# Multiobjective optimization Using an Adaptive Weighting Scheme

Shubhangi Deshpande[a], Layne T. Watson[a,b,c] , Robert A. Canfield[c]

[a]Department of Computer Science, Virginia Polytechnic Institute & State University, Blacksburg, VA

[b]Department of Mathematics, Virginia Polytechnic Institute & State University, Blacksburg, VA

[c]Department of Aerospace & Ocean Engineering, Virginia Polytechnic Institute & State University, Blacksburg, VA

**Abstract**

*Keywords:* multiobjective optimization; Pareto optimality; direct search method; DIRECT; MADS; surrogates; triangulation.

## 1. Introduction

Multiobjective optimization problems (MOPs) arise in practically every area of science and engineering where optimal decisions need to be made in the presence of two or more conflicting objectives. A decision maker has to take into account different criteria that need to be satisfied simultaneously.

Let $\mathbb{R}^n$ denote real $n$-dimensional Euclidean space. For vectors $x^{(1)}, x^{(2)} \in \mathbb{R}^n$ write $x^{(1)} < x^{(2)}$ ($x^{(1)} \leq x^{(2)}$) if $\forall i,\ x_i^{(1)} < x_i^{(2)}$ $\left(x_i^{(1)} \leq x_i^{(2)}\right)$, $i = 0, \ldots, n$. Let $l,\ u \in \mathbb{R}^n$ with $l < u$ and $B = \{x \in \mathbb{R}^n \mid l \leq x \leq u\}$. Given $p$ conflicting objective functions $f_i : B \to \mathbb{R}$, $i = 1, \ldots, p$, the multiobjective optimization problem is to simultaneously minimize all the $f_i$ over $B$.

For a nontrivial multiobjective optimization problem with $p$ conflicting objectives no single solution can optimize all $p$ objectives simultaneously. Hence, a new notion of optimality, known as Pareto optimality (the set of optimal solutions is known as the Pareto front), is generally used in the context of multiobjective optimization problems. Pareto optimality is generally defined using a dominance relation as follows: Given two decision vectors $x^{(1)}$ and $x^{(2)}$, $x^{(1)}$ is said to dominate $x^{(2)}$, denoted as $x^{(1)} \prec x^{(2)}$, if and only if $x^{(1)}$ is no worse than $x^{(2)}$ in all objectives, and $x^{(1)}$ is strictly better than $x^{(2)}$ in at least one objective. A point $x^* \in B$ is a globally Pareto optimal solution if and only if there does not exist any $x$ in the design space B such that $x \prec x^*$. A point $x^* \in B$ is a locally Pareto optimal solution if and only if for some $\epsilon > 0$ there does not exist any $x$ in the neighborhood $\left\{x \mid \|x - x^*\| < \epsilon\right\} \cap B$ of $x^*$ that dominates $x^*$.

The goal of a MOP is to find all the Pareto optimal solutions for all the conflicting objectives. In general, finding infinitely many solutions on a Pareto front is impossible, and finding even a large discrete subset is hard when the structure and/or the analytical form of the underlying objective functions is not known or is very complex (e.g., blackbox simulation optimization). An approximation to the true Pareto front is obtained as an alternative in such situations. The most common

approach is to approximate the Pareto front by a discrete set of solutions. The other difficulty is the efficient generation of well distributed Pareto optimal solutions. In real design problems a decision maker might be able to consider only a subset of the available solution set. In this context, having a good representation of the entire Pareto front is a crucial requirement in order to make an informed decision. Efficiency (in terms of the total number of true function evaluations) of a multiobjective optimization algorithm is a key factor in practical multidisciplinary design optimization problems where a design cycle involves time consuming and expensive computations for each discipline. For example, in a conceptual aircraft design process several different disciplines influence each other resulting in several interdisciplinary iterations to reach a feasible optimal design. The task becomes even more complicated when the objective functions are evaluated using expensive blackbox simulations where analytical form of the objective functions is not available or is very complex.

The aim of this paper is to propose a new MOP method that provides a well distributed representation of the Pareto front for multiobjective blackbox optimization with a minimal number of true function evaluations. The general idea is to systematically move towards the Pareto front at the end of every iteration by adaptively filling the gaps between the nondominated solutions obtained so far. The biobjective optimization method proposed in [9] by the authors was based on the scalarization scheme of Ryu et al. [17]. However, the same adaptive weighting scheme does not work for multiobjective problems with $p > 2$ objectives. The algorithm presented in this paper is an alternative approach for generalizing the adaptive weighting scheme of Ryu et al. [17] to problems with more than two objectives. The algorithm employs a hybrid approach using two derivative free direct search techniques — a deterministic global search algorithm DIRECT [14] for global exploration of the design space and a local direct search method MADS (mesh adaptive direct search) [1] to exploit the design space by fine tuning the potentially optimal regions returned by DIRECT and to speed up the convergence. Inexpensive surrogates for the objective functions are used in order to minimize the number of expensive simulation runs. The proposed method uses the global search algorithm DIRECT as a sampling method instead of using traditional random sampling (e.g., Latin hypercube sampling) or factorial designs are expensive in higher dimensional design spaces (see Section 4.1.4). Results show that the proposed method provides well distributed Pareto optimal points on the Pareto front for diverse types of problems. Another contribution is the use of a precise mathematical measure, known as star discrepancy, for the distribution of the Pareto optimal solutions. A biobjective version presented in [9] has been generalized in the present work for problems with more than two objectives. A detailed description is provided in Section 4.

The organization of this paper is as follows. Section 2 gives an overview of the existing multiobjective optimization approaches in general and the previous work by the authors on biobjective optimization problems in particular, and also provides background on surrogates and hybrid optimization approaches in the context of multiobjective optimization. Section 3 describes the proposed alternative scalarization scheme and presents the multiobjective optimization algorithm. Numerical evaluation on a set of test problems from the literature is presented, and results are discussed in Section 4. Section 5 offers concluding remarks.

## 2. Background

Many different methods have been suggested in the literature for solving multiobjective optimization problems. The solution approaches can be divided into two broad categories — evolutionary approaches and scalarization or aggregation approaches. Applying a Pareto based ranking scheme using genetic algorithms has become very common and popular in most of the evolutionary approaches (e.g., [7]), although some schemes are based on particle swarm intelligence and simulated annealing. An evolutionary approach yields a set of nondominated points at the end of each iteration, requires a very large number of true function evaluations, and does not guarantee optimality in general. Aggregate or scalarization approaches [10] are considered to be classical methods for solving multiobjective optimization problems. The general idea is to combine all objective functions to convert a multiobjective optimization problem into a single objective optimization problem. Thus each iteration yields a single solution by solving a single objective optimization problem. The most commonly used scalarization approach is the convex combination method where the multiobjective optimization problem is converted to a single optimization problem using a predefined weight vector. A major drawback of the convex combination method is that if the Pareto front has a nonconvex (or nonconcave) region then the method fails to produce any points in that region. Also uniformly distributed weights may not produce uniformly distributed optimal points on the front. To alleviate this problem an adaptive weighting scheme was suggested by Ryu et al. in their biobjective optimization algorithm PAWS [17]. PAWS has an efficient weighting scheme, however, the central composite design based sampling strategy used in PAWS is impractical in higher dimensional search domains. Deshpande et al. [9] proposed an algorithm for biobjective optimization that employs the adaptive weighting scheme of PAWS, but tries to overcome the limitation of PAWS with a novel sampling strategy and a global search method. The ordering property exploited in these algorithms ([17] and [9]) for the scalarization of the objectives does not work in higher dimensional objective spaces (i.e., problems with more than two objectives). Hence, an alternative strategy is proposed in this paper that works for problems with any number of objectives.

Other methods like normal boundary intersection [6] and its variations solve multiobjective optimization problems by constructing several aggregate objective functions. A scalarization method called BiMADS [2] solves a biobjective optimization problem through a series of single objective formulations using the direct search method MADS, and attempts to obtain a uniform coverage of the Pareto front, even in the case when the Pareto front is nonconvex or disjoint. However, results produced using BiMADS are dependent on the random sampling used in the algorithm (e.g., Latin hypercube sampling used for the very first run in NOMAD 3.5.1 may not produce consistent results across different computing systems), and BiMADS also exploits the ordering property available in the two–dimensional case, and hence is not applicable for problems with more than two objectives. Audet et al. suggested an alternate formulation in [2] to generalize the scalarization scheme to problems with more than two objectives.

Several real world scientific and engineering applications require analysis of spatially distributed data from expensive experiments or complex computer simulations that can take hours to run even on a fast workstation. This makes it difficult to explore and produce well distributed design combinations in high dimensional design spaces. Identifying the regions that contain good designs in such data-scarce domains by keeping the number of simulation runs to a minimum is a nontrivial problem. The proposed algorithm employs a systematic way to enrich an incomplete database by adaptively filling the gaps between the nondominated points obtained from available data. This paper deals with multiobjective optimization problems for such black box simulations where the structure of the objective functions is not known or is very complex. To tackle the problem of expensive simulation runs and to reduce the number of true function evaluations, computationally cheap(er) approximations (known as surrogates) of the underlying complex functions are used. Statistical sampling is a crucial part of the surrogate building process which becomes nontrivial in higher dimensional search domains. The novel idea proposed in [9] of using the optimization algorithm DIRECT as a sampling strategy is employed in the current work as well.

A popular approach in the optimization community, hybrid optimization, is where several different optimization techniques are combined to improve robustness and blend distinct strengths of different approaches [11]. This same notion has been extended to multiobjective optimization problems in [21]. The Pareto front approximation method of this paper is a hybrid approach using two direct search methods, DIRECT (to explore the design space globally) and MADS (to fine tune the potentially optimal regions returned by DIRECT), to balance between global and local searches and to improve the convergence rate.

## 3. The Multiobjective Optimization Algorithm

The proposed new method for multiobjective optimization applies to possibly nonsmooth functions. The ordering property (for biobjective pairs$\big(f_1(x), f_2(x)\big)$) available in two dimensions is exploited in the biobjective case to find the most isolated point from a set of nondominated points at each iteration. However, due to the lack of ordering in higher dimensions, the same strategy cannot be implemented in the multiobjective case. Hence, an alternative using the concept of triangulation, from simplical topology and computational geometry, is proposed in this paper. Section 3.1 describes this alternate approach, and the general scheme of the multiobjective optimization algorithm is presented in Section 3.2.

### 3.1. An alternative using Delaunay triangulation

In the biobjective optimization case, a point with the largest Euclidean distance from its neighbors is selected as the most isolated point provided that the nondominated data points are ordered in either of the two objectives. However, this notion of ordering does not work in higher dimensions. Hence, an alternative is proposed for identifying the most isolated point in a high dimensional ($p > 2$) objective space, using the concept of triangulation from simplical topology and computational geometry. Triangulation of a discrete set of points is a subdivision of the convex hull of the points into simplices (e.g., a set of triangles in two dimensions or tetrahedra in three dimensions). A frequently used and studied point set triangulation (in two dimensions) is the Delaunay triangulation, which has the property that no point in the point set falls in the interior of the circumcircle of any triangle in the triangulation. Once the triangulation is constructed for the current nondominated point set, the average distance of each vertex from its neighbors (given two vertices $v^{(1)}$ and $v^{(2)}$, $v^{(1)}$ is a neighbor of $v^{(2)}$ [and $v^{(2)}$ a neighbor of $v^{(1)}$] if there is an edge between $v^{(1)}$ and $v^{(2)}$ in the given triangulation) is computed, and the vertex with the largest average distance from its neighbors is then considered as the most isolated point for that iteration. This triangulation concept generalizes to $p$–simplices in $p$ dimensions (a triangle is a 2–simplex, a tetrahedron is a 3–simplex, etc.).The precise mathematical formulation of this process is presented in the next subsection.

### 3.2. The optimization algorithm

As a preprocessing step, a global search is performed using DIRECT to explore the design space in unsampled regions of a database (if the database is empty, the global exploration step becomes the data generation step).Since DIRECT is an optimization algorithm, the design space exploration is conducted in an intelligent manner by focusing the global search in potentially optimal regions

only. For this initialization step, $p+1$ single objective formulations using $p+1$ weight vectors $w$ are obtained — $p$ of which correspond to the individual optima of the $p$ objective functions and one more with equal preference given to all the objectives ($w_i = 1/p$, $i = 1, \ldots, p$, $\sum_{i=1}^{p} w_i = 1$). A peculiarity of DIRECT is that it never samples the boundary points of a design space and takes a large number of iterations to reach reasonably close to the boundary. On account of this behavior of DIRECT, the potentially optimal regions returned by DIRECT are fine tuned using MADS. For each of these $p+1$ single objective formulations using the data collected in DIRECT's global search an interpolating surrogate is built over the entire design space and optimized using MADS. These $p+1$ candidate solutions are evaluated and stored in the database.

The preprocessing step is a crucial part of the proposed algorithm. It helps in eliminating a subset of local Pareto optimal points and accelerates the process of moving to the Pareto front. However, it may not eliminate all the local Pareto optimal points (as the preprocessing is done using a fixed set of weights), and hence the algorithm may not always find the global Pareto front.

A set $X^0$ of nondominated points (with respect to all database points) is obtained at the end of the preprocessing step that serves as input for the iterative procedure of the algorithm that follows. Each iteration of the algorithm consists of three steps:(1) finding the most isolated point from the current nondominated point set, (2) constructing surrogates for the objective functions with the isolated point determined in Step 1 as the center, and (3) solving several single objective optimization problems to produce candidate Pareto solutions. At the beginning of the iterative process, a trust region radius $\Delta_0$, the trust region contraction parameter $\rho$, and the tolerance value $\tau$ for the trust region radius are initialized.

**Step 1: Isolated point determination**

Let $X^{(k)} = \left\{x^{(k,1)},\ldots,x^{(k,J_k)}\right\}$ be the set of nondominated points at the start of the iteration $k$ and $P^{(k)} = \left\{F\left(x^{(k,1)}\right), F\left(x^{(k,2)}\right), \ldots, F\left(x^{(k,J_k)}\right)\right\}$ be the corresponding objective space set, where $F(x) = \left(f_1(x), \ldots, f_p(x)\right)$ for $p$ objectives and $J_k$ is the cardinality of $X^{(k)}$ (and $P^{(k)}$). Note that for the very first iteration ($k = 0$) the nondominated set $X^{(0)}$ is the result of the preprocessing step. For a typical $p$-objective optimization problem the Pareto front is a $p-1$-dimensional manifold. Hence to construct a $p-1$-dimensional triangulation from a set of $p$-dimensional points, each point $(f_1, f_2, \ldots, f_p)$ is transformed to homogeneous coordinates $(f_1/f_p, f_2/f_p, \ldots, f_{p-1}/f_p, 1)$ assuming that the objectives are appropriately scaled. Essentially the triangulation is done in the $(p-1)$-dimensional real projective space $\mathbb{P}^{p-1}$. A Delaunay triangulation $DT$ is constructed using this

$(p-1)$-dimensional transformed data set. The gap $\delta_j^k$ between the nondominated points in the objective space set $P^{(k)}$ is computed for each point $F(x^{(k,j)})$, $j = 1, \ldots, J_k$, in $P^{(k)}$ by,

$$\delta_j^k = \frac{\sum_{i \in N_j} \|F\big(x^{(k,j)}\big) - F\big(x^{(k,i)}\big)\|_2}{N_j},$$

where $N_j = \big\{i \mid F\big(x^{(k,i)}\big)$ is a neighbor of $F\big(x^{(k,j)}\big)$ in DT$\big\}$.

The point $x^{(k,j)}$ with the largest $\delta_j^k$ value is selected as the most isolated point, and is used as the center point $\tilde{x}^{(k)}$ for a local model with a trust region radius $\Delta_k$.

In case of ties, choose the smallest index $j$.

If $N_j = \phi$ $(J_k = 1)$, then $\tilde{x}^{(k)}$ is a single point in $X^{(k)}$.

All the most isolated nondominated points points $\tilde{x}^{(k)}$ found, together with an associated trust region radius $\Delta_i$, are stored in a database. If $\tilde{x}^{(k)}$ happens to be a previously visited point (for $k > 0$, i.e., if for some $i < k$ $\tilde{x}^{(k)} = \tilde{x}^{(i)}$, the trust region radius is modified as $\Delta_k := \rho\Delta_i$, and if the new $\Delta_k$ value is greater than $\tau$, $\tilde{x}^{(k)}$ is the center point for the local model for the next iteration; otherwise $x_c^{(k)}$ is accepted as a Pareto optimal point (and excluded from further consideration as an isolated point) and the above procedure is repeated until a new isolated point is found. If $\tilde{x}^{(k)}$ is not a previously visited point, then $\tilde{x}^{(k)}$ and the trust region radius $\Delta_k := \Delta_0$ for the iteration $k$ are added to the database. If the point $\tilde{x}^{(k)}$ is some previously visited point then the algorithm failed to produce any better points (i.e., a point that dominates the center point) at the $i$th iteration. This behavior could be seen in either of two situations: the surrogate for the local trust region for the $i$th iteration was not accurate enough or $\tilde{x}^{(k)}$ is a locally Pareto optimal point. Reducing the trust region radius (and generating a new experimental design) would help in constructing a more accurate surrogate to check if any better points around $\tilde{x}^{(k)}$ exist.

In classic trust region algorithms a trust region is expanded when the surrogate is in reasonably good agreement with the true objective function. In the proposed algorithm the center point, and hence the trust region, at each iteration changes as a result of the process of identifying the most isolated point, and weights are redetermined at each iteration that changes the resulting objective function. As a result, the objective function changes at each iteration, and hence is not plausible that a good surrogate for one iteration is likely good for the next iteration as well. Hence, expansion of the trust region is omitted in the proposed algorithm.

**Step 2: Surrogates**

The algorithm uses a linear Shepard method (LSHEP from [18]) to approximate a response surface within a trust region. The isolated point $\tilde{x}^{(k)}$ determined in the previous step serves as the center

point for the local trust region with radius $\Delta_k$ taken as the intersection of the local trust region box $\left\{x \mid \|x - \tilde{x}^{(k)}\|_\infty \leq \Delta_k\right\}$ with the box $[l, u]$ defined by the variable bounds $l \leq x \leq u$. A tuning parameter $\epsilon$ for DIRECT controls the exploration mode for DIRECT (local versus global). The number of points to be sampled can be specified as an input to DIRECT, as for a Latin hypercube, the difference being deterministic adaptive rather than random point placement. The points to be sampled using DIRECT are based on the value of the aggregate objective constructed as a convex combination of the $p$ objective functions. At the end of a single run of DIRECT, a data set $D^{(k)} = \left\{\left(x, f_1(x), f_2(x), \ldots, f_p(x)\right) \mid x \in X_d^{(k)}\right\}$ of design samples $x \in X_d^{(k)}$ and function values $f_1(x)$, $f_2(x)$ ..., $f_p(x)$ is generated. $p$ LSHEP [18] surrogates $\tilde{f}_1$, $\tilde{f}_2$, ..., $\tilde{f}_p$ are built for the $p$ objective functions using the database $D^{(k)}$. At every iteration, several single objective optimization problems are formulated using convex combinations of these surrogates. $p$ of the weight combinations used have fixed values (e.g., (1, 0, ... ,0)) for each iteration to compute the individual optima of the $p$ surrogates. Additional weights are derived using an adaptive weighting scheme. Based on the value of the objective functions at the center point $\tilde{x}^{(k)}$ for an iteration and its neighbors, max{1, $N_k$} additional weight vector $w$ indexed by $N_k = \left\{i \mid F\left(x^{(k,i)}\right) \text{ is a neighbor of } F\left(\tilde{x}^{(k)}\right)\right\}$

If $J_k = 1$, $w = (, w_1, w_2, \ldots, w_p) = (1/p, 1/p, \ldots, 1/p)$. Now consider the case $J_k >= 2$ ($N_k \neq \phi$). Since $\tilde{x}^{(k)}$ and any neighbor $x^{(k,i)}$ are both nondominated (with respect to the set $X^{(k)}$, $\|F\left(\tilde{x}^{(k)}\right) - F\left(x^{(k,i)}\right)\| \neq 0$. For each $i \in N_k$, define a weight vector $w^{(i)}$ by

$$w_j^{(i)} = \begin{cases} 0, & \text{if } f_j\left(\tilde{x}^{(k)}\right) \text{-} f_j\left(x^{(k,i)}\right) = 0; \\ \frac{c_i}{\left|f_j\left(\tilde{x}^{(k)}\right) - f_j\left(x^{(k,i)}\right)\right|}, & \text{otherwise.} \end{cases}$$

where $c_i > 0$ is a normalization constant such that $\sum_{j=1}^{p} w_j^{(i)} = 1$.

Thus, there are $p + \max\{1, \mid N_k \mid\}$ weight vectors $w$ and corresponding surrogates constructed at each iteration.

**Step 3: Solving optimization problems**

Each of the surrogates constructed in Step 2 is optimized using MADS to get a candidate solution. Thus at the end of iteration $k$, $p + \max\{1, \mid N_k \mid\}$ candidate Pareto solutions are obtained, evaluated to get their true function values, and then nondominated points among those are selected. Let $X_*^{(k)}$ be the set of these nondominated points. An updated set $X^{(k+1)}$ of nondominated points is obtained at the end of iteration $k$ by comparing all points from the union $X^{(k)} \cup X_d^{(k)} \cup X_*^{(k)}$.

These three steps are iterated by the algorithm until a stopping condition is satisfied. In practice, termination is either determined by a fixed number of iterations, or when the value of the star discrepancy (discussed in detail in the next section) drops below a predefined threshold.

## 4. Numerical Evaluation

### 4.1. Performance Measures

### 4.2. Parameter Setup

### 4.3. Test Problems

### 4.4. Results and Discussion

## 5. Conclusion and Future Work

## References


[1] Audet C. and Dennis J. E., "Mesh adaptive direct search algorithms for constrained optimization," *SIAM Journal on Optimization*, Vol 17, No. 1, pp. 188-217, 2006

[2] Audet C., Savard G., and Zghal W., "A mesh adaptive direct search algorithm for multiobjective optimization," *European Journal of Operational Research*, Vol. 204, pp. 545-556, 2010

[3] Audet C., Savard G., and Zghal W., "Multiobjective optimization through a series of single objective formulations," *SIAM Journal of Optimization*, Vol. 19, pp. 188-210, 2008

[4] Caflisch R. E., "Monte Carlo and quasi-Monte Carlo methods," *Acta Numerica*, vol. 7, pp. 1–49, 1998

[5] Deb K., "Multiobjective genetic algorithms: problem difficulties and construction of test problems," *Evol. Comput.*, Vol. 7. pp. 205—230, 1999

[6] Das I. and Dennis J.E., "Normal-Boundary Intersection: A New Method for Generating the Pareto Surface in Nonlinear Multicriteria Optimization Problems ," *SIAM Journal on Optimization*, Vol. 8. pp. 631—657, 1998

[7] Deb, K., Pratap. A, Agarwal, S., and Meyarivan, T., "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transaction on Evolutionary Computation*, 6(2), pp. 181-197, 2002

[8] Digabel S. L. and Tribes C., *NOMAD user guide version 3.5.1*, Mar, 2012

[9] Deshpande S. G., Watson L. T., and Canfield R. A., "Biobjective Optimization using Direct Search Techniques ," in *IEEE SoutheastCon*, Jacksonville, FL, Apr 4-7, 2013

[10] Eichfelder G., *Adaptive Scalarization Methods in Multiobjective Optimization (Vector Optimization)*, Springer, 2008

[11] Gray G. A. and Fowler K. R. (2011), "Traditional and hybrid derivative-free optimization approaches for black box functions," *Computational Optimization, Methods and Algorithms*, 356

125-151

[12] He J., Watson L. T., Ramakrishnan N., Shaffer C. A., Verstak A., Jiang J., Bae K., and Tranter W. H., "Dynamic data structures for a direct search algorithm," *Computational Optimization and Applications*, Vol. 23, pp. 5–25, 2002

[13] Jones D. R., Perttunen C. D., and Stuckman B. E., "Lipschitzian optimization without the Lipschitz constant," *Journal of Optimization Theory and Application*, Vol. 79, No. 1, pp. 157-181, 1993

[14] Kugele S. C., Watson L. T., and Trosset M. W., "Multidimensional numerical integration for robust design optimization," in *ACM Southeast Regional Conference*, pp. 385-390, 2007

[15] Ryu J., Kim S., and Wan H., "Pareto front approximation with adaptive weighted sum method in multiobjective simulation optimization," in *Proceedings of the 2009 Winter Simulation Conference*, pp. 623-633, Austin, TX, USA, 2009

[16] Thacker W. I., Zhang J, Watson L. T., Birch J. B., Iyer M. A., Barry M. W., "Algorithm 905: SHEPPACK: modified Shepard algorithm for interpolation of scattered multivariate data," *ACM Trans. Math. Software*, vol. 37, pp. 1-20, 2010

[17] Veldhuizen D. A. and Lamont G. B, *Evolutionary Computation and Convergence to a Pareto Front*, Stanford University Bookstore, 1998

[18] Wang L., Ishida H., Hiroyashu T., and Miki M., "Examination of multiobjective optimization method for global search using DIRECT and GA," in *IEEE World Congress on Computational Intelligence*, pp 2446–2451, 2008