Handbook of Optimization Theory
Editors: Juan Varela and Sergio Acuna, pp. 1-25
ISBN 978-1-60876-500-3

Chapter 9

MULTIOBJECTIVE OPTIMIZATION: QUASI-EVEN GENERATION OF PARETO FRONTIER AND ITS LOCAL APPROXIMATION

Sergei V. Utyuzhnikov
School of Mechanical Aerospace and Civil Engineering,
University of Manchester, P.O. Box 88, Manchester, M60 1QD, UK

Abstract

In multidisciplinary optimization the designer needs to find a solution to an optimization problem that includes a number of usually contradicting criteria. Such a problem is mathematically related to the field of nonlinear vector optimization under constraints. It is well-known that the solution to this problem is far from unique and given by a Pareto surface. In the real-life design the decision-maker is able to analyze only several Pareto optimal (trade-off) solutions. Therefore, a well-distributed representation of the entire Pareto frontier is especially important. In the literature, there are only a few methods that are capable of even generating a Pareto frontier in a general formulation. They are compared to each other, while the main focus is devoted to a general strategy combining the advantages of the known algorithms. The approach is based on shrinking a search domain to generate a Pareto optimal solution in a selected area on the Pareto frontier. The search domain can be easily conducted in the general multidimensional formulation. The efficiency of the method is demonstrated on different test cases. For the problem in question, it is also important to carry out a local analysis. This provides an opportunity for a sensitivity analysis and local optimization. In general, the local approximation of a Pareto frontier is able to complement a quasi-even generated Pareto set.

1 Introduction

In the real-life design, the decision-maker (DM) has to take into account many different criteria such as low cost, manufacturability, long life and good performance which cannot be satisfied simultaneously. In fact, it is possible only to consider a trade-off among all (or almost all) criteria. The task becomes even more complicated because of additional constraints which always exist in practice.

Mathematically, the trade-off analysis can be formulated as a vector nonlinear optimization problem under constraints. Generally speaking, the solution of such a problem is not unique. It is natural to exclude from the consideration any design solution that can be improved without deterioration of any discipline and violation of the constraints; in other words, a solution that can be improved without any trade-off. This leads to the notion of a Pareto optimal solution [1]. Mathematically, each Pareto point is a solution of the multiobjective optimization (MOO) problem. A designer selects the ultimate solution among the Pareto set on the basis of additional requirements, which may be subjective. Generally speaking, it is desirable to have a sufficient number of Pareto points to represent the entire Pareto frontier in the objective space. Yet it is important the Pareto set to be evenly distributed, otherwise the generation of the Pareto set becomes inefficient.

Despite the existence of many numerical methods for vector nonlinear optimization, there are few methods potentially suitable for real-design applications. This is because the problem in question is usually very time-consuming. In many practical multidisciplinary optimization applications the design cycle includes time-consuming and expensive computations at each discipline. For example, in the aerospace industry this is most evident in the solution of the aerodynamics and stress analysis tasks. The solutions corresponding to these subtasks influence each other and usually demand iterations to reach the ultimate solution. Under such conditions, it is important to minimize the number of iterations required to find a Pareto optimal solution.

In general, the optimization methods can be split into two principle categories: classical, preference-based, methods and evolutionary genetic-based algorithms (GA). The classical methods usually use deterministic approaches, whereas GA usually use stochastic algorithms. It goes without saying that such a division is not strict and the combination of classical and GA methods is also possible. In classical MOO methods, vector optimization approaches are often reduced to the minimization of an aggregate objective function (AOF) (preference function), which includes a combination of objective (cost) functions. The simplest and most distributed example of the AOF is represented by a linear (weight) combination of the objective functions [1]. This method has many drawbacks related mainly to an uncertainty in weights coefficients. It may require many iterations to find the combination of the weights leading to a solution, which corresponds to the DM's expectations [2]. Furthermore, it is well known that this method can generate only the convex part of a Pareto surface [3] while real-life problems often result in non-convex Pareto frontiers. This drawback can be avoided by using either a more complex consideration of the AOF [4] or the weighted Tchebychev method [1]. However, the weights remain to be unknown functions of the objectives [2].

In real design, the DM is able to consider only a few possible solutions (Pareto points). In such a context, it is important to have a well-spread distribution of Pareto points to obtain maximum information on the Pareto surface at minimum (computational) cost. Das and Dennis [5] showed that an even spread of weights in the AOF does not necessary result in an even distribution of points in the Pareto set. In the literature, there are only a few methods that can be considered for even generating the entire Pareto frontier in the general multidimensional formulation [6].

1.1 Quasi-even Set Generators of Pareto Frontier

The Normally Boundary Intersection (NBI) Method [7], [8] is developed for generating a quasi-even distribution of Pareto solutions by Das and Dennis. The method has a clear geometrical interpretation. It is based on the well-known fact that a Pareto surface belongs to the boundary of feasible domain towards minimization of objective functions [1]. First, so-called anchor points are obtained in the objective space. An anchor point corresponds to the optimal value of one and only one objective function in the feasible domain. Thus, n objective functions provide up to n anchor points. Second, the utopia plane passing through the anchor points is obtained. The Pareto surface is then constructed by the intersection of lines normal to the utopia plane and the boundary of feasible domain. A single optimization problem with respect to one of the objective functions is solved along each line. An even distribution of Pareto points is provided by even distribution of lines orthogonal to the utopia plane. One can note that this is valid until cosine of the angle between such a line and the normal to the boundary of feasible space is not locally close to zero. The method appeared to generate non-Pareto and locally Pareto solutions that requires a filtering procedure [9], [10]. In addition, the NBI method might be non-robust because the feasible domain is reduced to a line.

The Normal Constraint (NC) Method [9], [12] by Messac *et al.* represents the modification of the NBI approach. The single optimization problem, used in the NC, is based only on inequality constraints. This modification makes the method more flexible and stable. Both methods may fail to generate Pareto solutions over the entire Pareto frontier [12] in a multidimensional case. The modification of the NC [12] partially eliminates this drawback. However, both methods may generate non-Pareto and locally Pareto solutions [9] although the NC does it less likely [12]. In [10], [6] it is shown that both the NBI and NC methods can be inefficient because of the significant number of redundant solutions. In the example given in [6], sixty six points on the utopia plane lead only to twenty four Pareto solutions. Another examples can be found in [10]. Both the methods can meet significant difficulties in the case of a disconnected frontier [10]. In particular, they do not always find the entire Pareto frontier. Meanwhile, one can note that a recent modification of the NC method [14] is able to improve these methods.

The Physical Programming (PP) Method is suggested in [13] by Messac. This method also generates Pareto points on both convex and non-convex Pareto frontiers as shown in [15]. The method does not use any weight coefficients and allows one to take into account the DM experience immediately. In the PP, the designer assigns each objective to one of the four categories (class-functions). The optimization is based on minimization of an AOF determined by the preference functions (class-functions). The algorithm given in [15] is able to generate an evenly distributed Pareto set. However, it contains a few free parameters. The optimal choice of those requires a preliminary information on the location of the Pareto frontier.

In [16] and [6], Utyuzhnikov *et al.* modify the PP to make it simpler and more efficient for practical applications. A simpler structure of the class-functions is suggested. The class-functions are generalized to shrink the search domain and make its location in space more optimal. This is critical for generating an even set of the Pareto frontier. The proposed modification combine the advantages of the PP, NBI and NC methods. One of the main

advantages of the approach [6] is that it does not provide non-Pareto solutions while local Pareto solutions may be easily recognized and removed. In [6] it is shown that the modified PP is able to generate a quasi-even Pareto set in the general formulation. It is proven that the method is able to capture an entire Pareto frontier in the general case. As will be shown in this Chapter, the algorithm suggested in [6] can be applied beyond the PP technique.

In contrast to the classical, preference-based, approaches, the class of evolutionary methods, such as genetic-based algorithms (GA), generate a set of Pareto solutions simultaneously (see, e.g., [17], [18]). This class of methods seems to be very promising for solving multiobjective problems. Unfortunately GA do not usually guarantee either the generation of a well-distributed Pareto set or the representation of the entire Pareto frontier. In [11], GA is combined with the generalized data envelopment analysis to remove dominated design alternatives. The method is capable to efficiently generating both convex and concave frontiers. All examples in [11] are obtained only for two objective functions. A modification of the evolutionary algorithm is suggested in [10] to generate a well-distributed Pareto set. The general problem of GA is that their efficiency significantly drops if the number of objective functions increases especially at the presence of constraints.

A global information on the Pareto frontier, gained from a well distributed Pareto set, can be complimented via a local analysis. If a local approximation of the Pareto frontier is available, then it can be used for a sensitivity (trade-off) analysis. In [22], Utyuzhnikov *et al.* derived the precise formula for the linear and quadratic approximations in the multidimensional formulation. They showed that the formulae [23], widely used in the literature for the linear approximation, should be corrected in the general formulation.

The Chapter is organized as follows. In Section 2 the general mathematical formulation of multiobjective optimization under constraints is given. Then, in Section 3 an algorithm for even-generating a Pareto set is described. The algorithm is given in the general formulation for an arbitrary number of design variables, objective functions and constraints. Some examples of the application of the algorithm are shown in Section 4. Local approximations of a smooth Pareto frontier in the objective space are derived in Section 5. Finally, an example of the local approximation is given in Section 6.

2 Multiobjective Optimization Problem. Pareto Optimal Solution

Assume that there are N design variables to be found. Then, we can introduce a design space $\mathbf{X} \subset \mathbb{R}^N$. Each element in the design space is represented by a design vector $\mathbf{x} = (x_1, x_2, \ldots, x_N)^T : \mathbf{x} \in \mathbf{X}$. Suppose that the quality of each combination of design variables is characterized by M objective (cost) functions. As such, along with the design space $\mathbf{X}$, we introduce the space of objective functions $\mathbf{Y} \subset \mathbb{R}^M$. Each element in the objective space $\mathbf{Y}$ is represented by a vector $\mathbf{y} = (y_1, y_2, \ldots, y_M)^T$, where $y_i = f_i(\mathbf{x})$, $f_i : \mathbb{R}^N \to \mathbb{R}^1$, $i = 1, 2, \ldots, M$. Thus, $\mathbf{X}$ is mapped onto $\mathbf{Y}$ by $\mathbf{f} \in \mathbb{R}^M : \mathbf{X} \mapsto \mathbf{Y}$.

Suppose that there are constraints, which are formulated via either equations or inequalities. Then, we arrive at the following multiobjective optimization problem under constraints:

$$\min[\mathbf{y}(\mathbf{x})] \tag{1}$$

subject to K inequality constraints

$$g_i(\mathbf{x}) \leq 0, \;\; i = 1,2,\ldots,K \tag{2}$$

and P equality constraints

$$h_j(\mathbf{x}) = 0, \;\; j = 1,2,\ldots,P. \tag{3}$$

The feasible design space $\mathbf{X}^*$ is defined as the set of design variables satisfying all the constraints (2) and (3). The feasible criterion (objective) space $\mathbf{Y}^*$ is defined as the set: $\{\mathbf{Y}(\mathbf{x})|\ \mathbf{x} \in \mathbf{X}^*\}$. Thus, the feasibility means no constraint is violated.

Definition. a design vector $\mathbf{a}$ $(\mathbf{a} \in \mathbf{X}^*)$ is called a Pareto optimum iff it does not exist any $\mathbf{b} \in \mathbf{X}^*$ such that

$$\mathbf{y}(\mathbf{b}) \leq \mathbf{y}(\mathbf{a})$$

and exist $l \leq M : y_l(\mathbf{b}) < y_l(\mathbf{a})$.

A design vector is called a local Pareto optimum if it is a Pareto optimum in its some vicinity.

In other words, a design vector is Pareto optimal if it cannot be improved with regard to any objective function without deterioration of at least one of the others. Due to constraints, we are not able to minimize all objectives simultaneously. Thus, the solution of the MOO problem (1), (2), (3) is not unique and any solution represents a trade-off between different objectives. Consider any element $\mathbf{x} \in \mathbf{X}$ such that vector $\mathbf{y}(\mathbf{x})$ belongs to the interior of the feasible space $\mathbf{Y}^*$ rather than its boundary. Obviously, such an element cannot be a Pareto solution. More precisely, the general solution of an MOO problem is represented by a Pareto surface, which always belongs to the boundary of the feasible space $\mathbf{Y}^*$. In the real-life design, as a rule, it is impossible to obtain the entire Pareto frontier. In fact, the entire Pareto frontier is not usually required.

In the next Section, we consider an algorithm that provides a well-distributed representation of the entire Pareto surface. We describe a strategy to seek the Pareto frontier based on a Directed Search Domain (DSD) algorithm, which was first applied for the modification of the PP method in [16] and [6]. One can see that the DSD approach can be used for very different search engines. The main idea of DSD is that we shrink the search domain to obtain a Pareto solution in a selected area of $\mathbf{Y}^*$. A well-spread distribution of the selected search domains should give us a quasi-even Pareto set.

3 Generation of a Well-Distributed Pareto Set. DSD Algorithm

3.1 Trade-off matrix. Utopia plane

For further consideration, we introduce the trade-off matrix T:

$$T = \begin{pmatrix} f_{1,\min} & f_{12} & \cdots & f_{1M} \\ f_{21} & f_{2,\min} & \cdots & f_{2M} \\ \cdots & \cdots & \cdots & \cdots \\ f_{M1} & f_{M2} & \cdots & f_{M,\min} \end{pmatrix}. \tag{4}$$

In the trade-off matrix T, an i-th row represents the coordinates of an anchor point $\boldsymbol{\mu}_i^*$ corresponding to the solution of single-optimization problem $\min f_i$ in the feasible criterion space $\mathbf{Y}^*$ (see, e.g.,[12]):

$$\boldsymbol{\mu}_i^* = f(\mathbf{X}^{i^*}), \quad \left\{\mathbf{X}^{i^*} : \mathbf{X}^{i^*} = \arg\min_{\mathbf{Y}^*} f_i\right\}. \tag{5}$$

In the feasible space $\mathbf{Y}^*$ we consider a hypercube H limiting the search domain. For that we define pseudo-nadir points [12]: $f_{i,\max} = \max_j f_{ij}$, where f_{ij} are the elements of the trade-off matrix T. Then, the hypercube H is represented by $H = [f_{1,\min} f_{1,max}] \times [f_{2,\min} f_{2,max}] \times \cdots \times [f_{M,\min} f_{M,max}]$. One can show that the hypercube H always contains the Pareto frontier [12].

Next, similar to the NC method, we introduce the utopia plane created by anchor points $\boldsymbol{\mu}_i^*$. One can show that the polygon spanned by all M vertexes $\boldsymbol{\mu}_i^*$ is convex. Then, any point $\mathbf{f}^*$, belonging to the interior of this polygon, is represented by:

$$\mathbf{f}^* = \sum_{i=1}^{M} \alpha_i \boldsymbol{\mu}_i^*. \tag{6}$$

Here, the parameters α_i satisfy the following conditions:

$$0 \le \alpha_i \le 1 \quad (i = 1, \ldots, M), \qquad \sum_{j=1}^{M} \alpha_j = 1. \tag{7}$$

As shown in [6], the definition of an anchor point can strongly affect the efficiency of the algorithm for Pareto set generation. The standard definition does not always lead to a uniquely determined point. To resolve this problem, in the next Section a modified lexicographic-based definition of the anchor point is given following [16] and [6]. It guarantees the uniqueness of the anchor point for each objective.

3.2 Modified anchor points

The standard definition of an anchor point (5) may lead to a non-uniqueness. If the solution of the problem (5) is not unique, then the point corresponding to the minimal value of the other objective functions is to be chosen. It may lead to the problem of trade-off minimization for the remaining objectives. To avoid this, priority in the minimization is introduced as follows. First, instead of space $\mathbf{Y}^*$, we consider domain $\mathbf{Y}^{**}$ that includes all ultimate points of $\mathbf{Y}^*$. Then, we minimize f_i, then f_{i+1} and so on up to f_{i-1}. Thus, we use the following lexicographic prioritization in a circular order: $i+1, i+2, \ldots, M, 1, 2, \ldots, i-1$. A $k-th$ prioritization assumes that the $k-th$ minimization must not violate all the previous $k-1$ ones. One can prove that all anchor points belong to the Pareto frontier. Indeed, the anchor points belong to the boundary of the feasible domain $\mathbf{Y}^*$ and no objectives can be improved without deterioration of any other objective. As soon as we know the coordinates of the anchor points we can determine the polygon (6), (7) on the utopia plane.

3.3 Reference points on the utopia plane

Next, in the objective space consider a search domain D: $f_i \le f_i^*$ $(i = 0,\ldots,M)$, where the vector $\mathbf{f}^*$ is determined by (6) and (7). The lower boundary in D can be quite arbitrary in each direction. However, the values of the lower boundaries must be small enough for the search domain to contain a part of the Pareto surface if possible. It is natural to require that $f_{i,\min} \le f_i \le f_i^*$ $(i = 0,\ldots,M)$, where $f_{i,\min}$ are determined by the trade-off matrix T in (4). The end of the vector $\mathbf{f}^*$ determines a reference point M: $M = (f_1^*, f_2^*, \ldots, f_M^*)^T$, which belongs to the interior of the utopia polygon (6) and (7) including its boundaries. Obviously, there is no guarantee that a local search domain D has any intersection with $\mathbf{Y}^*$. Then, and only then, we switch the search on the opposite direction: $f_i^* \le f_i$ $(i = 0,\ldots,M)$. In turn, it is natural to require the search domain is limited by $f_i^* \le f_i \le f_{i,\max}$ $(i = 0,\ldots,M)$, where $f_{i,\max}$ are determined by the trade-off matrix T.

A well-spread distribution of the search domains can be reached via an even distribution of the coefficients α_i in (7). An algorithm for calculating the coefficients α_i is given in [12]. The approach is based on an induction procedure. First, a uniform distribution of the coefficient α_1 is considered. From conditions (7) it is clear that the sum of the rest coefficients $\sum_1^M \alpha_j$ equals to $1-\alpha_1$ for each selected value of α_1. Then, we consider a uniform distribution of the coefficient α_2 for each of these variants. The algorithm is repeated until either the last coefficient α_M is reached or the sum of the coefficients already determined is equal to 1. In the latter case we set the remaining coefficients to be zero.

In contrast to the NC and NBI algorithms, the DSD approach allows us to represent the entire Pareto frontier considering only non-negative coefficients α_i, which satisfy conditions (7). As shown in [6], it is important to avoid redundant solutions.

A distribution of reference points leads to different search domains. As a result, one can expect generating different Pareto solutions. However, as shown in [6], it is not always the case. Although the search domain is limited for each case, it is large enough and the distribution of the Pareto set may be sensitive to the displacement of the box D along the utopia plane especially if the Pareto frontier is concave. A more efficient and flexible algorithm based on the introduction of new objective functions is described in the next Section. It is based on an affine transform of the coordinate system in the objective space. In the algorithm, to substantially shrink the search domain, an affine transform of the coordinate system is introduced in the objective space

3.4 Shrinking of search domain

Following to [6], let us introduce new objective functions $\widetilde{f}_i$ via an affine transform

$$\widetilde{f}_i = \sum_{j=1}^{M} f_j B_{ji} \;\; (i = 1,\ldots,M). \tag{8}$$

In the objective space $\mathbf{Y}$, it is equivalent to the introduction of a new coordinate system with the basis vectors

$$\mathbf{a}_i = \sum_{j=1}^{M} A_{ij}\mathbf{e}_j \;\; (i = 1,\ldots,M), \tag{9}$$

$$A^{-1} = B,$$

where $\mathbf{e}_j$ $(j = 1,\ldots,M)$ are the basis vectors of the original coordinate system.

For the new objective functions we can shrink the search domain to the following:

$$\widetilde{f}_i \le \widetilde{f}_i^* \ \ (i = 1,\ldots,M)), \tag{10}$$

where $\widetilde{f}_i^*$ is determined by the transform (8):

$$\widetilde{f}_i^* = \sum_{j=1}^{M} f_j^* B_{ji} \ \ (i = 1,\ldots,M). \tag{11}$$

Then, the search domain can be changed as shown in Figure 1. In particular, we can choose the basis vectors $\mathbf{a}_i$ $(i = 1,\ldots,M)$ to form an angle γ_c to a selected direction $\mathbf{l}$. In 2D case the matrices A and B can be easily determined:

$$A = \begin{pmatrix} \cos\gamma_- & \sin\gamma_- \\ \cos\gamma_+ & \sin\gamma_+ \end{pmatrix}, \quad B = \frac{1}{2\gamma_c} \begin{pmatrix} \sin\gamma_+ & -\sin\gamma_- \\ -\cos\gamma_+ & \sin\gamma_- \end{pmatrix}, \tag{12}$$

where $\gamma_+ = \gamma_n + \gamma_c$, $\gamma_- = \gamma_n - \gamma_c$, $\mathbf{l} = (\cos\gamma_n, \sin\gamma_n)^T$.

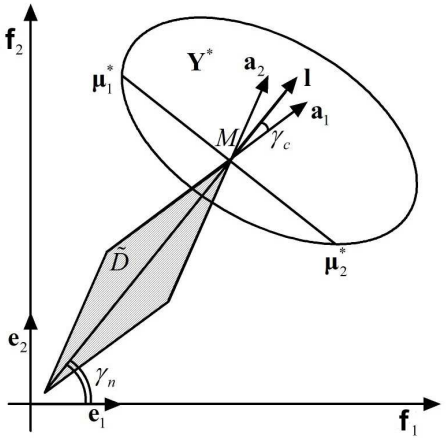

Figure 1: Search domain [6]

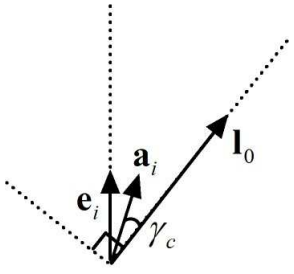

Figure 2: Basis vectors [6]

To extend this approach to a multidimensional space $\mathbb{R}^M$, we set the following conditions on the basis vectors $\mathbf{a}_i$:

$$(\mathbf{a}_i, \mathbf{l}) = \cos\gamma_c \ \ (i = 1, \ldots, M).$$

Here, $(\cdot,\cdot)$ corresponds to the inner product.

Thus, all the vectors $\mathbf{a}_i$ are parallel to the lateral area of the hypercone that has the angle γ_c and the axis directed along vector $\mathbf{l}$. It is important to guarantee a spread distribution of these vectors. Then, it is clear that the basis created by vectors $\mathbf{a}_i$ must not vanish. The following algorithm guarantees a fully uniform distribution of the basis vectors $\mathbf{a}_i$ $(i = 1, \ldots, M)$ around an axis $\mathbf{l}$ in $\mathbb{R}^M$.

First, we introduce vector $\mathbf{l}$:

$$\begin{aligned} \mathbf{l} &= \mathbf{l}_0, \\ \mathbf{l}_0 &= (l_0, l_0, \ldots, l_0)^T. \end{aligned} \tag{13}$$

If the vector $\mathbf{l}$ is unit, then it has coordinates:

$$l_0 \equiv \cos\gamma_0 = \frac{1}{\sqrt{M}}.$$

The basis vector $\mathbf{a}_i$ can be determined in the plane created by the vectors $\mathbf{e}_i$ and $\mathbf{l}_0$ (see Figure 1). One can show that

$$\mathbf{a}_i = \frac{\sin\gamma_c}{\sin\gamma_0}\,\mathbf{e}_i + \frac{\sin(\gamma_0 - \gamma_c)}{\sin\gamma_0}\,\mathbf{l}_0 \quad (i = 1, \ldots, M). \tag{14}$$

Obviously, the basis of the vectors $\mathbf{a}_i$ $(i = 1, \ldots, M)$ does not vanish. The basis vectors form a search cone similar to the 2D cone shown in Figure 1. It is clear that if $M = 2$ and $\gamma_0 = \pi/4$, then we obtain formula (12).

From (9), (13) and (14) we have

$$A = A_0 \equiv \frac{\sin\gamma_c}{\sin\gamma_0}\,I + \frac{\sin(\gamma_0 - \gamma_c)}{\sin\gamma_0}\,E, \tag{15}$$

where all elements of the matrix E are unities: $\|E_{ij}\| = 1$.

In (15), the angle γ_c is quite arbitrary. However, it should be small enough to provide shrinking the search domain. On the other hand, it should be not too small to avoid any stiffness related to that [6]. Finally, it is to be noted that the matrix A_0 is to be inverted to find the matrix B. However, it should be done only once because the matrix A_0 is the same for all the search domains.

Transform (8) allows us to shrink the search domain and focus on a much smaller area on the Pareto surface. It makes the algorithm more flexible and much less sensitive to the displacement of box D. It generates a "light beam" emitting from point M and highlighting a spot on the Pareto frontier [6]. As noted above, if no solution is found, the direction is switched on the opposite.

3.5 Arbitrary direction of search domain

The direction of the search domain along the lines parallel to the vector $\mathbf{l}_0$ can be sufficient. However, in the general case it is required to obtain the appropriate matrix A for an arbitrary unit vector $\mathbf{l}$. For this purpose, we consider a linear transform mapping the previous pattern in such a way that the vector $\mathbf{l}_0$ is mapped onto the vector $\mathbf{l}$. It can be obtained by multiplying both parts of equation (9) by an orthogonal matrix R:

$$R\mathbf{l}_0 = \mathbf{l}.$$

Next, we obtain the basis of vectors $\{\mathbf{a}'_i\}$ $(i = 1,\dots,M)$ uniformly distributed on the lateral area of the hypercone that has the axis parallel to the vector $\mathbf{l}$:

$$\mathbf{a}'_i = \frac{\sin\gamma_c}{\sin\gamma_0}\,\mathbf{e}'_i + \frac{\sin(\gamma_0-\gamma_c)}{\sin\gamma_0}\,\mathbf{l}\ \ (i = 1,\dots,M), \tag{16}$$

where $\mathbf{e}'_i = R\mathbf{e}_i$ are the basis vectors of the Cartesian coordinate system in which the vector $\mathbf{l}$ has equal coordinates. One can see that the columns of transition matrix R are the coordinates of the vectors $\mathbf{e}'_i$ $(i = 1,\dots,M)$ in the basis $\{\mathbf{e}_j\}$ $(j = 1,\dots,M)$. It is clear that all angles are preserved because the transform is orthogonal. In particular, $(\mathbf{a}'_i,\mathbf{l}) = \cos\gamma_0$. Hence, we obtain the matrix A in the general form:

$$A = A_0R^T = \frac{\sin\gamma_c}{\sin\gamma_0}\,R^T + \frac{\sin(\gamma_0-\gamma_c)}{\sin\gamma_0}\,E, \tag{17}$$

where $\|E_{ij}\| = \|l_j\|$.

If $\gamma_c = \gamma_0$, obviously $\mathbf{a}'_i = \mathbf{e}'_i$ that means the transform becomes orthogonal and is only reduced to a turn of the original Cartesian coordinate system. As such, the matrix A is orthogonal and $B = A^T$.

It is clear that in the general case $B = RA_0^{-1}$. The matrix A_0 is only assigned to the vector l_0. Thus, in the entire algorithm only matrix A_0 is to be inverted. It is important for multidimensional applications.

The general presentation requires the calculation of the orthogonal matrix R, the components of which must satisfy the following additional requirements:

$$\cos\gamma_0 \sum_{j=1}^{M} R_{ij} = l_i. \tag{18}$$

It is clear that the matrix R in (18) is not unique. The simplest way to obtain it is to consider the rotation from the vector $\mathbf{l}_0$ to the vector $\mathbf{l}$ in a Cartesian coordinate system related to these vectors:

$$R = DT_RD^T.$$

Here, T_R is an elementary rotation matrix describing the rotation in the plane created by the first two basis vectors

$$T_R = \begin{pmatrix} (\mathbf{l}_0,\mathbf{l}) & -\sqrt{1-(\mathbf{l}_0,\mathbf{l})} & 0 & \dots & 0 \\ \sqrt{1-(\mathbf{l}_0,\mathbf{l})} & (\mathbf{l}_0,\mathbf{l}) & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix}$$

and the matrix D is the transition matrix from the original basis to an orthogonal basis $\{\mathbf{b}_i\}$ assigned to the vectors $\mathbf{l}$ and $\mathbf{l}_0$. For example, it can be obtained as follows. First, we consider orthogonal unit vectors $\mathbf{b}_1 = \mathbf{l}_0$ and $\mathbf{b}_2 = (\mathbf{l} - (\mathbf{l},\mathbf{l}_0)\mathbf{l}_0) \,/\, \left(\sqrt{1-(\mathbf{l},\mathbf{l}_0)^2}\right)$. Then, we complement these two vectors by vectors $\{\mathbf{e}_i\}$ upon the Gram-Schmidt orthogonolization procedure. According to this procedure, each subsequent vector is to be orthogonal to all the previous.

More precisely, from all basis vectors $\{\mathbf{e}_i\}$ we eliminate two vectors that create the most minimal angle with the vector $\mathbf{l}$. Without loss of generality, assume that we retain vectors $\mathbf{e}_i$ $(i = 3,\ldots,M)$. Then, we have

$$\mathbf{b}_k = \frac{\mathbf{e}_k - \sum\limits_{i=1}^{i=k-1} (\mathbf{e}_k,\mathbf{b}_i)\mathbf{b}_i}{\sqrt{1 - \sum\limits_{i=1}^{i=k-1} (\mathbf{e}_k,\mathbf{b}_i)^2}} \quad (k = 3,\ldots,M).$$

Thus, the columns of the matrix D consist of the vectors $\mathbf{b}_i$ $(i = 1,\ldots,M)$:

$$D = (\mathbf{b}_1,\mathbf{b}_2,\ldots,\mathbf{b}_M).$$

It is clear that the obtained matrix D is orthogonal.

Thus, the direction of the search can be easily conducted. Finally, it is to be noted that the rotation from the vector $\mathbf{l}_0$ to a vector $\mathbf{l}$ makes sense only if the angle between the two vectors is big enough.

3.6 Rotation of search domain

The general representation of the matrix A given by equation (17) can be important for seeking the Pareto set nearby its boundary. If we consider orthogonal projection of the Pareto set onto the utopia hyperplane, the images of some Pareto points may not belong to the interior of the convex polygon (6), (7) spanned by the M vertexes $\boldsymbol{\mu}_i^*$. This fact was first noted in [12]. One of the possibilities to resolve this problem, suggested in [12] for the NC method, is based on the use of negative coefficients α_i. However, this can lead to too many redundant solutions [6]. Another opportunity is suggested in [6] and described below.

Let us consider the edge vectors of polygon (6), (7): $\boldsymbol{\nu}_i = \boldsymbol{\mu}_{i+1} - \boldsymbol{\mu}_i$ $(i = 1,\ldots,M-1)$. The point p_i belongs to a k-th edge of the polygon if and only if $\alpha_m = 0$ $(m \neq k, k+1)$. Assume that the vector $\mathbf{l}$ is related to the normal of the utopia hyperplane. Then, if the point M belongs to on an edge of the polygon, we rotate the vector $\mathbf{l}$ in the direction opposite to the polygon. In other words, $\mathbf{l}$ is changed in such a way that the orthogonal projection of the end of the vector, drawn from an edge, onto the utopia hyperplane does not fall in the interior of the polygon. For this purpose, in the utopia plane we introduce a unit vector which is the outer normal to the edge in question. The vector can be defined as:

$$\mathbf{s}_i = \frac{\boldsymbol{\nu}_{i+1} + \beta_i \boldsymbol{\nu}_i}{|\boldsymbol{\nu}_{i+1} + \beta_i \boldsymbol{\nu}_i|}, \quad \beta_i = -\frac{(\boldsymbol{\nu}_{i-1},\boldsymbol{\nu}_i)}{(\boldsymbol{\nu}_i,\boldsymbol{\nu}_i)}.$$

Then, the current vector $\mathbf{l}_r$ is determined via $\mathbf{s}_i$ and normal $\mathbf{n}$ to the utopia hyperplane towards the decrease of all objective functions as follows:

$$\mathbf{l}_r = \cos\theta_r \mathbf{n} + \sin\theta_r \mathbf{s}_i, \tag{19}$$

$$0 < \theta_r < \pi/2. \tag{20}$$

The angle θ_r is a parameter. Changing θ_r from 0 to $\pi/2$, the vector $\mathbf{l}_r$ is turned from the normal vector $\mathbf{n}$ to the vector $\mathbf{s}_i$ (see Figures 3a and 3b). Thus, we arrive at the following algorithm. If the reference point M belongs strictly to the interior of the polygon, then the vector $\mathbf{l}_r$ coincides with the normal $\mathbf{n}$. If point M is on an edge of the polygon, then an additional rotation of the vector may be required. To obtain an even distribution of the Pareto set, the number of additional points N_r related with the rotation of the vector $\mathbf{l}_r$ depends on the distance to the vertexes of the edge. For example, the rotation is not required at the anchor points. Generally speaking, it is reasonable to choose the maximal value of N_r at the center of an edge. The following evaluation of N_r is suggested for a k-th edge:

$$N_r = \text{integer}\,(4m\alpha_k\alpha_{k+1}) \quad (m \geq 1). \tag{21}$$

Finally, it is worth noting that this number can be substantially optimized if the information on the current local distribution of the Pareto set is taken into account. For example, as noted above, if a Pareto solution appears to be at an edge of the polygon, no additional rotation is needed and $N_r = 0$.

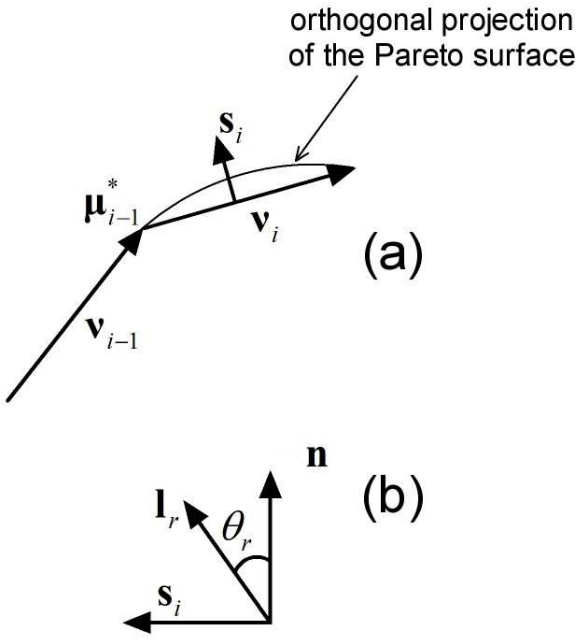

Figure 3: Rotation of search domain [6]

3.7 Finding a Pareto solution

Thus, the DSD algorithm gives us a well-spread distribution of search domains. In each selected search domain we seek only one Pareto solution. For that it is enough to introduce an aggregate objective function (AOF), which is a combination of objective functions. Obviously, the AOF cannot be arbitrary. Requirements to admissible AOF can be found in [19]. According to [19], an admissible AOF must be a locally coordinatewise increasing function of the objective functions. On the definition, a function is a coordinatewise increasing function if it is a strict monotonically increasing function with respect to each argument [20]. Partial examples of AOF are given in [13], [16] and [6] in the framework of PP method.

3.8 Filtering procedure

As noted in [6], the algorithm can generate local Pareto solutions. However, they can be removed via the filtering procedure based on the contact theorem. Consider the sketch of an example shown in Figure 4. A point P is a local Pareto solution rather than a global one. To filter local Pareto solutions, we put the reference point M of a search domain at a point considered as a candidate Pareto solution and set $A = I$ in (14), (15). If the point corresponds to a global Pareto solution (e.g., a point P'), then no any other solution can be obtained. It immediately follows from the contact theorem [21]. Thus, if $D \cap \mathbf{Y}^* = P$, then and only then the point P represents a Pareto solution. Thus, we have a criterion for verification if the solution is a global Pareto solution.

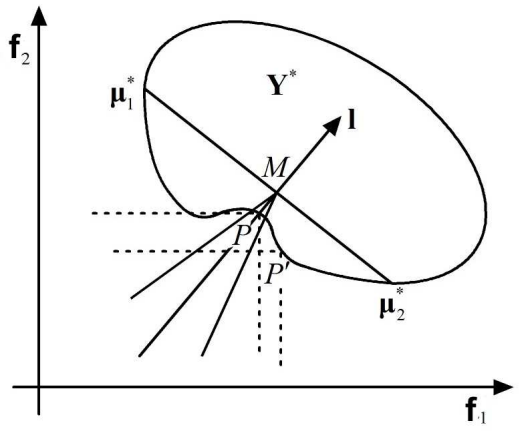

Figure 4: Filtering local Pareto solutions [6]

3.9 Scaling procedure

In the general formulation, to avoid undesirable severe skewing of the search domain in the algorithm, the objective functions should be preliminary scaled [1]:

$$f_i^{sc} = \frac{f_i - f_{i,\min}}{f_{i,\max} - f_{i,\min}}. \tag{22}$$

3.10 Step-by-step algorithm

The entire DSD method can shortly be formulated as a 12-step algorithm.

Step 1: apply the scaling procedure (22) to the objective functions.

Step 2: find anchor points according to the procedure described in Section 3.2.

Step 3: find the utopia polygon determined by (6) and (7).

Step 4: introduce a distribution of reference points according to Section 3.3.

Step 5: determine the angle γ_c and matrix A_0 according to (15).

Step 6: find matrix $B = A_0^{-1}$.

Step 7: identify the local search domain according to (10) and (11).

Step 8: find a local Pareto solution.

Step 9: if no solution is found, switch the direction of the search on the opposite.

Step 10: displace the search domain to another reference point.

Step 11: at the edges of the utopia polygon (6), (7), apply the rotation algorithm described in Section 3.6. Then, Steps 6-8 are repeated for matrix A determined by (17).

Step 12. apply the filtering procedure described in Section 3.8.

3.11 Efficiency of the algorithm

Thus, the algorithm described above is able to generate an entire well distributed Pareto set. This is achieved by solving a number of single-objective optimization problems for an AOF. The algorithm is efficient because the number of the single-objective problems solved mostly equals the number of the Pareto points obtained. The method can efficiently be applied in multidimensional case because most matrices are known explicitly in an analytical form. In the entire algorithm only matrix A_0 is to be inverted once. In addition, it is to be noted that the method can naturally be realized on parallel processors because each Pareto search can be done independently from the others.

4 Test Cases

The DSD algorithm was implemented in the PP method in [6]. In that paper, the efficiency of the approach is demonstrated on a number of test cases including comparison against the NBI, NC and original PP methods. The suggested algorithm performs quite well on multidimensional test cases, convex and concave frontiers.

4.1 Criterion of evenness

To compare different methods, the following criterion of evenness is suggested in [6]. It is based on a coefficient k_e characterizing how evenly a Pareto set is distributed on the Pareto surface. For this purpose, we introduce a curvilinear coordinate system $\{x^i\}$ $(i = 1,\dots,M-1)$ on the Pareto surface in the objective space. In the Riemann space $\mathbb{R}^{M-1}$, related to the Pareto surface, the Riemann metric is given by

$$d\mathbf{r}^2 = \sum_{i=1}^{M-1}\sum_{j=1}^{M-1} g_{ij} d\mathbf{x}^i d\mathbf{x}^j. \tag{23}$$

Then, the coefficient k_e is defined via the Hausdorff metric:

$$k_e = \frac{\max\limits_i \min\limits_j r_{ij}}{\min\limits_i \min\limits_j r_{ij}} \quad (i, j = 1, \ldots, N_p;\ i \neq j).$$

Here, N_p is the number of Pareto points, r_{ij} is the distance between an i-th and j-th Pareto points in metric (23): $r_{ij} = |\mathbf{x}_i - \mathbf{x}_j|$.

The coefficient k_e represents the ratio of the maximal possible distance between any two nearest Pareto points to the minimal one.

The following three relatively simple test cases from [6] demonstrate how the algorithm works.

Example 1. First, we consider a two-dimensional test case:

$$\min x, y \tag{24}$$

subject to a constraint

$$x^2 + y^2 \leq 1. \tag{25}$$

It is clear that in this test case the feasible domain $\mathbf{Y}^*$ corresponds to the unit circle and the Pareto surface is convex.

The solution of problem (24), (25) is represented by a segment of the unit circle as shown in Figure 5. The DSD algorithm provides an even representation of the Pareto frontier with $k_e = 1.6$ if $N_p = 11$. Without shrinking the search domain, if the angle $\gamma_c = \gamma_0 = 45^0$, the generated Pareto set turns out not to be well spread with $k_e = 5.6$. The difference between the two strategies is even more impressive if a similar test case with a concave Pareto frontier is tackled [6].

Example 2. Next, we consider an example of a concave Pareto frontier in $\mathbb{R}^3$:

$$\min x, y, z \tag{26}$$

subject to

$$\begin{aligned} x^2 + y^2 + z^2 &\geq 1, \\ x &> 0, \\ y &> 0, \\ z &> 0. \end{aligned} \tag{27}$$

In contrast to Example 1, the standard definition of an anchor point does not lead to a unique point. For example, the solution of a single-objective problem $\min x$ leads to a segment of the unit circle in the plane $x = 0$. Meanwhile, it is easy to see that the modified definition of the anchor point given in Section 3.2 results in only three anchor points: $(0,0,1)$, $(1,0,0)$ and $(0,1,0)$.

The entire orthogonal projection of the Pareto surface onto the utopia plane does not necessarily appear to be in the triangle created by the anchor points (see Figure 6). For this

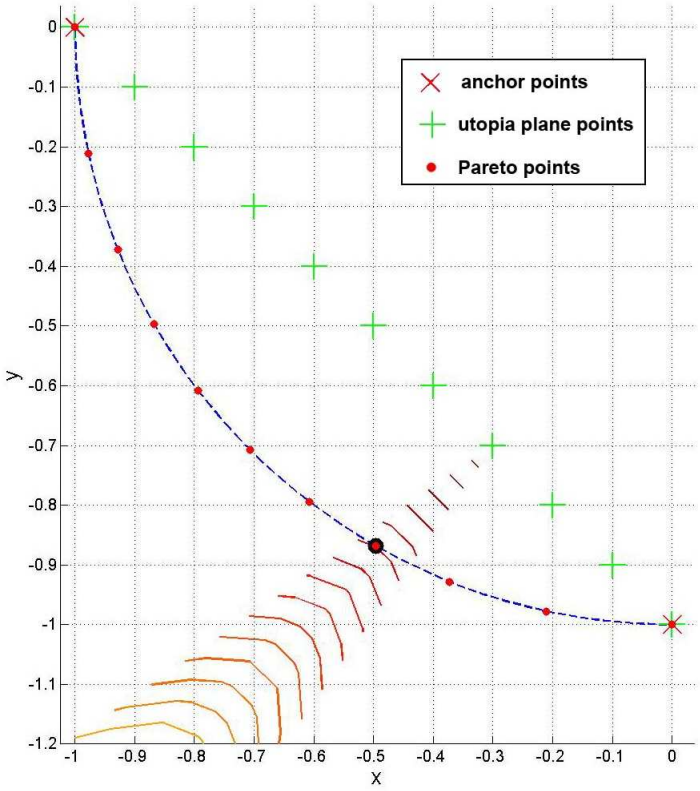

Figure 5: Segment-of-circle frontier [6]

reason, a method such as NBI, for example, is not able to catch the entire Pareto frontier if the coefficients α_i in (7) are not negative.

In [6], the rotation strategy described in Section 3.6 is used to generate a complete representation of the Pareto frontier. The utopia plane and reference points, distributed in the polygon (triangle) according to algorithm (6), (7), are given in Figure 7. Then, the generated Pareto set is shown in Figure 8.

As noted above, the definition of the anchor point might be very important for the efficiency of the algorithm. This statement is illustrated by several examples available in [6].

Example 3. The next test case, suggested in [9], includes a Pareto frontier with both convex and concave parts, which are created by three ellipsoid segments centered at the origin. The problem reads:

$$\min x, y$$

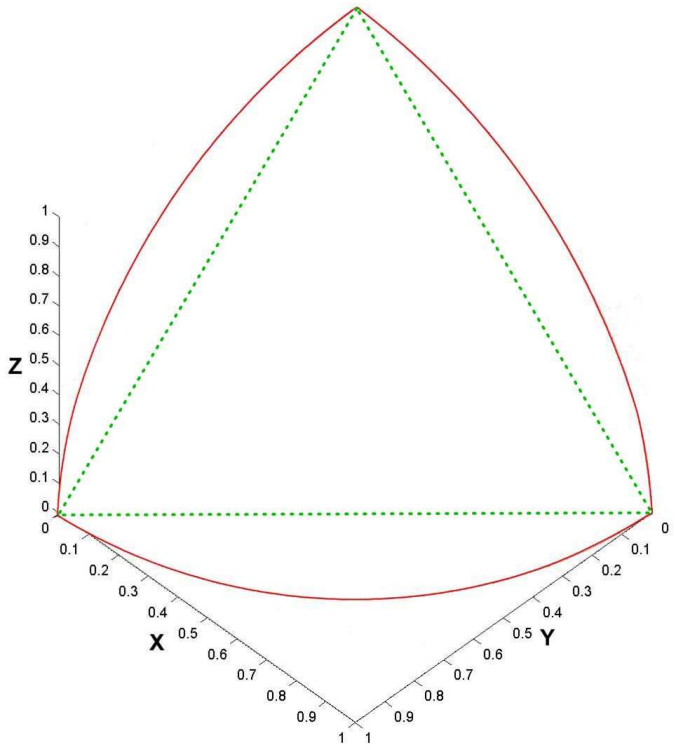

Figure 6: 3D test case. Orthogonal projection of Pareto surface [6]

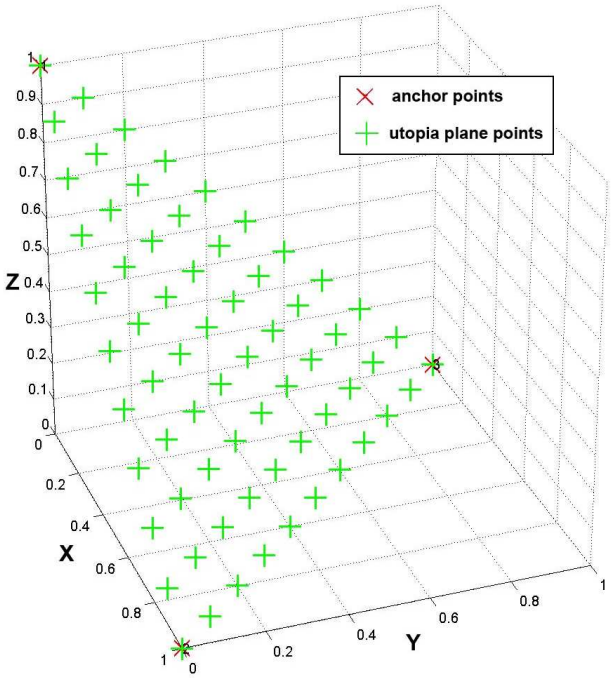

Figure 7: 3D test case. Utopia plane [6]

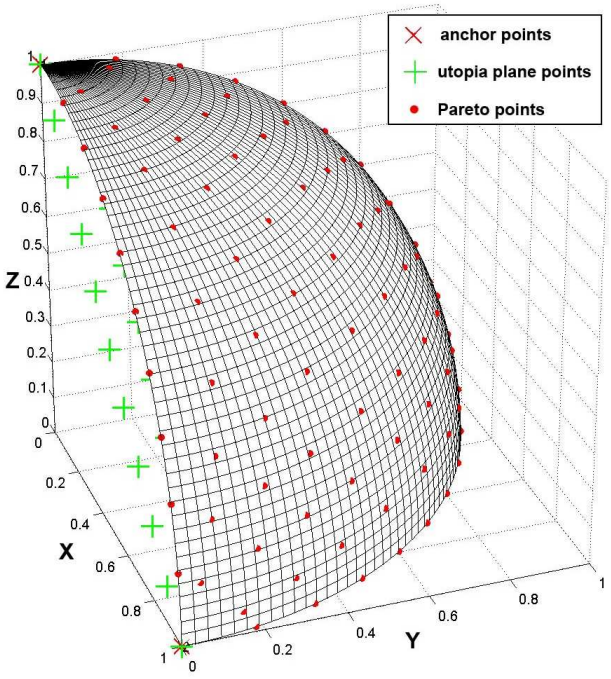

Figure 8: 3D test case. Pareto frontier [6]

subject to

$$\begin{aligned} x^2+(y/3)^2 &\geq 1,\\ x^4+y^4 &\geq 16,\\ (x/3)^3+y^3 &\geq 1,\\ 0 \leq x &\leq 2.9,\\ 0 \leq y &\leq 2.9. \end{aligned}$$

The exact Pareto curve is shown by the dashed line in Figure 9. It is important to note that the Pareto frontier is not smooth and is located on both sides of the utopia line. As shown in Figure 9, the algorithm is capable of capturing the entire frontier and generating a well-distributed Pareto set.

As soon as a quasi-even distributed Pareto set is available, the information on the Pareto optimal solutions can be complemented by a local approximation of the Pareto surface in the objective space. This is also important for a sensitivity analysis. A local approximation can be achieved by the either linear or quadratic approximation described in the next Section. It is based on the results obtained in [22]. In that paper it is shown that a linear approximation known in the literature is not applicable in the general formulation. A corrected linear approximation is suggested. It is proven the approximation to be accurate in a multidimensional case.

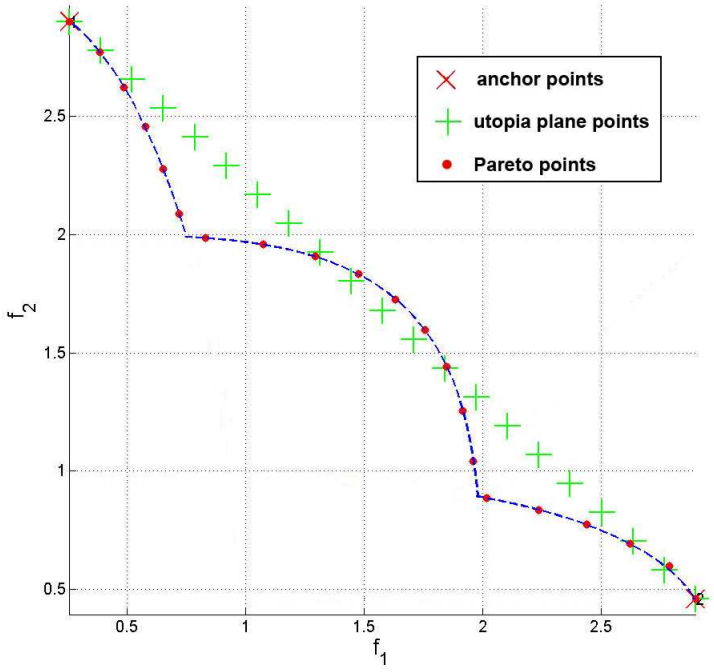

Figure 9: Hyperellipsoid frontier [6]

5 Pareto Local Approximation

Consider a Pareto solution and assume that in its vicinity the Pareto surface is smooth enough. To obtain a local approximation of the Pareto surface, we should identify only active constraints and consider their local approximation [23]. It is clear that in the general case not all constraints (2), (3) are necessarily active on the Pareto frontier. A constraint is said to be active at a Pareto point $\mathbf{x}^*$ of the design space $\mathbf{X}$ if a strict equality holds at this point [23]. We suppose that if some constraints are active at a Pareto point, then they remain active in its vicinity.

Without loss of generality, we assume that all constraints (2), (3) are active. Note the set of active constraints as $\mathbf{G} \in \mathbb{R}^I$, where $I = K + P$. Then, at the given point $\mathbf{x}^*$ of the feasible design space $\mathbf{X}^*$ we have:

$$\mathbf{G}(\mathbf{x}^*) = 0. \tag{28}$$

If $\mathbf{G} \in C^1(\mathbb{R}^I)$, then the constraints can be linearized:

$$J(\mathbf{x} - \mathbf{x}^*) = \mathbf{0}.$$

where J is the Jacobian of the active constraints set at $\mathbf{x}^* : J = \nabla\mathbf{G}$.

A point $\mathbf{x}^*$ is said to be regular if all gradients of the active constraints are linearly independent [21]. It is clear that at a regular point $\operatorname{rank} J = I$. In our further analysis we consider regular points.

In the objective space $\mathbf{Y}$ let the Pareto surface be represented by:

$$S(\mathbf{y}) = 0 \tag{29}$$

and $S \in C^2(\mathbb{R}^I)$ in a vicinity of $\mathbf{x}^*$.

The gradient of any differentiable function F at point $\mathbf{x}^*$ under constraints is given by the reduced gradient formula [25]:

$$\nabla F_{|S_l} = P\nabla F. \tag{30}$$

Here, S_l is the hyperplane tangent to $\mathbf{X}^*$ in the design space:

$$S_l = \{\mathbf{x}|\ J(\mathbf{x}-\mathbf{x}^*) = 0\},$$

and P is the projection matrix onto hyperplane S_l:

$$P = I - J^T(JJ^T)^{-1}J.$$

Then, in the objective space the tangential derivatives of the function F on the boundary of the feasible domain $\mathbf{Y}^*$ are give by

$$\frac{dF}{df_i} = \frac{dF}{d\mathbf{x}}_{|S_l}\frac{d\mathbf{x}}{df_i} \quad (i = 1,\ldots,M). \tag{31}$$

In equality (31) the meaning of the right-hand side is the following. The first term represents the reduced gradient (30), whereas the second term gives the derivative of the design vector $\mathbf{x}$ with respect to an objective function f_i along the direction that is tangent to the Pareto surface. The latter term can be represented via the gradients of the objective functions in the design space.

One can prove that the columns of matrix $P\nabla f, \mathbf{f} = (f_1, f_2, \ldots, f_M)^T$ are linearly dependent [22]. Without loss of generality, we assume that the first $n_f < M$ columns are linearly independent and represented by $P\nabla\widetilde{\mathbf{f}}$.

Thus,

$$d\widetilde{\mathbf{f}} = (P\nabla\widetilde{\mathbf{f}})^T d\mathbf{x}, \tag{32}$$

where matrix $P\nabla\widetilde{\mathbf{f}} \equiv (P\nabla\widetilde{f}_1, P\nabla\widetilde{f}_2, \ldots, P\nabla\widetilde{f}_{n_f})$ has all the columns linearly independent.

Next, we represent $d\mathbf{x}$ as

$$d\mathbf{x} = Ad\widetilde{f}. \tag{33}$$

To find the matrix A, we multiply both sides of equation (33) by $(P\nabla\widetilde{\mathbf{f}})^T$. Then, we obtain:

$$((P\nabla\widetilde{\mathbf{f}})^T d\mathbf{x})^T Ad\widetilde{\mathbf{f}} = d\widetilde{\mathbf{f}}$$

and

$$A = P\nabla\widetilde{\mathbf{f}}\left[(P\nabla\widetilde{\mathbf{f}})^T P\nabla\widetilde{\mathbf{f}}\right]^{-1}. \tag{34}$$

It is easy to see that the inverse matrix $[(P\nabla\widetilde{\mathbf{f}})^T(P\nabla\widetilde{\mathbf{f}})]^{-1}$ in (34) is always non-singular because all the vectors $P\nabla f_i$ $(i = 1,\ldots,n_f)$ are linearly independent. Thus, the matrix A is the right-hand generalized inverse matrix to $(P\nabla\widetilde{\mathbf{f}})^T$. From the definition of matrix A

it follows that $(P\widetilde{\nabla \mathbf{f}})^T A = I$ and $A_i^T P\widetilde{\nabla \mathbf{f}}_j = \delta_{ij}$, where I is the unit matrix and δ_{ij} is the Kronecker symbol.

From equation (34) it follows that $PA = A$ and in equation (32) $d\mathbf{x}$ belongs to the tangent plane S_l at the Pareto point. Thus,

$$\frac{d\mathbf{x}}{d\widetilde{\mathbf{f}}} A = P\widetilde{\nabla \mathbf{f}} \left[(P\widetilde{\nabla \mathbf{f}})^T P\widetilde{\nabla \mathbf{f}} \right]^{-1} \tag{35}$$

and for any $i \leq n_f$

$$\frac{d\mathbf{x}}{df_i} = \mathbf{A}_i,$$

where $A = (\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_{n_f})$. Then, from equations (30), (31) and (35), we obtain for any $i \leq n_f$:

$$\frac{dF}{df_i} = (P\nabla F)^T \mathbf{A}_i = \mathbf{A}_i^T P\nabla F = \mathbf{A}_i^T \nabla F.$$

In particular, if $F = f_j$ $(n_f < j < M)$, then we get the first order derivative of objective f_j with respect to objective f_i on the Pareto surface. Hence,

$$\frac{df_j}{df_i} = \mathbf{A}_i^T \nabla f_j \ \ (0 \leq i \leq n_f, \ \ _f < j \leq M). \tag{36}$$

As soon as we know the tangential derivatives (36), we are able to obtain a linear local approximation of the surface S.

One can derive the formula of sensitivity of objective f_j with respect to objective f_i along the greatest feasible descent direction for objective f_i [23]:

$$\frac{df_j}{df_i} = \frac{(P\nabla f_j, P\nabla f_i)}{(P\nabla f_i, P\nabla f_i)} \equiv \frac{(f_j, P\nabla f_i)}{(f_i, P\nabla f_i)}. \tag{37}$$

It is to be noted that formulae (37) is usually used as a linear approximation of the Pareto surface [24]. However, formulae (37) exactly corresponds to the linear approximation if and only if either $n_f = 1$ or the vectors $P\widetilde{\nabla \mathbf{f}}$ create an orthogonal basis. In the latter case the matrix $(P\widetilde{\nabla \mathbf{f}})^T P\widetilde{\nabla \mathbf{f}}$ is diagonal. If there are only two objectives functions, (36) and (37) always coincide because $n_f = 1$. However, in the general case, formulae (37) is not always applicable. This is demonstrated in the next Section.

On the Pareto surface in the objective space the operator of the first derivative can be defined as:

$$\frac{d}{df_i} = \mathbf{A}_i^T \nabla.$$

By applying this operator to the first order derivative, we arrive at the reduced Hessian:

$$\frac{d^2F}{df_i df_j} = \mathbf{A}_i^T \nabla (\mathbf{A}_j^T \nabla F) \approx \mathbf{A}_i^T \nabla^2 \mathbf{A}_j \ \ (0 \leq i, j \leq n_f). \tag{38}$$

Thus, we obtain a local approximation of the Pareto surface. It can be represented by either a linear hyperplane:

$$\sum_{i=1}^{n_f} \frac{dS}{df_i} \Delta f_i = 0 \tag{39}$$

or a quadratic surface:

$$\sum_{i=1}^{n_f} \frac{dS}{df_i} \Delta f_i + \frac{1}{2} \sum_{j,k=1}^{n_f} \frac{d^2 S}{df_j df_k} \Delta f_j \Delta f_k = 0, \tag{40}$$

where $\Delta \mathbf{f} = \mathbf{f} - \mathbf{f}(\mathbf{x}^*)$.

Approximations (39) and (40) can be rewritten with respect to the trade-off relations between the objective functions as follows:

$$f_p = f_p^* + \sum_{1}^{n_f} \frac{df_p}{df_i} \Delta f_i \ \ (p = n_f + 1, \ldots, M) \tag{41}$$

and

$$f_p = f_p^* + \sum_{i=1}^{n_f} \frac{df_p}{df_i} \Delta f_i + \frac{1}{2} \sum_{j,k=1}^{n_f} H_{jk}^{(p)} \Delta f_j \Delta f_k \ \ (p = n_f + 1, \ldots, M), \tag{42}$$

where

$$H_{jk}^{(p)} = \frac{d^2 f_p}{df_j df_k}.$$

In [23], to obtain a quadratic approximation, it is suggested to evaluate the reduced Hessian matrix H_{ij} via a least-squared minimization using the Pareto set generated around the original Pareto point. However, in the case of a well distributed Pareto set the accuracy of this approximation might not be sufficient. The determination of the reduced Hessian (38) is entirely based on the local values in vicinity of a Pareto point. One should note that the approximations (41) and (42) derived in [22] precisely correspond to the first three terms of the Taylor expansion in the general case.

It is clear that a local approximation of the Pareto surface allows us to carry out a sensitivity analysis and trade-off between different local Pareto optimal solutions.

The considered local approximation assumes that the Pareto surface under study is smooth enough. The case of a non-differentiable Pareto frontier and its local analysis is addressed in [22].

6 Example of a Local Approximation

Following [22], in this section, we compare the linear approximations of the Pareto surface (41) based on the derivatives (36) and (37). For this purpose we consider the optimization problem (26), (27).

It is easy to see that the first order tangent derivatives to the Pareto surface are given by:

$$\begin{aligned} \frac{df_3}{df_1} &= \frac{dz}{dx} = \frac{-x}{\sqrt{1-x^2-y^2}}, \\ \frac{df_3}{df_2} &= \frac{dz}{dy} = \frac{-y}{\sqrt{1-x^2-y^2}}. \end{aligned} \tag{43}$$

Next, we apply formulas (36) and (37). It is clear that on the Pareto surface the matrices J, P and A are given by:

$$J = [-2x, -2y, -2z],$$
$$P = \begin{pmatrix} y^2+z^2 & -xy & -xz \\ -xy & x^2+z^2 & -yz \\ -xz & -yz & x^2+y^2 \end{pmatrix}, \tag{44}$$
$$A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -x/z & -y/z \end{pmatrix}.$$

Then, from (37) we obtain

$$\frac{df_3}{df_1}_{|Eq.37} = \frac{-xz}{\sqrt{x^2+y^2}} = \frac{(x^2+y^2-1)x}{(1-x^2)\sqrt{1-x^2-y^2}}, \tag{45}$$
$$\frac{df_3}{df_2}_{|Eq.37} = \frac{-yz}{\sqrt{x^2+y^2}} = \frac{(x^2+y^2-1)y}{(1-y^2)\sqrt{1-x^2-y^2}}.$$

One can see that the derivatives (45) do not coincide with the exact derivatives (43). Thus, the approach [23], based on (37), leads only to an approximate linear approximation. This is demonstrated in Figure 10.

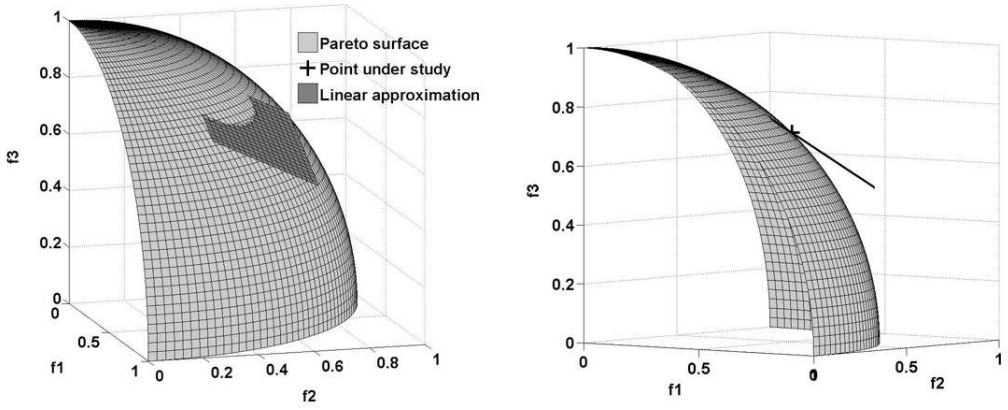

Figure 10: Linear approximation based on [23] (from [22])

In turn, from equation (36) we obtain the exact first order derivatives (43). It is easy to verify that equation (38) gives the exact second order derivatives. The linear approximation based on equations (36) and (41) is shown in Figure 11.

More examples including industrial applications are available in [22].

7 Conclusion

The DSD algorithm provides an efficient way for quasi-even generating a Pareto set in a quite arbitrary multidimensional formulation. The approach is based on shrinking a

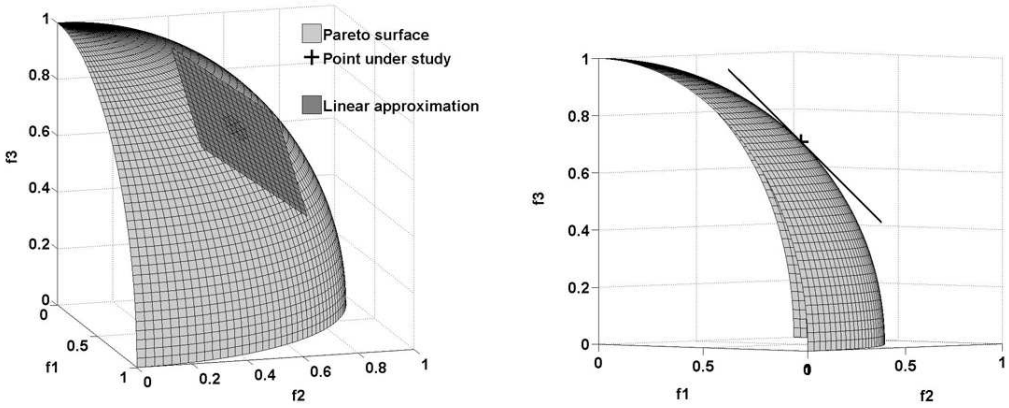

Figure 11: Linear approximation [22]

search domain. The orientation of the search domain in space can be easily conducted. The approach can be combined with different search engines. The use of the DSD algorithm with the PP technique [6] demonstrates that the approach is capable to generating both convex and concave Pareto sets and filter local Pareto solutions. The algorithm can efficiently be complemented by a local first- or second-order approximation of the Pareto frontier described in [22].

References

[1] Miettinen, K. M. *Nonlinear Multiobjective Optimization*. Boston: Kluwer Academic, 1999.

[2] Messac, A. *AIAA Journal*. 2000, 38 (1), 155–163.

[3] Koski, J. *Communications in Applied Numerical Methods*. 1985, 1, 333–337.

[4] Athan, T. W.; Papalambros, P. Y. *Engineering Optimization*. 1996, 27, 155–176.

[5] Das, I.; Dennis, J. E. *Structural Optimization*. 1997, 14, 63–69.

[6] Utyuzhnikov, S. V.; Fantini, P.; Guenov, M. D. *Journal of Computational and Applied Mathematics*. 2009, 223 (2), 820–841.

[7] Das, I.; Dennis, J. E. Normal-Boundary Intersection: A New Method for Generating the Pareto Surface in Nonlinear Multicriteria Optimization Problems, *SIAM Journal of Meter Design Optimization Problems*, In Proceeding of ASME Design Automation Conference, pp. 77–89, Montreal, Quebec, Canada, Sept. 17–20, 1989.

[8] Das, I. An Improved Technique for Choosing Parameters for Pareto Surface Generation Using Normal-Boundary Intersection, WCSMO-3 Proceedings, Buffalo, NY, March, 1999.

[9] Messac, A.; Ismail-Yahaya, A.; Mattson, C. A. *Structural and Multidisciplinary Optimization*. 2003, 25 (2), 86–98.

[10] Shukla, P. K.; Deb, K. *European Journal of Operational Research*. 2007, 181, 1630–1652.

[11] Yun, Y. B.; Nakayam, H.; Tanino, T.; Arakawa, M. *European Journal Operational Research*. 2001, 129, 586-595.

[12] Messac, A.; Mattson, C. *AIAA Journal*. 2004, 42 (10), 2101–2111.

[13] Messac, A. *AIAA Journal*. 1996, 34 (1), 149–158.

[14] Sanchis, J.; Martnínez, M.; Blasco, X.; Salcedo, J.V. *Structural and Multidisciplinary Optimization*. 2008, 36 (5), 537–546.

[15] Messac, A.; Mattson, C. *Optimization and Engineering*. 2002, 3, 431–450.

[16] Utyuzhnikov, S. V.; Fantini, P.; Guenov, M. D. Numerical method for generating the entire Pareto frontier in multiobjective optimization, Proceedings of Eurogen'2005, Munich, September 12-14, 2005.

[17] Collette, Y.; Siarry, P. Multiobjective Optimization: Principles and Case Studies, Berlin, Heidelberg, New York: Springer, 2003.

[18] Deb, K. Multi-objective Optimization Using Evolutionary Algorithms. Chichester, J. Wiley & Sons, 2001.

[19] Messac, A.; Melachrinoudis, A.; Sukam, C. *Optimization and Engineering Journal*. 2000, 1 (2), 171–188.

[20] Steuer, E. R. Multiple Criteria Optimization: Theory, Computation, and Application. Melbourne, Florida: Krieger, 1989.

[21] Vincent, T. L.; Grantham, W. J. Optimality in Parametric Systems. New York: John Wiley & Sons, 1981.

[22] Utyuzhnikov, S. V.; Maginot, J.; Guenov, M. D. *Journal of Engineering Optimization*. 2008, 40 (9), 821–847.

[23] Tappeta, R. V.; Renaud, J. E. Interactive multi-objective optimization procedure. AIAA 99-1207, April 1999.

[24] Tappeta, R. V.; Renaud, J. E.; Messac, A.; Sundararaj, G. J. *AIAA Journal*. 2000, 38 (5), 917–926.

[25] Fletcher, R. Practical Methods of Optimization. New York: John Wiley & Sons, 1989.