. R. Motta, S. A. M. Silvana, P. Lyra, A modified NBI and NC method for the solution of n-multiobjective optimization problems, Struct. Multidiscip. Optim. 46 (2) (2012) 239–259.
37. E. Zitzler, K. Deb, L. Thiele, Comparison of multiobjective evolutionary algorithms: Empirical results, Evol. Comput. 8 (2) (2000) 173–195.
38. A. Caponio, F. Neri, Integrating cross-dominance adaptation in multi-objective memetic algorithms, Springer, 2009.
39. J.-Y. Lin, Y.-P. Chen, Analysis on the collaboration between global search and local search in memetic computation, Evolutionary Computation, IEEE Transactions on 15 (5) (2011) 608–623.
40. J. Knowles, D. Corne, Memetic algorithms for multiobjective optimization: issues, methods and prospects, in: Recent advances in memetic algorithms, Springer, 2005, pp. 313–352.

41. P. Shukla, On gradient based local search methods in unconstrained evolutionary multi-objective mptimization, in: S. O. et al. (Ed.), EMO 2007, 2007, pp. 96–110.
42. Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, W. Liu, S. Tiwari, Multi-objective o?ptimisation test instances for the CEC 2009 special session and competition, Working Report CES-887, School of Computer Science and Electrical Engineering, University of Essex, revised on 20/04/2009 (2008).
43. Q. Zhang, W. Liu, H. Li, The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances, in: Evolutionary Computation, 2009. CEC'09. IEEE Congress on, IEEE, 2009, pp. 203–208.
44. O. Schütze, X. Esquivel, A. Lara, C. A. C. Coello, Using the averaged Hausdorff distance as a performance measure in evolutionary multi-objective optimization, IEEE Transactions on Evolutionary Computation 16 (4) (2012) 504–522.
45. E. Zitzler, Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications, Ph.D. thesis, ETH Zurich, Switzerland (1999).

Table 1. Δ_1 values and standard deviations obtained by GS and SNS on several benchmark MOPs. Hereby, n denotes the number of decision variables, k the number of objectives, MM multi-modality, and D a disconnected Pareto front of the problem. The results are averaged over 30 independent runs with a budget of 1,000 function calls, and the best value is displayed in bold.

Problem	GS	SNS	
ZDT 1	3.67746	**2.53825**	$n = 30$, $k = 2$
(std.dev)	(0.25024)	(0.38357)	
ZDT 2	4.25890	**3.04861**	$n = 30$, $k = 2$
(std.dev)	(0.26437)	(0.45873)	
ZDT 3	3.65511	**2.58236**	$n = 30$, $k = 2$, MM, D
(std.dev)	(0.29973)	(0.36736)	
ZDT 4	**490.89280**	592.40396	$n = 10$, $k = 2$, MM
(std.dev)	(19.12823)	(3.22944)	
ZDT 6	11.46799	**8.62762**	$n = 10$, $k = 2$, MM
(std.dev)	(0.41380)	(1.07383)	
DTLZ 1	109.98199	**41.62222**	$n = 6$, $k = 2$, MM
(std.dev)	(21.23505)	(10.26279)	
DTLZ 2	0.41768	**0.20813**	$n = 11$, $k = 2$, MM
(std.dev)	(0.04080)	(0.10050)	
DTLZ 3	723.45940	**153.65992**	$n = 11$, $k = 2$, MM
(std.dev)	(58.65155)	(26.21516)	
DTLZ 4	1.28054	**0.43433**	$n = 11$, $k = 2$
(std.dev)	(0.20119)	(0.21565)	
Fonseca	0.75282	**0.66993**	$n = 10$, $k = 2$
(std.dev)	(0.04957)	(0.15155)	
Kursawe	42.40731	**39.07086**	$n = 3$, $k = 2$
(std.dev)	(0.04730)	(1.42769)	
Poloni	13.913745388	**7.0603285851**	$n = 2$, $k = 2$, MM, D
(std.dev)	(0.4184712873)	(0.6042983239)	
Viennet	3.11828	**2.39613**	$n = 2$, $k = 3$, D
(std.dev)	(0.04413)	(0.11528)	
Viennet 2	0.71608	**0.21305**	$n = 2$, $k = 3$
(std.dev)	(0.01536)	(0.08833)	

Table 2. Results on Δ_2 and *Hypervolume* indicators for the ZDT and UF problems. These results show the average and standard deviation value over 30 independent runs. In the table, MD abbreviates MOEA/D.

	Δ_2			*Hypervolume*		
Problem	MD	MD/DDS	MD/DS	MD	MD/DDS	MD/DS
ZDT 1	0.52723	**0.47869**	**0.41636**	114.6688	**115.8445**	**116.6319**
(std.dev)	(0.19902)	(0.22428)	(0.22606)	(1.5695)	(1.7952)	(1.8084)
ZDT 2	0.91824	**0.74658**	**0.66601**	110.4825	**112.1839**	**113.0544**
(std.dev)	(0.42883)	(0.4998)	(0.49617)	(3.2805)	(3.9638)	(4.0418)
ZDT 3	0.81688	**0.77392**	**0.71203**	119.552	**120.4113**	**121.2138**
(std.dev)	(0.25095)	(0.25119)	(0.27828)	(2.6732)	(2.7322)	(2.8983)
ZDT 4	7.6429	**7.4658**	**7.5273**	40.2909	**42.1296**	**41.5298**
(std.dev)	(2.813)	(2.8661)	(2.8202)	(30.3759)	(30.8694)	(30.4259)

	Δ_2			*Hypervolume*		
Problem	MD	MD/DDS	MD/DS	MD	MD/DDS	MD/DS
UF1	0.0355	**0.0167**	**0.0198**	0.9663	**0.9789**	**0.9763**
(std.dev.)	(0.0015)	(0.0008)	(0.0005)	(0.0018)	(0.0013)	(0.0011)
UF2	**0.0277**	0.0280	**0.0243**	0.9722	**0.9731**	**0.9784**
(std.dev.)	(0.00178)	(0.0025)	(0.0015)	(0.0020)	(0.0022)	(0.0015)
UF3	0.0925	**0.0668**	**0.0798**	0.9124	**0.9170**	**0.9192**
(std.dev.)	(0.0034)	(0.0024)	(0.0023)	(0.0055)	(0.0034)	(0.0033)
UF4	0.0878	**0.0822**	**0.0797**	0.9186	**0.9261**	**0.9266**
(std.dev.)	(0.0006)	(0.0004)	(0.0077)	(0.0008)	(0.0006)	(0.0095)
UF5	0.8261	**1.1005**	**0.9493**	0.7672	**0.7047**	**0.7198**
(std.dev.)	(0.0166)	(0.02086)	(0.0197)	(0.0050)	(0.0047)	(0.0053)
UF6	0.2918	**0.2417**	**0.2527**	0.8325	**0.8537**	**0.8505**
(std.dev.)	(0.0158)	(0.0115)	(0.0140)	(0.0073)	(0.0060)	(0.0064)
UF7	0.0258	**0.0148**	**0.0137**	0.9703	**0.9844**	**0.9854**
(std.dev.)	(0.0035)	(0.0005)	(0.0005)	(0.0041)	(0.0063)	(0.0066)
UF8	0.2441	**0.1230**	0.1967	0.9494	**0.96642**	0.9529
(std.dev.)	(0.1326)	(0.0674)	(0.1349)	(0.0185)	(0.0086)	(0.0181)
UF9	0.3385	**0.2652**	0.3535	0.9644	0.9760	**0.9800**
(std.dev.)	(0.1004)	(0.0877)	(0.1337)	(0.0198)	(0.0228)	(0.2072)
UF10	2.5567	**2.3465**	2.9263	0.0543	**0.0807**	0.0537
(std.dev.)	(0.3437)	(0.3277)	(0.2498)	(0.0390)	(0.0666)	(0.0343)
CONVEX	2.2122	2.1851	**2.0629**	0.9556	0.9560	**0.9709**
(std.dev.)	(0.0691)	(0.0481)	(0.0579)	(0.0026)	(0.0016)	(0.0051)