HCS: A New Local Search Strategy for Memetic Multi-Objective Evolutionary Algorithms

Adriana Lara[1], Gustavo Sanchez[2],
Carlos A. Coello Coello[1⋆], and Oliver Schütze[1]

[1] CINVESTAV-IPN, Computer Science Department, Mexico City, Mexico
{alara,ccoello,schuetze}@cs.cinvestav.mx,
[2] Simon Bolivar University, Caracas, Venezuela
gsanchez@usb.ve

Abstract. In this paper we propose and investigate a new local search strategy for multi-objective memetic algorithms. More precisely, we suggest a novel iterative search procedure, the HCS (*Hill Climber with Sidestep*), which is designed for the treatment of multi-objective optimization problems, and show further on two possible ways to integrate the HCS into a given evolutionary strategy leading to new memetic (or hybrid) algorithms. The pecularity of the HCS is that it is intended to be capable both of moving toward and along the (local) Pareto set depending on the distance of the current iterate toward this set. The local search procedure utilizes the geometry of the directional cones of such optimization problems and works with or without gradient information. Finally, we present some numerical results on some well-known benchmark problems indicating the strength of the local search strategy as a standalone algorithm as well as its benefit when used within a MOEA. For the latter we use the state of the art algorithms NSGA-II and SPEA2 as base MOEAs.

Key words: multi-objective optimization, heuristic search, memetic algorithm, hill climber, Pareto set

1 Introduction

In a variety of applications in industry and finance one is faced with the problem that several objectives have to be optimized concurrently leading to a *multi-objective optimization problem* (MOP). As a general example, two common goals in product design are certainly to maximize the quality of the product and to minimize its cost. Since these two goals are typically contradicting, it comes as no surprise that the solution set—the so-called *Pareto set*—of an MOP does in general not consist of one single solution but rather of an entire set of solutions (see Section 2 for a more detailed discussion).

⋆ The third author is also affiliated to the UMI-LAFMIA 3175 CNRS

For the computation of the Pareto set of a given MOP there exist several classes of algorithms. There exist, for instance, a variety of mathematical programming techniques such as scalarization methods (see e.g., [40, 17, 11] and references therein) or multi-objective continuation methods [24] which are in general very efficient in finding single solutions—the most prominent example is probably Newton's method which is used within continuation methods and which has local quadratic convergence [42]—or even entire sets of solutions but which may have trouble in finding the entire (global) Pareto set in certain cases. In contrast, there are global methods including multi-objective evolutionary algorithms (MOEAs) [12, 10] or subdivision techniques [55, 15] which accomplish the 'global task' exceedingly but offer in turn (much) slower convergence rates compared to the algorithms mentioned above.
Another class of algorithms are the *memetic* (or hybrid) algorithms, i.e., algorithms which hybridize MOEAs with local search strategies (see Section 2.2 for an overview of existing methods). This is done in order to obtain an algorithm which offers on one side the globality and robustness of the evolutionary approach, but on the other side also an improved overall performance by the inclusion of well directed local search.
The scope of this paper is to contribute to the last category of algorithms. To be more precise, we propose a new point-wise local search prodecure, the *Hill Climber with Sidestep* (HCS), which is capable of moving both toward (using hill climber techniques) and along (sidestep) the Pareto set according to the distance of the current iterate to this set. In particular the automatic switch of the movement represents a novelty which makes the operator universally applicable within any given MOEA. We present the HCS as local search procedure and demonstrate on two examples that it can be beneficial to integrate the HCS into a MOEA. According to the classification made in [58] the resulting algorithms NSGA-II-HCS and SPEA2-HCS, which are based on NSGA-II and SPEA2, are exploitation-embedded hybrid methods.

The remainder of this paper is organized as follows: In Section 2, we state some theoretical background and give an overview on existing memetic MOEAs (MEMOEAs). In Section 3, we introduce the underlying ideas of the HCS and propose two realizations, a gradient free version and one version which exploits gradient information, both presented as standalone algorithms. In Section 4 we address the integration of the HCS into a MOEA and propose two possible memetic strategies where NSGA-II and SPEA2 are used as base MOEAs. In Section 5, we show some numerical results on both the HCS as a standalone algorithm as well as on the memetic strategies. Finally, some conclusions are drawn in Section 6.

2 Background

Here we briefly describe the background required for this paper: we introduce to the notion of multi-objective optimization and give an overview on existing memetic strategies for the numerical treatment of such problems.

2.1 Multi-objective Optimization (MOO)

In a variety of applications in industry and finance a problem arises that several objective functions have to be optimized concurrently leading to *multi-objective optimization problems* (MOPs). In the following we consider continuous MOPs which are of the following form:

$$\min_{x \in Q} \{F(x)\}, \tag{MOP}$$

where $Q \subset \mathbb{R}^n$ is the domain and the function F is defined as the vector of the objective functions

$$F : Q \to \mathbb{R}^k, \qquad F(x) = (f_1(x), \ldots, f_k(x)),$$

and where each $f_i : Q \to \mathbb{R}$ is continuous. In this work we will mainly consider the unconstrained case (i.e., $Q = \mathbb{R}^n$) but will give some possible modifications of the algorithms in case Q is defined by inequality constraints such as box constraints.
Central for the treatment of MOPs is the concept of the optimality of a point $x \in Q$ which is not analogue to the scalar objective case ($k = 1$). In the multi-objective case ($k > 1$) the concept of *dominance* is used which dates back over a century and was proposed first by Pareto [44].

Definition 1. *(a) Let $v, w \in \mathbb{R}^k$. Then the vector v is* less than *w ($v <_p w$), if $v_i < w_i$ for all $i \in \{1, \ldots, k\}$. The relation $\leq_p$ is defined analogously.*
(b) 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 $F(x) \leq_p F(y)$ and $F(x) \neq F(y)$, else y is called non-dominated by x.*
(c) A point $x \in Q$ is called Pareto optimal *or a* Pareto point *if there is no $y \in Q$ which dominates x.*

In case all the objectives $f_i, i = 1, \ldots, k$, of the MOP are differentiable the following theorem of Kuhn and Tucker [38] states a necessary condition for Pareto optimality for unconstrained MOPs. For a more general formulation of the theorem we refer e.g. to [40].

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*

$$\sum_{i=1}^{k} \alpha_i \nabla f_i(x^*) = 0. \tag{1}$$

The theorem claims that the vector of zeros can be written as a convex combination of the gradients of the objectives at every Pareto point. Obviously, (1) is not a sufficient condition for Pareto optimality. On the other hand, points satisfying (1) are certainly 'Pareto candidates'.

Definition 2. *A point $x \in \mathbb{R}^n$ is called a* Karush–Kuhn–Tucker point[3] *(KKT–point) if there exist scalars $\alpha_1, \ldots, \alpha_k \geq 0$ such that $\sum_{i=1}^{k} \alpha_i = 1$ and that Equation (1) is satisfied.*

The set of all (globally) Pareto optimal solutions is called the *Pareto set*, denoted by P_Q. It has been shown that this set typically—i.e., under mild regularity assumptions—forms a $(k-1)$-dimensional object [24]. The image of the Pareto set, $F(P_Q)$, is called the *Pareto front*. Since we are involving local search strategies in our work we have to take also locally optimal points into consideration. In the following, let $\mathcal{P}$ be the set of local Pareto points. In case the MOP is differentiable, $\mathcal{P}$ can be considered as the set of KKT–points.

Theorem 2 ([49]). *Let (MOP) be given and $q : \mathbb{R}^n \to \mathbb{R}^n$ be defined by*

$$q(x) = \sum_{i=1}^{k} \hat{\alpha}_i \nabla f_i(x), \tag{2}$$

where $\hat{\alpha}$ is a solution of

$$\min_{\alpha \in \mathbb{R}^k} \left\{ \| \sum_{i=1}^{k} \alpha_i \nabla f_i(x) \|_2^2 ; \alpha_i \geq 0, i = 1, \ldots, k, \sum_{i=1}^{k} \alpha_i = 1 \right\}. \tag{3}$$

Then either $q(x) = 0$ or $-q(x)$ is a descent direction for all objective functions $f_1, \ldots, f_k$ in x.

The theorem states that for every point $x \in Q$ which is not a KKT–point a descent direction (i.e., a direction where all objectives' values can be improved) can be found by solving the quadratic optimization problem (3). In case $q(x) = 0$ the point x is a KKT–point. Thus, a test for optimality has to be performed automatically when computing the descent direction for a given point $x \in Q$.

2.2 Memetic Strategies in MOO

Hybridization of MOEAs with local search algorithms has been investigated for more than twelve years, starting shortly after the first MOEAs were proposed [36, 10]. One of the first MEMOEAs for models on discrete domains was presented in [29, 30] as a 'Multi-Objective Genetic Local Search' (MOGLS) approach. The authors proposed to use the local search method after classical variation operators are applied. A randomly drawn scalarizing function is used to assign fitness for parent selection.

[3] Named after the works of Karush [33] and Kuhn & Tucker [38].

Jaszkiewicz [32] proposed an algorithm called the Pareto Memetic Algorithm (PMA). This algorithm uses an unbounded 'current set' of solutions (CS) and from this selects a small 'temporary population' (TP) that comprises the best solutions with respect to a scalarizing function. Then TP is used to generate offspring by crossover. Jaszkiewicz suggests that scalarizing functions are particularly better at encouraging diversity than dominance ranking methods used in most MOEAs.

Another important MEMOEA, called M-PAES, was proposed in [35]. Unlike Ishibuchi's and Jaszkiewicz's approaches, M-PAES does not use scalarizing functions, but employs instead a Pareto ranking based selection coupled with a grid-type partition of the objective space. Two archives are used: one that maintains the global non-dominated solutions and the other that is used as the comparison set for the local search phase.

In [41], the authors proposed a local search process with a generalized replacement rule. Ordinary two-replacement rules based on the dominance relation are usually employed in a local search for multiobjective optimization. One is to replace a current solution with a solution which dominates it. The other is to replace the solution with a solution which is not dominated by it. The movable area with the first rule is very small when the number of objectives is large. On the other hand, it is too huge to move efficiently with the latter. The authors generalize these extreme rules by counting the number of improved objectives for a given candidate.

Caponio and Neri [8] proposed the *Cross Dominant Multi-Objective Memetic Algorithm* (CDMOMA), which consists of the NSGA-II combined with two local search engines: a multi-objective implementation of the Rosenbrock algorithm [47], which performs very small movements, and the Pareto Domination Multi-Objective Simulated Annealing (PDMOSA) approach proposed in [61], which performs a more global exploration. The main idea of this approach is to use the mutual dominance between non-dominated solutions belonging to consecutive generations (this is called cross-dominance by the authors) as a parameter that indicates the degree of improvement achieved. Such value is applied with certain probability (based on a generalized Wigner semicircle distribution) to decide which of the two local search engines to apply. CDMOMA was found to have a similar or better performance than both NSGA-II and SPEA2 in several benchmark problems and in a real-world electrical engineering problem.

Soliman et al. [60] proposed a memetic version of a co-evolutionary multi-objective differential evolution (CMODE-MEM) approach, which evolves both a population of solutions and promising search directions. The fitness of a search direction is based on its capability to improve solutions. Local search is applied to a portion of the population after each generation. The performance of CMODE-MEM was assessed using several benchmark problems, and results were compared with respect to NSGA-II, NSDE [27, 28] and CMODE (without local search). The results indicated that the two versions of CMODE (with and without local search) were the best overall performers.

In [62, 64, 65, 63], methods are presented which are hybrids of evolutionary search algorithms and multi-agent strategies where the task of the agents is to perform the local search.

The continuous case—i.e., continuous objectives defined on a continuous domain—was explicitly first explored in [19], where a neighborhood search was applied to NSGA-II [13]. In their initial work, the authors applied the local search only after NSGA-II had ended. To do this, the authors applied a local search using a weighted sum of objectives. The weights were computed for each solution based on its location in the Pareto front such that the direction of improvement is roughly in the direction perpendicular to the Pareto front. Later works compare this approach with the same local search method being applied after every generation. Evidently, they found that the added computational workload impacted efficiency.

In [25] a gradient based local algorithm (Sequential Quadratic Programming (SQP)), was used in combination with NSGA-II and SPEA [72] to solve the ZDT benchmark suite [70]. The authors stated that if there are no local Pareto fronts, the hybrid MOEA has faster convergence toward the true Pareto front than the original one, either in terms of the objective function evaluations or in terms of the CPU time consumed (since a gradient based algorithm is utilized, the sole usage of the number of function calls as a basis for a comparison can be misleading). Furthermore, they found that the hybridization technique does not decrease the solution diversity.

In [1] three different local search techniques were hybridized with MOGA: simulated annealing, hill climbing and tabu search. The three hybrid algorithms were applied to ZDT problems and compared to the standard MOGA considering the same number of function evaluations. An adaptive mechanism was proposed to determine the size of the neighborhood for each individual. The authors claim that the 'MOGA - Hill climbing' was outperforming the standalone MOGA and the MOGA hybridized with the other local search techniques. They noted also that the process of fine-tuning the non-dominated individuals resulted in unwanted genetic drift and premature convergence: none of the hybrids were dedicated for distribution enhancement.

In [46], the authors proposed a hybrid technique that combines the robustness of MOGA-II [45] with the accuracy and speed of NBI-NLPQLP, an accurate and fast converging algorithm based on a classical gradient method. The methodology consists of starting with a preliminary robust MOGA-II run, then isolating each single portion of the Pareto curve as an independent problem, each of which is treated with an independent accurate NBI-NLPQLP run.

In [68] the proposed local search process employs quadratic approximations for all objective functions. The samples gathered by the algorithm along the evolutionary process are used to fit these quadratic approximations around the point selected for local search. After that, a locally improved solution is estimated from the quadratic associated problem. The hybridization of the procedure is demonstrated with SPEA 2 [71].

A succesful hybrid approach was proposed in [26]. The authors proposed the algorithm MO-CMA-ES, a multi-objective CMA-ES [21], which combines the strategy parameter adaptation of evolutionary strategies with a multi-objective selection based on non-dominated sorting. The MO-CMA-ES is independent of the chosen coordinate system and its behavior does not change if the search space is translated, rotated, and/or rescaled. The authors claim that MO-CMA-ES significantly outperforms NSGA-II on all but one of the considered test problems: the NSGA-II is faster only on the ZDT4 problem where the optima form a regular axis-parallel grid, because NSGA-II heavily exploits this kind of separability.

In [67], a novel evolutionary algorithm (EA) for constrained optimization problems is presented: the so-called hybrid constrained optimization EA (HCOEA). The algorithm combines multi-objective optimization with global and local search processes. In performing the global search, a niching genetic algorithm based on tournament selection is used. Meanwhile, the best infeasible individual replacement scheme is used as a local search operator for the purpose of guiding the population toward the feasible region of the search space. During the evolutionary process, the global search model effectively promotes high population diversity, and the local search model remarkably accelerates the convergence speed. HCOEA was tested on 13 benchmark functions, and the experimental results suggest that it is more robust and efficient than other state-of-the-art algorithms in terms of the selected performance metrics.

The use of gradient based hill climbing methods within NSGA-II has been proposed and studied in [57]. By this, the authors were able to accelerate the convergence of NSGA-II.

Finally, in [55, 52, 22, 53], hybrids can be found were heuristic methods are coupled with multi-objective continuation methods.

Concluding, it can be said that so far many authors have reported succesfull hybridizations of local search techniques with genetic algorithms. However, to the best of the authors knowledge, there exist basically three crucial questions which remain open in the design of (single- or multi-objective) memetic strategies (e.g., [23, 59, 37, 31, 39, 4]): Where shall a local search process be hybridized with a genetic algorithm? Which individuals should be fine-tuned and how much? And when shall the local refinement be applied?
In the following, we give a particular—but not all-embracing—response to these last questions.

3 HCS: The Hill Climber with Sidestep

In the following, we propose a novel iterative local search procedure, the HCS, which is designed to be uses within a memetic strategy. For sake of a better understanding, we present here the method as standalone algorithm. The integration of the HCS into MOEAs and the resulting modifications will be addressed in the next section.

Before we can come to the design of such a strategy, we have to ask ourselves what are the requirements for an iterative search procedure $\Phi : \mathbb{R}^n \to \mathbb{R}^n$ with

$$x_{l+1} = \Phi(x_l), \tag{4}$$

where $x_0 \in \mathbb{R}^n$ is a given initial solution and $\{x_i\}_{i \in \mathbb{N}_0}$ is the resulting sequence of iterates. Note that we are dealing with a point-wise iteration—i.e., input and output of Φ are a single points of the domain—, and not with a population based strategy. We are of the opinion that such a 'wish list' on Φ for the treatment of MOPs includes the following tasks:

(a) Φ should generate an improvement of the current iterate x_l if this one is not already 'close' to the $\mathcal{P}$, i.e., a point x_{l+1} with $x_{l+1} \prec x_l$.
(b) In case the current iterate x_l is already 'close' to $\mathcal{P}$, a search *along* $\mathcal{P}$ would be desired.
(c) The switch between the situations described in (a) and (b) should be done *automatically* according to the position of the current iterate x_l.
(d) The process should work with or without gradient information (whether or not provided by the model).
(e) The process should be capable of handling constraints of the MOP.

In (a) the 'classical' task of a hill climber as known for single-objective optimization problems [16, 34, 43, 48, 18, 39] is described. Item (b) contains a peculiarity of multi-objective optimization, namely that there is—using the climbing metaphor—no single mountain top but rather an entire ridge of mountain tops which forms $\mathcal{P}$ (respectively a set of ridges in case $\mathcal{P}$ is diconnected). The generation of such a point x_{l+1} can be regarded as a 'sidestep' relative to the current iterate x_l in the upward movement of the hill climber. Important for the efficiency of Φ within a memetic strategy is item (c), i.e., the capability to decide if case (a) or (b) is more appropriate.
In the following we describe two variants of such a function Φ which aims to fulfill the above wish list: one version of the HCS which is gradient free, and another version which involves gradient information.

3.1 HCS without Using Gradient Information

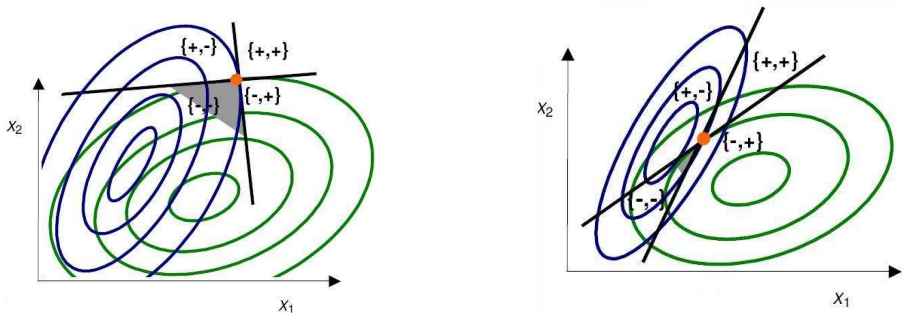
First, we describe the HCS algorithm for the case in which no gradient information is available, since that seems to be more relevant for common real-world engineering problems which is the main area of application for MOEAs. We concentrate here on the unconstrained case and possible modifications of the algorithm for the treatment of MOPs with inequality constraints are given below.

The method we describe here is based on the geometry of multi-objective optimization which has been studied in [7]. This work gives a good insight into the structure of such problems by analyzing the geometry of the directional cones of candidate solutions at different stages of the optimization process: when a point x_0 is 'far away' from any local Pareto optimal solution, the gradients'

objectives are typically aligned and the descent cone is almost equal to the half-spaces associated with each objective. Therefore, for a randomly chosen search direction ν, there is a nearly 50 % chance that this direction is a descent direction at x_0 (i.e., there exits an $h_0 \in \mathbb{R}_+$ such that $F(x_0 + h_0\nu) <_p F(x_0)$). If on the other side a point x_0 is 'close' to the Pareto set, the individual gradients are almost contradictory (compare also to the famous theorem of Kuhn and Tucker [38] which holds for points on $\mathcal{P}$), and thus, the size of the descent cone is extremely narrow, resulting in a small probability for a randomly chosen vector to be a descent direction. The two scenarios are depicted in Figure 1 for the bi-objective case. Hereby, $\{-,-\}$ and $\{+,+\}$ denote the descent and ascent cone, respectively. The symbol $\{-,+\}$ indicates that in this direction an improvement according to f_1 can be achieved while the values of f_2 will increase. To be more precise, if all objectives are differentiable, then the following equivalence holds for a search direction ν at a point x_0:

$$\nu \in \{-,+\} \quad \Longleftrightarrow \quad \langle \nabla f_1(x_0), \nu \rangle < 0 \quad \text{and} \quad \langle \nabla f_2(x_0), \nu \rangle > 0, \tag{5}$$

where $\langle \cdot, \cdot \rangle$ denotes the standard scalar product. Analogous statements hold for $\{+,-\}$.



(a) (b)

Fig. 1. The descent cone (shaded) for an MOP with 2 parameters and 2 objectives during initial (a) and final (b) stages of convergence. The descent cone shrinks to zero during the final stages of convergence. The figure is taken from [4].

The gradient free HCS is constructed on the basis of these observations. Given a point $x_0 \in Q$, the next iterate x_1 is selected as follows: a further point $\tilde{x}_1$ is chosen randomly from a neighborhood of x_0, say $\tilde{x}_1 \in B(x_0, r)$ with

$$B(x_0, r) := \{x \in \mathbb{R}^n : x_{0,i} - r_i \leq x_i \leq x_{0,i} + r_i \; \forall i = 1, .., n\}, \tag{6}$$

where $r \in \mathbb{R}^n_+$ is a given (problem dependent) radius. If $\tilde{x}_1 \prec x_0$, then $\nu := \tilde{x}_1 - x_0$ is a descent direction[4] at x_0, and along it a 'better' candidate can be searched, for example via line search methods (see below for one possible realization). If $x_0 \prec \tilde{x}_1$ the same procedure can be applied to the opposite direction (i.e., along $\nu := x_0 - \tilde{x}_1$) and starting with $\tilde{x}_1$. If x_0 is 'far away' from any local solution, the chance is, by the above discussion, quite high that domination occurs, either $\tilde{x}_1 \prec x_0$ or $x_0 \prec \tilde{x}_1$. If x_0 and $\tilde{x}_1$ are mutually non-dominating, the process will be repeated with further candidates $\tilde{x}_2, \tilde{x}_3, \ldots \in B(x_0, r)$. If only mutually non-dominated solutions $(\tilde{x}_i, x_0)$ are found within N_{nd} steps, this indicates, using the above observation, that the point x_0 is already near to the (local) Pareto set, and hence it is desirable to search along this set. This is because even if a descent direction would be available further improvements would very likely be negligible, and, hence, it is advisable to seek for further regions of the Pareto set. To perform such a sidestep it would be desirable to use the accumulated information obtained by the unsuccessful trials. Fundamental for the algorithm we present here is the fact that the 'unsuccessful' search directions $\nu_{i,1} := \tilde{x}_i - x_0$ and $\nu_{i,2} := x_0 - \tilde{x}_i = -\nu_{i,1}$ are located in the diversity cones. Further, there exists the following relation of $\nu_{i,1}$ and $\nu_{i,2}$: if $\nu_{i,1}$ is, for example, in the cone $\{+,-\}$, then $\nu_{i,2}$ is the opposite cone $\{-,+\}$ which is a direct consequence of (5). This holds for bi-objective MOPs, the general k-objective case is analogue.

Based on these observations we propose the following search directions. First we address the bi-objective case. If, for example, a search along $\{-,+\}$ after N_{nd} unsuccessful trials is sought, we propose to use the following one which uses the previous information:

$$\nu_{acc} = \frac{1}{N_{nd}} \sum_{i=1}^{N_{nd}} s_i \frac{\tilde{x}_i - x_0}{\|\tilde{x}_i - x_0\|}, \tag{7}$$

where

$$s_i = \begin{cases} 1 & \text{if } f_1(\tilde{x}_i) < f_1(x_0) \\ \text{-1} & \text{else} \end{cases} \tag{8}$$

By construction, ν_{acc} is in $\{-,+\}$, and by the averaging of the search directions we aim to obtain a direction which is 'perpendicular' to the (small) descent cone. Note that in this case ν_{acc} is indeed a 'sidestep' to the upward movement of the hill climbing process as desired, but this search direction does not neccessarily have to point along the Pareto set (see next subsection for a better guided search). A similar strategy for the search can be done for a general number k of objectives, however, leading to a larger variety for the search direction. For instance, for $k = 3$, there are six diversity cones which can be grouped by reflection as follows:

[4] In the sense that there exists a $\bar{t} \in \mathbb{R}_+$ such that $f_i(x_0 + \bar{t}\nu) < f_i(x_0)$, $i = 1, \ldots, k$, but not in the 'classical' sense, i.e., in case f_i is differentiable $\nabla f_i(x_0)^T \nu < 0$ is not guaranteed.

$$\begin{array}{lll} \{+,-,-\} & \text{and} & \{-,+,+\}\,, \\ \{+,-,+\} & \text{and} & \{-,+,-\}\,, \\ \{+,+,-\} & \text{and} & \{-,+,+\}\,. \end{array} \tag{9}$$

That is, for $k = 3$ there are three different groups of cones in which search directions can be divided (The sidesteps performed in Algorithm 6 of Section 4 are based on this idea). For a general k there are a total of $2^{k-1} - 1$ different groups making it less likely to find a perpendicular direction due to averaging within N_{nd} trials and within one of these cones. Alternatively to (7) one can e.g. use the accumulated information by taking the average search direction over *all* search directions as follows:

$$\nu_{acc} = \frac{1}{N_{nd}} \sum_{i=1}^{N_{nd}} \frac{\tilde{x}_i - x_0}{\|\tilde{x}_i - x_0\|}, \tag{10}$$

This direction has previously been proposed as a local guide for a multi-objective particle swarm algorithm in [6]. Note that this is a heuristic that does *not* guarantee that ν_{acc} indeed points to a diversity cone. In fact, it can happen that this vector points to the descent or ascent cone, though the probability for this is low for points x_0 'near' to a local solution due to the narrowness of these cones. However, in both cases Algorithm 1 acts like a classical hill climber—i.e., it searches for better points—which is still in the scope of the procedure (though the improvements may not be significant due to the vicinity of the current iterate to $\mathcal{P}$).

A pseudocode of the HCS for the bi-objective case which uses the strategies described above and the sidestep heuristic (7) is given in Algorithm 1. In the following we provide details for possible realizations of the line search and the handling of the constraints.

Sidestep direction The direction for the sidestep is determined by the value of i_0 (see line 5 and lines 15-20 of Algorithm 1). For simplicity, in Algorithm 1 the value of i_0 is chosen at random. In order to introduce an orientation to the search, the following modifications can be done in the bi-objective case: in the beginning, i_0 is fixed to 1 for the following iteration steps. When the sidestep (line 23 of Algorithm 1) has been performed N_s times during the run of an algorithm, this indicates that the current iteration is already near to the (local) Pareto set, and this vector is stored in x_{temp}. If in the following no improvements can be achieved according to f_1 within a given number N_i of sidesteps, the HCS 'jumps' back to x_{temp}, and a similar process is started but aiming for improvements according to f_2. That is, i_0 is set to -1 for the following steps. A possible stopping criterion, hence, could be to stop the process when no improvements can be achieved according to f_2 within another N_i sidesteps along $\{+,-\}$ (this has in fact been chosen as the stopping criterion in Section 5.1).

Algorithm 1 HCS1 (without using gradient information for $k = 2$)

Require: starting point $x_0 \in Q$, radius $r \in \mathbb{R}^n_+$, number N_{nd} of trials, MOP with $k = 2$

Ensure: sequence $\{x_l\}_{l\in\mathbb{N}}$ of candidate solutions

$a := (0, \ldots, 0) \in \mathbb{R}^n$
$nondom := 0$
$x_0^1 := x_0$
for $l = 1, 2, \ldots$ **do**
    set $x_l^1 := x_{l-1}^1$ and choose $x_l^2 \in B(x_l^1, r)$ at random
    choose $i_0 \in \{1, 2\}$ at random
    if $x_l^1 \prec x_l^2$ **then**
        $\nu_l := x_l^1 - x_l^2$
        compute $t_l \in \mathbb{R}_+$ and set $x_l^1 := x_l^2 + t_l\nu_l$.
        $nondom := 0, \quad a := (0, \ldots, 0)$
    else if $x_l^2 \prec x_l^1$ **then**
        proceed analogous to case "$x_l^1 \prec x_l^2$" with
        $\nu_l := x_l^2 - x_l^1$ and $x_l^1 := x_l^1 + t_l\nu_l$.
    else
        if $f_{i_0}(x_l^2) < f_{i_0}(x_l^1)$ **then**
            $s_l := 1$
        else
            $s_l := -1$
        end if
        $a := a + \frac{s_l}{N_{nd}} \frac{x_l^2 - x_l^1}{\|x_l^2 - x_l^1\|}$
        $nondom := nondom + 1$
        if $nondom = N_{nd}$ **then**
            compute $\tilde{t}_l \in \mathbb{R}_+$ and set $x_l^1 := x_l^1 + \tilde{t}_l a$.
            $nondom := 0, \quad a := (0, \ldots, 0)$
        end if
    end if
end for

Computation of t_l The situation is that we are given two points, say $x_0, x_1 \in \mathbb{R}^n$, such that $x_1 \prec x_0$. That is, there exists a subsequence $\{i_1, \ldots, i_l\} \subset \{1, \ldots, k\}$ with

$$f_{i_j}(x_1) < f_{i_j}(x_0), \quad j = 1, \ldots, l,$$

and thus, $\nu := x_1 - x_0$ is a descent direction for all f_{i_j}'s at the point x_0. For this (single objective) case there exist various strategies to perform the line search (see e.g., [16, 56]). One crucial problem is to find a good initial guess t^* for a suitable step size (which is e.g. given by 1 when using Newton's method). In case t^* is not already sufficient the step size can for instance be fine tuned by backtracking methods ([16]). Since x_1 can be very close to x_0 the distance $\|x_1 - x_0\|$ can thus not always serve as a good choice, and standard methods to obtain the initial guess do not apply. Instead, we propose the following heuristic to compute t^*. To capture the idea we begin with the scalar case, i.e., we are given a function $f : \mathbb{R} \to \mathbb{R}$, and values $t_0, t_1 \in \mathbb{R}$ with $t_0 < t_1$ and $f(t_0) < f(t_1)$. We define $\Delta := t_1 - t_0$, $t_l := t_0$, $t_m := t_1$, and $t_r := t_0 + 2\Delta$ and check if

$$\frac{f(t_m) - f(t_l)}{t_m - t_l} < \frac{f(t_r) - f(t_m)}{t_r - t_m}. \tag{11}$$

If the above equation is true, we suggest to approximate f by a quadratic polynomial $p(t) = at^2 + bt + c$ (the values of a, b, and c can be derived explicitly by the interpolation conditions $p(t_l) = f(t_l)$, $p(t_m) = f(t_m)$, and $p(t_r) = f(t_r)$, see [16]). The reason for (11) is because if the term in (11) is true then p is convex (see Figure 2 for an example), and thus, it is guaranteed that the extreme point of p, $t_p^* = -\frac{b}{2a}$, is a minimizer and takes its value in (t_0, ∞). Hence, t_p^* can be chosen as a guess for the minimizer of f. (This idea to approximate f locally by a quadratic polynomial was first proposed by Armijo [3]).
If (11) is false, quadratic approximation may not yield a useful result (in fact, in that case t_p^* may be negative), and we suggest to check condition (11) with the new data $t_l := t_m$, $t_m := t_r$, and $t_r := t_l + t_l + 4\Delta$ (i.e., doubling the step size for t_r). This process will be repeated until the boundary of the domain ∂Q is reached (in that case, take the maximal step size t_{max} as describe below) or (11) is true. The process will stop after a few iterations. If t_p^* is too large (i.e., if $f(t_p^*) > f(t_1)$), smaller step sizes can be found via backtracking.
In Algorithm 2, this idea is tranferred to the multi-objective case. Hereby, $f_{\nu,i}$ denotes the restriction of objective f_i to the line $x_0 + \mathbb{R}\nu$, i.e.,

$$f_{\nu,i}(t) = f_i(x_0 + t\nu), \tag{12}$$

and *quad_approx* the method to find the minimizer of the quadratic polynomial as described above.
Note that this step size control differs from the one presented in [51] since the initial guesses as described in [51] are restricted to the range $t^* \in (0, 2\|x_1 - x_0\|_2]$ which may be too small if x_1 is near to x_0.

Computation of $\tilde{t}_l$ We are given a point $x_0 \in \mathbb{R}^n$ and the search direction $a = \sum_{i=1}^{N_{nd}} s_i(\tilde{x}_i - x_0)/\|\tilde{x}_i - x_0\|$ (or alternatively direction (10)) with $\tilde{x}_i \in$

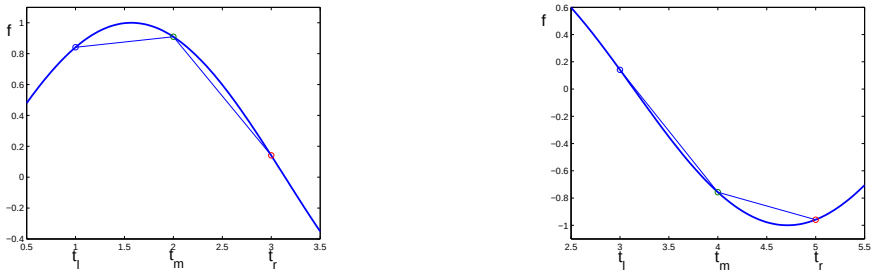


Fig. 2. The term in Equation (11) is false for t_l, t_m, and t_r in the left figure and true in the right figure. In the latter case, the quadratic polynomial is convex.

Algorithm 2 $t^* := hc\_step(x_0, x_1)$

Require: $x_0, x_1 \in \mathbb{R}^n$ with $x_1 \prec x_0$, maximal number of trials N_{max}
Ensure: step size t^* for the hill climber

$I := \{i \in \{1, \ldots, k\} : f_i(x_1) < f_i(x_0)\}$
$\nu := x_1 - x_0$
$\Delta := \|x_1 - x_0\|_2$
$t_l := 0$, $t_m := \Delta$, $t_r := 2\Delta$
for $j = 1, \ldots, N_{max}$ **do**
  if $\exists i \in I$: (11) is true for t_l, t_m, t_r and $f_{\nu,i}$ **then**
    for all $i \in I$ **do**
      if (11) is true for t_l, t_m, t_r and $f_{\nu,i}$ **then**
        $t_i^* := quad\_approx(t_l, t_m, t_r, f_{\nu,i})$
      else
        $t_i^* := \infty$
      end if
    end for
    return $t^* := \min_{i=1,\ldots,k} t_i^*$
  else
    $t_l := t_m$, $t_m := l_r$, $t_r := 2 * t_r$
  end if
end for
return $t^* := t_m$

$B(x_0, r)$, $i = 1, \ldots, N_{nd}$, and such that $(x_0, \tilde{x}_i), i = 1, \ldots, N_{nd}$, are mutually non-dominating. For this situation, we propose to proceed analogously to [50], where a step size strategy for multi-objective continuation methods is suggested: given a target value $\epsilon_y \in \mathbb{R}_+$—e.g., the minimal value which makes two solutions distinguishable from a practical point of view—, the task is to compute a new candidate $x_{new} = x_0 + \tilde{t}a$ such that

$$\|F(x_0) - F(x_{new})\|_\infty \approx \epsilon_y \tag{13}$$

In case F is Lipschitz continuous there exists an $L \geq 0$ such that

$$\|F(x) - F(y)\| \leq L\|x - y\|, \quad \forall x, y \in Q. \tag{14}$$

This constant can be estimated around x_0 by

$$L_{x_0} := \|DF(x_0)\|_\infty = \max_{i=1,\ldots,k} \|\nabla f_i(x_0)\|_1,$$

where $DF(x_0)$ denotes the Hessian of F at x_0 and $\nabla f_i(x_0)$ the gradient of the i-th objective at x_0. In case the derivatives of F are not given (which is considered in this section) the accumulated information can be used to compute the estimation

$$\tilde{L}_{x_0} := \max_{i=1,\ldots,N_{nd}} \frac{\|F(x_0) - F(\tilde{x}_i)\|_\infty}{\|x_0 - \tilde{x}_i\|_\infty},$$

since the $\tilde{x}_i$'s are near to x_0. Combining (13), (14) and using the estimation L_{x_0} leads to the step size control

$$x_{new} = x_0 + \frac{\epsilon_y}{L_{x_0}} \frac{a}{\|a\|_\infty}. \tag{15}$$

Handling constraints In the course of the computation it can occur that iterates are generated which are not inside the feasible domain Q. That is, we are faced with the situation that $x_0 \in Q$ and $x_1 := x_0 + h_0\nu \notin Q$, where ν is the search direction. In that case we propose to proceed analogously to the well-known bisection method for root finding in order to backtrack from the current iterate x_1 to the feasible set:
let $in_0 := x_0 \in Q$ and $out_0 := x_1 \notin Q$ and $m_0 := in_0 + 0.5(out_0 - in_0) = x_0 + \frac{h_0}{2}\nu$. If $m_0 \in Q$ set $in_1 := m_0$, else $out_1 := m_0$. Proceeding in an analogous way, one obtains a sequence $\{in_i\}_{i\in\mathbb{N}}$ of feasible points which converges linearly to the boundary ∂Q of the feasible set. One can, for example, stop this process with an $i_0 \in \mathbb{N}$ such that $\|out_{i_0} - in_{i_0}\|_\infty \leq tol$, obtaining a point in_{i_0} with maximal distance tol to ∂Q. See Algorithm 3 for one possible realization. Note that by this procedure no function evaluation has to be spent (though a feasibility test may also be of relevant numerical effort in some cases).

In case the domain Q is given by box constraints, i.e., if Q can be written as

$$Q = \{x \in \mathbb{R}^n : l_i \leq x_i \leq u_i,\ i = 1, \ldots, n\}, \tag{16}$$

where $l, u \in \mathbb{R}^n$ with $l \leq_p u$, the backtracking can be performed in one step: given a point $x_0 \in Q$ and a search direction ν the maximal step size h_{max} such that $x_0 + h_{max}\nu \in Q$ can be computed as shown in Algorithm 4.

Algorithm 3 Backtracking to Feasible Region

Require: $x_0 \in Q$, $x_1 = x_0 + h_0\nu \notin Q$, $tol \in \mathbb{R}_+$
Ensure: $\tilde{x} \in \overline{x_0 x_1} \cap Q$ with $\inf_{b \in \partial Q} \|b - \tilde{x}\| < tol$

$in_0 := x_0$
$out_0 := x_1$
$i := 0$
while $\|out_i - in_i\| \geq tol$ **do**
    $m_i := in_i + \frac{1}{2}(out_i - in_i)$
    if $m_i \in Q$ **then**
        $in_{i+1} := m_i$
        $out_{i+1} := out_i$
    else
        $in_{i+1} := in_i$
        $out_{i+1} := m_i$
    end if
    $i := i + 1$
end while
return $\tilde{x} := in_i$

Algorithm 4 $h_{max} := ComputeHmax(x_0, \nu, l, u)$

Require: feasible point $x_0 \in Q$, search direction $\nu \in \mathbb{R}^n \backslash \{0\}$, lower and upper bounds $l, u \in \mathbb{R}^n$
Ensure: maximal step size h_{max} such that $x_0 + h_{max}\nu \in Q$

for $i = 1, \ldots,$n **do**
    if $v_i > 0$ **then**
        $d_i := (u_i - x_i)/v_i$
    else if $v_i < 0$ **then**
        $d_i := -(x_i - l_i)/v_i$
    else
        $d_i := \infty$
    end if
end for
$h_{max} := \min_{i=1,\ldots,n} d_i$

Design parameters We agree that a realization of Algorithm 1 may include a variety of design parameters which may be difficult to tune and adapt to a particular problem. However, if the suggestions made in this paper are taken merely the values for four design parameters have to be chosen (see Table 1): the parameter r defines the neigborhood search of the procedure. Since this neighorbood search is used to find a search direction which is afterwards coupled with a step size control, the value of r is not that important, but should be 'small' to guarantee a local search. N_{nd} is the value which determines the number of directions which have to be averaged in order to choose the sidestep direction. In general, a larger value of N_{nd} leads to a 'better' sidestep (in the sense that the search is performed orthogonal to the upward movement), but will in turn increase the cost of the search. We have experienced that a low value N_{nd}, say 5 to 10, already gives satisfactory results, the 'accuracy' of the search does not seem to influence the performance of the HCS (unless the second derivatives of the objectives are available, see below). The value of ϵ_y is problem dependent but can be given quite easily in a real world application (see discussion above Equation (13)). Finally, the tolerance *tol* has to be adjusted for constrained MOPs. The choice of this value is also problem dependent and has to be chosen in every algorithm dealing with constraints.

Table 1. Design parameters that are required for the realization of the gradient free HCS algorithm.

Parameter	**Description**
r	Radius for neighborhood search (Alg. 1)
N_{nd}	Number of trials for the hill climber before the sidestep is performed (Alg. 1)
ϵ_y	Desired distance (in image space) for the sidestep (7)
tol	Tolerance value used for the backtracking in Alg. 2

3.2 HCS Using Gradient Information

In this section we discuss possible modifications which can be made to increase the performance of the HCS in case the MOP is sufficiently smooth. It will turn out that the resulting algorithm is more efficient (see Section 5), but in turn, more information of the model is required.
Here we describe one possible realization of the HCS using the descent direction presented in Theorem 2 for the hill climber and some elements from multi-objective continuation for the sidestep:

Given a point $x \in \mathbb{R}^n$ the quadratic optimization problem (3) can be solved leading to the vector $\hat{\alpha}$. In case

$$\| \sum_{i=1}^{k} \hat{\alpha}_i \nabla f_i(x) \|_2^2 \geq \epsilon_{\mathcal{P}}, \tag{17}$$

i.e., if the square of the norm of the weighted gradients is larger than a given threshold $\epsilon_{\mathcal{P}} \in \mathbb{R}_+$, the candidate solution x can be considered to be 'away' from $\mathcal{P}$, and thus, it makes sense to seek for a dominating solution. For this, the descent direction (2) can be taken together with a suitable step size control. For the latter the step size control described above can be taken, or—probably better—a step size control which uses gradient information as e.g. described in [16] or the one presented in [15]. If the value of the term in (17) is less than $\epsilon_{\mathcal{P}}$, this indicates that x is already in the vicinity of $\mathcal{P}$. In that case one can lean elements from (multi-objective) continuation [24, 2] to perform a search along $\mathcal{P}$. To do this, we assume for simplicity that we are given a KKT–point $\hat{x}$ and the according weight $\hat{\alpha}$ obtained by (3). Then the point $(\hat{x}, \hat{\alpha}) \in \mathbb{R}^{n+k}$ is obviously contained in the zero set of the auxiliary function $\tilde{F} : \mathbb{R}^{n+k} \to \mathbb{R}^{n+1}$ of the given MOP which is defined as follows:

$$\tilde{F}(x, \alpha) = \begin{pmatrix} \sum_{i=1}^{k} \alpha_i \nabla f_i(x) \\ \sum_{i=1}^{k} \alpha_i - 1 \end{pmatrix}. \tag{18}$$

In [24] it has been shown that the zero set $\tilde{F}^{-1}(0)$ can be linearized around $\hat{x}$ by using a QU-factorization of $\tilde{F}'(\hat{x}, \hat{\alpha})^T$, i.e., the transposed of the Jacobian matrix of $\tilde{F}$ at $(\hat{x}, \hat{\alpha})$. To be more precise, given a factorization

$$\tilde{F}'(\hat{x}, \hat{\alpha})^T = QU \in \mathbb{R}^{(n+k)\times(n+k)}, \tag{19}$$

where $Q = (Q_N, Q_K) \in \mathbb{R}^{(n+k)\times(n+k)}$ is orthogonal with $Q_N \in \mathbb{R}^{(n+k)\times(n+1)}$ and $Q_K \in \mathbb{R}^{(n+k)\times(k-1)}$, the column vectors of Q_K form—under some mild regularity assumptions on $\tilde{F}^{-1}(0)$ at $(\hat{x}, \hat{\alpha})$, see [24]—an orthonormal basis of the tangent space of $\tilde{F}^{-1}(0)$. Hence, it can be expected that each column vector $q_i \in Q_K$, $i = 1, \ldots, k-1$, points (locally) along $\mathcal{P}$ and is thus well suited for a sidestep direction. The step size control can in this case taken exactly as proposed in Equation (15) since the setting for that case was the same. In fact, since the search direction q_i is indeed pointing along $\mathcal{P}$, the results will be more accurate than for an averaged direction such as (7) or (10).
Algorithm 5 presents a procedure which is based on the above discussion. Note that this is one possible realization and that there exist certainly other possible ways leading, however, to similar results. For instance, alternatively to the descent direction used in Algorithm 5 the ones proposed in [17] and [5] can be taken. Further, the vicinity test (17) can be changed, though alternative conditions will most likely also be based on Theorem 1. Finally, the movement along

$\mathcal{P}$ can be realized by predictor-corrector methods [24, 2] which consist, roughly speaking, of a repeated application of a predictor step obtained by a linearization of $\tilde{F}^{-1}(0)$ as in (19) and a corrector step which is done via a Gauss-Newton method.
Note that the HCS is proposed for the unconstrained case. While an extension to the constrained case for the hill climber is possible (see, e.g., [17] for possible modifications) this does not hold for the movement along the Pareto set (i.e., the sidestep). Though it is possible to extend system (18) by equality constraints (e.g., by introducing slack variables to transform the inequality constraints into equality constraints) this could lead to effiency problems in the numerical treatment [24]. Hence, we restrict ourselves here to the unconstrained case.

As it will be shown in Section 5 the performance of the gradient based HCS in terms of convergence is better than its gradient free version, but this improvement does not come for free: for the descent direction all objectives' gradients have to be available (or approximated), and to perform the linearization of $\mathcal{P}$ even all second derivatives are required.

Algorithm 5 HCS2 (Using Gradient Information)

Require: starting point $x_0 \in Q$
Ensure: sequence $\{x_l\}_{l\in\mathbb{N}}$ of candidate solutions
for $l = 0, 1, 2, \ldots$ **do**
  compute the solution $\hat{\alpha}$ of (3) for x_l.
  if $\| \sum_{i=1}^{k} \hat{\alpha}_i \nabla f_i(x_l) \|_2^2 \geq \epsilon_{\mathcal{P}}$ **then**
    $\nu_l := -q(x_l)$
    compute $t_l \in \mathbb{R}_+$ and set $x_{l+1} := x_l + t_l \nu_l$
  else
    compute $\tilde{F}'(\hat{x}, \hat{\alpha})^T = (Q_N, Q_K)U$ as in (19)
    choose a column vector $\tilde{q} \in Q_K$ at random
    compute $\tilde{t}_l \in \mathbb{R}_+$ and set $x_{l+1} := x_l + \tilde{t}_l \tilde{q}$.
  end if
end for

Design parameters Analogue to the gradient free version of the HCS, the values of some design parameters have to be chosen for the realization of Algorithm 5. ϵ_y and *tol* are as discussed above, and N_{nd} and r are not needed due to the accuracy of the gradient based search. A new parameter, compared to the gradient free version of the HCS, is the threshold $\epsilon_{\mathcal{P}}$ for the vicinity test of a given candidate solution to $\mathcal{P}$. This value is certainly problem dependent, but it can be made 'small' due to the convergence properties of the hill climber (e.g., [17]).

Table 2. Design parameters that are required for the realization of the HCS algorithm which involves gradient information.

Parameter	Description
ϵ_y	Desired distance (in image space) for the sidestep (7)
tol	Tolerance value used for the backtracking in Alg. 2
$\epsilon_{\mathcal{P}}$	Threshold for the vicinity test (17)

4 Use of the HCS within MOEAs

Here we address the integration of the HCS into a given MOEA. For this, we present some modifications required on the standalone version of the HCS to be able to be coupled efficiently with an evolutionary algorithm and discuss the cost of the procedure. Finally, we present two particular hybrids where NSGA-II and SPEA2 are used as base MOEAs.

4.1 Modifying the HCS

In Algorithms 1 and 5 the HCS is presented as standalone algorithm generating an infinite sequence of candidate solutions which is certainly not applicable when coupling it with a MOEA. To support the search of the latter algorithm, it is rather advisable to stop the iteration after a few iterations (denote this parameter by *maxiter*). In case the HCS finds only a sequence of dominating solutions (i.e., by the hill climber) merely the last dominating solution (denoted by x_d) has to be returned since the other intermetiate solutions are all dominated by x_d and are thus not important for the current population of the MOEA. In case the sidestep is performed, which indicates that the iterates are near to the (local) Pareto set, the iteratation can be stopped even before *maxiter* is reached. The second modification of HCS compared to the standalone version presented above that we suggest is to perform the sidestep in each diversity direction which has been bound during the local search. This is due to the fact that the sidestep is the expensive part of the HCS (in terms of function calls, see also the discussion below), and hence all accumulated information should be exploited. Thus, the modified HCS will return in that case the dominating solution x_d (if not equal with the initial solution x_0) and further maximal $2^k - 2$ sidestep solutions in all diversity directions of x_d, depending on how many diversity directions of x_p have been found within the N_{nd} 'unsuccessful' trials.
Algorithm 6 shows such a modification of the HCS1 for $k = 3$. Hereby,

$$C(x, s_1, s_2, s_3), \tag{20}$$

where $s_i \in \{+,-\}, i = 1,2,3$, denotes the diversity cone at a point x. For instance, it is

$$y \in C(x,+,+,-) :\Leftrightarrow \{f_1(y) > f_1(x) \text{ and } f_2(y) > f_2(x) \text{ and } f_3(y) < f_3(x)\} \tag{21}$$

The algorithm requires the starting point x_0 and returns the set X_{new} which can consist of one candidate solution (i.e., the result of the hill climber x_d) up to seven candidate solutions (x_d plus candidates in all the six diversity directions (9) of x_d).
For HCS2, the modifications described above are much easier to handle: if the sidestep is performed (i.e., if equation (17) is false) the sidestep solutions can be chosen as

$$x_+^i := x_d + h_i q_i, \quad x_-^i := x_d - h_i q_i, \tag{22}$$

for all column vectors q_i of Q_K, which leads to $2k-2$ new candidate solutions.

4.2 Cost of the HCS

Crucial for the efficient usage of the HCS within a MOEA is the knowledge of its cost. Here we measure the cost of one step of the modified HCS as described above (i.e., for $maxiter = 1$). Unfortunately, the different algorithms use different information (mainly different gradient information) of the model. For sake of comparison, we measure the cost of the HCS in terms of required function calls (to be more precise, we measure the running time for a function call and neglect the memory requirement). That is, to measure HCS2 we have to find an equivalent in terms of function calls for the computation or approximation of the derivative $\nabla f(x)$ and the second derivative $\nabla^2 f(x)$ of a function $f : \mathbb{R}^n \to \mathbb{R}$ at a point x. If for instance automatic differentiation (AD) is used to compute the derivatives, we can estimate 5 function calls for the derivative call and $4 + 6n$ function call for the second derivative [20]. These values change when using finite differences (FD). If for instance the forward difference quotient

$$\frac{\partial f}{\partial x_i}(x) \approx \frac{f(x_1,\ldots,x_i+\delta_i,\ldots,x_n) - f(x_1,\ldots,x_n)}{\delta_i}, \quad i \in \{1,\ldots,n\} \tag{23}$$

where $\delta_i \in \mathbb{R}_+$ is a small value, is used to estimate the gradient, apparently n function calls are required. The central difference quotient leads to more accurate approximations, but does in turn require $2n$ function calls ([20]). A forward difference quotient approximation of the second derivative requires a total of n^2 function calls (and $2n^2$ or $4n^2$ function calls when using the central difference quotient, depending on how the rule is applied). Finally, we have to estimate the number of function calls required for a line search for the hill climber. Here we take the value of 3 obtained by our observations.

A call of HCS1 requires at least four function calls: one for the local search around x_0. If the new candidate solution is either dominating or dominated by x_0—which is very likely in the early stage of the optimization process—the next

Algorithm 6 HCS1 (for use within MOEAs for $k = 3$)

Require: maximal number of iterations *maxiter*, rest as in Alg. 1
Ensure: set of candidate solutions X_{new}

1: $L_1 := 0$, $L_2 := 0$, $L_3 := 0$
2: $a_1 := 0 \in \mathbb{R}^n$, $a_2 := 0 \in \mathbb{R}^n$, $a_3 := 0 \in \mathbb{R}^n$
3: $no\_a_1 := 0$, $no\_a_2 := 0$, $no\_a_3 := 0$
4: $nondom := 0$
5: $x_{1,0} := x_0$
6: **for** $i = 1, 2, \ldots, maxiter$ **do**
7: **for** $j = 1, 2, \ldots, N_{nd}$ **do**
8: choose $x_2 \in B(x_{1,i-1}, r)$ at random
9: **if** $x_{1,i-1} \prec x_2$ **then**
10: compute $t \in \mathbb{R}_+$ as in Alg. 1 (l. 6-10), set $x_{1,i} := x_2 + t(x_{1,i-1} - x_2)$.
11: $nondom := 0, \quad a_1 = a_2 = a_3 = 0$
12: **continue**
13: **else if** $x_2 \prec x_{1,i-1}$ **then**
14: comp. $t \in \mathbb{R}_+$ as in Alg. 1 (l. 11-13), set $x_{1,i} := x_{1,i-1} + t(x_2 - x_{1,i-1})$.
15: $nondom := 0, \quad a_1 = a_2 = a_3 = 0$
16: **continue**
17: **else**
18: $nondom := nondom + 1$
19: **if** $x_2 \in C(x_{1,i-1}, -, -, +)$ **then**
20: $a_1 := a_1 + (x_2 - x_{1,i-1})/\|x_2 - x_{1,i-1}\|_\infty$
21: $no\_a_1 := no\_a_1 + 1$
22: $L_1 := max(L_1, \|F(x_2) - F(x_{1,i-1})\|_\infty / \|x_2 - x_{1,i-1}\|_\infty)$
23: **end if**
24: **if** $x_2 \in C(x_{1,i-1}, +, +, -)$ **then**
25: $a_1 := a_1 + (x_{1,i-1} - x_2)/\|x_{1,i-1} - x_2\|_\infty$
26: $no\_a_1 := no\_a_1 + 1$
27: $L_1 := max(L_1, \|F(x_{1,i-1}) - F(x_2)\|_\infty / \|x_{1,i-1} - x_2\|_\infty)$
28: **end if**
29: **if** $x_2 \in C(x_{1,i-1}, -, +, -)$ **then**
30: update a_2, no_a_1, and L_2 analogue to lines 19 –22.
31: **end if**
32: **if** $x_2 \in C(x_{1,i-1}, +, -, +)$ **then**
33: update a_2, no_a_2, and L_2 analogue to lines 24 –27.
34: **end if**
35: **if** $x_2 \in C(x_{1,i-1}, +, -, -)$ **then**
36: update a_3, no_a_3, and L_3 analogue to lines 19 –22.
37: **end if**
38: **if** $x_2 \in C(x_{1,i-1}, -, +, +)$ **then**
39: update a_3, no_a_3, and L_3 analogue to lines 24 –27.
40: **end if**
41: **end if**
42: **end for**
43: $X_{new} := \{x_{1,i}\}$ ▷ perform sidesteps and return
44: **if** $no\_a_1 > 0$ **then**
45: $\nu_1 := a_1/\|a_1\|_\infty, \quad h_1 := \epsilon_y / L_1$
46: $x_+^{(1)} := x_{i,N_{nd}} + h_1\nu_1, \quad x_-^{(1)} := x_{i,N_{nd}} - h_1\nu_1$
47: $X_{new} := X_{new} \cup \{x_+^{(1)}, x_-^{(1)}\}$
48: **end if**
49: **if** $no\_a_2 > 0$ **then**
50: compute $x_+^{(2)}, x_-^{(2)}$ analogue to lines 44–47, set $X_{new} := X_{new} \cup \{x_+^{(2)}, x_-^{(2)}\}$
51: **end if**
52: **if** $no\_a_3 > 0$ **then**
53: compute $x_+^{(3)}, x_-^{(3)}$ analogue to lines 44–47, set $X_{new} := X_{new} \cup \{x_+^{(3)}, x_-^{(3)}\}$
54: **end if**
55: **return**
56: **end for**
57: $X_{new} := \{x_{1,maxiter}\}$ ▷ return dominating solution

point is found via line search resulting in 4 function calls due to our assumptions. When a sidestep, the most expensive event of the HCS, is performed, this means that first N_{nd} trials have been made around x_0, and then candidates in maximal $2^k - 2$ directions are computed (for each one function call is required, see (15)) leading to a total of $N_{nd} + 2^k - 2$ function calls.
The HCS2 needs for the realization of the hill climber the gradients of all k objectives, the solution of (QOP) (which we do not count here since k is typically low, and thus, the quadratic problem is easy to solve with standard techniques) and one line search. This makes $5k+3$ function calls when using AD and $kn+3$ function calls when using FD (here we assume the forward difference method). For a sidestep, k gradients and the second derivative of $\sum_{i=1}^{k} \alpha_i f_i(x_d)$ have to be computed, and further $2k-2$ sidestep candidates are produced. This leads to $6n + 7k + 2$ function calls when using AD and to $n^2 + k(n+2) - 2$ function calls when using FD.

Table 3. Cost of one step of the HCS measured in function calls. To convert the derivative calls in HCS2 into function calls we have used values based on automatic differentiation (AD) and finite differences (FD).

Method	No. of function calls required
HCS1	from 4 to $N_{nd} + 2^k - 2$
HC2 (AD)	$5k + 3$
HC2 (FD)	$kn + 3$
HCS2 (AD)	from $5k + 3$ to $6n + 7k + 2$
HCS2 (FD)	from $kn + 3$ to $n^2 + k(n+2) - 2$

Table 3 summarizes the cost of the different algorithms using the different conversion rules. Hereby, HC2 denotes the hill climber as presented in Algorithm 5 but without the sidestep operator (i.e., for $\epsilon_{\mathcal{P}} = 0$). Table 4 gives the numerical values for $k = 3$, and $N_{nd} = 3$, and $n_1 = 10$ and $n_2 = 30$. It is obvious that FD should only be used for models with moderate dimensional parameter space since else the cost of the HCS2 will get tremendous. On the other side, note that the cost of HCS1 is independent of n and thus relatively inexpensive, in particular in higher dimensions.

4.3 Integration into MOEAs

The questions which remains open is how to integrate the HCS into a given MOEA in order to obtain an efficient memetic strategy. Here we make first steps to answer this problem and propose hybrids with the state of the art MOEAs NSGA-II and SPEA2. The numerical results in the next section show that the combination is advantageous, however, we think that more effort has to be done to obtain a universal and self adaptive memetic algorithm which is beyond the

Table 4. Numerical values for the cost of the HCS algorithms for the settings (a) $n_1 = 10$, $k = 3$, $N_{nd} = 3$ and (b) $n_2 = 10$, $k = 3$, $N_{nd} = 3$. See Table 3 for details.

Method	No. of function calls required for $n_1 = 10$, $k = 3$, $N_{nd} = 3$	No. of function calls required for $n_2 = 30$, $k = 3$, $N_{nd} = 3$
HCS1	from 4 to 9	from 4 to 9
HC2 (AD)	18	18
HC2 (FD)	33	93
HCS2 (AD)	from 18 to 83	from 18 to 203
HCS2 (FD)	from 33 to 138	from 93 to 996

scope of this paper.

The modified HCS can be written in the shorthand form as

$$P_{HCS} = HCS\ (x_0), \tag{24}$$

where x_0 is a given point (e.g., coming from the current population of the MOEA) and P_{HCS} is the output set. Given a probability p_{HCS} for the application of the procedure on an individual of a population, the operator can be defined set-wise as

$$P_{HCS} = HCS\ (P, p_{HCS}), \tag{25}$$

where P denotes a given population. By doing so, the HCS can be interpreted as a particular mutation operator, and thus, can in principle be integrated into any given MOEA with little effort. However, this should be handled with care since the efficiency of the resulting hybrid depends (among others) on (a) which elements of the population the HCS is applied to, and (b) the balance of the genetic search and the HCS (see also Section 2.2). As an example for (a) we have observed that if the HCS is merely applied on elements of the external archive in a combination with SPEA2, that this 'elitism approach' has a negative effect on the diversity of the population, at least in early stages of the search. Even the application of the sidestep could not compensate this effect, since it is applied on a few, possibly closely located solutions. Problem (b) is another typical problem when designing memetic strategies (probably first reported in [31] in the context of multi-objective optimization), and in particular in our setting due to the relatively high cost of the HCS compared to classical mutation operators. Most important for the effect and the cost of the HCS are the parameters *maxiter* and N_{nd} (for HCS1). In general, it can be said that if both values of *maxiter* and N_{nd} are high, the local improvement of a point x_0 will be nearly optimal (i.e., the elements of P_{HCS} will be near to local solutions). This can be advantageous for uni-modal models but can in turn reduce the efficiency of the entire search algorithm for multi-modal models due to the high cost of the HCS and the relatively high chance that the search gets stuck in a local (and not global) solution. If the values of *maxiter* and N_{nd} are low, the local improvements in one application of HCS will typically be sub-optimal. However,

the choice of low values offers on the other hand two advantages: first, the HCS spends less function calls for unpromising starting points. That is apparently also the case for promising starting points but we have observed that it is advantageous to repeat the local search more often instead to spend the function calls for single solutions (future populations contain points which are at least as good as the point x_0 from the current population). The second advantage is that the population is not disturbed by drastic improvements of single solutions which may cause trouble in elitist strategies ([35]).
The next question is the choice of the probability p_{HCS} to apply the HCS. Due to the cost of the HCS a low value seems to be advisable which also coincides with our observations. Further, we suggest not to apply the HCS in every generation in order not to disturb the efficient but highly sensitive interplay of the different operators of the MOEA (as e.g. done in [69]).
To summarize, we suggest low values for the parameters $maxiter$ and N_{nd} which influence efficiency and cost of one application of the HCS, and a low value for the probability p_{HCS} of its application which influences the overall cost of the local search. See next section for particular choices of these values.
In the following, we propose two particular combinations where we use NSGA-II and SPEA2 as base MOEAs.

NSGA-II-HCS As discussed above, crucial are the questions when and to which elements the local search has to be applied within a given MOEA. For NSGA-II, we suggest to perform the local search (i.e., HCS) only on the best individuals of a given generation. This is made in order to find leader individuals to pull the entire population to better solutions during the search. This exclusive search can be done since the diversity of the best (i.e., non-dominated) solutions is typically quite high, also in early stages of the search. Thus, it is likely to generate well-spread leader individuals from the beginning of the search which helps to pull the population to the entire Pareto set.
Algorithm 7 shows an algorithm which combines NSGA-II with HCS. Hereby, the procedures 'Fast Non-Dominated Sort', 'Crowding Distance Assignment' and 'Generate Child Population' are well known as parts of the NSGA-II, a thorough discussion can be found in [13].

Algorithm 7 applies the local search each s generation after reproduction. The local search is applied only to non-dominated individuals, and, due to the cost of the procedure, is performed with a certain (low) probability. After having computed the improved solutions of local search, the regular operations of ranking and crowding are used as in NSGA-II.
Contrary to [51], where the local search has been applied after 75 percent of a given budget B of function calls have been spent, we have observed that it is advantageous to apply the HCS in all stages of the search to pull the population permanently to the Pareto set. In fact, we propose here that the local search should be evenly distributed over the run of the algorithm. This guideline and the choice to take only non-dominated solutions as starting points for the HCS has an implication on the rule to choose p_{HCS}: the number of non-dominated

Algorithm 7 NSGA-II-HCS

```
1: procedure NSGA-II-HCS(N,G, p_HCS, s)
2:     Generate Random Population P (size N).
3:     Evaluate Objective Values.
4:     Fast Non-Dominated Sort
5:     Crowding Distance Assignment
6:     Generate Child Population P_offs
7:     for i := 1, ..., G do
8:         Using P := P ∪ P_offs:
9:         if mod(i,s)==0 then
10:            LocalSearch(p_HCS)
11:        end if
12:        Fast Non-Dominated Sort
13:        Crowding Distance Assignment
14:        Generate Child Population P_offs
15:    end for
16: end procedure

17: procedure LocalSearch(p_HCS)
18:    for all a ∈ P do
19:        if ∄b ∈ P such that b ≺ a then
20:            A_a = HCS({a}, p_HCS)
21:            P := P ∪ A_a
22:        end if
23:    end for
24: end procedure
```

points (or rank 0 solutions) is typically very low at the beginning of the search, further on increasing, and from a certain stage of the process the number of non-dominated solutions is nearly constant (i.e., equal to the population size). A constant value of p_{HCS} would hence lead to a permanent growth of the fraction of the local search within the memetic strategy, at least in the beginning of the search. To counteract to this effect it seems to be better to start with a relatively high probability p_{max} and to decrease this value during the search process until a prescribed (low) probability p_{min} is reached. This value is then chosen for the remainder of the run of the algorithm.
For the computations presented in the next section we have used the following strategy which is based on the above considerations: starting with the probability $p_{HCS}(0) := p_{max}$ the local search probabilities for the subsequent generations are updated as follows

$$p_{HCS}(i) = max \left\{ \frac{-2(p_{max} - p_{min})}{B} \sum_{j=1}^{i} fc(j) + p_{max} \, , \, p_{min} \right\}, \qquad (26)$$

where $fc(j)$ denotes the number of function calls spent in the j-th generation. Hereby, the first expression in (26) is a linear term in the number of function calls spent. Its value is p_{max} for zero function calls (i.e., at the beginning of the search) and p_{min} for $B/2$. That is, after at least 50 percent of a given budget has been spent, the local search probability for future generations is constantly set to p_{min} (i.e., p_{min} times the population size is the number of HCS calls one is willing to spend in average per generation in advanced stages of the search).

SPEA2-HCS Unlike above, where NSGA-II is used as base MOEA, we have observed that for a hybridization with SPEA2 it is not always beneficial to apply the HCS only to members of the archive which consists only of non-dominated solutions. This is because the archive can—in particular in early stages of the search—consist of few, and probably not well spread solutions (which changes with increasing number of iterations). Thus, for a hybrid of HCS with SPEA2, we suggest to apply the local search operator to members of the mating pool, i.e., also to dominated solutions. Consequently, we propose by the above discussion to set p_{HCS} constant since the size of the mating pool does not change. See Algorithm 8 for a pseudocode of SPEA2-HCS.

5 Results and Discussions

Here we present and discuss some numerical results for the HCS as well as for the two memetic strategies in order to demonstrate the strength of both the HCS as standalone algorithm as well as its benefit as a local search procedure within a given MOEA. The MOPs we have used here are listed in Table 5. All computations have been done using the programming language MATLAB[5].

[5] `https://www.mathworks.com`

Table 5. The MOPs under investigation in this work. Hereby, $\tilde{k} = n - k + 1$.

CONV1
$f_1(x) = (x_1 - 1)^4 + \sum_{j=2}^{n}(x_j - 1)^2$
$f_2(x) = \sum_{j=1}^{n}(x_j + 1)^2$
CONV2
$f_i(x) = \sum_{\substack{j=1 \\ j \neq i}}^{n} (x_j - a_j)^2 + (x_i - a_i)^4,\ i = 1, 2, 3$
$a_1 = (1, \ldots, 1) \in \mathbb{R}^n$
$a_2 = (-1, \ldots, -1) \in \mathbb{R}^n$
$a_3 = (1, -1, 1, -1 \ldots) \in \mathbb{R}^n$
ZDT4
$f_1(x) = x_1$
$f_2(x) = g(x)(1 - \sqrt{f_1/g(x)})$
$g(x) = 1 + 10(n-1) + \sum_{i=2}^{n}(x_i^2 - 10cos(4\pi x_i))$
$0 \leq x_1 \leq 1, \quad -5 \leq x_i \leq 5,\ i = 2, \ldots, n$
DTLZ1
$f_1(x) = \frac{1}{2}x_1 x_2 \ldots x_{k-1}(1 + g(x))$
$f_2(x) = \frac{1}{2}x_1 x_2 \ldots (1 - x_{k-1})(1 + g(x))$
$\vdots$
$f_{k-1}(x) = \frac{1}{2}x_1(1 - x_2)(1 + g(x))$
$f_k(x) = \frac{1}{2}(1 - x_1)(1 + g(x))$
$g(x) = 100\left[\tilde{k} + \sum_{i=k}^{n}(x_i - \frac{1}{2})^2 - \cos(20\pi(x_i - \frac{1}{2}))\right]$
$0 \leq x_i \leq 1,\ i = 1, \ldots, n$
DTLZ2
$f_1(x) = \cos(\frac{x_1\pi}{2})\cos(\frac{x_2\pi}{2})\ldots\cos(\frac{x_{k-1}\pi}{2})(1 + g(x))$
$f_2(x) = \cos(\frac{x_1\pi}{2})\cos(\frac{x_2\pi}{2})\ldots\sin(\frac{x_{k-1}\pi}{2})(1 + g(x))$
$\vdots$
$f_{k-1}(x) = \cos(\frac{x_1\pi}{2})\sin(\frac{x_2\pi}{2})(1 + g(x))$
$f_k(x) = \sin(\frac{x_1\pi}{2})(1 + g(x))$
$g(x) = \sum_{i=k}^{n}(x_i - \frac{1}{2})^2$
$0 \leq x_i \leq 1,\ i = 1, \ldots, n$
DTLZ3
$f_1(x) = \cos(\frac{x_1\pi}{2})\cos(\frac{x_2\pi}{2})\ldots\cos(\frac{x_{k-1}\pi}{2})(1 + g(x))$
$f_2(x) = \cos(\frac{x_1\pi}{2})\cos(\frac{x_2\pi}{2})\ldots\sin(\frac{x_{k-1}\pi}{2})(1 + g(x))$
$f_{k-1}(x) = \cos(\frac{x_1\pi}{2})\sin(\frac{x_2\pi}{2})(1 + g(x))$
$f_k(x) = \sin(\frac{x_1\pi}{2})(1 + g(x))$
$g(x) = 100\left[\tilde{k} + \sum_{i=k}^{n}(x_i - \frac{1}{2})^2 - \cos(\alpha\pi(x_i - \frac{1}{2}))\right]$
$\alpha = 20$
$0 \leq x_i \leq 1,\ i = 1, \ldots, n$

Algorithm 8 SPEA2–HCS

1: Generate initial population $P_0 \subset Q$ and set $A_0 := \emptyset$, $\bar{P}_0 := \emptyset$.
2: **for** $k = 0, 1, \ldots, N_{maxiter}$ **do**
3: $\overline{P}_{k+1} :=$ non-dominated solutions of $P_k \cup A_k$
4: Set $A_{k+1} :=$ non-dominated solutions of $\overline{P}_{k+1}$
5: Calculate fitness values of individuals in $\overline{P}_{k+1}$
6: Perform tournament selection in $\overline{P}_{k+1}$ to fill the mating pool
7: Apply crossover, mutation and the local search operators (HCS) to the mating pool.
8: Denote the resulting population by P_{k+1}.
9: **end for**

5.1 HCS as a Standalone Algorithm

Since the two variants of the HCS as described in Algorithm 1 (which we will denote by HCS1 in this section) and in Algorithm 5 (denoted by HCS2) have no orientation in the search along the Pareto set, we have modified it for bi-objective models in the following way in order to demonstrate its potential (see also discussion in Section 3.1): the HCS—i.e., both variants—is started as described above. If the current iterate x_p is close enough to $\mathcal{P}$ such that the sidestep procedure can start (taking $N_s = 5$), first improvements according to f_1 are sought (leading to a 'left up' movement from $F(x_{temp})$ along the Pareto front). If no improvements according to f_1 can be obtained, an analogue 'right down' movement is performed starting again from x_{temp}. This is intended so 'screen' the entire connected component of $\mathcal{P}$ which is near to x_{temp}.
However, since this orientation is not needed within the use of a MOEA because in that case only few iterates are being computed from a given starting point, these modifications are only done within this subsection.

In the following we will test HCS1 and HCS2 on a convex model (i.e., a model which does not contain local minima where the local search can get stuck) and we will investigate both the unconstrained and the constrained case. Then we will consider a multi-modal and constrained model (ZDT4).
Consider the MOP CONV1. The Pareto set of this model which is equal to $\mathcal{P}$ is located within $[-1, 1]^n$. First, we turn our attention to the unconstrained case: Figure 3 shows two results obtained by the modified algorithms HCS1 and HCS2 with dimension $n = 10$ and domain $Q = [-5, 5]^{10}$. In both cases the same starting point x_0 has been chosen. Since $\mathcal{P}$ is located within Q, no constraint handling techniques had to be applied in order to generate the sequence. For HCS1 a total of 1693 function calls had to be spent in order to get this result. For HCS2, 207 function calls, 60 evaluations of the gradient and 192 evaluations of the Hessian were required (which are both given analytically. A conversion due to Section 4.2 would lead to 13,095 function calls). Figure 3 shows some qualitative differences as anticipated from the design of the different algorithms: HCS2 converges faster (in this case three iterates—not counting the function calls—were needed to reach $\mathcal{P}$ while HCS1 needed 6 iterations) and the non-

dominated front is much better compared to the results obtained by the gradient free version HCS1 (In fact, the solution of HCS2 is practically identical to the true Pareto front). However, both results are satisfying since both non-dominated fronts represent a good approximation of the Pareto front with reasonable effort.

Next we consider the constrained case. Figure 4 shows a numerical result from the HCS1 where we have used dimension $n = 2$ and for the domain $Q = [0.5, 1.5] \times [1, 2]$. The Pareto set is given by $P_Q = [0.5, 1] \times \{1\}$ and thus included in the boundary of Q. The figures show that also in this case the HCS1 is capable of approaching the solution set, and moving along it further on. However, a total of 997 function calls had to be spent in this setting, that is, more in comparison to the unconstrained case (note that the dimension of the model is much lower in the latter case).

Finally, we consider the problem ZDT4, which is a highly nonlinear and multimodal model. Figure 5 shows two results in image space for two different initial solutions $x_0, z_0 \in Q = [0, 1] \times [-5, 5]^9$ and for the two variants of the HCS. As anticipated, the results for both algorithms and starting points differ significantly since the HCS is a local strategy and ZDT4 contains many local Pareto fronts. However, both procedures also in this case are able to explore a part of the local Pareto front which is located 'near' to the image of the initial solution.

5.2 HCS Coupled with a MOEA

Here we make some comparisons of the two state of the art MOEAs NSGA-II and SPEA2 with their respective hybrid variants NSGA-II-HCS and SPEA2-HCS in order to demonstrate the possible benefit of the HCS when applied within a MOEA. Since we are dealing in this section with MOPs where the Pareto set is located at the boundary of the domain, we have used for these models a modification of HCS2 which acts just as a hill climber. That is, the search along the Pareto set is not performed (the value $\epsilon_{\mathcal{P}}$ is set to 0). To be conform with our notations and to avoid confusions, we denote this algorithm by $HC2$.

In order to evaluate the performance of the algorithms we have used the Generational Distance [66], the Inverted Generational Distance [9] and the Two Set Coverage Measure [70] as indicators. A brief description follows.

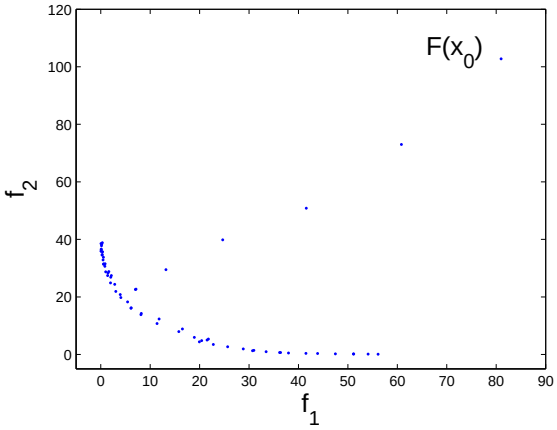
Denote by δ_i the minimum Euclidean distance from the image $F(x_i)$ of a given point $x_i, i = 1, \ldots, n$, to the true Pareto front $F(P_Q)$. The Generational Distance (GD) of a set (population) $P = \{x_1, \ldots, x_n\}$ is defined as

$$GD = \frac{1}{n}\sqrt{\sum_{i=1}^{n} \delta_i^2} \tag{27}$$

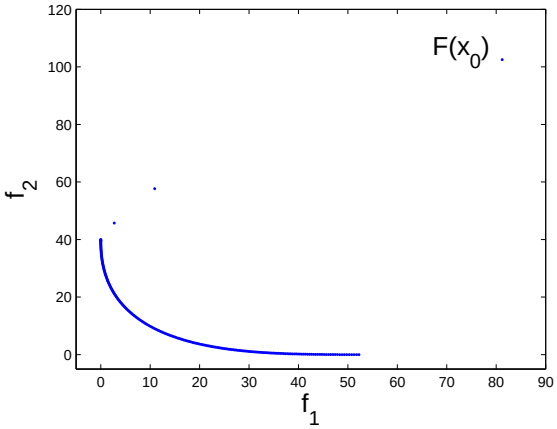
The Inverted Generational Distance (IGD) is analogous to GD but measured from $F(P_Q)$ to $F(P)$.

Given two finite subsets A and B of $\mathbb{R}^n$ the Two Set Coverage Measure is defined as

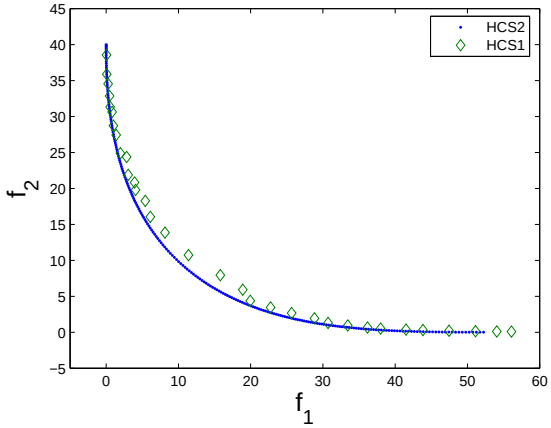
$$SC(A, B) = A \prec B = \frac{|\{b \in B \text{ such that } \exists a \in A \text{ with } a \prec b\}|}{|B|} \tag{28}$$



(a) Solution HCS1

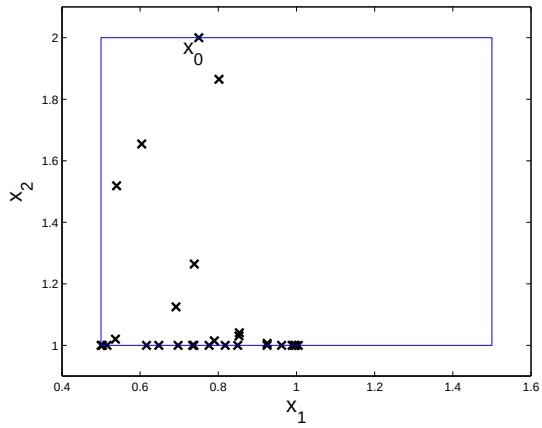


(b) Solution HCS2

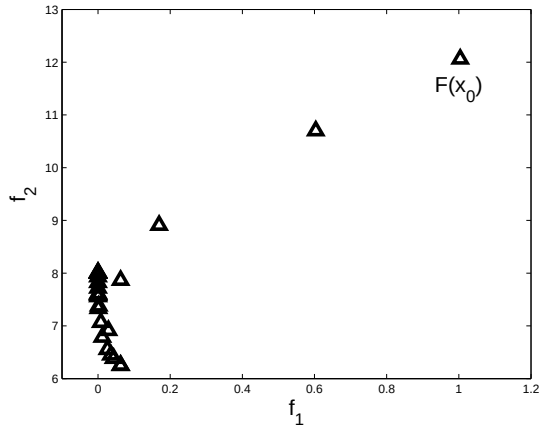


(c) Comparison Non-dominated Fronts

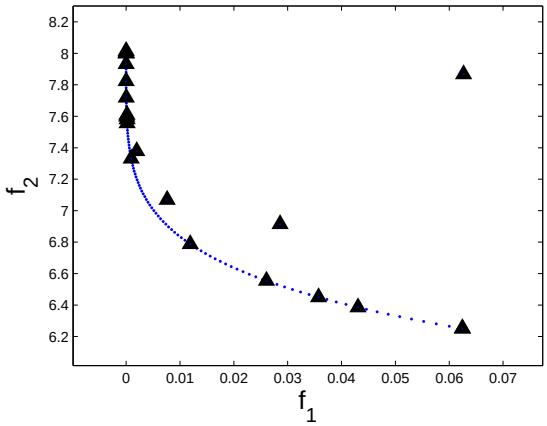
Fig. 3. Numerical result of HCS for MOP CONV1 with $Q = [-5, 5]^{10}$ in objective space (unconstrained case).



(a) Parameter Space

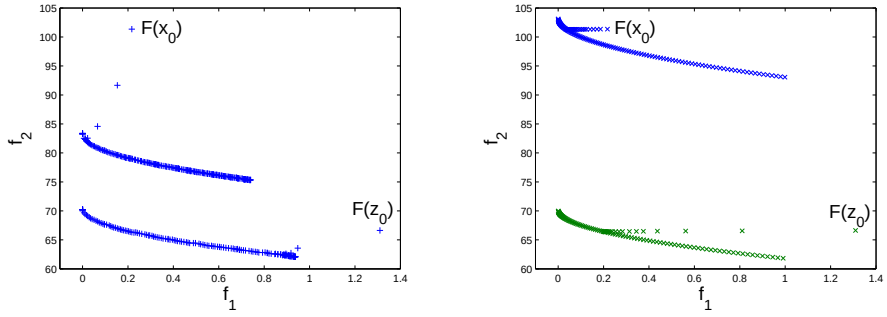


(b) Image Space



(c) Zoom Image Space and Pareto Front

Fig. 4. Numerical result of HCS1 for MOP CONV1 with $Q = [0.5, 1.5] \times [1, 2]$ (constrained case).



(a) Result HCS1 (b) Result HCS2

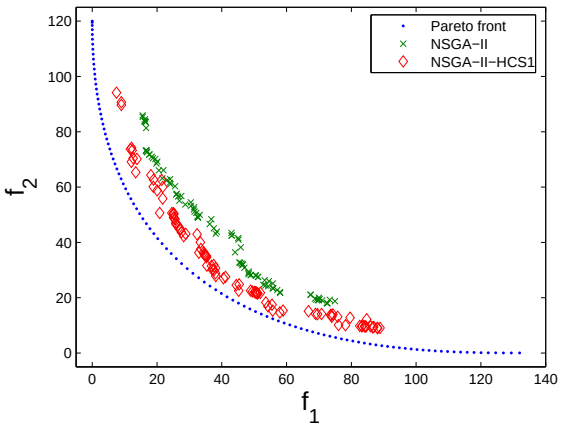
Fig. 5. Numerical result of HCS1 and HCS2 for MOP ZDT4 in objective space for two initial solutions x_0 and z_0.

If $A \prec B = 1$, it means that all the elements of B are dominated by at least one element of A. If on the other hand $A \prec B = 0$, it means that no element of B is dominated by any element of A. Since the Two Set Coverage Metric is not symmetric always both values $SC(A, B)$ and $SC(B, A)$ have to be taken into account.

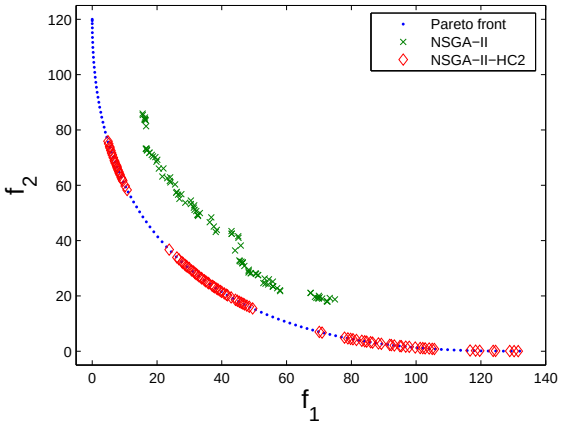
Convex Models First we consider the convex and thus uni-modal models CONV1 ($k = 2$ objectives) and CONV2 ($k = 3$) which are taken from [15], and set the dimension of the parameter space to $n = 30$. For both models we are interested in the unconstrained case (where the Pareto set does not intersect with the boundary of the domain) and in the constrained case. Since the Pareto set of both unconstrained problems (that is, for $Q = \mathbb{R}^{30}$) is located within $[-1, 1]^{30}$ we have chosen to take the domains $Q_u := [-5, 5]^{30}$ and $Q_c = [-1, 1] \times [1, 2]^{29}$ for the unconstrained and the constrained model, respectively. Denote the resulting models by CONV1-U and CONV1-C (analogue for CONV2).
Tables 7 and 8 show averaged numerical results obtained on the convex models using SPEA2, NSGA-II and the according memetic strategies. For the realization we have used the parameter values displayed in Table 6 and a budget of $B = 10,000$ function calls. B is chosen relatively low in order to obtain significant differences in the indicator values. For the two cases where we hybridize the base MOEA with the HCS the following observations can be made: the values of the Set Coverage and the GD improve when the HCS1 is used for additional local search. The values are even much better when using HC2 or HCS2 for the local search (the improvement is roughly one order of magnitude). The latter is certainly due to the fact that we are dealing with convex models: the gradient based search—though much more costly than HCS1 for $n = 30$, see Table 3—leads to great improvements of given initial points which do not get stuck at local solutions. Thus, the population is pulled to the 'right' set at any stage of

the optimization process. For IGD the results are not that conclusive, however, improvements can be observed.
In Figure 6, one result of NSGA-II and their memetic variants is plotted which reflects the above discussion: Figure 6 (a) shows the effect of the HCS1, i.e., better convergence and spread than the result of NSGA-II due to the two search directions of HCS1. Convergence is on the other side much better in Figures 6 (b) and (c) where gradient information is used. When comparing the latter two figures, the effect of the sidestep get visible: The spread in Figure 6 (c) is apparently better than in Figure 6 (b), where the solutions fall into clusters.
Table 9 gives an impression on the overall cost of the HCS within the search procedure. The table shows the amount of calls of the hill climber and sidestep procedures of HCS within SPEA2-HCS used to obtain the results in Figure 6. HCS1 used 22 percent of the total budget B of SPEA2-HCS1. The relatively large amount of sidestep calls is due to the low value of N_{nd} (=3). Larger values of N_{nd} would yield in less sidestep calls and in turn more hill climber calls. The cheapest local search operator is HC2 with a portion of 13 percent of the function calls since this operator merely computes the gradients to perform the hill climber. The sidestep operator is much more costly since the second derivatives are involved as Table 9 shows for HCS2: this local search operator spent 61 percent of B. The better result in Figure 6 (c) compared to Figure 6 (b) in terms of spread was obtained by merely 20 sidestep calls which, however, used quite a lot of function calls for this.
Concluding, it can be said that on the convex models (two and three objectives, constrained and unconstrained) a combination of the two base MOEAs with the HCS variants improves on one hand the overall performance of the search. On the other hand considerations of the cost of the operators show that its application should be handled with care since else the HCS can take the lion's share of the budget which results in a risk to decrease the overall performance (note that we have assumed B to be constant).

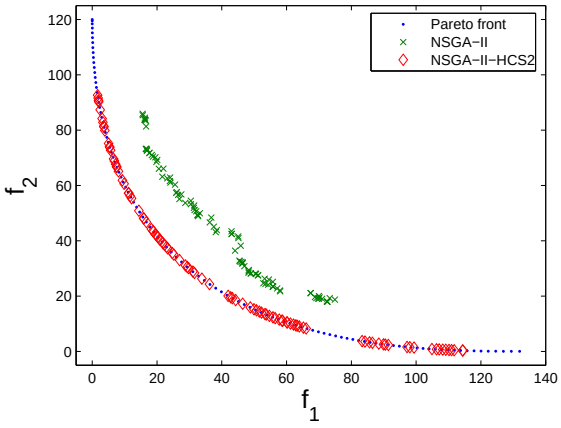
DTLZ for Three Objectives Finally, we consider the MOPs DTLZ1, DTLZ2 and DTLZ3 (see [14] and Table 5), where we have chosen $n = 30$ for the dimension of the parameter space, $k = 3$ objectives, and the domain $Q = [0, 1]^{30}$ for all models.
Tables 11 and 12 show averaged numerical results obtained on the test functions by the MOEAs and MEMOEAs under consideration. Hereby, we have used the parameter values shown in Table 10 and a budget of $B_1 = 100,000$ function calls for the multi-modal models DTLZ1 and DTLZ3 and a budget of $B_2 = 10,000$ for the uni-modal model DTLZ2 (this has again been done in order to prevent too small values of the indicators). The conclusions which can be drawn from the results are not as straightforward as for the convex case. While a similar trend as for the convex case can be observed for DTLZ2, this does not hold for the multi-modal models. Apparently, the two MEMOEAs which use HC2 can not compete with their base MOEAs (however, to be fair the construction of the models already suggests that gradient information is worthless. Thus, it is



(a) NSGA-II vs. NSGA-II-HCS1



(b) NSGA-II vs. NSGA-II-HC2



(c) NSGA-II vs. NSGA-II-HCS2

Fig. 6. Numerical result for CONV1-U using NSGA-II and the three memetic strategies NSGA-II-HCS1, NSGA-II-HC2, and NSGA-II-HCS2. The best result (spread *and* convergence) in this case was obtained by SPEA2-HCS2.

Table 6. Parameter values used for SPEA2 and NSGA-II and the memetic strategies SPEA2-HCS and NSGA-II-HCS on the convex problems.

	SPEA2-HCS		NSGA-II-HCS	
Parameters	*unbounded*	*bounded*	*unbounded*	*bounded*
N_{pop}	100	100	100	100
N_a	100	100	-	-
η_c	-	-	15	15
η_m	-	-	20	20
p_c	0.9	0.9	0.9	0.9
p_m	0.1	0.1	1/30	1/30
p_{HCS1}	0.2	0.2	-	-
s_{HCS1}	5	5	10	10
p_{HCS2}	0.1	0.1	-	-
s_{HCS2}	10	10	10	10
ε_y	5	2	5	5
$\varepsilon_{\mathcal{P}}$	0.0001	0.0001	0.0001	0.0001
r	0.1	0.02	0.1	0.1
maxiter	10	10	10	10
N_{nd}	5	5	3	3
tol	10^{-4}	10^{-4}	10^{-4}	10^{-4}

rather a question of the choice of the model than an indication of a general failure of the HC2). On the other hand, SPEA2-HCS1 outperforms its base MOEA SPEA2 significantly on these highly multi-modal models. Such an improvement, however, can not be observed from NSGA-II-HCS1 and NSGA-II. The latter MOEA is already performing very good on these MOPs which makes it hard for a local search strategy such as the HCS—which causes extra cost—to improve the overall performance, in particular on such complex problems.

Based on the numerical results presented in this section, it can be said that both variants of the standalone HCS accomplish their task, i.e., they are capable of moving toward and along the Pareto set. That is, by using the HCS, entire connected components of the (local) Pareto set can be explored starting with one single solution. Furthermore, it has been shown that the hybridization of the HCS with a given MOEAs can improve the overall performance of the base MOEA. More precisely, satisfying results have yet been obtained for uni-modal MOPs. The results on multi-modal models, however, are so far not that conclusive in general but depending on the base MOEA. Due to the relative high cost of the HCS and the natural handicap of local search methods for multi-modal problems the balance of local and genetic search (such as for DTLZ1 and DTLZ3) is a challenging task. To handle this efficiently, adaptive strategies are indispensable, and further research should be done in that direction.

Table 7. Numerical results of SPEA2 and the memetic strategies SPEA2-HCS on the convex problems using dimension $n = 30$ and performing 10,000 function calls. Statistics were gathered from 30 independent runs.

	Indicators		
Problems		GD	IGD
CONV1-U	SPEA2(A)	3.5762(0.9190)	2.5469(0.3460)
	SPEA2-HCS1(B)	1.9399(0.5721)	2.2846(0.5483)
	SPEA2-HCS2(C)	0.1139(0.0505)	1.8377(0.9405)
	SPEA2-HC2(D)	0.1483(0.0463)	1.8792(0.9371)
CONV1-C	SPEA2(A)	6.8446(2.7245)	1.2491(0.1268)
	SPEA2-HCS1(B)	6.3508(1.9602)	1.1985(0.1225)
	SPEA2-HC2(C)	0.2800(1.1503)	0.4268(0.0069)
CONV2-U	SPEA2(A)	3.5762(0.9190)	2.5469(0.3460)
	SPEA2-HCS1(B)	1.9399(0.5721)	2.2846(0.5483)
	SPEA2-HCS2(C)	0.1139(0.0505)	1.8377(0.9405)
	SPEA2-HC2(D)	0.1483(0.0463)	1.8792(0.9371)
CONV2-C	SPEA2(A)	1.7887(0.4642)	0.4882(0.1086)
	SPEA2-HCS1(B)	1.6342(0.7454)	0.4341(0.1303)
	SPEA2-HC2(C)	0.4902(0.1789)	0.2774(0.1440)

Set Coverage CONV1-U					
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$	$D \prec A$	$A \prec D$
0.7834(0.2726)	0.0984(0.1929)	0.9742(0.0755)	0(0)	0.9770(0.0753)	0(0)

Set Coverage CONV1-C			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
0.4881(0.4119)	0.2932(0.3604)	1(0)	0(0)

Set Coverage CONV2-U					
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$	$D \prec A$	$A \prec D$
0.5602(0.2900)	0.1223(0.1405)	0.8963(0.2118)	0(0)	0.9893(0.0324)	0(0)

Set Coverage CONV2-C			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
0.6426(0.3622)	0.1796 (0.2752)	1(0)	0(0)

Table 8. Numerical results of NSGA-II and the memetic strategies NSGA-II-HCS on the convex problems using dimension $n = 30$ and performing 10,000 function calls. Statistics were gathered from 30 independent runs.

	Indicators		
Problems		GD	IGD
CONV1-U	NSGA-II(A)	1.2847(0.2258)	1.6994(0.2843)
	NSGA-II-HCS1(B)	0.5661(0.0938)	1.1123(0.4191)
	NSGA-II-HCS2(C)	0.0606(0.0054)	1.5931(0.9827)
	NSGA-II-HC2(D)	0.0590(0.0048)	1.3167(0.8445)
CONV1-C	NSGA-II(A)	1.3747(0.1687)	1.0594(0.1027)
	NSGA-II-HCS1(B)	0.1143(0.0417)	0.3470(0.0661)
	NSGA-II-HC2(C)	0.0063(0.0041)	0.0386(0.0540)
CONV2-U	NSGA-II(A)	2.1814(0.4247)	0.4618(0.0652)
	NSGA-II-HCS1(B)	1.1465(0.1249)	0.4533(0.0784)
	NSGA-II-HCS2(C)	0.1041(0.0133)	0.3815(0.1582)
	NSGA-II-HC2(D)	0.1171(0.0136)	0.3374(0.1466)
CONV2-C	NSGA-II(A)	2.1165(0.3591)	1.4540(0.1725)
	NSGA-II-HCS1(B)	0.4333(0.1673)	0.5873(0.0794)
	NSGA-II-HC2(C)	0.0127(0.0114)	0.4232(0.1127)

Set Coverage CONV1-U					
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$	$D \prec A$	$A \prec D$
0.9516(0.0977)	0.0218(0.0597)	0.9412(0.0825)	0.0032(0.0155)	0.9418(0.1144)	0(0)

Set Coverage CONV1-C			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
1(0)	0(0)	1(0)	0(0)

Set Coverage CONV2-U					
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$	$D \prec A$	$A \prec D$
0.8433(0.1572)	0.0103(0.0332)	0.9397(0.1233)	0(0)	0.9633(0.1035)	0(0)

Set Coverage CONV2-C			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
1(0)	0 (0)	1(0)	0(0)

Table 9. Cost of the HCS variants within SPEA2-HCS in one run on CONV1-U (the run which produced the result displayed in Figure 6).

Method	Hill climber calls (no sidestep)	Sidestep calls (w or w/o sidestep)	Function calls (total)
HCS1	7	68	2229
HC2	69	0	1262
HCS2	51	20	6145

Table 10. Parameter values used for SPEA2 and NSGA-II and the memetic strategies SPEA2-HCS and NSGA-II-HCS on the DTLZ functions.

Parameters	SPEA2-HCS	NSGA-II-HCS
N_{pop}	200	200
N_a	100	-
η_c	-	15
η_m	-	20
p_c	0.9	0.9
p_m	0.01	1/30
p_{HCS1}	0.3	-
p_{HCS2}	0.3	-
s	10	10
ε_y	1	5
$\varepsilon_{\mathcal{P}}$	0.0001	0.0001
r	0.05	0.1
maxiter	5	10
N_{nd}	5	3
tol	10^{-4}	10^{-4}

Table 11. Numerical results of SPEA2 and the memetic strategies SPEA2-HCS on the DTLZ benchmarks using dimensions $n = 30$, $k = 3$, and performing 100,000 function calls. Statistics were gathered from 30 independent runs.

	Indicators		
Problems		GD	IGD
$DTLZ1$	SPEA2(A)	22.7984(1.6658)	2.4303(0.4716)
	SPEA2-HCS1(B)	12.2165(3.9738)	1.5389(0.2132)
	SPEA2-HC2(C)	48.06789(7.5157)	1.3770(0.4404)
$DTLZ2$	SPEA2(A)	0.0529(0.0102)	0.0011(0.0001)
	SPEA2-HCS1(B)	0.0342(0.0404)	0.0007(0.0001)
	SPEA2-HC2(C)	0.0557(0.0164)	0.0012(0.0001)
$DTLZ3$	SPEA2(A)	218.3484(9.9639)	10.3885(2.2814)
	SPEA2-HCS1(B)	28.1974(3.8434)	3.2152(0.3890)
	SPEA2-HCS2(C)	198.8947(27.5283)	3.6925(0.7106)

Set Coverage DTLZ1			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
0.9789(0.0286)	0.0157(0.0227)	0.6925(0.2840)	0.2100(0.2710)

Set Coverage DTLZ2			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
0.7820(0.0779)	0.0040(0.0089)	0.2060(0.0888)	0.3660(0.1740)

Set Coverage DTLZ3			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
1(0)	0(0)	0.7905(0.0862)	0.0877(0.0586)

Table 12. Numerical results of NSGA-II and the memetic strategies NSGA-II-HCS on the DTLZ benchmarks using dimensions $n = 30$, $k = 3$, and performing 100,000 function calls. Statistics were gathered from 30 independent runs.

	Indicators		
Problems		GD	IGD
$DTLZ1$	NSGA-II(A)	8.5024(1.2081)	1.5998(0.1530)
	NSGA-II-HCS1(B)	10.4444(1.1505)	1.6856(0.2202)
	NSGA-II-HC2(C)	18.1555(4.6705)	1.0096(0.1899)
$DTLZ2$	NSGA-II(A)	0.0363(0.0041)	0.0022(0.0002)
	NSGA-II-HCS1(B)	0.0293(0.0033)	0.0018(0.0001)
	NSGA-II-HC2(C)	0.0097(0.0039)	0.0005(0)
$DTLZ3$	NSGA-II(A)	17.6126(3.5630)	3.2696(0.7721)
	NSGA-II-HCS1(B)	16.9381(2.9723)	2.9951(0.6861)
	NSGA-II-HC2(C)	24.6485(3.6724)	2.5127(0.4320)

Set Coverage DTLZ1			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
0.1610(0.1376)	0.6035(0.2565)	0.4885(0.1354)	0.3960(0.1627)

Set Coverage DTLZ2			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
0.7125(0.1869)	0.1235(0.0790)	0.9715(0.0284)	0.0105(0.0116)

Set Coverage DTLZ3			
$B \prec A$	$A \prec B$	$C \prec A$	$A \prec C$
0.5010(0.3365)	0.3965(0.3243)	0.4435(0.3225)	0.4550(0.2857)

6 Conclusions and Future Work

We have proposed a novel point-wise iterative search procedure, the Hill Climber with Sidestep (HCS), which is designed for the local search of a given multi-objective optimization problem. The HCS is intended to be capable to moving both toward and along the set of (local) Pareto points, depending on the location of the current iterate. We have proposed two variants of the HCS, a gradient free version (HCS1) and one version which exploits gradient information (HCS2). Both algorithms can be used as standalone algorithms to explore parts of the Pareto set starting with one single solution and are able to handle constraints of the model to some extend. Further, we have addressed the problem of integrating the HCS into a given MOEA in order to obtain a novel memetic strategy. As examples, we have proposed the two algorithms (or family of algorithms) SPEA2-HCS and NSGA-II-HCS which are derived from SPEA2 and NSGA-II, respectively. Finally, we have presented some numerical results indicating the efficiency of the HCS as a standalone algorithm and its benefit when beeing integrated into a MOEA. More precisely, the results of SPEA2-HCS and NSGA-II-HCS show that the combination as proposed here is advantageous in many cases. However, it has to be mentioned that for this the design parameters of the HCS and the balance between local and genetic search has to be chosen properly, which is so far not done in an adaptive manner.

For future work, there are some interesting topics which can be addressed to advance the present work. In the first place, there is probably the design of an adaptive strategy to choose the design parameters as discussed above. Further, the coupling of local and global search could be improved. For this, an adaptive choice of the local search probability would be desirable (as done in [8]), or other techniques to reduce the computational complexity, e.g., by involving the information of the current population into the HCS. Another interesting topic would be to develop a version of HCS2 which can move efficiently along the Pareto set even if it is contained in the boundary of the domain which would allow for a more general use of the algorithm. Finally, it is conceivable to apply and advance the standalone HCS to other contexts. Areas which seem to be promising for that are numerical path-following (see [54] for a first study) or dynamic multi-objective optimization.

Acknowledgements

The authors thank the anonymous reviewers for their valuable comments which greatly helped them to improve the contents of the paper. The third author gratefully acknowledges support from the CONACyT project no. 45683-Y.

References

1. S. F. Adra, I. Griffin, and P. J. Fleming. Hybrid Multiobjective Genetic Algorithm with a New Adaptive Local Search Process. In Hans-Georg Beyer et al.,

editor, *2005 Genetic and Evolutionary Computation Conference (GECCO'2005)*, volume 1, pages 1009–1010, New York, USA, June 2005. ACM Press.
2. E. L. Allgower and K. Georg. *Numerical Continuation Methods*. Springer, 1990.
3. L. Armijo. Minimization of functions having Lipschitz-continuous first partial derivatives. *Pacific Journal of Mathematics*, 16:1–3, 1966.
4. N. K. Bambha, S. S. Bhattacharyya, J. Teich, and E. Zitzler. Systematic integration of parameterized local search into evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 8(2):137–155, 2004.
5. P. A. N. Bosman and E. D. de Jong. Exploiting Gradient Information in Numerical Multi-Objective Evolutionary Optimization. In Hans-Georg Beyer et al., editor, *2005 Genetic and Evolutionary Computation Conference (GECCO'2005)*, volume 1, pages 755–762, New York, USA, June 2005. ACM Press.
6. J. Branke and S. Mostaghim. About Selecting the Personal Best in Multi-Objective Particle Swarm Optimization. In Thomas Philip Runarsson, Hans-Georg Beyer, Edmund Burke, Juan J. Merelo-Guervós, L. Darrell Whitley, and Xin Yao, editors, *Parallel Problem Solving from Nature - PPSN IX, 9th International Conference*, pages 523–532. Springer. Lecture Notes in Computer Science Vol. 4193, Reykjavik, Iceland, September 2006.
7. M. Brown and R. E. Smith. Directed Multi-Objective Optimisation. *International Journal of Computers, Systems and Signals*, 6(1):3–17, 2005.
8. A. Caponio and F. Neri. Integrating cross-dominance adaption in multi-objective memetic algorithms. In C.-K. Goh, Y.-S. Ong, and K. C. Tan, editors, *Multi-Objective Memetic Algorithms*, pages 325–351. Springer, Studies in Computational Intelligence , Vol. 171, 2009.
9. C. A. Coello Coello and N. Cruz Cortés. Solving Multiobjective Optimization Problems using an Artificial Immune System. *Genetic Programming and Evolvable Machines*, 6(2):163–190, June 2005.
10. C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer, New York, second edition, September 2007. ISBN 978-0-387-33254-3.
11. I. Das and J. Dennis. *Normal-boundary intersection*: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems. *SIAM Journal of Optimization*, 8:631–657, 1998.
12. K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Chichester, UK, 2001. ISBN 0-471-87339-X.
13. K. Deb, A. Pratap, Sameer Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.
14. K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable Test Problems for Evolutionary Multiobjective Optimization. In Ajith Abraham, Lakhmi Jain, and Robert Goldberg, editors, *Evolutionary Multiobjective Optimization. Theoretical Advances and Applications*, pages 105–145. Springer, USA, 2005.
15. M. Dellnitz, O. Schütze, and T. Hestermeyer. Covering Pareto sets by multilevel subdivision techniques. *Journal of Optimization Theory and Applications*, 124:113–155, 2005.
16. J.E. Dennis and R.B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, 1983.
17. J. Fliege and B. Fux Svaiter. Steepest descent methods for multicriteria optimization. *Mathematical Methods of Operations Research*, 51(3):479–494, 2000.
18. F. Glover and M. Laguna. *Tabu Search*. Springer, 1997.

19. T. Goel and K. Deb. Hybrid Methods for Multi-Objective Evolutionary Algorithms. In Lipo Wang, Kay Chen Tan, Takeshi Furuhashi, Jong-Hwan Kim, and Xin Yao, editors, *Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning (SEAL'02)*, volume 1, pages 188–192, Orchid Country Club, Singapore, November 2002. Nanyang Technical University.
20. A. Griewank. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. Number 19 in Frontiers in Appl. Math. SIAM, Philadelphia, PA, 2000.
21. N. Hansen and A. Ostermeier. Completely Derandomized Self-adaptation in Evolution Strategies. *Evolutionary Computation*, 9(2):159–195, Summer 2001.
22. K. Harada, J. Sakuma, and S. Kobayashi. Uniform Sampling of Local Pareto-Optimal Solution Curves by Pareto Path Following and its Applications in Multi-objective GA. In Dirk Thierens, editor, *2007 Genetic and Evolutionary Computation Conference (GECCO'2007)*, volume 1, pages 813–820, London, UK, July 2007. ACM Press.
23. W. E. Hart. *Adaptive Global Optimization with Local Search*. PhD thesis, Univeristy of California, San Diego. USA, 1994.
24. C. Hillermeier. *Nonlinear Multiobjective Optimization - A Generalized Homotopy Approach*. Birkhäuser, 2001.
25. X. Hu, Z. Huang, and Z. Wang. Hybridization of the Multi-Objective Evolutionary Algorithms and the Gradient-based Algorithms. In *Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003)*, volume 2, pages 870–877, Canberra, Australia, December 2003. IEEE Press.
26. C. Igel, N. Hansen, and S. Roth. Covariance Matrix Adaptation for Multi-objective Optimization. *Evolutionary Computation*, 15(1):1–28, Spring 2007.
27. A. W. Iorio and X. Li. Solving rotated multi-objective optimization problems using differential evolution. In *AI 2004: Advances in Artificial Intelligence, Proceedings*, pages 861–872. Springer-Verlag, Lecture Notes in Artificial Intelligence Vol. 3339, 2004.
28. A. W. Iorio and X. Li. Rotated problems and rotationally invariant crossover in evolutionary multi-objective optimization. *International Journal of Computational Intelligence and Applications*, 7(2):149–186, 2008.
29. H. Ishibuchi and T. Murata. Multi-Objective Genetic Local Search Algorithm. In Toshio Fukuda and Takeshi Furuhashi, editors, *Proceedings of the 1996 International Conference on Evolutionary Computation*, pages 119–124, Nagoya, Japan, 1996. IEEE.
30. H. Ishibuchi and T. Murata. A multi-objective genetic local search algorithm and its application to flowshop scheduling. *IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews*, 28(3):392–403, August 1998.
31. H. Ishibuchi, T. Yoshida, and T. Murata. Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. *IEEE Transactions on Evolutionary Computation*, 7(2):204–223, 2003.
32. A. Jaszkiewicz. Do Multiple-Objective Metaheuristics Deliver on Their Promises? a Computational Experiment on the Set-Covering Problem. *IEEE Transactions on Evolutionary Computation*, 7(2):133–143, April 2003.
33. W. E. Karush. *Minima of functions of several variables with inequalities as side conditions*. PhD thesis, University of Chicago, 1939.
34. S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science. New Series*, 220(4598):671–680, 1983-05-13.

35. J. Knowles and D. Corne. M-PAES: A Memetic Algorithm for Multiobjective Optimization. In *2000 Congress on Evolutionary Computation*, volume 1, pages 325–332, Piscataway, New Jersey, July 2000. IEEE Service Center.
36. J. Knowles and D. Corne. Memetic Algorithms for Multiobjective Optimization: Issues, Methods and Prospects. In William E. Hart, N. Krasnogor, and J.E. Smith, editors, *Recent Advances in Memetic Algorithms*, pages 313–352. Springer. Studies in Fuzziness and Soft Computing, Vol. 166, 2005.
37. N. Krasnogor. *Studies on the Theory and Design Space of Memetic Algorithms*. PhD thesis, University of the West of England, Bristol, United Kingdom, 2002.
38. H. Kuhn and A. Tucker. Nonlinear programming. In J. Neumann, editor, *Proceeding of the 2nd Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492, 1951.
39. M. Lozano, F. Herrera, N. Krasnogor, and D. Molina. Real-coded memetic algorithms with crossover hill-climbing. *Evolutionary Computation*, 12(3):273–302, 2004.
40. K. Miettinen. *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers, Boston, Massachusetts, 1999.
41. T. Murata, S. Kaige, and H. Ishibuchi. Generalization of Dominance Relation-Based Replacement Rules for Memetic EMO Algorithms. In Erick Cantú-Paz et al., editor, *Genetic and Evolutionary Computation—GECCO 2003. Proceedings, Part I*, pages 1234–1245. Springer. Lecture Notes in Computer Science Vol. 2723, July 2003.
42. J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer, New York, 1999.
43. U.-M. O'Reilly and F. Oppacher. Hybridized crossover-based search techniques for program discovery. In *Proceedings of the 1995 World Conference on Evolutionary Computation*, volume 2, pages 573–578, Perth, Australia, 1995. IEEE Press.
44. V. Pareto. *Manual of Political Economy*. The MacMillan Press, 1971 (original edition in French in 1927).
45. S. Poles, E. Rigoni, and T. Robič. MOGA-II Performance on Noisy Optimization Problems. In Bogdan Filipič and Jurij Šilc, editors, *Bioinspired Optimization Methods and Their Applications. Proceedings of the International Conference on Bioinspired Optimization Methods and their Applications, BIOMA 2004*, pages 51–62. Jožef Stefan Institute, Ljubljana, Slovenia, October 2004.
46. E. Rigoni and S. Poles. NBI and MOGA-II, two complementary algorithms for Multi-Objective optimization. In *Dagstuhl Seminar Proceedings 04461. Practical Approaches to Multi-Objective Optimization*, pages 1–22, 2005.
47. H.H. Rosenbrock. An automatic method for finding the greatest or least value of a function. *The Computer Journal*, 3(3):175–184, 1960.
48. H. Satoh, M. Yamamura, and S. Kobayashi. Minimal generation gap model for gas considering both exploration and exploitation. In *Iizuka '96*, volume 2, pages 494–497, Fukuoka, Japan, 1996.
49. S. Schäffler, R. Schultz, and K. Weinzierl. A stochastic method for the solution of unconstrained vector optimization problems. *Journal of Optimization Theory and Applications*, 114(1):209–222, 2002.
50. O. Schuetze, M. Laumanns, E. Tantar, C. A. Coello Coello, and E.-G. Talbi. Convergence of Stochastic Search Algorithms to Gap-Free Pareto Front Approximations. In Dirk Thierens, editor, *2007 Genetic and Evolutionary Computation Conference (GECCO'2007)*, volume 1, pages 892–899, London, UK, July 2007. ACM Press.

51. O. Schuetze, G. Sanchez, and C. A. Coello Coello. A new memetic strategy for the numerical treatment of multi-objective optimization problems. In *2008 Genetic and Evolutionary Computation Conference (GECCO'2008)*, pages 705–712. ACM Press, Atlanta, USA, July 2008. ISBN 978-1-60558-131-6.
52. O. Schütze. *Set Oriented Methods for Global Optimization*. PhD thesis, University of Paderborn, 2004. <http://ubdata.uni-paderborn.de/ediss/17/2004/schuetze/>.
53. O. Schütze, C. A. Coello Coello, S. Mostaghim, E.-G. Talbi, and M. Dellnitz. Hybridizing Evolutionary Strategies with Continuation Methods for Solving Multi-Objective Problems. *Engineering Optimization*, 40(5):383–402, May 2008.
54. O. Schütze, A. Lara, and C. A. Coello Coello. Evolutionary continuation methods for optimization problems. Proceedings of Genetic and Evolutionary Computation Conference, (GECCO 2009, to appear), 2009.
55. O. Schütze, S. Mostaghim, M. Dellnitz, and J. Teich. Covering Pareto Sets by Multilevel Evolutionary Subdivision Techniques. In Carlos M. Fonseca, Peter J. Fleming, Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele, editors, *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003*, pages 118–132, Faro, Portugal, April 2003. Springer. Lecture Notes in Computer Science. Volume 2632.
56. O. Schütze, E.-G. Talbi, C. A. Coello Coello, L. V. Santana-Quintero, and G. Toscano Pulido. A Memetic PSO Algorithm for Scalar Optimization Problems. In *Proceedings of the 2007 IEEE Swarm Intelligence Symposium (SIS 2007)*, pages 128–134, Honolulu, Hawaii, USA, April 2007. IEEE Press.
57. K. Sindhya, K. Deb, and K. Miettinen. A Local Search Based Evolutionary Multi-objective Optimization Approach for Fast and Accurate Convergence. In Günter Rudolph, Thomas Jansen, Simon Lucas, Carlo Poloni, and Nicola Beume, editors, *Parallel Problem Solving from Nature–PPSN X*, pages 815–824. Springer. Lecture Notes in Computer Science Vol. 5199, Dortmund, Germany, September 2008.
58. A. Sinha, Y.-P. Chen, and D. E. Goldberg. Designing Efficient Genetic and Evolutionary Algorithm Hybrids. In William E. Hart, N. Krasnogor, and J.E. Smith, editors, *Recent Advances in Memetic Algorithms*, pages 259–288. Springer. Studies in Fuzziness and Soft Computing, Vol. 166, 2005.
59. J. E. Smith and T. C. Fogarty. Operator and parameter adaption in genetic algorithms. *Soft Computing*, 1(2):81–87, 1997.
60. O. Soliman, L. T. Bui, and H. Abbass. A memetic coevolutionary multi-objective diffierential evolution algorithm. In C.-K. Goh, Y.-S. Ong, and K. C. Tan, editors, *Multi-Objective Memetic Algorithms*, pages 325–351. Springer, Studies in Computational Intelligence , Vol. 171, 2009.
61. B. Suman. Study of simulated annealing based algorithms for multiobjective optimization of a constrained problem. *Computers & Chemical Engineering*, 28:1849–1871, 2004.
62. M. Vasile. A Behavior-based Meta-Heuristic for Robust Global Trajectory Optimization. In *2007 IEEE Congress on Evolutionary Computation (CEC'2007)*, pages 494–497, Singapore, 2007. IEEE Press.
63. M. Vasile. Hybrid behavioral-based multiobjective space trajectory optimization. In C.-K. Goh, Y.-S. Ong, and K. C. Tan, editors, *Multi-Objective Memetic Algorithms*, pages 231–254. Springer, Studies in Computational Intelligence , Vol. 171, 2009.
64. M. Vasile and M. Locatelli. A hybrid multiagent approach for global trajectory optimization. *Journal of Global Optimization*, 2008. (in press) DOI:10.1007/s10898-008-9329-3.

65. M. Vasile and C. Maddock. Design of optimal spacecraft-asteroid formations through a hybrid global optimization approach. *International Journal of Intelligent Computing and Cybernetics*, 1(2):239–268, 2008.
66. D. A. Van Veldhuizen. *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. PhD thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio, May 1999.
67. Y. Wang, Z. Cai, G. Guo, and Y. Zhou. Multiobjective optimization and hybrid evolutionary algorithm to solve constrained optimization problems. *IEEE Transactions on Systems, Man and Cybernetics Part B–Cybernetics*, 37(3):560–575, June 2007.
68. E. F. Wanner, F. G. Guimaraes, R. H.C. Takahashi, and P. J. Fleming. A Quadratic Approximation-Based Local Search Procedure for Multiobjective Genetic Algorithms. In *2006 IEEE Congress on Evolutionary Computation (CEC'2006)*, pages 3361–3368, Vancouver, BC, Canada, July 2006. IEEE.
69. Y. Yin, T. Okabe, and B. Sendhoff. Adapting weighted aggregation for multiobjective evolution strategies. In *Evolutionary Multi-Criterion Optimization. First International Conference, EMO 2001*, pages 96–110. Springer. Lecture Notes in Computer Science. Volume 1993, 2001.
70. E. Zitzler, K. Deb, and L. Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2):173–195, Summer 2000.
71. E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. In K. Giannakoglou, D. Tsahalis, J. Periaux, P. Papailou, and T. Fogarty, editors, *EUROGEN 2001. Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*, pages 95–100, Athens, Greece, 2002.
72. E. Zitzler and L. Thiele. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, November 1999.