Post-Pareto Optimality Methods For The Analysis Of Large Pareto Sets In Multi-Objective Optimization

Victor Manuel Carrillo

University of Texas at El Paso, vcarrill@uacj.mx

Follow this and additional works at: https://digitalcommons.utep.edu/open_etd

Part of the Applied Mathematics Commons, and the Engineering Commons

Recommended Citation

https://digitalcommons.utep.edu/open_etd/1596

This is brought to you for free and open access by DigitalCommons@UTEP. It has been accepted for inclusion in Open Access Theses & Dissertations by an authorized administrator of DigitalCommons@UTEP. For more information, please contact lweber@utep.edu.
POST-PARETO OPTIMALITY METHODS FOR THE ANALYSIS OF LARGE PARETO SETS IN MULTI-OBJECTIVE OPTIMIZATION

VICTOR M. CARRILLO

Computational Science

Approved:

___________________________
Heidi Taboada, Ph.D., Chair

___________________________
Jose F. Espiritu, Ph.D.

___________________________
Vinod Kumar, Ph.D.

___________________________
Noe Vargas Hernandez, Ph.D.

___________________________
Benjamin C. Flores, Ph.D.
Dean of the Graduate School
Dedication

To the loving memory of my mother Maria Elena
POST-PARETO OPTIMALITY METHODS FOR THE ANALYSIS OF LARGE PARETO SETS IN MULTI-OBJECTIVE OPTIMIZATION

by

VICTOR M. CARRILLO, B.Sc., M.Sc.

DISSERTATION
Presented to the Faculty of the Graduate School of The University of Texas at El Paso
in Partial Fulfillment
of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

Computational Science
THE UNIVERSITY OF TEXAS AT EL PASO
December 2013
Acknowledgments

I would like to express my sincere gratitude to my advisor Dr. Heidi Taboada for her continuous support throughout my Ph.D studies. I really thank her for her patience, motivation, enthusiasm, and knowledge. Her guidance helped me all the time during my research work, and during the writing of this dissertation. Also, I would like to thank Dr. Jose Espiritu, Dr. Vinod Kumar and Dr. Noe Vargas Hernandez for being part of my dissertation committee and for the time they dedicated to review this work. Moreover, I would like to thank some of my fellow Ph. D students: Luis Basurto, Richard Medina, Daniel Yehedego, Oswaldo Aguirre and Nicolas Lopez. They each helped make my time in the PhD program more bearable, enjoyable and interesting.

Additionally, I would like to thank PROMEP for supporting me with a scholarship for doctoral studies. I will not finish without thanking my wife Rocio wholeheartedly; she always supported me and provided me with the strength I needed to complete this important stage of my life. Last but not least, I dedicate this dissertation to my children and grandchildren. This work is a proof of perseverance, and I hope it encourages you all to not dismay or give up when facing challenges along your life.
Abstract

Multiple objective optimization involves optimization in the presence of more than one (conflicting) criteria. Multi-criteria optimization problems arise in a variety of real-world applications. The main difference between single and multi-objective optimization is that in multi-objective optimization there is usually no single optimal solution, but a set of equally good alternatives with different trade-offs, also known as Pareto-optimal solutions. Evolutionary multi-objective optimization algorithms are widely used for solving optimization problems with multiple conflicting objectives. With the use of multi objective evolutionary algorithms, multiple objective optimization has become a two-part problem. First, the multiple objective optimization problem needs to be formulated and successfully solved using a multi objective evolutionary algorithm. Then, a non-dominated set also known as efficient or Pareto frontier needs to be analyzed to select a solution to the problem. This can represent a challenging task to the decision-maker because this set can contain a large number of solutions. Therefore there exist needs for efficient methods that can reduce the size of the Pareto-optimal set to facilitate decision-making. This decision-making stage is usually known as the post-Pareto analysis stage.

This thesis presents four different methods to perform post-Pareto analysis. The first method is the generalization of a method known as the non-numerical ranking preferences method. This method can help decision makers reduce the number of design possibilities to small subsets that clearly reflect the decision maker’s objective function preferences without having to provide specific weight values. Previous research has only presented the application of the non-numerical ranking preferences method using three and four objective
functions but had not been generalized to the case of $n$ objective functions. The work presented in this thesis expands the non-numerical ranking preferences method. The second method presented in this thesis uses a non-uniform weight generator method to reduce the size of the Pareto-optimal set.

A third method Sweeping cones is introduced to reduce the size of the Pareto set. Geometrically speaking, this method projects all of the objective function values and weights into the space over a unit radius sphere, and then sweeping cones are used to capture Pareto points that reflect decision-maker’s preferences. The fourth and last method developed called Orthogonal Search for post-Pareto optimality. This method generates a decreasing succession of mesh points guided by what is called an ideal direction. All methods have been tested on different problem instances to show their performance.
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acknowledgments</td>
<td>v</td>
</tr>
<tr>
<td>Abstract</td>
<td>vi</td>
</tr>
<tr>
<td>Table of Contents</td>
<td>viii</td>
</tr>
<tr>
<td>Chapter 1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>Chapter 2 Multi-objective optimization</td>
<td>4</td>
</tr>
<tr>
<td>2.1 Single and multi-objective optimization</td>
<td>4</td>
</tr>
<tr>
<td>2.2 Pareto-optimal set</td>
<td>5</td>
</tr>
<tr>
<td>2.3 Non dominance relation</td>
<td>7</td>
</tr>
<tr>
<td>2.4 Conditions for Pareto optimality</td>
<td>9</td>
</tr>
<tr>
<td>2.5 Classical methods for Pareto optimality</td>
<td>13</td>
</tr>
<tr>
<td>2.5.1 Weighted sum method</td>
<td>13</td>
</tr>
<tr>
<td>2.5.2 $\epsilon$-Constraint method</td>
<td>15</td>
</tr>
<tr>
<td>2.5.3 Weighted Metric Methods</td>
<td>17</td>
</tr>
<tr>
<td>2.5.4 Rotated Weighted Metric Method</td>
<td>20</td>
</tr>
<tr>
<td>2.5.5 Utility function method</td>
<td>21</td>
</tr>
<tr>
<td>2.7 Evolutionary algorithms</td>
<td>24</td>
</tr>
<tr>
<td>Chapter 3 Post-Pareto optimality methods</td>
<td>31</td>
</tr>
<tr>
<td>3.1 Compromise Programming Approach</td>
<td>31</td>
</tr>
<tr>
<td>3.2 Marginal Rate of Substitution Approach</td>
<td>32</td>
</tr>
<tr>
<td>3.3 Pseudo-Weight Vector Approach</td>
<td>33</td>
</tr>
<tr>
<td>3.4 Non-numerical Preferences Method</td>
<td>33</td>
</tr>
<tr>
<td>3.5 $k$-Means clustering</td>
<td>37</td>
</tr>
<tr>
<td>3.6 Greedy reduction</td>
<td>39</td>
</tr>
<tr>
<td>Chapter</td>
<td>Section</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>7</td>
<td>7.1 Sweeping cones method</td>
</tr>
<tr>
<td>7</td>
<td>7.2 Geometric interpretation of the sweeping cones technique</td>
</tr>
<tr>
<td></td>
<td>Summary</td>
</tr>
<tr>
<td>8</td>
<td>Orthogonal search methodology to prune Pareto fronts</td>
</tr>
<tr>
<td>8</td>
<td>8.1 Orthogonal search method</td>
</tr>
<tr>
<td></td>
<td>Summary</td>
</tr>
<tr>
<td>9</td>
<td>A hybrid technique for pruning Pareto fronts</td>
</tr>
<tr>
<td>9</td>
<td>9.1 Orthogonal search implementation</td>
</tr>
<tr>
<td>9</td>
<td>9.2 Non-numerical ranking preferences application</td>
</tr>
<tr>
<td></td>
<td>Conclusions</td>
</tr>
<tr>
<td>10</td>
<td>Future Research</td>
</tr>
<tr>
<td>References</td>
<td></td>
</tr>
<tr>
<td>Curriculum Vita</td>
<td></td>
</tr>
</tbody>
</table>
List of Tables

Table 1: Data in $\mathbb{R}^3$ to check non-dominance on points A, B, C, D, E, and F........................................8
Table 2: Comparing A against the rest of the solutions A, B, C, D.................................................................8
Table 3: Comparing B against the rest of the solutions A, B, C, D.................................................................8
Table 4: Comparing C against the rest of the solutions A, B, C, D.................................................................8
Table 5: Comparing D against the rest of the solutions A, B, C, D.................................................................8
Table 6: Chromosomes mating pool........................................................................................................27
Table 7: Chromosomes mating pool after selection.........................................................................................28
Table 8: Chromosomes mating pool after crossover.....................................................................................28
Table 9: Chromosomes mating pool after mutation.........................................................................................29
Table 10: Mating pool for a single point mutation..........................................................................................30
Table 11: Disadvantages for $k$-means and hierarchical algorithms..............................................................42
Table 12: Reduced Pareto set with 5 weights and threshold value $\alpha=0.2$......................................................52
Table 13: Reduced Pareto set with 5 weights and threshold value $\alpha=0.3$......................................................52
Table 14: Reduced Pareto set with 5 weights and threshold value $\alpha=0.4$......................................................53
Table 15: Reduced Pareto set with 7 weights and threshold value $\alpha=0.2$......................................................61
Table 16: Reduced Pareto set with 7 weights and threshold value $\alpha=0.3$......................................................61
Table 17: Reduced Pareto set with 7 weights and threshold value $\alpha=0.4$......................................................62
Table 18: 75 non-dominated solutions............................................................................................................67
Table 19: Pruned values with non-uniform weight generator.............................................................................67
Table 20: Pruned values with non-numerical ranking preferences method....................................................68
Table 21: Finite sequences obtained with the conditional weight generator method...............................71
Table 22: Size 3 sequences obtained with the conditional weight generator method...............................73
Table 23: Pareto front from a redundancy allocation problem 75 solutions........................................77
Table 24: Pruned values with a post Pareto approach using a conditional uniform weight
generator method for a 750 run.................................................................78
Table 25: Pruned values with a post Pareto approach using a conditional uniform weight
generator method for a 7,500 run.................................................................78
Table 26: Pruned values with a post Pareto approach using a conditional uniform weight
generator method for a 75,000 run.................................................................79
Table 27: Pruned values with a post Pareto approach using a conditional uniform weight
generator method for a 750,000 run.................................................................79
Table 28: Tabulated values for pruned values with a post Pareto approach using a conditional uniform weight generator method for a 750,000 run........................................79
Table 29: Six most likely optimal points after minimizing the composite function with the non-
uniform weight generator approach.................................................................80
Table 30: Seven most likely optimal points after minimizing the composite function using the non-numerical ranking preferences method.....................................................80
Table 31: 75 non-dominated solutions........................................................................84
Table 32: Pruned values with the sweeping cones approach........................................85
Table 33: Minimum values for tables of pruned values with the sweeping cones approach......86
Table 34: Features of minimum values for tables obtained with sweeping cones approach……86
Table 35: Pareto nested sets obtained with orthogonal search…………………………………99
Table 36: Subsets pruned and shared points ,where point 360 is the closest to Ideal point……105
Table 37: Statistics of table 36………………………………………………………………………………106
Table 38: Subsets pruned sizes, and distances from the closest and farthest points to the Ideal point (0,0,0)…………………………………………………………………………………………107
Table 39: Pruned subset for a 178, 00 run ………………………………………………………………107
Table 40: Comparison of the four points obtained in a 178,000 run to the closest point to the ideal point…………………………………………………………………………………………107
Table 41: Comparison to the closest point……………………………………………………………108
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>Pareto-optimal set and Pareto-front representation</td>
<td>5</td>
</tr>
<tr>
<td>Figure 2</td>
<td>Pareto fronts for minimization and maximization problems</td>
<td>6</td>
</tr>
<tr>
<td>Figure 3</td>
<td>3D graph of the Pareto front</td>
<td>7</td>
</tr>
<tr>
<td>Figure 4</td>
<td>Graph of weak Pareto point’s sets</td>
<td>9</td>
</tr>
<tr>
<td>Figure 5</td>
<td>Pareto front example 3</td>
<td>11</td>
</tr>
<tr>
<td>Figure 6</td>
<td>Subset of Pareto-front example 4</td>
<td>13</td>
</tr>
<tr>
<td>Figure 7</td>
<td>Weighted sum method on a convex Pareto-front</td>
<td>15</td>
</tr>
<tr>
<td>Figure 8</td>
<td>Disadvantage of the weighted method for non-convexity</td>
<td>15</td>
</tr>
<tr>
<td>Figure 9</td>
<td>Pareto front point’s example 5</td>
<td>15</td>
</tr>
<tr>
<td>Figure 10</td>
<td>Pareto front points selected with $\varepsilon$ - Constraint Method</td>
<td>16</td>
</tr>
<tr>
<td>Figure 11</td>
<td>Weighted metric method illustration with $p = 1$</td>
<td>18</td>
</tr>
<tr>
<td>Figure 12</td>
<td>Weighted metric method illustration with $p = 2$</td>
<td>19</td>
</tr>
<tr>
<td>Figure 13</td>
<td>Pareto front subset for example 7</td>
<td>20</td>
</tr>
<tr>
<td>Figure 14</td>
<td>Weighted metric method illustration with $p = \infty$</td>
<td>20</td>
</tr>
<tr>
<td>Figure 15</td>
<td>Rotated weight method with $p = 2$</td>
<td>21</td>
</tr>
<tr>
<td>Figure 16</td>
<td>Utility function method contours</td>
<td>22</td>
</tr>
<tr>
<td>Figure 17</td>
<td>Evolutionary algorithm optimal search</td>
<td>25</td>
</tr>
<tr>
<td>Figure 18</td>
<td>Chromosomes before crossover</td>
<td>28</td>
</tr>
<tr>
<td>Figure 19</td>
<td>Chromosomes after crossover</td>
<td>28</td>
</tr>
<tr>
<td>Figure 20</td>
<td>Chromosomes before mutation</td>
<td>29</td>
</tr>
<tr>
<td>Figure 21</td>
<td>Chromosomes after mutation</td>
<td>29</td>
</tr>
</tbody>
</table>
Figure 22: One complete iteration for one single point ......................................................... 29
Figure 23: Compromise programming approach method illustration .................................... 32
Figure 24: Marginal rate of substitution approach method illustration ................................. 32
Figure 25: Pseudo vector approach ......................................................................................... 33
Figure 26: Weight region geometric representation ............................................................... 35
Figure 27: Graph of pruned values in table 11 ...................................................................... 52
Figure 28: Graph of pruned values in table 12 ...................................................................... 53
Figure 29: Graph of pruned values in table 14 ...................................................................... 62
Figure 30: Graph of pruned values in table 15 ...................................................................... 62
Figure 31: Graph of pruned values in table 24 ...................................................................... 78
Figure 32: Graph of pruned values in table 25 ...................................................................... 78
Figure 33: Graph of pruned values in table 26 ...................................................................... 79
Figure 34: Graph of pruned values in table 27 ...................................................................... 79
Figure 35: Graph of sweeping cones ..................................................................................... 83
Figure 36: Graph of sweeping cones ..................................................................................... 83
Figure 37: Three random sweeping cones .......................................................................... 84
Figure 38: Three random sweeping cones .......................................................................... 84
Figure 39: Pareto front .......................................................................................................... 84
Figure 40: Minimum point obtained from subset 1 ............................................................... 86
Figure 41: Minimum point obtained from subset 2 ............................................................... 86
Figure 42: Minimum point obtained from subset 3 ............................................................... 86
Figure 65: DTLZ7 Pareto reduced to 240 points from top to bottom view...............................99
Figure 66: DTLZ7 Pareto reduced to 87 points.................................................................100
Figure 67: DTLZ7 Pareto 87 points from top to bottom view..........................................100
Figure 68: Pareto-optimal front (red dots obtained with NSGA2)....................................102
Figure 69: Normalized Pareto-front alpha= 1.................................................................103
Figure 70: Pruned Pareto-front for an alpha=0.5............................................................103
Figure 71: Pruned 4401 points for an alpha=0.3 ............................................................103
Figure 72: Pruned 4283 points for alpha=0.2.................................................................103
Figure 73: Pruned 178 points for alpha=0.1.................................................................104
Figure 74: Pruned 41 points for alpha=0.05.................................................................104
Figure 75: Pruned 24 points for alpha=0.04.................................................................104
Figure 76: Pruned 14 points for alpha=0.03.................................................................104
Figure 77: Multiple Knee Pareto front...........................................................................110
Chapter 1
Introduction

Many real-world engineering optimization problems are implicitly or explicitly multi-objective in nature (Mietinnen, 1999). Multiple objective optimization problems can be found in many different areas in nature and fields, such as economics, biology, engineering, etc. Almost all the optimization problems in real life involve more than one objective to be optimized, and normally those objectives are in conflict with each other. Such a situation is very easy to observe in real life. For instance, a very high performance product is also a high cost item, and a customer always seeks merchandise with high performance, but with a low cost.

Generally, there are two primary approaches to solve multiple objective optimization problems: mathematical methods and meta-heuristic methods. The first approach involves the aggregation of the attributes into a linear combination of the objective functions, also known as scalarization (Nakayama et al., 2009). Some examples of mathematical methods used to solve multiple objective optimization problems are the weighted sum method, goal programming, the lexicographic method, and utility functions among others (Collete & Siarry, 2003). The second general approach involves populating a number of feasible solutions along the Pareto frontier using local search methods, evolutionary algorithms, or other means. When using this second general approach, the final solution to the multi-objective optimization problem is a set of non-dominated solutions also called Pareto-optimal set (Branke et al., 2008).

Pareto-optimality is defined in terms of the non-dominance principle. According to this principle, all solutions obtained in the Pareto set are said to be optimal in the sense that no solution dominates any other solution in the solution space. Without loss of generality, for a
minimization problem, a solution \( x_1 \) dominates a solution \( x_2 \) if and only if, the two following conditions are true:

\[
x_1 \text{ is no worse than } x_2 \text{ in all objectives, i.e. } f(x_1) \leq f(x_2) \quad \forall \ i, \ i \in \{1,2,\ldots,n\}
\]

\[
x_1 \text{ is strictly better than } x_2 \text{ for at least one objective, i.e. } f(x_1) < f(x_2) \text{ for at least one } i
\]

Therefore, the solution of a multiple objective optimization is a set of non-dominated solutions, also known as Pareto set or Pareto front. Usually, the Pareto set contains a large number (in some cases, thousands) of solutions. From the decision-maker perspective, consideration, and evaluation, of all the non-dominated solutions can be prohibitive and inefficient. The selection of solutions from the Pareto set is called post-Pareto optimality analysis. And, this decision-making stage is the main focus of my research work. Even though several methods for solving multi-objective optimization problems have been developed and studied in the past, very little prior work has been done on the post-Pareto optimality stage.

In this dissertation four methods are presented for post-Pareto analysis. The first method is a generalization of the non-numerical ranking preferences method proposed by Taboada and Coit (2006). The second method is an approach based on the \( n \)-values uniform random number generation method, developed by Carrillo & Taboada (2011). The third method is called the Sweeping Cones technique. This method is introduced to reduce the size of the Pareto set. Geometrically speaking, this method projects all of the objective function values and weights into the space over a unit radius sphere, and then sweeping cones are used to capture the Pareto points that reflect decision-maker’s preferences. The fourth and last method developed is called Orthogonal Search for post-Pareto optimality. This method generates a decreasing succession of mesh points guided by what is called an ideal direction.
This work is divided into ten chapters. Chapter 1 is an overview of how the post-optimality analysis has been addressed in the past, as well as the methodologies and objectives considered in previous research. Chapter 2 presents a comprehensive review of the theory behind multiple objective optimization. This includes some of the classical multi-objective optimization methods, and an introduction to evolutionary algorithms. Chapter 3 is dedicated to present the most recent post-Pareto optimality methods. Chapter 4 presents the development of the generalization of the non-numerical ranking preferences method. Chapter 5 presents an approach for post-Pareto analysis, based on the principles of the non-numerical ranking preferences method (NNRPM) using a different weight generation method. In Chapter 6, a new weight generation method and its performance is tested with the well-known multiple objective redundancy allocation problem. The Pareto front of this problem was obtained from Taboada & Coit, 2006. In Chapter 7, the Sweeping Cones technique is developed for post Pareto analysis. This technique projects the Pareto front points over a unitary sphere, and with the help of random cones selects subsets of prioritized points. In Chapter 8, a last post-Pareto analysis approach is developed based on an Orthogonal Search. Chapter 9 presents a hybrid technique for pruning Pareto fronts. Chapter 10 presents conclusions and discusses some ideas for future research.
Chapter 2

Multi-objective optimization

This chapter provides the elements, terminology and concepts of multi-objective optimization of the two general existent approaches for solving multi-criteria problems: the classical mathematical methods and the meta-heuristic approaches.

2.1 Single and Multi-objective Optimization

Optimization refers to finding one or more feasible solutions which correspond to critical values of one or more objective functions. If an optimization problem comprises only one objective function, the task of finding an optimal solution is called single objective optimization.

When an optimization problem contains more than one objective function, the procedure to find a solution to this problem is known as Multi-objective optimization. (Marler & Arora, 2004)

Without loss of generality any multi-objective optimization problem can be presented as a minimization problem as shown in Equation 1:

\[
\text{Minimize } f(x) = (f_1(x), ..., f_n(x)) \\
\text{subject to } g_j(x) \leq 0; j = 1, ..., m \\
x \in R^k
\]

Where: \( n \) is the number of objective functions \( f_n(x) \), \( m \) is the number of inequality constraints \( g_j(x) \), \( x \in R^k \) is a vector of design variables, and \( f_n(x) \in R^k \) is a vector of \( n \) objective functions, where \( f_i: R^k \rightarrow R \) for \( i = 1, ..., n \). The feasible or decision space is defined as \( X = \{x \in R^k | g_j(x) \leq 0; j = 1, ..., m\} \). The objective space is defined as the image of the
feasible decision space $X$ i.e. $Z = \{f(x)|x \in X\}$. Having just a single solution that optimizes all objective functions simultaneously is difficult, and such a solution usually does not exist.

For example, consider the following bi-objective optimization problem

$$\text{Minimize } f_1(x)$$

$$\text{Minimize } f_2(x)$$

$$\text{s.t. } g_j(x) \leq 0; j = 1, \ldots, m \quad x \in R^3$$

The image of the decision space under the objective functions is shown in Figure 1. From this perspective it is hard, if not impossible to decide which is or are the optimum values of the problem. The yellow points in the purple region represent optimal solutions $f(x^*) = (f_1(x^*), f_2(x^*))$ and the yellow points in the blue region, are the corresponding arguments denoted by $x^*$.

![Figure 1 Pareto-optimal set and Pareto-front representations (yellow dots)](image)

In a simple optimization problem, the notion of optimality is direct. The search is for the best (the minimum or the maximum) value of an objective function. In a multi-objective optimization problem, the notion of optimality is not as obvious as can be seen from Figure 1.

### 2.2 Pareto optimal set

Due to the above argument, a new definition of optimality had to be created; a definition that accounted for all the criteria. The concept of Pareto optimality is based on the definition of
the non-dominance principle. This set of optimal solutions is known as the Pareto optimal set or set of the non-dominated solutions. The complementary set of solutions is called the dominated solutions set. The image of the Pareto-optimal set is called the Pareto front (Mietinnen, 1999).

**Example 1:** Let us assume that each one of the objective spaces in Figure 2 correspond to each one of the four multi-objective optimization problems. The red line corresponds to their respective Pareto fronts.

a) $\text{Min } f_1(x), \text{Min } f_2(x) \text{ s.t. } g_j(x) \leq 0 \text{ for } j = 1, \ldots, m \text{ } x \in \mathbb{R}^2$

b) $\text{Min } f_1(x), \text{Max } f_2(x) \text{ s.t. } g_j(x) \leq 0 \text{ for } j = 1, \ldots, m \text{ } x \in \mathbb{R}^2$

c) $\text{Max } f_1(x), \text{Min } f_2(x) \text{ s.t. } g_j(x) \leq 0 \text{ for } j = 1, \ldots, m \text{ } x \in \mathbb{R}^2$

d) $\text{Max } f_1(x), \text{Max } f_2(x) \text{ s.t. } g_j(x) \leq 0 \text{ for } j = 1, \ldots, m \text{ } x \in \mathbb{R}^2$

![Figure 2 Pareto fronts represented as red curves.](image)

**Example 2:** For a three objective minimization problem the Pareto front is shown in Figure 3

$$\text{Min } f_1(x), \text{Min } f_2(x), \text{Min } f_2(x) \text{ s.t. } g_j(x) \leq 0 \text{ for } j = 1, \ldots, m \text{ } x \in \mathbb{R}^2$$

6
Consequently, the idea of Pareto optimality is used to describe solutions for MOP problems. A solution point is Pareto optimal if it is not possible to move from that point and improve at least one objective function without detriment to any other objective function. Alternatively, a point is weakly Pareto optimal if it is not possible to move from that point and improve (or worsen depending either minimizing or maximizing) all objective functions simultaneously. (Figure 1 and Figure 2 source: http://www.lania.mx/~ccoello/EMOO/)

2.3 Pareto optimality and Non-dominance relation

The Pareto dominance relation was originally proposed by Edgeworth in 1881, and later generalized by Pareto in 1896. The formal definition of non-dominance is:

**Definition 1** A point $x^* \in X$ is called a Pareto optimal or non-dominated point if and only if

$$f_i(x^*) \leq f_i(x) \text{ for all } i \text{ and there is at least an index } j \text{ such that } f_i(x^*) < f_j(x)$$

To illustrate the above definition a simple exercise is solved for calculating the non-dominated points of a set of six vectors.

**Example 2** Identify the non-dominated set from the following points where all objectives are minimized (Deb, 2004):
Table 1 Data in $\mathbb{R}^3$ to check non-dominance

<table>
<thead>
<tr>
<th>Solution</th>
<th>$f_1$</th>
<th>$f_2$</th>
<th>$f_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>5</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>C</td>
<td>3</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>D</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>F</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

Solution: Comparing all values of row A vs. row B in Table 1, A does not dominate B since $3 > 1$ in the second column. Similarly comparing row A against all the remaining rows $t$ A dominates points C, E and F as shown in Table 2. Continuing with the same procedure table 3, table 4, and table 5 show B, C, D and dominance, and by the property of transitivity for dominance.

Table 2 Point A dominance

<table>
<thead>
<tr>
<th>Comparing</th>
<th>A against the rest of the solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-B</td>
<td>✔️ x ✔️ Non Dominates B</td>
</tr>
<tr>
<td>A-C</td>
<td>✔️ ✔️ ✔️ A dominates C</td>
</tr>
<tr>
<td>A-D</td>
<td>✔️ x ✔️ Non Dominates D</td>
</tr>
<tr>
<td>A-E</td>
<td>✔️ ✔️ ✔️ A dominates E</td>
</tr>
<tr>
<td>A-F</td>
<td>✔️ ✔️ ✔️ A dominates F</td>
</tr>
</tbody>
</table>

Table 3 Point B dominance

<table>
<thead>
<tr>
<th>Comparing</th>
<th>B against the rest of the solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>B-A</td>
<td>x ✔️ ✔️ Non Dominates A</td>
</tr>
<tr>
<td>B-C</td>
<td>x ✔️ ✔️ Non Dominates C</td>
</tr>
<tr>
<td>B-D</td>
<td>x ✔️ ✔️ Non Dominates D</td>
</tr>
<tr>
<td>B-E</td>
<td>x x ✔️ Non Dominates E</td>
</tr>
<tr>
<td>B-F</td>
<td>x ✔️ ✔️ Non Dominates F</td>
</tr>
</tbody>
</table>

Table 4 Point C dominance

<table>
<thead>
<tr>
<th>Comparing</th>
<th>C against the rest of the solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>C-A</td>
<td>x x ✔️ Non Dominates A</td>
</tr>
<tr>
<td>C-B</td>
<td>✔️ x ✔️ Non Dominates B</td>
</tr>
<tr>
<td>C-D</td>
<td>x x ✔️ Non Dominates D</td>
</tr>
<tr>
<td>C-E</td>
<td>✔️ x ✔️ Non Dominates E</td>
</tr>
<tr>
<td>C-F</td>
<td>✔️ ✔️ ✔️ Non Dominates F</td>
</tr>
</tbody>
</table>

Table 5 Point D dominance

<table>
<thead>
<tr>
<th>Comparing</th>
<th>D against the rest of the solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>D-A</td>
<td>✔️ ✔️ ✔️ Non Dominates A</td>
</tr>
<tr>
<td>D-B</td>
<td>✔️ ✔️ ✔️ Non Dominates B</td>
</tr>
<tr>
<td>D-C</td>
<td>✔️ ✔️ ✔️ D dominates C</td>
</tr>
<tr>
<td>D-E</td>
<td>✔️ ✔️ ✔️ D dominates E</td>
</tr>
<tr>
<td>D-F</td>
<td>✔️ ✔️ ✔️ D dominates F</td>
</tr>
</tbody>
</table>

Finally is found that A, B and D points are non-dominated, thus the Pareto-front is the subset of points \{A, B, D\}.
There is a class of non-dominated points that are classified as weak non-dominated or weak Pareto solutions. This collection of points usually poses a dilemma for the decision maker because of its size.

**Definition 2** A \( x^* \in X \) is called a weak Pareto solution if \( f_i(x^*) < f_i(x) \) for all \( i \). In figure 4 (Nakayama, H., et al., 2009) is an example of an infinity set of weak Pareto solutions.

![Figure 4](image-url) 

**Figure 4** Flat borders are weak Pareto solutions

Generating Pareto optimal solutions plays an important role in Multi-objective optimization and mathematically the problem is considered to be solved when the Pareto optimal set is found. However, this is not always enough. It is desirable to obtain one solution. This means that the Pareto optimal solutions must be put in a complete order. This is why is needed a decision maker preference structure; that is, finding the Pareto optimal solutions that best satisfies the decision maker.

**2.4 Conditions for Pareto-Optimality**

In this section are stated the optimality conditions for the Multi-objective optimization problem given in Equation 3 and a couple of examples are presented to apply these concepts:
\[ \text{Minimize } f(x) = (f_1(x), \ldots, f_n(x)) \]
\[ \text{s.t. } g_j(x) \leq 0; j = 1, \ldots, m \]
\[ x \in R^k \]

It is assumed that all objective and constraint functions are continuously differentiable

The following statement is known as the necessary condition for Pareto-optimality

**Theorem 1** (Fritz-John necessary condition) If \( x^* \) is a Pareto-optimal point then there exist vectors \( \lambda \geq 0, \mu \geq 0 \) (where \( \lambda \in R^k, \mu \in R^m \) and \( \lambda \neq 0, \mu \neq 0 \)) such that the following conditions are satisfied:

1. \( \sum_{i=1}^{k} \lambda_i \nabla f_i(x^*) + \sum_{j=1}^{m} \mu_j \nabla g_j(x^*) = 0 \)

2. \( \mu_j g_j(x^*) = 0 \text{ for all } j = 1, \ldots, m \)

For an unconstrained Multi-objective optimization problem the above necessary conditions are reduced to \( \sum_{i=1}^{k} \lambda_i \nabla f_i(x^*) = 0 \), or in matrix form:

\[
\begin{bmatrix}
\frac{\partial f_1(x^*)}{\partial x_1} & \ldots & \frac{\partial f_k(x^*)}{\partial x_1} \\
\vdots & \ddots & \vdots \\
\frac{\partial f_1(x^*)}{\partial x_n} & \ldots & \frac{\partial f_k(x^*)}{\partial x_n}
\end{bmatrix}
\begin{bmatrix}
\lambda_1 \\
\vdots \\
\lambda_k
\end{bmatrix} = \begin{bmatrix}
0 \\
\vdots \\
0
\end{bmatrix} \tag{5}
\]

If the above matrix is square \( n \times n \) and \( x^* \) is a Pareto optimal point then is enough to compute the Jacobian matrix determinant:

\[
\det \begin{bmatrix}
\frac{\partial f_1(x^*)}{\partial x_1} & \ldots & \frac{\partial f_n(x^*)}{\partial x_1} \\
\vdots & \ddots & \vdots \\
\frac{\partial f_1(x^*)}{\partial x_n} & \ldots & \frac{\partial f_n(x^*)}{\partial x_n}
\end{bmatrix} = 0 \tag{6}
\]
Example 3 (Deb, 1999) Identify the candidate Pareto-optimal points solutions and the candidate Pareto-front for the following problem:

\[
Min \ f_1(x_1, x_2) = x_1^4 - 4x_1x_2 \\
Min \ f_2(x_1, x_2) = x_1 + 2x_2^2
\]

Solution: Note that there are no constraints, therefore the determinant of the Jacobian matrix we must be calculated and equated to zero to obtain the candidate Pareto-optimal points.

Indeed,

\[
\begin{vmatrix}
\frac{\partial f_1(x^*)}{\partial x_1} & \frac{\partial f_1(x^*)}{\partial x_2} \\
\frac{\partial f_2(x^*)}{\partial x_1} & \frac{\partial f_2(x^*)}{\partial x_2}
\end{vmatrix} = \begin{vmatrix}
4x_1^3 - 4x_2 & -4x_1 \\
1 & 4x_2
\end{vmatrix} = 16x_1^3 - 416x_2^2 + 4x_1 = 0
\]

Solving for \( x_1 \) and \( x_2 \) the candidate Pareto-optimal points is the set

\[
\left\{ (x_1, x_2) : x_2 = \frac{x_1 \pm \sqrt{x_1^6 + x_1}}{2} \right\}
\]

Figure 5 Pareto front points example 3

The following theorem states sufficient conditions for a solution to be Pareto-optimal for convex functions.
**Theorem 2** Karush-Kuhn-Tucker (KKT) sufficient condition for Pareto-optimality

Let the objective functions of the MOOP in equation (1) be convex and the constraint functions concave. Let be the objective and constraint functions continuously differentiable at a feasible solution \( x^* \). A sufficient condition for \( x^* \) to be a Pareto-optimal is that there exist vectors \( \lambda > 0 \) and \( \mu \geq 0 \) (where \( \lambda \in R^k \), \( \mu \in R^m \)) such that the following equations are true:

1. \( \sum_{i=1}^{k} \lambda_i \nabla f_i(x^*) + \sum_{j=1}^{m} \mu_j \nabla g_j(x^*) = 0 \) 

[7]

2. \( \mu_j g_j(x^*) = 0 \) for all \( j = 1, ..., m \)

**Example 4** (Deb, 2004) for the following problem

\[
\begin{align*}
\text{Min } & f_1(x_1, x_2) = x_1 \\
\text{Min } & f_2(x_1, x_2) = x_2 \\
\text{s.t. } & g(x_1, x_2) = (x_1 - 2)^2 + (x_2 - 2)^2 \leq 4
\end{align*}
\]

Find the Pareto-optimal region.

The KKT conditions are:

\[
\sum_{i=1}^{k} \lambda_i \nabla f_i(x^*) + \sum_{j=1}^{m} \mu_j \nabla g_j(x^*) = \lambda_1 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \lambda_2 \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \mu \begin{bmatrix} 2(x_1 - 2) \\ 2(x_1 - 2) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}
\]

Thus solving the system of equations

\[
\begin{align*}
\lambda_1 + 2\mu(x_1 - 2) &= 0 \\
\lambda_2 + 2\mu(x_2 - 2) &= 0 \\
\mu[(x_1 - 2)^2 + (x_2 - 2)^2 - 4] &= 0
\end{align*}
\]

the Pareto-optimal region is \( \{(x_1, x_2) : x_1 = \frac{16\mu^2 - \lambda_1^2}{4\mu^2} , x_2 = \frac{\lambda_2^2}{4\mu^2} , \text{ with } 0 < \lambda_1 \leq 4\mu, \mu > 0 \} \)

A subset of the Pareto-optimal front is Figure 6
Note that it is hard for the decision maker to select from that large set of alternatives in the Pareto front. On the other hand, solving MOP’s using the optimality conditions set in many cases a hard task in terms of computational effort that several methods had to be developed for the search of the Pareto optimal points trying to avoid such difficulty. In next section are shown various approaches that were developed based in the reduction of the original multi-objective optimization problem into a single objective optimization and some examples for illustrating their application.

2.5 Classical methods for Pareto-optimality (*)

(*)Classical methods is a phrase coined by Deb, Kalyanmoy to distinguish from evolutionary methods.

A Multi-objective optimization problem with more than two objectives is hard to solve with just the optimality conditions presented in last section. Thereby multi-objective optimization problems were solved by scalarization before the evolutionary methods were sufficiently developed. Scalarization means that objective functions of the problem are transformed into a single (scalar) function or a real valued function i.e. a transformation $F: \mathbb{R}^n \rightarrow \mathbb{R}$ is applied to $f(x)$ such that problem (1) is now written as:
Once the Multi-objective optimization problem has been scalarized, the single nonlinear optimization methods can be applied. One of the most used methods is the weighting technique.

### 2.5.1 Weighted Sum Method

The weights of the objective functions are usually chosen in proportion to its relative importance in the problem. Otherwise they are selected at random.

\[
\begin{align*}
\text{Minimize} & \quad F(x) = \sum_{i=1}^{k} w_i f_i(x) \\
\text{s.t.} & \quad g_j(x) \leq 0 \quad ; j = 1, \ldots, m \\
& \quad x \in \mathbb{R}^n \\
\end{align*}
\]

The optimal points obtained when solving the above problem with single nonlinear optimization are Pareto-optimal points due to the following theorem:

**Theorem 3** The solution to the problem by equation (9) is Pareto-optimal if all of the weights are positive.

In figures 6 and 7 is depicted the search of Pareto points using the weighted sum. The source of all figures in this chapter excepting for the examples is: “Classic methods for multi-objective optimization” by Giuseppe Narzizi.
**Example 5:** Given the problem:

\[
\begin{align*}
\text{Min } f_1(x, y) &= x^3 + y^2 \\
\text{Min } f_2(x_1, x_2) &= y^2 - 4x
\end{align*}
\]

find the Pareto-optimal set using the weights \( w, 1 - w \) where \( w > 0 \). Solution: restating the problem as:

\[
\text{Min } f(x, y) = w(x^3 + y^2) + (1-w)(y^2 - 4x)
\]

, calculating the stability points

\[
\frac{\partial f}{\partial x} = 3x^2w - 4(1 - w) \\
\frac{\partial f}{\partial y} = 2yw + 2y(1 - w)
\]

, solving the equations

\[
3x^2w - 4(1 - w) = 0 \\
2yw + 2y(1 - w) = 0
\]

The Pareto-optimal set is:

\[
\left\{ x = 2\sqrt{\frac{1-w}{3w}}, \quad y = 0 : w > 0 \right\}
\]

with graph in figure 1.4

---

**Figure 7** Weighted sum method on a convex Pareto-front

**Figure 8** Disadvantage of the weighted method for non-convexity

**Figure 9** Pareto front for example 5
This method is the simplest way to solve a MOP. For problems having a convex Pareto-optimal front it is guaranteed finding solutions on the entire Pareto-optimal set. However if the Pareto front is non-convex it cannot provide solutions among sunken parts.

2.5.2 \( \epsilon \) - Constraint Method

This method alleviates the difficulties faced by the weighted sum when solving problems with non-convex objective spaces Figure 10. The method is:

\[
\begin{align*}
\text{Minimize} & \quad f_\mu(x) \\
\text{s.t.} & \quad f_i(x) \leq \epsilon_i, \quad i = 1, \ldots, k, \quad i \neq \mu \\
& \quad g_j(x) \leq 0, \quad j = 1, \ldots, m \\
& \quad h_l(x) = 0, \quad l = 1, \ldots, r \quad x \in R^3 
\end{align*}
\]

(10)

For convex or non-convex objective space problems the following theorem supports the utility of the \( \epsilon \) - Constraint Method:

**Theorem 4:** The unique solution of the \( \epsilon \) - constraint problem stated in equation (10) is a Pareto-optimal for any given upper bound vector \( \epsilon = (\epsilon_1, \ldots, \epsilon_{t-1}, \epsilon_{t+1}, \ldots, \epsilon_k) \).

Distinct Pareto-optimal solutions can be found by using different \( \epsilon_t \) values. The method can be used for problems having convex or non-convex objective spaces, but has the disadvantage that...
solutions depend on the chosen vector $\epsilon$. It must be chosen in the interval $\min f_\mu \leq \epsilon \leq \max f_\mu$
.

**Example 6:** Find the optimal solution for the function in terms of $\epsilon$, using Karush-Kuhn-Tucker optimality conditions:

$$
\text{Minimize } f_1(x, y) = y^2 - 4x
$$

$$
\text{Minimize } f_2(x, y) = x^3 + y^2
$$

The following reformulation is made

$$
\text{Min } f(x, y) = y^2 - 4x
\text{ s.t. } g(x, y) = x^3 + y^2 \leq \epsilon
$$

Solution: Let $L(x, y) = f(x, y) + \mu g(x, y)$ be the Lagrangian function associated. Let us find the stationary points of the Lagrangian function:

$$
\nabla L(x, y) = \nabla f(x, y) + \mu \nabla g(x, y)
$$

$$
= \begin{bmatrix} -4 \\ 2y \end{bmatrix} + \mu \begin{bmatrix} 3x^2 \\ 2y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}
$$

(11)

Solving equations (11) we get the Lagrangian stationary points: $x = \pm \frac{2}{\sqrt{3\mu}}$, $y = 0$.

$\mu = \frac{4}{3\sqrt[3]{\epsilon_1^2}}$ which implies that there is only one stationary point to consider: $x = \sqrt[3]{\epsilon_1}$, $y = 0$

Let us check the point $(\sqrt[3]{\epsilon_1}, 0)$ is candidate to be optimum points. Let us check the Lagrangian function:

$$
z^T \nabla^2 L(\sqrt[3]{\epsilon_1}, 0) z = [z_1 \ z_2] \begin{bmatrix} \frac{6\sqrt[3]{\epsilon_1}}{4} & 0 \\ 0 & \frac{6\sqrt[3]{\epsilon_1}}{4} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = 6z_1^2 \sqrt[3]{\epsilon_1} + 4z_2^2 > 0 \ \forall (z_1, z_2) \neq 0
$$

Therefore the optimal solutions are all the points $(\sqrt[3]{\epsilon_1}, 0)$ for any $\epsilon > 0$ value selected.
2.5.3 Weighted Metric Methods

Instead of using a weighted sum of the objectives, other means of combining multiple objectives into a single objective can also be used. For this purpose, weighted metrics such as $l_p$ and $l_\infty$ distance metrics are used. For non-negative weights, the weighted $l_p$ metric from the ideal solution $z^*$ can be minimized as follows:

$$
\text{Minimize } l_p(x) = \left( \sum_{i=1}^{k} w_i \left| f_i(x) - z^*_i \right|^p \right)^{1/p}
$$

s.t. $g_j(x) \leq 0 \quad j = 1, \ldots, m$

$h_l(x) = 0 \quad l = 1, \ldots, r \quad x \in \mathbb{R}^n$

For $p = 1$ $l_1(x) = \sum_{i=1}^{k} w_i \left| f_i(x) - z^*_i \right|$ the scalarized function is equivalent to the weighted sum approach.

For $p = 2$ $l_2(x) = \left( \sum_{i=1}^{k} w_i \left| f_i(x) - z^*_i \right|^2 \right)^{1/2}$ the scalarized function is a weighted distance to the ideal solution $z^*$.
Example 7: Consider the following problem (Deb, 2004):

\[
\begin{align*}
\text{Min } f_1(x, y) &= x^3 + y^2 \\
\text{Min } f_2(x, y) &= 5(y^2 - x)
\end{align*}
\]

Use the weighted \( l_2 \) distance metric, find the Pareto-optimal solutions corresponding to the weight vector \((w_1, w_2) = (1, 0)\).

Solution: for the weights proposed and taking \( z^* = (0, 0) \) as ideal point, and restating the problem as \( \text{Min } f_1(x, y) = (x^3 + y^2)^2 \) where \((x, y) \in \mathbb{R}^2\), its critical points are:

\[
\left\{ \left(x, \sqrt{-x^3}\right) : x < 0 \right\}
\]

The Hessian is \( \mathbf{H}(x, y) = \begin{bmatrix} 30x^4 + 12xy^2 & 12xy^2 \\ 12xy^2 & 4x^3 + 12y^2 \end{bmatrix} \) therefore

\[
\mathbf{H}(x, \sqrt{-x^3}) = \begin{bmatrix} 18x^4 & -12x^4 \\ -12x^4 & -8x^3 \end{bmatrix}
\]

since

\[
\det \left[ \mathbf{H}_{ii}(x, \sqrt{-x^3}) \right] = 18x^4 > 0 \quad \text{for } x \neq 0
\]

and

\[
\det \left[ \mathbf{H}(x, \sqrt{-x^3}) \right] = -144x^7(x+1) > 0 \iff -1 < x < 0
\]

the Pareto-optimal solutions is
\[
\left\{ \left( x, \sqrt{-x^3} \right) : x < 0 \right\}
\]
and the corresponding Pareto-front subset is
\[
\left\{ (f_1, f_2) : f_1 = x^3 + (-x)^{3/2}, f_2 = 5(-x)^{3/2} - 5x < 0 \text{ for } -1 < x < 0 \right\},
\]
with plot in Figure 13.

![Figure 13 Pareto front subset for example 7](image)

For \( p = \infty \), \( l_2 \) can be reduced to the Weighted Tchebycheff problem, Figure 14:

Minimize \( l_{\infty}(x) = \max_{1 \leq i \leq k} \left\{ w_i \left| f_i(x) - z_i^* \right| \right\} \)

s.t. \( g_j(x) \leq 0 \quad j = 1, \ldots, m \)

\( h_l(x) = 0 \quad l = 1, \ldots, r \)

\( x \in \mathbb{R}^n \)  

(13)

![Figure 14 Weighted metric method with \( p = \infty \)](image)

The weighted Tchebycheff method guarantees finding each and every Pareto-optimal solution when \( z^* \) is a utopian objective vector (Mietinen, K., 1999). To applying the method, objective
functions must be normalized. A negative aspect is that requires knowledge of the maximum and minimum values of the objective functions

2.5.4 Rotated Weighted Metric Method

Instead of directly using the \( l_\infty \) metric as it is stated in last section, the \( l_p \) metric can be applied with an arbitrary rotation from the ideal point. Let us assume that \( R \) is the square rotating axis matrix therefore the system after rotation can be expressed without loss of generality equally to the original formulation.

\[
\text{Minimize } l_p(x) = \left( \sum_{i=1}^{k} w_i \left| f_i(x) - z_i^* \right|^p \right)^{1/p}
\]

\[ s.t. \quad g_j(x) \leq 0 \quad j = 1, \ldots, m \]
\[ h_l(x) = 0 \quad l = 1, \ldots, r \quad x \in \mathbb{R}^n \]

By using different rotation matrices, the above functions can be minimized. For case \( p=2 \) the metric can be written as:

\[
I_p(x) = \left[ \left( (f_i(x) - z_i^*)' C (f_i(x) - z_i^*) \right) \right]^{1/2}
\]

where the rotation matrix is

\[
R = \begin{bmatrix}
\cos \alpha & \sin \alpha \\
-\sin \alpha & \cos \alpha \\
\end{bmatrix}
\]

and

\[
C = \begin{bmatrix}
\cos \alpha & \sin \alpha \\
-\sin \alpha & \cos \alpha \\
\end{bmatrix}^T \begin{bmatrix}
w & 0 \\
0 & w_2 \\
\end{bmatrix} \begin{bmatrix}
\cos \alpha & \sin \alpha \\
-\sin \alpha & \cos \alpha \\
\end{bmatrix}
\]

The rotation matrix will transform the objective axis into another set of axes. With this procedure it is easier to attain the sunken parts of the Pareto-optimal front. Unfortunately, this approach has many parameters to cope with and in certain problems could reach dominated points so there is no guarantee of getting always Pareto-optimal points, Fig 15.

\[ \text{Figure 15 Rotated weight method with } p = 2 \]
2.5.5 Utility or Value Function Method

To use this method, the user provides a mathematical function $U : R^m \rightarrow R$ which relates all the objectives functions. The task is to maximize the Utility function as follows:

$$\text{Maximize } U(f(x))$$

s.t. $g_j(x) \leq 0 \quad j = 1, \ldots, m$

$h_l(x) = 0 \quad l = 1, \ldots, r$

$x \in R^n$ \hspace{1cm} (15)

Rosenthal (1985) stated that the utility function must be strictly decreasing before it can be used in Multi-objective optimization. Miettinen, K., (1999) proves the following theorem which makes meaningful this method, Figure 16:

**Theorem 5:** Let the utility function $U : R^m \rightarrow R$ be strictly decreasing. If $U$ attains its maximum at $f^* = f(x^*)$. Then $f^*$ is Pareto-optimal

This method is simple and ideal, if the user has at hand a utility function, but it requires users to come up with an adequate utility function and avoid using over-simplified utility functions.
Example 8: For the following two-objective problem (Deb, 2004)

\[ \begin{align*}
\text{Min} & \quad f_1(x, y) = x^2 + y^2 \\
\text{Min} & \quad f_2(x_1, x_2) = 5 + y^2 - x \\
\text{s.t.} & \quad g_1(x_1, x_2) = x + 5 \geq 0 \\
& \quad g_2(x_1, x_2) = 5 - y \geq 0
\end{align*} \]

use the utility function \( U = 50 - f_1 - f_2 \) and find the Pareto-optimal solution.

Solution: Calculate the reduced Lagrangian from the K.K.T sufficiency conditions. Since the utility function \( U \) must to maximized, without loss of generality it can be written as
\[ F = 50 - f_1 - f_2. \]

The restated problem:

\[ \begin{align*}
\text{Max} & \quad U = 45 - x^2 - 2y^2 + x \\
\text{s.t.} & \quad g_1(x_1, x_2) = x + 5 \geq 0 \\
& \quad g_2(x_1, x_2) = 5 - y \geq 0
\end{align*} \]

\((x^*, y^*) = (-5, 5)\) and lagrangian values \( \mu_1 = 6 \), \( \mu_2 = 10 \); the gradients
\[ \nabla g_1(-5, 5) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \nabla g_2(-5, 5) = \begin{bmatrix} 0 \\ -1 \end{bmatrix} \] are linearly independent. And the Hessian satisfies
\[ z'\nabla^2 L(x, y)z = z' \begin{bmatrix} -2 & 0 \\ 0 & -2 \end{bmatrix} z = -(z_1^2 + z_2^2) < 0 \quad \forall \ (z_1, z_2) \neq (0,0) \] such that \( z'\nabla g_i(-5, 5) = 0 \), and thereby \((x^*, y^*) = (-5, 5)\) is a Pareto-optimal of the original problem.

To close this section it is worth to mention that other very important methods are The Goal programming methods, the lexicographic method, the weighted Goal programming, and The Min-Max method, which are not described here.

One common characteristic of multi-objective optimization algorithms with single objective optimization problems is that Pareto-optimal points search is through updating a single solution per iteration and that mainly use a deterministic transition rule to approach the optimum solution.
Every classical optimization algorithm although convergent is designed to solve a specific type of problem (Reklaitis et al., 1983). The Geometric method is designed to solve only constrained polynomial-type objective functions, but cannot be applied easily to solve other types of problems. The conjugate gradient methods work well in problems having one optimal solution, but not expected to work well in problems having multiple optimal solutions. The Frank-Wolfe's successive linear programming method Reklaitis et al., (1983) works efficiently on a constrained linear function, but on nonlinear objective functions its performance depends mainly on the initial conditions. Thus, one algorithm may be best suited for one class of problems but not for those problems outside that class. This requires users to know a number of optimization algorithms to solve different optimization problems.

Another search approach to obtain Pareto-optimal solutions is with evolutionary algorithms (EA’s). Evolutionary algorithms generate sets of Pareto-optimal points or non-dominated points.

In next section an outline of one evolutionary algorithm (EA) will be presented. This algorithm is in the group of the most actually used.

2.6 Evolutionary Algorithms

In the 1960s, several researchers independently suggested adopting the principles of natural evolution, in particular Darwin’s theory of the survival of the fittest, for optimization. These pioneers were Lawrence Fogel, John H. Holland, Ingo Rechenberg, and Hans-Paul Schwefel. One distinguishing feature of these so-called evolutionary algorithms (EAs) is that they work with a population of solutions. This is of particular advantage in the case of multi-objective optimization, as they can search for several Pareto optimal solutions simultaneously in one run as shown in Figure 17, providing the DM with a set of alternatives to choose from.
Evolutionary algorithms mimic natural evolutionary principles in the search and optimization procedures as: evolution, reproduction, mutation recombination and selection. Some Evolutionary Algorithms are: Genetic algorithms (GA), evolutionary expression programming (EP), gene expression programming (GEP), evolution strategy (ES), and neuro-evolution (NE). Some related techniques are swarm algorithms as: ant colony, bee swarm, cuckoo search etc. Through their differences is a common technique to better understand the characteristics of classical optimization and evolutionary algorithms. Indeed let us start with the classical optimization methods (COM) features:

**Classical optimization methods are classified as**

a) Direct methods and

b) Gradient based methods

In direct search methods, only the objective functions and constraints are used to guide the search strategy; whereas gradient methods require first and second order derivatives for the objective functions and constraints.

Common difficulties with direct and gradient methods are:

Convergence to an optimal solution depends on the chosen initial solution; most algorithms tend to get stuck to a suboptimal solution; an algorithm solving one optimization problem may not be
solving a different one; every classical optimization algorithm is designed to solve a specific type of problem: geometric programming solves only polynomial. Most classical optimization methods use the point to point approach where only one solution gets updated to a new solution by iteration so the advantages of parallel systems cannot be fully exploited.

On the other hand we have that:

Evolutionary algorithms (EA's) work with a population of solutions instead of a single solution; EA's do not require any auxiliary information except the objective functions values; EA's use probabilistic rules to guide their search; EA's are robust since can be applied to a wide variety of problems, and finally as early mentioned EA's can be easily used in parallel systems. One striking difference between classical optimization methods (COM) and EA's is in their convergence theory development. Whereas in COM there is plenty of convergence results that supports it. In EA's, until recently Yu et al., 2001 proved necessary and sufficient conditions on EA's convergence with Markov's chain theory.

Next is described the working principle of EA's with a Genetic Algorithm example for a can design.

**Example 9:** The minimization problem of a can design cost material is:

\[
\min f(d, h) = c \left( \frac{\pi d^2}{2} + \pi dh \right)
\]

s.t. \( g(d, h) = \frac{\pi d^2 h}{4} \geq 300 \)

\( 0 \leq d \leq 31 \text{ cm}, 0 \leq h \leq 31 \text{ cm} \)

To use a Genetic Algorithm, the objective function and constraint \( g \) must be represented as binary strings. For the sake of simplicity it is assumed that parameters \( d \) and \( h \) are integers. However GA's are not restricted to use only integer values.
Also will be used five bits to code each of the decision variables \( d \) and \( h \), which makes the overall string made out a \( (d, h) \) 10 bites long as is shown below:

If \( (d, h) = (8, 6)_{10} \) where each one of the 0 and 1 values are called "genes" and the whole zeros and ones string is known as "chromosome". Afterwards choose a \( (d, h) = (8, 10)_{10} \) chromosome and calculate its fitness. In most cases the fitness is calculated with the objective function as \( f(8, 6) = 16 \). The first step of the GA is to generate randomly a population of feasible points and calculate its fitness as shown before. Assume that six chromosomes are generated and its fitness calculated as in the next table:

<table>
<thead>
<tr>
<th>Values</th>
<th>Chromosomes Mating Pool</th>
<th>Fitness ( f(d, h) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(9,11)</td>
<td>1001 1011</td>
<td>28</td>
</tr>
<tr>
<td>(13,9)</td>
<td>1101 1001</td>
<td>46</td>
</tr>
<tr>
<td>(12,6)</td>
<td>1100 0110</td>
<td>29</td>
</tr>
<tr>
<td>(5,8)</td>
<td>0101 1000</td>
<td>11</td>
</tr>
<tr>
<td>(7,4)</td>
<td>0111 0100</td>
<td>11</td>
</tr>
<tr>
<td>(6,13)</td>
<td>0110 1101</td>
<td>20</td>
</tr>
</tbody>
</table>

Now the reproduction or selection operator is applied to the chromosomes in Table 6.

The primary objective of the reproduction operator is to make duplicates of good solutions and eliminate bad solutions in a population, while keeping the population size constant. Some common methods for selection are: Tournament selection, proportionate selection and ranking selection. According to Goldberg, (1991) tournament selection has better or equivalent convergence and computational time complexity properties compared to any other selection operator. For this example a tournament selection is applied. Tournament selection consists on choosing at random a pair of chromosomes, and compares their fitness. The best of each pair is kept in the matting pool discarding the worst and duplicating the best chromosomes in order to keep the population size. In the following table are the results of the selection:
Table 7 Mating pool after selection

<table>
<thead>
<tr>
<th>Values</th>
<th>Chromosomes</th>
<th>Fitness $f(d, h)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(5,8)</td>
<td>0101 1000</td>
<td>11</td>
</tr>
<tr>
<td>(7,4)</td>
<td>0111 0100</td>
<td>11</td>
</tr>
<tr>
<td>(6,13)</td>
<td>0110 1101</td>
<td>20</td>
</tr>
<tr>
<td>(5,8)</td>
<td>0101 1000</td>
<td>11</td>
</tr>
<tr>
<td>(7,4)</td>
<td>0111 0100</td>
<td>11</td>
</tr>
<tr>
<td>(6,13)</td>
<td>0110 1101</td>
<td>20</td>
</tr>
</tbody>
</table>

Know the crossover operator is applied picking at random a pair of strings from the matting pool and some portions of the strings are exchanged between them in order to create two new strings.

Let us select two strings and do a crossover exchanging 7 bits from the right side from each one of the chromosomes

(5, 8) 01 01 1000 --------> 01 10 1000 = (6, 8)
(6, 13) 01 10 1101 --------> 01 01 1101 = (5, 13)

Figure 18 Chromosomes before crossover

Figure 19 Chromosomes after crossover

A next pair is selected and crossover until a new population of the same size is obtained.

Table 8 Mating pool after crossover

<table>
<thead>
<tr>
<th>Values</th>
<th>Chromosomes</th>
<th>Fitness $f(d, h)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(6,8)</td>
<td>0110 1000</td>
<td>13</td>
</tr>
<tr>
<td>(7,4)</td>
<td>0111 0100</td>
<td>11</td>
</tr>
<tr>
<td>(5,13)</td>
<td>0101 1101</td>
<td>16</td>
</tr>
<tr>
<td>(6,8)</td>
<td>0110 1000</td>
<td>13</td>
</tr>
<tr>
<td>(7,4)</td>
<td>0111 0100</td>
<td>11</td>
</tr>
<tr>
<td>(5,13)</td>
<td>0101 1101</td>
<td>16</td>
</tr>
</tbody>
</table>

For the next step the mutation operator is applied to each one of the strings in the matting pool exchanging one bit into a zero or one and finally a stopping criterion is applied. If it is satisfied the process stops or continues starting all over as showed starting with calculating the fitness of the new population. Mutating a pair of strings is obtained Figures 20 and 21:
The procedure could stops or the cycle starts all over if a termination condition is not satisfied.

A summary of a GA procedure is:

1. Choose a string representation scheme for the variable values,
2. Create a population of strings at random,
3. Assign a fitness value to each one of the solutions or strings,
4. Check a termination condition. If it is not satisfied go to the next step. Otherwise end the procedure
5. Modify the population solutions by the main operators: crossover, mutation, selection and go to step 4 to check the termination condition.

For one string or "off-spring" obtained after passing under a complete iteration from its fitness value we can see that instead of improving, the value increased Figure 18.

The result of the complete cycle for string (6, 8) is shown in Table 10.
To avoid that the fitness of the population-best solution does not deteriorate, an elite-preserving operator was introduced to favor the elites of a population giving an opportunity to be carried over the next generations. Rudolph 1996 proved that GAs converge to the global optimal solution of some functions in the presence of elitism (Deb, K., Multi-objective using Evolutionary Algorithms, Wiley 2001).

Some of the more common EAs mentioned in the literature are:

- Strength Pareto Evolutionary Algorithm (SPEA) by Zitzler & Thiele (1998)
- Strength Pareto Evolutionary Algorithm (SPEA 2) by Zitzler (2001)
- Pareto Archived Evolution Strategy (PAES) by Knowles (2000)
- Non-dominated Sorting Genetic Algorithm (NSGA II) by Deb, K., (2001)
- Adaptive Pareto Algorithm (APA). Dumitrescu (2001)

In the next chapter will be shown some methods developed to the further analysis of Pareto optimal sets obtained with Evolutionary algorithms.
Chapter 3

Post Pareto optimality methods

In Chapter 2 were described the two most general approaches known to solve a multiple objective optimization problems: The classic approach and the evolutionary algorithm (EA) approach. In the last approach, a set of non-dominated points or Pareto-optimal set is obtained with the help of evolutionary algorithms. A non-dominated set can be formed by numerous solutions and equally acceptable for selection, but it can be difficult for a decision maker that choice. In this chapter our interest is to present several methods for choosing optimal solutions from a Pareto-optimal set and comment about their advantages and disadvantages.

Post-Pareto Optimality Techniques

Once a Pareto-optimal set is obtained, some decision-maker considerations are used to choose a solution or a subset of trade-off points. Some of the following methods are frequently used in classical multi-objective over the search space optimization.

3.1 Compromise Programming Approach:

This method picks a solution that is located at a minimum distance from a reference point or ideal point which will be named “A”, Yu (1973), Zeleny (1973). Usually it is the vector whose components are the minimum values for each one of the objective functions for a multi-objective optimization problem under consideration i.e., \( A = (f_1^*, f_2^*, ..., f_n^*) \) where \( f_i^* = \min/\max\{f_i: f_i \in \text{Pareto front}\} \) where \( S \) is the search space. Some of the metrics to calculate distances are:

\[
d(f, z) = \left( \sum_{i=1}^{n} |f_i - z_i| \right)^{1/p} \quad l_p \text{ metric}
\]  

(16)
\[ d(f, z) = \max_{\substack{i \in \mathbb{R}^n \setminus \{x^*\} \in \mathbb{R}^n}} \frac{f_i - z_i}{\max_{x \in S} \{f_i - z_i\}} \quad \text{Tchebycheff metric}, \]

\[ f = (f_1, \ldots, f_n) \in \text{Pareto front} \]

Black dots on Figure 23 are optimum points.

\textbf{Figure 23} Ideal point and selected optimal solution

\textbf{3.2 Marginal Rate of Substitution Approach}

The marginal rate of substitution is the amount of improvement in one objective function which can be obtained by sacrificing a unit decrement in any other objective function, Miettinen, k., (1999), the solution having the maximum marginal rate of substitution or "knee point", Branke, \textit{et al.}, (2004) is the one chosen by this method. Since pair-wise comparisons have to be made with all m objectives for each Pareto-Optimal solution, this method may be computationally expensive Deb, K., (2004).

\textbf{Figure 24} Chosen solution or "knee point" with maximum marginal rate of substitution
3.3 Pseudo-Weight Vector Approach

In this approach, a pseudo-weight vector is calculated for each obtained solution.

From the Pareto-Optimal solutions obtained, the weights are calculated as:

\[
 w_i = \frac{\left( f_i^{\text{max}} - f_i \right)}{\sum_{i=1}^m \left( f_i^{\text{max}} - f_i^{\text{min}} \right)}
\]

By definition is clear that \( \sum_{i=1}^m w_i = 1 \) \( \text{(18)} \)

This equation calculates the relative distance of the solution from the worst value in each objective function and finds the solution, as shown in the next figure. The interesting part is that Equation 18 allows a way to compute a relative trade off weight vector for a solution situated in the non convex region of the Pareto-optimal region. Once the weight vectors are calculated, a simple strategy would be to choose the solution closer to a DM preferred Deb, K., (2004).

![Figure 25 Chosen solution for pseudo weight approach](image)

3.4 Non-numerical Ranking Preferences Method

This is a post-Pareto optimality method proposed and developed by Taboada et al., (2006) to obtain a finite sequence of values for a weighting function which is used in a pruning algorithm, that reduces a Pareto-optimal set (previously obtained with an evolutionary algorithm) into an acceptable and manageable size for the decision maker. One example of ranking objective
functions was developed by Carrillo et al., (2011). The non-numerical preferences method ranks the objective functions in a non-numerical way as is shown: Let us assume that, for a decision-maker the objective function $f_1(x)$ is more important than objective $f_2(x)$; the objective $f_2(x)$ is more important than objective $f_3(x)$ and so on for the remaining objective functions, ending up with the non-numerical ranking: $f_1 > f_2 > \cdots > f_n$. The relation "$ > " is a binary relation, and is called preference relation, or preference order. Taking into account the preference order, random weights are generated in the numerical order $w_1 > w_2 > \cdots > w_n$ suggested by the ranked objective functions, and a weight function

$$f_{\text{composite}}(x) = w_1 f_1(x) + w_2 f_2(x) + \cdots + w_n f_n(x),$$

where $\sum_{i=1}^{n} w_i$ and $0 < w_i < 1$ for all $i$ \hspace{1cm} (19)

is created. Note that under the conditions in Equation 19, the composite function becomes a weighted average. This procedure is clearly advantageous because there is no need to provide specific weight values; the only requirement is to provide the preferences ranking of the objective functions. In a broader sense, this method is a pseudo-ranking scheme that accommodates preferences, but it is different from assigning preselected weights or utility functions. The weight values are sampled from the region shown in Figure 21 with the help of a $f_w(x)$ probability density function (p.d.f.) or weight function that will be demonstrated how to obtain it for the three dimensional case.
To obtain the probability density function that generates the ordered weights $w_i$ for the construction of the function in Equation 19 it is assumed to have the form as in Equation 20 where the normalization constant $c$ must be found out.

$$f_w(w_1, w_2, w_3) = \begin{cases} c, & w_i > w_2 > w_3 \\ 0, & \text{otherwise} \end{cases}$$

(20)

where $w_1 + w_2 + w_3 = 1$ and $0 \leq w_i \leq 1$

To obtain the normalization constant $c$ the probability density function (p.d.f.) in Equation 20 can be reduced to a two dimensional case Equation 21 simplifying the calculations.

$$f_w(w_1, w_2) = \begin{cases} c, & w_i > w_2 > 1 - w_i - w_2 \\ 0, & \text{otherwise} \end{cases}$$

(21)

its p.d.f. is Equation 22

$$f_w(w_1, w_2) = \begin{cases} 0, & 0 \leq w_i < \frac{1}{3} \\ 12, & \frac{1}{2} \leq w_i \leq w_2 \leq \frac{1}{2} & \text{and} \frac{1}{2} \leq w_i < \frac{1}{2} \\ 12, & \frac{1}{2} \leq w_i \leq 1 - w_i \leq \frac{1}{2} & \text{and} \frac{1}{2} \leq w_i < 1 \end{cases}$$

(22)

Thus its cumulative distribution function (c.d.f) is Equation 23:
Finally is possible to calculate $w_1$ using the probability integral transformation Equation 24

$$w_1 = F_{w_1}^{-1}(u) = \begin{cases} 0 & , 0 \leq w_1 < \frac{1}{3} \\ \frac{\sqrt{u} + 1}{3} & , 0 \leq u < \frac{1}{4} \\ 1 - \frac{1 - u}{3} & , \frac{1}{4} \leq u < 1 \end{cases}$$

Similarly to calculate $w_2$, the inverse probability transformation of its (c.d.f) is derived to obtain Equation 25

$$w_2 = F_{w_2}^{-1}(w_1) = \begin{cases} 0 & , 0 \leq w_1 < \frac{1}{3} \\ \frac{\left(3w_1 - 1\right)u + 1 - w_1}{2} & , 0 \leq u \leq 1 \quad \frac{1}{3} \leq w_1 < \frac{1}{2} \\ \frac{\left(1 - w_1\right)(u + 1)}{2} & , 0 \leq u \leq 1 \quad \frac{1}{2} \leq w_1 < 1 \end{cases}$$

To finalize the process solve for $w_3$ from the equation $w_1 + w_2 + w_3 = 1$. Once the weights are obtained, the algorithm used to prune Pareto optimal solutions is shown below:

3.4.1 Pseudo code

Step 1. Rank the objective component functions:

$$f_1 > f_2 > \cdots > f_n \Rightarrow w_1 > w_2 > \cdots > w_n$$

Step 2. Convert the original problem into a minimization problem and scale the $f_i$'s

Step 3. Randomly generates weights based on the following scheme:

(i) Randomly generate $u \in [0, 1]$
(ii) Generate \( w_i = F^{-1}_w(u) = \begin{cases} 
0, & \text{otherwise} \\
\frac{\sqrt{u} + 1}{3}, & 0 \leq u < \frac{1}{4} \\
1 - \frac{1-u}{3}, & \frac{1}{4} \leq u < 1 
\end{cases} \)

(iii) Calculate \( w_i = \frac{\sqrt{u} + 1}{3} \) if \( 0 \leq u < \frac{1}{4} \) and get \( w_2 = \frac{(3w_i - 1)u + (1-w_i)}{2} \);

(iv) otherwise calculate \( w_i = 1 - \frac{1-u}{3} \) if \( \frac{1}{4} \leq u \leq 1 \) and get \( w_2 = \frac{(u+1)(1-w_i)}{2} \);

(v) whatever path is selected above calculate, calculate \( w_3 = 1 - w_i - w_2 \).

(vi) Repeat all steps until \( n \) iterations to generate sets of weights \( w_1, w_2 \text{ and } w_3 \).

Step 4. Sum weighted objectives to form:

\[ f_{\text{composite}}(x) = w_1f_1(x) + w_2f_2(x) + \cdots + w_nf_n(x) \]

Step 5. Record the best solution from the weight combination.

Step 6. Repeat Steps 2-5, 10,000 times (user defined) and the best solution for that combination is identified.

The final result is a “pruned” Pareto optimal set. This method has been observed to achieve a 90% reduction of the entire Pareto optimal set, as reported by Coit & Baheranwala (2006).

### 3.5 Pruning using \( k \)-means clustering

This approach is more useful for users that find difficult in specifying any objective function preference. The result is a pruned Pareto set from which the decision maker then only needs to consider \( k \) particular solutions. The \( k \)-means clustering algorithm is well known for its efficiency in clustering data sets. The grouping is done by calculating the centroid for each cluster, and assigning each observation to the group with the closest centroid. A recurrent problem that many clustering algorithms encounter is the choice of the number of clusters.
In order to determine the optimal number of $k$ clusters, Rousseeuw (1987), Rousseeuw et al., (1989) suggested a cluster validity technique, the silhouette plot, to evaluate the quality of a clustering allocation, independently of the clustering technique that is used. An approach to prune Pareto-optimal set was proposed by Taboada & Coit (2006) using this technique.

**Proposed Approach:**

1. Obtain the entire Pareto-optimal set or sub-set of solutions by using a multiple-objective evolutionary algorithm (MOEA) or by another means.
2. Apply the $k$-means algorithm to form clusters on the solutions contained in the Pareto set.
3. To determine the “optimal” number of clusters, $k$, in this set, silhouette plots are used. A value of the silhouette width, $s(i)$, is obtained for several values of $k$. The clustering with the highest average silhouette width is selected as the “optimal” number of clusters in the Pareto-optimal set.
4. For each cluster, select a representative solution. To do this, the solution that is closest to its respective cluster centroid is chosen as a good representative solution.
5. Analyze the results. At this point, the decision-maker can either:
   5.1 Analyze the “knee” cluster. The suggestion is to focus on the cluster that has solutions that conform to the “knee” region. The “knee” (Branke et al., 2004) is formed by those solutions of the Pareto-optimal front where a small improvement in one objective would lead to a large deterioration in at least one other objective. Moreover, from this “knee” cluster the decision maker can select a promising solution for system implementation. This would be the solution closest to the ideal, or utopian solution of the multiple objective problem, in a standardized space.
   5.2 Analyze the $k$ representative solutions and/or select the most promising solutions among this $k$ set, selecting the solution closest to the ideal point.
An advantage of this approach is that the user does not have previously selected the number of cluster since the silhouette plot gives a number of clusters to be selected. It reduced in a 96% a Pareto optimal set; nevertheless more tests must be done to make sure such rate pattern continues.

### 3.6 Greedy Reduction (GR) algorithm

The Greedy Reduction (GR) algorithm is useful for obtaining subsets of large, Pareto-optimal sets. Selection of these subsets is based on maximizing a composite function of the vector of percentile ordinal rankings of the Pareto optima within the larger set. Nevertheless, if the percentile vectors are non-dominated, the Greedy algorithm is not always optimal. On the other hand, the GR algorithm executes in linear time in the worst case.

### 3.7 Local Search with ASF (Achievement Scalarizing function)

*Padhye et al.*, (2009), proposed a mutation driven, hill climbing local search to refine the solutions from a Pareto-optimal set previously obtained with an evolutionary algorithm. Local search usually considers a non-dominated solution and tries to improve it by utilizing a construction of some single objective function. In their study the following achievement scalarizing function (ASF) was used, and a starting point $z = f(y)$ usually known as a reference point for local search.

$$\min\max_{x \in S \subset \mathbb{R}^n} \frac{f_i - z_i}{f_i^{\max} - f_i^{\min}} + \rho \sum_{j=1}^{M} \frac{f_j - z_j}{f_j^{\max} - f_j^{\min}}$$

(26)

By minimizing ASF solutions are projected on the Pareto-front and convergence can be guaranteed. Improvement was obtained, assuring that the solutions found are good estimates for true Pareto-solutions. Improvement was obtained, guaranteeing that the solutions found are good estimates for true Pareto-solutions.
3.8 A Two Stage Approach Pareto Analysis

In 2008, Li et al proposed a two stage approach for selecting optimal solutions from a Pareto optimal set, previously obtained with any of the two NSGA or NSGA-II evolutionary algorithms. At the first stage, Pareto-optimal solutions are classified into several clusters by applying an artificial neural network method called self-organizing map (SOM). In the second stage, non-efficient solutions are eliminated from each cluster, and representative efficient solutions are identified through the data envelopment analysis (DEA) method which is a special multiple objective selection optimization (MOSO) approach. The self-organizing map is a clustering method that has several advantages, compared to the k-means classification method.

Unlike the k-means method that only minimizes the mean squared error, in terms of the Euclidian distance, the SOM approach measures the similarity by the Euclidian distance as well as the angle between the input vectors by updating the weight vectors iteratively. Such training process results in the topological preservation from the input vectors to the output lattice map. Because of those advantages, SOM is utilized in the proposed two-stage approach to classify a Pareto optimal solution set. Once the Pareto set has been clustered with SOM, the data envelopment method (DEA) is applied to reduce the size of each cluster to a manageable collection of optimal points for the decision maker. DEA is a linear programming technique for measuring the relative performance of the decision making units. In a Multi-objective optimization problem, the decision variables can be considered as decision making units. Such performance is measured with relative efficiency formula:

\[
RE = \frac{\text{weighted sum of inputs}}{\text{weighted sum of outputs}}
\]  

For a problem involving \( l \) decision making units, each of which has \( m \) inputs and \( n \) outputs, the relative efficiency of the \( k \)-th DMU can be expressed as:

\[
RE_k = \frac{\text{weighted sum of inputs}_k}{\text{weighted sum of outputs}_k}
\]
where \( u_j \) and \( v_i \) are the weights for the outputs and inputs, respectively.

The strategy proposed for pruning the clusters obtained in the first stage is maximizing the relative efficiency of the DMU’s as shown in formula (26):

$$
\text{max } RE_{k_0} = \sum_{j=1}^{n} u_j y_{jk_0}
$$

subject to

$$
\sum_{i=1}^{m} v_i x_{ik} = 1,
$$

$$
\sum_{j=1}^{n} u_j y_{jk_0} - \sum_{i=1}^{m} v_i x_{ik} \leq 1
$$

$$
u_j, v_i \geq \varepsilon > 0, \quad i = 1, 2, ..., m \quad j = 1, 2, ..., n$$

\( \varepsilon \) is a small positive value. To obtain the performance of the entire set of variables, it is necessary to solve \( n \) times the above linear problem. Finally, the best DMU’s are those with relative efficiency equal to one, and the Pareto optimal vectors with relative efficiency equal to one for all its entries are the selected optimal solutions from the clustered Pareto-optimal set by the SOM method.

### 3.9 A Clustering Method Based on Dynamic Self Organizing Trees for Post-Pareto Optimality Analysis

This algorithm offers two main advantages: there is no need to provide an initial number of clusters, and at each hierarchical level, the algorithm optimizes the number of clusters, and can reassign data from previous hierarchical levels in order to rearrange missed clustered data. The clustering method is the dynamically growing self organizing tree (DGSOT) algorithm. DGSOT is a hierarchical clustering method than has some advantages over other well-known clustering methods. The DGSOT algorithm constructs a hierarchical tree from top to bottom by division. At each hierarchical level, the algorithm optimizes the number of clusters, and can reassign data.
from previous hierarchical levels in order to rearrange non clustered data. Each leaf of the tree represents a cluster; each cluster is a subset of non-dominated solutions from the original set of solutions. Therefore, each leaf in the final tree is a non-dominated solution.

Hierarchical algorithms find successive clusters using previously established clusters. These algorithms usually are either agglomerative ("bottom-up") or divisive ("top-down"). Agglomerative algorithms begin with each element as a separate cluster and merge the element into successively larger clusters. Divisive algorithms begin with the whole set, and proceed to divide it into successively smaller clusters (like DGSOT). A more detailed description of these methods is presented by Fung (2001). Some of the most important disadvantages of both methods are presented in Table 11.

Table 11 Disadvantages for k-means and hierarchical algorithms

<table>
<thead>
<tr>
<th>K-Means</th>
<th>Hierarchical clustering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of k clusters have to be predefined</td>
<td>It does not provide a discrete number of clusters. Clusters have to be defined with cut-offs</td>
</tr>
<tr>
<td>Fixed number of clusters can make it difficult to predict what k should be</td>
<td>Cannot return to previously hierarchical level to reassign misclustered data</td>
</tr>
<tr>
<td>Different initial partitions can result in different final clusters. It is helpful to rerun the program using different values of k to compare results</td>
<td>Selection of split points is critical</td>
</tr>
</tbody>
</table>

3.10 Visualization technique for Pareto Front

Efficient visualization techniques for Pareto Front and Set analyses are needed for helping decision makers in the selection task. The Pareto front supplies a set of solutions where the designer has to look for the best choice according to his preferences. Visualization techniques often play a key role in helping DM, but they have important restrictions for more than two-dimensional Pareto fronts. A graphical representation, called Level Diagrams, for n-dimensional Pareto front analysis was proposed by Blasco et al., (2008). Level Diagrams consists of representing each objective and design parameter on separate diagrams. This new technique is
based on two key points: classification of Pareto front points according to their proximity to ideal points, measured with a specific norm of normalized objectives (several norms can be used); and synchronization of objective and parameter diagrams. Level Diagrams can be colored, so establishing a visual representation of preferences that can help the decision-maker.

The Level Diagrams tool is based on the classification of the Pareto front approximation according to the proximity to the ideal point. For this classification a norm is applied to evaluate the distance to the ideal point. Different norms could be used to obtain different characteristics of the diagrams, the most common are:

(a) Block Norm: \( \|f(x)\|_1 = \sum_{i=1}^{n} |f_i(x)| \)

(b) Euclidean Norm: \( \|f(x)\|_2 = \sum_{i=1}^{n} f_i(x)^2 \)

(c) Maximum Norm: \( \|f(x)\|_\infty = max\{f_i(x) | i = 1, ..., n\} \). Each norm gives a different point of view of the Pareto front shape, for instance: Euclidean norm supply an accurate evaluation of the conventional geometrical distance to the ideal point, and then offer a better view of the true shape. Maximum norm can supply information about the worst objective for a specific point, and is useful for trade-off analysis between different objectives. An increment in this norm directly reveals a worsening of at least one of the objectives. To plot Level diagrams, the points of the Pareto front are sorted in ascending order of the value of the norm of the objective function vector \( \|f(x)\|_x \) whatever norm is used. Afterwards \( f_i(x) \) and decision variable \( x_j \) are graphed against \( \|f(x)\|_x \) . On the vertical axis \( \|f(x)\|_x \) is drawn vs \( f_i(x) \) on the horizontal axis. The same procedure is used to graph \( \|f(x)\|_x \) vs \( x_j \).

The fundamental idea is classification by layers, and synchronous representation of all objectives and parameters. It is shown that this level diagrams representation enables a good analysis of the Pareto front, and so provides an excellent tool to help decision-making.
Chapter 4
Non-numerical ranking preferences method generalization

In Chapter 3, several methods to analyze Pareto-optimal sets were presented; their advantages and disadvantages were discussed as well. The fourth method described in Chapter 3 is the non-numerical ranking preferences method which was originally developed to reduce the size of large Pareto-optimal sets. This method was presented by Taboada et al., (2007) for cases with three and four objective functions. However, authors did not present steps to generalize their method. This method can be applied only when the decision maker can prioritize the relative importance of the objective functions without having to give specific weight values. In this chapter the method to generate weights will be generalized for any number of weights, making this method feasible to be applied in multi-objective optimization problems with any number of objective functions. Pair of examples is presented to show its performance. The strength of this method is precisely that the decision-maker only ranks non-numerically (in order of relative importance) the objective functions but does not have to provide specific weight values. Section 4.1 presents the generalization of the non-numerical preferences method. Examples are presented to show how the method works in the generation of five and seven weights respectively.

4.1 Generalization of the non-numerical preferences method to n number of weights.

The generalization of the non-numerical ranking preferences method was proposed by Carrillo & Taboada, (2012). The method starts with the search of a probability density function which provides the information to obtain the cumulative distribution functions that with the help of the probability transformation theorem each one of the searched weights are obtained.
Iterative method for n weights.

1. Find the normalization constant $c$ for the probability density function (p.d.f.) as shown in Equation 30

$$f(w_1,...,w_n) = \begin{cases} c, & w_1 > w_2 > \cdots > w_n \\ 0, & \text{otherwise} \end{cases} \quad \text{where } \sum_{i=1}^{n} w_i = 1 \text{ and } 0 < w_i < 1 \text{ for all } i$$

$$W = (w_1,...,w_n)$$

2. Get the marginal probability density function

$$f(w_n) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} c \, dw_1 \cdots dw_{n-1}$$

3. Obtain its corresponding cumulative distribution function

$$F_n(w_n) = \int_{-\infty}^{w_n} f_w(s) \, ds \quad \text{And solve for } w_n \text{ in } u = F_n(w_n)$$

4. To calculate $w_{n-1}$ obtain its conditional probability density function

$$f(w_{n-1}|w_n) = \frac{f(w_{n-1},w_n)}{f(w_n)}$$

and from its conditional cumulative distribution function (c.d.f.) $F(w_{n-1}|w_n)$ solve for

$$w_{n-1} = F^{-1}(u|w_n), \quad \text{where } u = F(w_{n-1}|w_n) \text{ is uniformly distributed on } (0,1)$$

5. Continue up to

$$f(w_2|w_3\cdots w_n) = \frac{f(w_2,w_3,\cdots,w_n)}{f(w_3,\cdots,w_n)}$$

and from its conditional distribution

$$F(w_2|w_3\cdots w_n) \quad \text{solve for } \quad w_2 = F^{-1}(u|w_3\cdots w_n) \quad \text{where } \quad u = F(w_2|w_3\cdots w_n) \text{ is uniformly distributed in } (0,1)$$

6. Finally solve for

$$w_1 = 1 - \sum_{i=2}^{n} w_i$$

Remark: The integration order to obtain the weights is determined by the integration region $R$ given by the constraint $w_1 > w_2 > \cdots > w_n, \sum_{i=1}^{n} w_i = 1 \text{ and } 0 < w_i < 1 \text{ for all } i$. Since this region can be restated in several algebraic forms, there are as many normalizing $c$ constants in
Equation 30, as algebraic representations for region \( R \). Therefore numerous probability density functions can be found. Moreover, the constant \( c \) in Equation 30 must have a value that the integral probability theorem can be applied to solve for all of the weights values searched. Hence, it is very unlikely to obtain a general algorithm to produce any desired number of weights. For this reason interested users must develop each algorithm for each case as shown in the next section for the case of five weights.

4.2 Generalization of the non-numerical ranking preferences method: 5 weights example

Equation 33 shows the probability density function to be calculated for this case.

\[
f(W) = \begin{cases} 
  c, & w_i > w_j > w_k > w_l > w_m \\
  0, & \text{otherwise}
\end{cases} \\
p.d.f.
\]

where \( w_i + w_j + w_k + w_l + w_m = 1 \) and \( 0 \leq w_i \leq 1 \)
and \( W = (w_i, w_j, w_k, w_l, w_m) \)

One algebraic expression for the set

\[
w_i > w_j > w_k > w_l > w_m \quad \text{where} \quad \sum_{i=1}^{5} w_i = 1 \quad \text{and} \quad 1 > w_i > 0
\]

is the region shown in Equation 34.

\[
0 < w_3 < \frac{1}{5} ; w_5 < w_4 < \frac{1-w_5}{4}
\]

\[
\frac{1}{2}(1-2w_5-w_3) \leq w_3 < 1-3w_4 - w_5
\]

\[
\frac{1}{2}(1- w_1 - w_3 - w_5) < w_2 < 1 - w_1 - 2w_4 - w_5
\]

\[
w_3 = 1 - w_1 - w_2 - w_4 - w_5
\]

To find the normalization constant \( c \) let us calculate the multiple integral over the region described in Equation 34.
\[
\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f_w(w_1, w_2, w_4) dw_1 dw_2 dw_4 dw_5 = \\
\int_{0}^{1/5} \int_{w_3}^{1-3w_4-w_3} \int_{1/2(1-w_4-w_5)}^{1-2w_4-w_5} \int_{1/2(1-w_4-w_5)}^{1} 1 dw_2 dw_4 dw_5 dw_3 = \frac{1}{3840}
\]

Thus \( c = 3840 \)

Therefore, its joint probability density function can be written as Equation 35:

\[
f(w_1, w_2, w_4, w_5) = \begin{cases} 
0 < w_5 < \frac{1}{5}, & w_5 < w_4 < \frac{1-w_5}{4} \\
3840, \quad \frac{1}{2}(1-2w_4-w_5) \leq w_1 < 1-3w_4-w_5 \\
\frac{1}{2}(1-w_1-w_4-w_5) < w_2 < 1-w_1-2w_4-w_5 \\
0, \quad otherwise
\end{cases}
\]

To obtain the weights in increasing order starting with the lowest weight value \( w_5 \), its marginal probability density function must be calculated.

The marginal p.d.f. of \( f_{w_5} \) is \( f_{w_5}(w_5) = \int_{w_3}^{1-3w_4-w_3} \int_{1/2(1-w_4-w_5)}^{1-2w_4-w_5} \int_{1/2(1-w_4-w_5)}^{1} 3840 dw_2 dw_4 dw_3. \)

Which is shown in Equation 36:

\[
f_{w_5}(w_5) = \begin{cases} 
20(1-5w_5)^3 , & 0 < w_5 < \frac{1}{5} \\
0 , & otherwise
\end{cases}
\]

To generate \( w_5 \) its cumulative distribution function (c.d.f.) is calculated to create an equation for solving for \( w_5 \), as is shown in Equation 37.

The \( w_5 \) corresponding c.d.f. is shown in Equation 37:

\[
F_{w_5}(w_5) = \int_{-\infty}^{w_5} f_{w_5}(s) ds = \begin{cases} 
1-(1-5w_5)^4 , & 0 < w_5 < \frac{1}{5} \\
0 , & otherwise
\end{cases}
\]
Applying the probability integral transformation theorem (DeGroot, M., 1985) to solve for $w_3$ from Equation 37:

$$w_3 = \frac{1-(1-u)^{\frac{1}{3}}}{5} = F^{-1}_{w_3}(u) \text{ where } u \text{ is a uniform random variable } U \sim (0,1) \quad (38)$$

To generate the weight $w_4$, the joint p.d.f $f(w_4, w_5) = 240(4w+m-1)^2$ is derived. Afterwards the conditional probability density function of $w_4$ given $w_5$ is derived to calculate the conditional cumulative distribution function for $w_4$ given $w_5$. In Equation 39 is shown the first conditional p.d.f obtained.

$$f(w_4|w_5) = \frac{f(w_4, w_5)}{f(w_5)} \text{ which is }$$

$$f(w_4|w_5) = \begin{cases} 
\frac{12(4w_4 + w_5 - 1)^2}{(1-5w_5)^3}, & 0 < w_5 < \frac{1}{5}, \quad w_5 < w_4 < \frac{1-w_5}{4} \\
0, & \text{otherwise} 
\end{cases} \quad (39)$$

Afterwards the cumulative distribution function for $w_4$ given $w_5$ is presented in Equation 40.

$$F(w_4|w_5) = \begin{cases} 
1+\frac{(4w_4 + w_5 - 1)^3}{(1-5w_5)^3}, & 0 < w_5 < \frac{1}{5} \\
\frac{w_5 - w_4}{4}, & w_5 < w_4 < \frac{1-w_5}{4} \\
0, & \text{otherwise} 
\end{cases} \quad (40)$$

Thus $w_4 = \frac{u-1}{4}(1-5w_5) + w_5 = F^{-1}_{w_4|w_5}(u)$ where $u$ is a uniform random variable $U \sim (0,1)$

To calculate $w_1$, we need to obtain its conditional probability density function given $w_3$ and $w_4$ as shown in Equation 41.
\[
f(w_1|w_2, w_5) = \frac{f(w_1, w_2, w_5)}{f(w_2, w_5)}
\]

\[
f(w_1, w_2, w_5) = \begin{cases} 
1920(1-w_1^3w_4-w_5), & 0 < w_5 < \frac{1}{5}, \quad w_5 < w_2 < \frac{1-w_5}{4} \\
\frac{1}{2}(1-2w_1-w_5) \leq w_1 < 1-3w_4-w_5, & \text{otherwise}
\end{cases}
\]

\[
f(w_1|w_2, w_5) = \begin{cases} 
\frac{8(1-w_1^3w_4-w_5)}{(4w_1+m-1)^2}, & 0 < w_5 < \frac{1}{5}, \quad w_5 < w_2 < \frac{1-w_5}{4} \\
\frac{1}{2}(1-2w_1-w_5) \leq w_1 < 1-3w_4-w_5, & \text{otherwise}
\end{cases}
\]

The corresponding cumulative distribution function is Equation 42:

\[
F_{w_1}(w_1|w_2, w_5) = \begin{cases} 
\frac{1-(1-w_1^3w_4-w_5)^2}{(4w_1+m-1)^2}, & 0 < w_5 < \frac{1}{5}, \quad w_5 < w_2 < \frac{1-w_5}{4} \\
\frac{1}{2}(1-2w_1-w_5) \leq w_1 < 1-3w_4-w_5, & \text{otherwise}
\end{cases}
\]

By the probability integral transformation theorem, \( u = F_{w_1}(w_1|w_2, w_5) \) is a uniform random variable on \((0,1)\), thus solving for \( w_1 \) we obtain Equation 43

\[
w_1 = F_{w_1|w_2, w_5}^{-1}(u) = 1-w_5-3w_4-(4w_4+w_5-1)\sqrt{1-u}
\]

To find \( w_2 \) the conditional probability density function is calculated as shown in Equation 43.

\[
f(w_2|w_1, w_4, w_5) = \frac{f(w_1, w_2, w_4, w_5)}{f(w_1, w_4, w_5)}
\]
And the conditional probability density function given \( w_1, w_4, w_5 \) is shown in Equation 45.

\[
f (w_2 | w_1, w_4, w_5) = \begin{cases} 
2 & \frac{1}{2} (1 - 2w_4 - w_5) \leq w_1 < 1 - 3w_4 - w_5 \\
0 & \text{otherwise} 
\end{cases}
\]

Finally its conditional cumulative distribution function is given in Equation 46.

\[
F \left( w_2 \bigg| w_1, w_4, w_5 \right) = \int_{\frac{1}{2}(1-w_1-w_4-w_5)}^{1} \frac{2ds}{(1-w_1-w_5-3w_4)} 
\]

Solving for \( w_2 \) from Equation 46,

\[
u = \frac{1-w_1-2w_2-w_4-w_5}{w_1+3w_4+w_5-1}
\]

obtaining:

\[
w_2 = \frac{(u+1)(1-w_1-w_4-w_5)-2uw_4}{2}
\]

The last weight \( w_3 \) is solved from equation \( w_1 + w_2 + w_3 + w_4 + w_5 = 1 \).

Finally the five weights algorithm can be written in the following pseudo code:

\[ \text{4.2.1 Pseudo code for the non-numerical ranking preferences method for five weights} \]

1) Randomly generate an \( u \in (0, 1) \)
2) Compute \( w_3 = \frac{1-(1-u)^{\frac{1}{3}}}{5} \)

3) Randomly generate another \( u \in (0,1) \)

\[
\text{Compute } w_4 = \sqrt[3]{u-1}(1-5w_3)^3 - m + 1
\]

4) Randomly generate another \( u \in (0,1) \)

\[
\text{compute } w_i = 1-w_5-3w_4-(4w_4+w_5-1)\sqrt{1-u}
\]

5) Randomly generate another \( u \in (0,1) \)

\[
\text{Calculate } w_2 = \frac{(u+1)(1-w_i-w_3-w_4)-2uw_4}{2}
\]

6) Calculate \( w_3 = 1-w_1-w_2-w_4-w_3 \)

7) Repeat all steps until \( n \) iterations to generate an \( n \)-weights set of \( w_1, w_2, w_3, w_4 \) and \( w_5 \).

**Example 4.1**

The Pareto set to test the non-numerical ranking preferences generalized algorithm was obtained from a car-sales file provided by the SPSS software program; which consist of 115 cars brand with nine features: vehicle type, price in thousands, engine size, horsepower, fuel capacity, fuel efficiency, length, width and curb weight. For the Pareto analysis the first six of the nine features where renamed as: car, price, engine size, horsepower, fuel capacity and miles per gallon. The original file was reduced to 42 non-dominated values or Pareto optimal solutions, for the problem of minimizing: car price(price), engine size (engines) and maximizing: horsepower (horsepw), fuel capacity(fuelcap) and miles per gallon(m.p.g) using the non-numerical ranking.
preferences method for 5 objective functions and utilizing three different threshold values for the composite function below.

\[ f_{\text{composite}} = w_1 f_1(x) + \cdots + w_5 f_5(x) < \alpha \text{ threshold} \]

\[ w_1 > w_2 > w_3 > w_4 > w_5 \text{ where } \sum_{i=1}^{5} w_i = 1 \text{ and } 1 > w_i > 0 \]

The results obtained are shown in Tables 12, 13 and 14.

**Table 12** Reduced Pareto set with 5 weight and \( \alpha=0.2 \)

<table>
<thead>
<tr>
<th>car</th>
<th>pruned for alpha=0.2</th>
<th>price</th>
<th>engines</th>
<th>horsepw</th>
<th>fuelcap</th>
<th>mpg</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>Chevrolet Metro</td>
<td>9.235</td>
<td>1</td>
<td>55</td>
<td>10.3</td>
<td>45</td>
</tr>
</tbody>
</table>

**Table 13** Reduced Pareto set with 5 weight and \( \alpha=0.3 \)

<table>
<thead>
<tr>
<th>car</th>
<th>pruned for alpha=0.3</th>
<th>price</th>
<th>engines</th>
<th>horsepw</th>
<th>fuelcap</th>
<th>mpg</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>Chevrolet Metro</td>
<td>9.235</td>
<td>1</td>
<td>55</td>
<td>10.3</td>
<td>45</td>
</tr>
<tr>
<td>44</td>
<td>Honda Civic</td>
<td>12.885</td>
<td>1.6</td>
<td>106</td>
<td>11.9</td>
<td>32</td>
</tr>
<tr>
<td>46</td>
<td>Hyundai Accent</td>
<td>9.699</td>
<td>1.5</td>
<td>92</td>
<td>11.9</td>
<td>31</td>
</tr>
<tr>
<td>74</td>
<td>Nissan Sentra</td>
<td>13.499</td>
<td>1.8</td>
<td>126</td>
<td>13.2</td>
<td>30</td>
</tr>
<tr>
<td>94</td>
<td>Saturn SL</td>
<td>10.685</td>
<td>1.9</td>
<td>100</td>
<td>12.1</td>
<td>33</td>
</tr>
<tr>
<td>100</td>
<td>Toyota Corolla</td>
<td>13.108</td>
<td>1.8</td>
<td>120</td>
<td>13.2</td>
<td>33</td>
</tr>
</tbody>
</table>

**Figure 27** Graph of pruned values in table 11
Table 14 Reduced Pareto set with 5 weight and $\alpha=0.4$

<table>
<thead>
<tr>
<th>car</th>
<th>pruned for alpha=0.4</th>
<th>price</th>
<th>engines</th>
<th>mpg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chevrolet Malibu</td>
<td>16.535</td>
<td>3.1</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>Chevrolet Metro</td>
<td>9.235</td>
<td>1</td>
<td>45</td>
<td></td>
</tr>
<tr>
<td>Chrysler Concorde</td>
<td>22.245</td>
<td>2.7</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>Dodge Intrepid</td>
<td>22.505</td>
<td>2.7</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>Ford Contour</td>
<td>17.035</td>
<td>2.5</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>Honda Civic</td>
<td>12.885</td>
<td>1.6</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>Hyundai Accent</td>
<td>9.699</td>
<td>1.5</td>
<td>31</td>
<td></td>
</tr>
<tr>
<td>Hyundai Elantra</td>
<td>11.799</td>
<td>2</td>
<td>27</td>
<td></td>
</tr>
<tr>
<td>Hyundai Sonata</td>
<td>14.999</td>
<td>2.4</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>Mitsubishi Eclipse</td>
<td>19.047</td>
<td>2.4</td>
<td>24</td>
<td></td>
</tr>
<tr>
<td>Nissan Sentra</td>
<td>20.39</td>
<td>2.4</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>Nissan Altima</td>
<td>26.249</td>
<td>3</td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>Oldsmobile Alero</td>
<td>18.27</td>
<td>2.4</td>
<td>27</td>
<td></td>
</tr>
<tr>
<td>Saturn SL</td>
<td>10.685</td>
<td>1.9</td>
<td>33</td>
<td></td>
</tr>
<tr>
<td>Toyota Corolla</td>
<td>13.108</td>
<td>1.8</td>
<td>33</td>
<td></td>
</tr>
<tr>
<td>Toyota Celica</td>
<td>16.875</td>
<td>1.8</td>
<td>31</td>
<td></td>
</tr>
<tr>
<td>Volkswagen Passat</td>
<td>21.2</td>
<td>1.8</td>
<td>27</td>
<td></td>
</tr>
<tr>
<td>Volvo S40</td>
<td>23.4</td>
<td>1.9</td>
<td>25</td>
<td></td>
</tr>
</tbody>
</table>

Figure 28 Graph of pruned values in table 12

According to the results shown in Table 1, the pruned solution is the most inexpensive car that provides the best mileages per gallon, the smallest engine, minimum horsepower and minimum fuel capacity. Tables 13 and 14 show a more ample variety of solutions that the decision maker can analyze.
4.3 Development of a seven weights algorithm

In a similar form, as in the previous section, a probability density function for the seven weights case is derived and demonstrated. Let \( f(W) \) be the probability density function we are interested to find as shown in Equation 48.

\[
    f(W) = \begin{cases} 
    c, & w_i > w_2 > w_3 > w_4 > w_5 > w_6 > w_7 \\
    0, & \text{otherwise} 
    \end{cases} 
\]

where \( w_1 + w_2 + w_3 + w_4 + w_5 + w_6 + w_7 = 1 \) and \( 0 < w_i < 1 \); \( W = (w_1, w_2, w_3, w_4, w_5, w_6, w_7) \)

To find out the constant \( c \), Equation 48 can be reduced to a six weight case solving the seventh weight from the equality \( \sum_{i=1}^{7} w_i = 1 \). Primarily an algebraic expression for the integration region \( R \) formed by the constraint \( w_i > w_2 > w_3 > w_4 > w_5 > w_6 > w_7 \) where \( \sum_{i=1}^{7} w_i = 1 \) must be obtained to integrate and calculate the \( c \) value or normalization constant as is called in probabilistic terms

Once the integration region \( R \) is restated as the set of weights \( w_i \) that satisfy \( w_1 > w_2 > w_3 > w_4 > w_5 > w_6 > 1 - \sum_{i=1}^{7} w_i = 1 \) one region satisfying the last inequalities is shown in Equation 49.

\[
    \frac{1}{7} < w_i \leq \frac{1}{6} ; \quad \frac{1-w_i}{6} < w_2 < w_i ; \quad \frac{1}{4} \left(1-w_i-w_2\right) < w_3 < w_2 \\
    \frac{1}{4} \left(1-w_1-w_2-w_3\right) < w_4 < w_3 ; \quad \frac{1}{3} \left(1-w_1-w_2-w_3-w_4\right) < w_5 < w_4 \\
    \frac{1}{2} \left(1-w_1-w_2-w_3-w_4-w_5\right) < w_6 < w_5 \\
    w_7 = 1 - w_1 - w_2 - w_3 - w_4 - w_5 - w_6
\]

To find the normalization constant \( c \), integrate the restated p.d.f in Equation 49 for six weights
\[
\int_{\gamma} \int_{\gamma} \int_{\gamma} \int_{\gamma} f_W(W) \, dw_6, dw_5, dw_4, dw_3, dw_1 = \int_{\gamma} \int_{\gamma} \int_{\gamma} \int_{\gamma} c \, dw_6, dw_5, dw_4, dw_3, dw_1 = \frac{c}{90720(6)^6} = 1
\]

Thus the normalization constant is \( c = 90720(6)^6 \)

Therefore the six weights restated p.d.f. can be written as Equation 50.

\[
f(w_1, w_2, w_3, w_4, w_5, w_6) = \begin{cases} 
\frac{1}{7} < w_1 \leq \frac{1}{6} & \quad \frac{1}{4} < w_2 < \frac{1}{3} (1 - w_1 - w_2) < w_3 < w_2 \\
\frac{1}{4} (1 - w_1 - w_2 - w_3) < w_4 < \frac{1}{3} (1 - w_1 - w_2 - w_3 - w_4) < w_5 < w_4 \\
\frac{1}{2} (1 - w_1 - w_2 - w_3 - w_4 - w_5) < w_6 < w_5 \\
0, & \quad \text{otherwise}
\end{cases}
\]

To generate the first weight \( w_1 \), its marginal probability density function and cumulative distribution functions must be calculated.

The marginal p.d.f. of \( f_{w_1} \) is

\[
f(w_1) = \int_{\gamma} \int_{\gamma} \int_{\gamma} \int_{\gamma} c \, dw_6, dw_5, dw_4, dw_3, dw_1 = \frac{c}{2160(7w_1 - 1)^5}, \frac{1}{7} \leq w_1 \leq \frac{1}{6}
\]

The p.d.f. is:

\[
f(w_1) = \begin{cases} 
\frac{c}{2160(7w_1 - 1)^5}, & \quad \frac{1}{7} \leq w_1 \leq \frac{1}{6} \\
0, & \quad \text{otherwise}
\end{cases}
\]

And its corresponding c.d.f is Equation 52

\[
F(w_1) = \int_{-\infty}^{w_1} f_{w_1}(s) \, ds = \begin{cases} 
\frac{c}{6(7)(2160)} (7w_1 - 1)^5, & \frac{1}{7} \leq w_1 \leq \frac{1}{6} \\
0, & \quad \text{otherwise}
\end{cases}
\]

By the probability Integral transformation theorem, states that

"\( F(w_1) = u \) is a uniform random variable \( U \sim (0,1) \)".

Solving for \( w_1 \) the first weight is obtained as is shown in Equation 53.
\[ w_i = \frac{6 + \sqrt[4]{u}}{42} = F_{w_i}^{-1}(u) \]  

(53)

To generate \( w_2 \), the marginal p.d.f for \( w_1, w_2 \) is

\[ f(w_1, w_2) = \frac{c(6w_2 + w_1 - 1)^4}{360} \]

To calculate \( w_2 \) we need to obtain its conditional p.d.f given \( w_1 \) Equation 54

\[ f(w_2 | w_1) = \frac{f(w_1, w_2)}{f(w_1)} \text{ which is} 
\]

\[ f(w_2 | w_1) = \begin{cases} 
\frac{c(6w_2 + w_1 - 1)^4}{(7w_1 - 1)^3}, & \frac{1}{7} < w_1 \leq \frac{1}{6} : \frac{1-w_1}{6} < w_2 < w_1 \\
0, & \text{otherwise}
\end{cases} \]  

(54)

The \( w_2 \) cumulative distribution given that \( w_1 \) is shown in Equation 55.

\[ F(w_2 | w_1) = \begin{cases} 
\frac{(6w_2 + w_1 - 1)^5}{5(7w_1 - 1)^3}, & \frac{1}{7} < w_1 \leq \frac{1}{6} : \frac{1-w_1}{6} < w_2 < w_1 \\
0, & \text{otherwise}
\end{cases} \]  

(55)

Thus \( w_2 = \frac{1-5w_1 + (7w_1 - 1)\sqrt[4]{5u}}{6} = F_{w_2 | w_1}^{-1}(u) \) where \( u \) is a uniform random variable \( U \sim (0,1) \)

To calculate \( w_3 \), is need to obtain its conditional probability density function given \( w_1 \) and \( w_2 \) as in Equation 56
\[
f(w_1|w_2) = \frac{f(w_1, w_2)}{f(w_2)}\]

\[
f(w_1, w_2, w_3) = \begin{cases} 
\frac{c}{18}(5w_3 + w_1 + w_2 - 1)^3 & : \frac{1}{7} < w_1 \leq \frac{1}{6} ; \frac{1}{6} < w_2 < w_1 \\
: \frac{1}{4}(1-w_1 - w_2) < w_3 < w_2 \\
0 & \text{otherwise}
\end{cases}
\]

The corresponding cumulative distribution function is shown in Equation 57.

\[
F(w_3|w_1, w_2) = \begin{cases} 
\frac{1}{7} < w_1 \leq \frac{1}{6} ; \frac{1}{6} < w_2 < w_1 & \\
: \frac{1}{4}(1-w_1 - w_2) < w_3 < w_2 \\
0 & \text{otherwise}
\end{cases}
\]

By the inverse probability transformation \( u = F(w_3|w_1, w_2) \) is a uniform random variable on \((0,1)\), thus solving for \( w_4 \) we obtain Equation 58.

\[
w_3 = F^{-1}_{w_3|w_1, w_2}(u) = \frac{1-w_1 - w_2 + u(6w_2 + w_1 - 1)}{5}
\]

The next weight to find is \( w_4 \). Its p.d.f. is shown in Equation 59

\[
f(w_4|w_1, w_2, w_3) = \frac{f(w_1, w_2, w_3, w_4)}{f(w_1, w_2, w_3)} = \begin{cases} 
\frac{3(4w_4 + w_1 + w_2 + w_3 - 1)^3}{(5w_3 + w_2 + w_1 - 1)^3} & : \frac{1}{7} < w_1 \leq \frac{1}{6} ; \frac{1}{6} < w_2 < w_1 \\
: \frac{1}{4}(1-w_1 - w_2) < w_3 < w_2 \\
0 & \text{otherwise}
\end{cases}
\]
Finally the corresponding cumulative distribution function is shown in Equation 60.

\[
F(w_4 | w_1, w_2, w_3) = \begin{cases} 
\frac{(4w_4 + w_1 + w_2 + w_3 - 1)^3}{4(5w_3 + w_2 + w_1 - 1)^3}, & \frac{1}{7} < w_i \leq \frac{1}{6}; \frac{1-w_i}{6} < w_2 < w_i \\
\frac{1}{4}(1-w_i-w_2) < w_3 < w_2 \\
\frac{1}{4}(1-w_i-w_2-w_3) < w_4 < w_3 \\
0, & \text{otherwise}
\end{cases}
\] (60)

Solving for \(w_4\) from Equation 61, \(u = \frac{(4w_4 + w_1 + w_2 + w_3 - 1)^3}{4(5w_3 + w_2 + w_1 - 1)^3}\) (61)

The solution is \(w_4 = \frac{(1-w_i-w_2-w_3)+\sqrt[4]{4u(5w_3 + w_2 + w_1 - 1)}}{4}\) (62)

The next weight to find is \(w_5\), calculate its conditional probability function given that four previous weights are known. In Equation 63 is written the conditional probability distribution.

\[
f(w_5 | w_1, w_2, w_3, w_4) = \frac{f(w_1, w_2, w_3, w_4, w_5)}{f(w_1, w_2, w_3, w_4)} = \begin{cases} 
\frac{3(3w_4 + w_1 + w_2 + w_3 - 1)}{(4w_4 + w_3 + w_2 + w_1 - 1)^2}, & \frac{1}{7} < w_i \leq \frac{1}{6}; \frac{1-w_i}{6} < w_2 < w_i \\
\frac{1}{4}(1-w_i-w_2) < w_3 < w_2 \\
\frac{1}{4}(1-w_i-w_2-w_3) < w_4 < w_3 \\
\frac{1}{3}(1-w_i-w_2-w_3-w_4) < w_5 < w_4 \\
0, & \text{otherwise}
\end{cases}
\] (63)

Thus its cumulative distribution function is in Equation 64:
Solving for \( w_5 \) in the Equation 65,

\[
u = \frac{(3w_5 + w_4 + w_3 + w_2 + w_1 - 1)^2}{2(4w_4 + w_3 + w_2 + w_1 - 1)^2}\]  

(65)

the fifth weight is shown in Equation 66

\[
w_5 = \frac{\sqrt{2u (4w_4 + w_3 + w_2 + w_1 - 1) + 1 - w_4 - w_3 - w_2 - w_1}}{3}\]  

(66)

The next step is to find weight \( w_6 \). Similarly to last procedure its conditional cumulative probability function is shown in Equation 67:

\[
f\left(w_6 | w_1, w_2, w_3, w_4, w_5\right) = \frac{f\left(w_1, w_2, w_3, w_4, w_5, w_6\right)}{f\left(w_1, w_2, w_3, w_4, w_5\right)} \times \begin{cases} 1 & \frac{1}{7} < w_3 \leq \frac{1}{6}; \frac{1-w_1}{6} < w_2 < w_3; \\
1/4 (1-w_1-w_2) < w_2 < w_3; \\
1/4 (1-w_1-w_2-w_3) < w_4 < w_3; \\
1/3 (1-w_1-w_2-w_3-w_4) < w_5 < w_4; \\
1/2 (1-w_1-w_2-w_3-w_4-w_5) < w_6 < w_5; \\
0, \text{ otherwise} \end{cases}\]  

(67)
Its cumulative distribution function is shown in Equation 68.

\[
F(\{w_1, w_2, w_3, w_4, w_5, w_6\}) = \begin{cases} 
\frac{1}{7} < w_i \leq \frac{1}{6} ; & \frac{1}{6} < w_j < w_i; \\
\frac{1}{4}(1-w_i-w_j) < w_j < w_i \\
\frac{1}{3}(1-w_i-w_j-w_k) < w_k < w_j \\
\frac{1}{2}(1-w_i-w_j-w_k-w_l) < w_l < w_k \\
0, & \text{otherwise}
\end{cases}
\]  

(68)

, to conclude solving for \( w_6 \) in the Equation 69

\[
u = \frac{(2w_6 + w_2 + w_3 + w_4 + w_5 - 1)}{(3w_3 + w_4 + w_5 + w_6 - 1)}
\]  

(69)

we get: \( w_6 = \frac{2nu_2 + (u_3 + u_4 + u_5 + u_6 - 1)(u - 1)}{2} \), and the last weight is obtained from: \( w_7 = 1 - w_1 - w_2 - w_3 - w_4 - w_5 - w_6 \)

### 4.3.1 Seven Weights algorithm Pseudo code

Finally the algorithm’s pseudo code to randomly generate the collection of seven weights \( w_1, w_2, w_3, w_4, w_5, w_6 \ and w_7 \) is:

1) Randomly generate an \( u \in (0,1) \)

2) Compute \( w_i = 6 + \frac{4\sqrt{u}}{42} \)

3) Compute
\[ w_2 = \frac{1 - 5w_1 + (7w_1 - 1)\sqrt{5u}}{6} \]

4) Compute
\[ w_3 = \frac{1 - w_1 - w_2 + u(6w_2 + w_1 - 1)}{5} \]

5) Compute
\[ w_4 = \frac{(1 - w_1 - w_2 - w_3) + \sqrt{4u}(5w_3 + w_2 + w_1 - 1)}{4} \]

6) Compute
\[ w_5 = \frac{\sqrt{2u}(4w_4 + w_3 + w_2 + w_1 - 1) + 1 - w_4 - w_3 - w_2 - w_1}{3} \]

7) Compute
\[ w_6 = \frac{2u w_5 + (w_4 + w_3 + w_2 + w_1 - 1)(u - 1)}{2} \]

Finally calculate
\[ w_7 = 1 - w_1 - w_2 - w_3 - w_4 - w_5 - w_6 \]

8) Repeat all steps until \( n \) iterations to generate a weights set of \( w_1, w_2, w_3, w_4, w_5 \) and \( w_6 \).

**Example 4.2**

As in the first Example 4.1 for the five weights case, the same cars-sale file will be tested adding two additional features, width and curb weight. The pruned subsets for three threshold values are:

**Table 15 Reduced Pareto set with 7 weights and \( \alpha=0.2 \)**

<table>
<thead>
<tr>
<th>Car</th>
<th>price</th>
<th>engines</th>
<th>horsepower</th>
<th>fuelcap</th>
<th>mpg</th>
<th>width</th>
<th>curb_wgt</th>
</tr>
</thead>
<tbody>
<tr>
<td>25 Chevrolet Metro</td>
<td>9.235</td>
<td>1</td>
<td>55</td>
<td>10.3</td>
<td>45</td>
<td>62.6</td>
<td>1.895</td>
</tr>
</tbody>
</table>

Again the cheapest and most gas saving car is the Chevy metro

**Table 16 Reduced Pareto set with 7 weights and \( \alpha=0.3 \)**

<table>
<thead>
<tr>
<th>Car</th>
<th>price</th>
<th>engines</th>
<th>horsepower</th>
<th>fuelcap</th>
<th>mpg</th>
<th>width</th>
<th>curb_wgt</th>
</tr>
</thead>
<tbody>
<tr>
<td>25 Chevrolet Metro</td>
<td>9.235</td>
<td>1</td>
<td>55</td>
<td>10.3</td>
<td>45</td>
<td>62.6</td>
<td>1.895</td>
</tr>
<tr>
<td>46 Hyundai Accent</td>
<td>9.699</td>
<td>1.5</td>
<td>92</td>
<td>11.9</td>
<td>31</td>
<td>65.7</td>
<td>2.24</td>
</tr>
</tbody>
</table>
For the second threshold value $\alpha=0.3$ there is a difference in the number of cars compared with the five weights case such that simplifies the decision maker car selection. Finally for the last threshold value $\alpha=0.4$ a considerable car reduction is observed comparing to the five case.

Table 17 Reduced Pareto set with 7 weights and $\alpha=0.4$

<table>
<thead>
<tr>
<th>car</th>
<th>price</th>
<th>engines</th>
<th>horsepow</th>
<th>fuelcap</th>
<th>mpg</th>
<th>width</th>
<th>curb_wgt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chevrolet</td>
<td>Metro</td>
<td>9.235</td>
<td>1</td>
<td>55</td>
<td>10.3</td>
<td>45</td>
<td>62.6</td>
</tr>
<tr>
<td>Honda</td>
<td>Civic</td>
<td>12.885</td>
<td>1.6</td>
<td>106</td>
<td>11.9</td>
<td>32</td>
<td>67.1</td>
</tr>
<tr>
<td>Hyundai</td>
<td>Accent</td>
<td>9.699</td>
<td>1.5</td>
<td>92</td>
<td>11.9</td>
<td>31</td>
<td>65.7</td>
</tr>
<tr>
<td>Hyundai</td>
<td>Elantra</td>
<td>11.799</td>
<td>2</td>
<td>140</td>
<td>14.5</td>
<td>27</td>
<td>66.9</td>
</tr>
<tr>
<td>Nissan</td>
<td>Sentra</td>
<td>13.499</td>
<td>1.8</td>
<td>126</td>
<td>13.2</td>
<td>30</td>
<td>67.3</td>
</tr>
<tr>
<td>Saab</td>
<td>3-Sep</td>
<td>12.535</td>
<td>1.9</td>
<td>100</td>
<td>12.1</td>
<td>33</td>
<td>66.4</td>
</tr>
<tr>
<td>Saturn</td>
<td>SL</td>
<td>10.605</td>
<td>1.9</td>
<td>100</td>
<td>12.1</td>
<td>33</td>
<td>66.4</td>
</tr>
<tr>
<td>Toyota</td>
<td>Celica</td>
<td>16.875</td>
<td>1.8</td>
<td>140</td>
<td>14.5</td>
<td>31</td>
<td>68.3</td>
</tr>
</tbody>
</table>
Conclusions: This weight generator method has the advantage of providing the decision maker a linear computational time complexity algorithm for the ranked weights search. However the larger the number of objective functions in a non-numerical ranking multi-objective optimization problem, the larger the mathematical complexity to obtain the algorithm to generate the weights required. This posed a new challenge: The creation of weight generators with a simple algorithmic structure. Fortunately such goal was accomplished; two weight generators were developed. Both methods will be demonstrated in the following two chapters. In the next chapter the first weight generator is proposed and developed by Carrillo & Taboada, (2012) and applied in a Pareto analysis.
Chapter 5

A post-Pareto approach using a non-uniform weight generator method

So far there exist two general approaches to solve multiple objective problems. The first approach involves the aggregation of all the objective functions into a single composite objective function. Mathematical methods such as the weighted sum method, goal programming, or utility functions are methods that pertain to this general approach. The output of this method is a single solution. On the other hand, there is the multiple objective evolutionary algorithm approach that offer the decision maker a set of trade off solutions usually called non dominated solutions or, Pareto-optimal solutions. This set is usually very large and the decision maker faces the problem of reducing the size of this set to have a manageable number of solutions to analyze. This chapter presents an additional post-Pareto approach to prune the non-dominated set of solutions obtained by multiple objective evolutionary algorithms due to Carrillo & Taboada (2012). The proposed approach uses a non-uniform weight generator method to reduce the size of the Pareto-optimal set. This weight generator is the first one of the two mentioned at the end of chapter 4. Its development starts in section 5.1. In section 5.2 the algorithm pseudo code is presented followed by an example to show the performance of the method.

5.1 A general method to create n-increasingly ordered non-uniform distributed random numbers

In this section is developed the algorithm that generates n uniformly distributed weights that are the basis for the non-numerical ranking weights needed for the scalar function

\[ f_{\text{composite}} = w_1 f_1 + w_2 f_2 + w_3 f_3. \]
To obtain the weights, first a finite sequence of $0 < x_1 < x_2 < \cdots < x_n < n$ positive values is produced from $X_1, \ldots, X_n$ random variables uniformly distributed $X_i \sim U(i-1,i)$; subsequently the ratio $w_k = \frac{x_k}{M}$ for $k = 1, \ldots, n$ is calculated where $M = \sum_{i=1}^{n} x_i$. Finally the sequence of weights $w_1, w_2, w_3, \ldots, w_n$ is obtained by the ratio $w_k = \frac{x_k}{M}$ for $k = 1, \ldots, n$. The mathematical foundation of this method is demonstrated as is shown below:

For the $X_1, \ldots, X_n$ random variables uniformly distributed $X_i \sim U(i-1,i)$ It’s corresponding probability density functions are as shown in Equation 70:

$$f(x_i) = \begin{cases} 1 & \text{if } x_i \in (i-1,i) \\ 0 & \text{otherwise} \end{cases} \quad \text{for } i = 1, \ldots, n \quad (70)$$

Since $X_1, \ldots, X_n$ are independent, its joint probability density function is as shown in Equation 71

$$f(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} f(x_i) = \begin{cases} 1 & \text{if } x_i \in (i-1,i) \text{ for } i = 1, \ldots, n \\ 0 & \text{otherwise} \end{cases} \quad (71)$$

Next by multiple integration it is proven that the joint function in Equation 71 is a probability density since its summation equals

$$\int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x_1, x_2, \ldots, x_n) \, dx_1 \, dx_2 \cdots dx_n = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} 1 = 1$$

To generate each one of the $X_i$ values such that $X_1 < X_2 < \cdots < X_n$ the marginal probability density functions (p.d.f) and cumulative distribution (c.d.f.) for each one of the random variables $X_i$ must be calculated respectively.

According to Equation 70 the p.d.f for the first random variable $X_1$ is as Equation 72:

$$f(x_i) = \begin{cases} 1 & \text{if } x_i \in (0,1) \\ 0 & \text{otherwise} \end{cases} \quad (72)$$
and its cumulative distribution function is:

\[ F(x_i) = \int_{0}^{x} ds = x_i \]

By the probability integral transformation theorem \( F(x_i) = u_i \) has a uniform distribution \( u \sim U(0,1) \). Therefore the first is \( x_i = u_i \). To obtain the second value \( x_2 \) given that first value \( x_1 \) it is already known, the conditional probability density function must be calculated as is shown in Equation 73:

\[
f(x_2 \mid x_i) = \frac{f(x_1, x_2)}{f(x_i)} = \begin{cases} 
1 & \text{if } x_i \in (i-1, i) \text{ for } i = 1, 2 \\
0 & \text{otherwise}
\end{cases}
\]

Similarly as \( x_1 \) was obtained, the cumulative distribution function of \( x_2 \) given that first value \( x_1 \) it is already known must be calculated:

\[ F(x_2 \mid x_i) = \int_{x_1}^{x_2} ds = x_2 - 1 = u_2 \text{ where } u_2 \in U(0,1) \]

Solving for \( x_2 \) the second value \( x_2 = u_2 + 1 \) is obtained to be included in the increasing sequence of \( x_2 \) values needed to construct the collection of weights for the composite function. \( f_{\text{composite}} = w_1f_1 + w_2f_2 + w_3f_3 \)

Since this is an iterative procedure for the \( k \)-th case the c.d.f is:

\[ F(x_k \mid x_1, x_2, \ldots , x_{k-1}) = \int_{x_{k-1}}^{x_k} ds = x_k - (k - 1) = u_k \text{ where } u_k \in U(0,1) \]

Solving for \( x_k \) we get the general iterative formula Equation 74 for each one of the \( x_k \) values:

\[ x_k = u_k + (k - 1) \]

For the values \( k = 1, 2, 3, \ldots, n \) the \( x_k \) sequence obtained is:

\[ x_1 = u_1 < x_2 = u_2 + 1 < x_3 = u_3 + 2 < \cdots < x_k = u_k + (k - 1) < \cdots < x_n = u_n + (n - 1) \]
The weighting search ends up scaling the \( x_i \) values previously obtained, a sequence of weights in \((0, 1)\) is generated:

Let be \( M = \sum_{k=1}^{n} x_k \), and then we can obtain a non-uniformly distributed and increasing sequence of weights \( \{w_k\}_{k=1}^{n} \) such that \( 0 < w_1 < w_2 < \ldots < w_n < 1 \) and \( \sum_{k=1}^{n} w_k = 1 \) as desired and shown in Equation 75.

\[
\frac{x_k}{M} \quad \text{for} \quad k = 1, \ldots, n \tag{75}
\]

Summarizing the above procedure, the pseudo code for the weight generator is:

### 5.2 Non-uniform weights pseudo code Algorithm

To obtain the \( w_1, w_2, \ldots, w_n \) values, follow the steps:

1. Randomly generate an \( u_i \in (0,1) \)
2. \( x_1 = u_1 + (1-1) \)
3. Randomly generate an \( u_2 \in (0,1) \)
4. \( x_2 = u_2 + (2-1) \)
5. Continue the iteration according to the formula \( x_k = u_k + (k-1) \) for \( k = 1, \ldots, n \)
6. Finally calculate \( M = \sum_{k=1}^{n} x_k \) and \( w_k = \frac{x_k}{M} \) for \( k = 1, \ldots, n \)

**Example:** To show the performance of the non-uniform weight generator the algorithm will be applied over a set of non-dominated solutions. The Pareto set was obtained from the work presented by Taboada & Coit (2007). The problem solved in that paper is a well-known problem called redundancy allocation problem (RAP). The RAP refers to a system of \( s \) subsystems in series. For each subsystem, there are \( m_i \) functionally equivalent components, with different levels
of cost, weight, reliability and other characteristics, which may be selected. There is an unlimited supply of each of the \(m_i\) choices. The objective of the problem is to find how many components to set in parallel in each subsystem and of which supplier in order to optimize three different objectives. For this specific case the objectives considered were: reliability cost and weight. Table 18 shows the Pareto-set of solutions obtained in. The Pareto sets consists of 75 solutions.

<table>
<thead>
<tr>
<th>Solution</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reliability</td>
<td>0.682</td>
<td>0.72</td>
<td>0.757</td>
<td>0.789</td>
<td>0.843</td>
<td>0.86</td>
<td>0.86</td>
<td>0.864</td>
<td>0.875</td>
<td>0.877</td>
<td>0.882</td>
<td>0.883</td>
<td>0.888</td>
<td>0.893</td>
<td>0.901</td>
</tr>
<tr>
<td>Cost</td>
<td>13</td>
<td>16</td>
<td>19</td>
<td>17</td>
<td>19</td>
<td>21</td>
<td>21</td>
<td>23</td>
<td>31</td>
<td>23</td>
<td>25</td>
<td>27</td>
<td>29</td>
<td>33</td>
<td>34</td>
</tr>
<tr>
<td>Weight</td>
<td>19</td>
<td>24</td>
<td>23</td>
<td>25</td>
<td>28</td>
<td>31</td>
<td>36</td>
<td>34</td>
<td>20</td>
<td>39</td>
<td>42</td>
<td>55</td>
<td>58</td>
<td>24</td>
<td>24</td>
</tr>
<tr>
<td>Solution</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
</tr>
<tr>
<td>Reliability</td>
<td>0.902</td>
<td>0.916</td>
<td>0.917</td>
<td>0.92</td>
<td>0.926</td>
<td>0.934</td>
<td>0.936</td>
<td>0.943</td>
<td>0.945</td>
<td>0.945</td>
<td>0.962</td>
<td>0.962</td>
<td>0.964</td>
<td>0.965</td>
<td>0.97</td>
</tr>
<tr>
<td>Cost</td>
<td>35</td>
<td>33</td>
<td>34</td>
<td>36</td>
<td>37</td>
<td>35</td>
<td>36</td>
<td>36</td>
<td>38</td>
<td>39</td>
<td>38</td>
<td>39</td>
<td>39</td>
<td>40</td>
<td>41</td>
</tr>
<tr>
<td>Weight</td>
<td>22</td>
<td>28</td>
<td>27</td>
<td>28</td>
<td>26</td>
<td>32</td>
<td>31</td>
<td>32</td>
<td>29</td>
<td>30</td>
<td>36</td>
<td>34</td>
<td>35</td>
<td>40</td>
<td>39</td>
</tr>
<tr>
<td>Solution</td>
<td>33</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
<td>40</td>
<td>41</td>
<td>42</td>
<td>43</td>
<td>44</td>
<td>45</td>
</tr>
<tr>
<td>Reliability</td>
<td>0.973</td>
<td>0.973</td>
<td>0.975</td>
<td>0.98</td>
<td>0.981</td>
<td>0.982</td>
<td>0.982</td>
<td>0.983</td>
<td>0.983</td>
<td>0.983</td>
<td>0.984</td>
<td>0.985</td>
<td>0.986</td>
<td>0.986</td>
<td>0.99</td>
</tr>
<tr>
<td>Cost</td>
<td>42</td>
<td>43</td>
<td>42</td>
<td>44</td>
<td>43</td>
<td>52</td>
<td>45</td>
<td>53</td>
<td>54</td>
<td>46</td>
<td>50</td>
<td>48</td>
<td>54</td>
<td>55</td>
<td>56</td>
</tr>
<tr>
<td>Weight</td>
<td>34</td>
<td>32</td>
<td>41</td>
<td>38</td>
<td>47</td>
<td>35</td>
<td>45</td>
<td>33</td>
<td>39</td>
<td>46</td>
<td>42</td>
<td>44</td>
<td>43</td>
<td>41</td>
<td>37</td>
</tr>
<tr>
<td>Solution</td>
<td>46</td>
<td>47</td>
<td>48</td>
<td>49</td>
<td>50</td>
<td>51</td>
<td>52</td>
<td>53</td>
<td>54</td>
<td>55</td>
<td>56</td>
<td>57</td>
<td>58</td>
<td>59</td>
<td>60</td>
</tr>
<tr>
<td>Reliability</td>
<td>0.991</td>
<td>0.992</td>
<td>0.992</td>
<td>0.994</td>
<td>0.995</td>
<td>0.998</td>
<td>0.998</td>
<td>0.998</td>
<td>0.999</td>
<td>0.999</td>
<td>0.999</td>
<td>0.999</td>
<td>0.999</td>
<td>0.999</td>
<td>0.999</td>
</tr>
<tr>
<td>Cost</td>
<td>58</td>
<td>59</td>
<td>57</td>
<td>58</td>
<td>60</td>
<td>68</td>
<td>69</td>
<td>73</td>
<td>75</td>
<td>72</td>
<td>72</td>
<td>79</td>
<td>80</td>
<td>82</td>
<td>82</td>
</tr>
<tr>
<td>Weight</td>
<td>41</td>
<td>37</td>
<td>50</td>
<td>45</td>
<td>49</td>
<td>45</td>
<td>43</td>
<td>55</td>
<td>47</td>
<td>67</td>
<td>51</td>
<td>69</td>
<td>69</td>
<td>73</td>
<td>82</td>
</tr>
<tr>
<td>Solution</td>
<td>61</td>
<td>62</td>
<td>63</td>
<td>64</td>
<td>65</td>
<td>66</td>
<td>67</td>
<td>68</td>
<td>69</td>
<td>70</td>
<td>71</td>
<td>72</td>
<td>73</td>
<td>74</td>
<td>75</td>
</tr>
<tr>
<td>Reliability</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Cost</td>
<td>81</td>
<td>64</td>
<td>70</td>
<td>74</td>
<td>68</td>
<td>76</td>
<td>85</td>
<td>87</td>
<td>88</td>
<td>90</td>
<td>94</td>
<td>119</td>
<td>121</td>
<td>125</td>
<td>143</td>
</tr>
<tr>
<td>Weight</td>
<td>77</td>
<td>85</td>
<td>89</td>
<td>91</td>
<td>92</td>
<td>94</td>
<td>81</td>
<td>85</td>
<td>85</td>
<td>89</td>
<td>105</td>
<td>110</td>
<td>113</td>
<td>121</td>
<td>129</td>
</tr>
</tbody>
</table>

The algorithm applied to prune the data above in Table 18 once the weights were generated is as follows: Calculate \(f^{\text{composite}} = w_1f_1 + w_2f_2 + w_nf_n\) for \(k=100,000\) weights triplets and bound the collection of points that satisfy \(f < \alpha\) threshold.

Several thresholds were used with values from [0.063 to 0.1] as shown in Table 19.

<table>
<thead>
<tr>
<th>Thresholds</th>
<th>Solutions obtained with the non-uniform weights generator and several threshold values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Th&lt;0.1</td>
<td>32</td>
</tr>
<tr>
<td>Th&lt;0.08</td>
<td>38</td>
</tr>
<tr>
<td>Th&lt;0.075</td>
<td>45</td>
</tr>
<tr>
<td>Th&lt;0.07</td>
<td>45</td>
</tr>
<tr>
<td>Th&lt;0.065</td>
<td>47</td>
</tr>
<tr>
<td>Th&lt;0.064</td>
<td>47</td>
</tr>
<tr>
<td>Th&lt;0.063</td>
<td>0</td>
</tr>
</tbody>
</table>
From Table 19 we can accept that an optimal solution closer to the ideal point (0,0,0) is case 47 with reliability 0.9921; cost 59 and weight 37 according to Table 16. Any other subset has solutions with larger reliability including cost and weight, but also for higher thresholds solutions with lower reliability, cost and weight as well. Is clear that if the alpha threshold is increased, the optimal-Pareto solutions subset enlarges, which complicates the solutions selection for the decision maker. On the other hand the results obtained by Taboada & Coit (2007) uses a different weight generator, and values selection was done calculating the minimum value of each set of the $f$ values calculated for every 10,000 random vectors $w = (w_1, w_2, ..., w_n)$. It is important to mention that these weights were produced following the same non-numerical preferences procedure. Results are in Table 20:

**Table 20** Pruned solutions RAP, 2007

<table>
<thead>
<tr>
<th>Solutions obtained with the non-numerical ranking preferences method</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
</tr>
<tr>
<td>49</td>
</tr>
<tr>
<td>70</td>
</tr>
</tbody>
</table>

When comparing results with Table 19, we can see that both sets share a 22% of exact solutions and a 60% of close points.

Conclusions: The approach used to prune the Pareto front for the redundancy allocation problem through the ranking preferences method contains a non-uniform weight generator plus a boundary technique for reducing the Pareto front, creating an embedded sequence of decreasing sub-Pareto sets that converges to a an empty set if the threshold values tend to zero. The results were compared with the obtained by Taboada & Coit, (2007). The difference is explained by the different procedures used in the final step when pruning the Pareto front.
Meanwhile the former approach utilize a frequency counting of the minimum values for each one of the composite values, the new approach exposed in this chapter uses boundaries suggested by decreasing thresholds values, which is not as exhaustive as compared to the tabulated form. In next chapter the second weight generator method is presented.
Chapter 6

A Post Pareto Approach using a conditional uniform weight generator method

6.1 Introduction

As it was presented in Chapter 4, the non-numerical ranking preferences method was generalized to be able to consider \( n \) different types of objective functions. Although the non-numerical ranking preferences method has been tested and proved in a different number of instances, it still presents a challenge to any decision maker which is the obtaining of the probability density function, the cumulative distribution function and the inverse of the cumulative distribution function for the general uncertain weight function. To obtain the above mentioned equations becomes harder as the number of objective functions to be considered increases. The above mentioned equations are only used to generate the random but ranked weights that later are used in the composite objective function. A conditional uniform weight generator has been developed to overcome these difficulties found in the non-numerical ranking preferences method. The conditional uniform weight generator is presented in this Chapter and its performance is tested using the well-known multi-objective redundancy allocation problem (MORAP). Sections 6.2 and 6.3 introduce the concept of the conditional weight generator method with two and three weights, respectively.

6.2 Conditional weight generator method: Generation of two random but ranked weights.

This section presents the generation of two random but ranked weights using the conditional weight generator method. The two numbers are generated such that \( 0 < x_2 < x_1 < 1 \). Every random variable \( x_i \) is conditionally uniformly distributed in the previous random
variable $x_{i-1}$. Indeed let $x_1$ be a uniformly distributed random variable over $(0, 1)$, and let $x_2$ be a uniformly distributed random variable over $(0, x_1)$ for a given previously first value $x_1$ randomly selected. Therefore the joint probability density of $x_1$ and $x_2$ is given by Equation 76

$$f(x_1, x_2) = \begin{cases} \frac{1}{x_1}, & 0 < x_2 < x_1 < 1 \\ 0, & \text{otherwise} \end{cases} \quad (76)$$

From the above statement, the marginal density of $x_1$ is given by Equation 77

$$f(x_1) = \begin{cases} 1, & 0 < x_1 < 1 \\ 0, & \text{otherwise} \end{cases} \quad (77)$$

Indeed, the constraint $0 < x_2 < x_1 < 1$ is the same as the region $R = \{(x_1, x_2) : 0 < x_1 < 1, x_2 < x_1 < 1\}$.

Then $f(x_1) = \int_0^{x_1} f(x_1, x_2) dx_2 = \int_0^{x_1} \frac{1}{x_1} dx_2 = \frac{x_1}{x_1} = 1$ where $0 < x_1 < 1$ \quad (78)

Since the density of $x_2$ given $x_1$ is uniform on $(0, x_1)$, then the conditional density of $x_2$ is given by

$$f(x_2 | x_1) = \frac{f(x_1, x_2)}{f(x_1)} = f(x_1, x_2) = \begin{cases} \frac{1}{x_1}, & 0 < x_2 < x_1 \\ 0, & \text{otherwise} \end{cases} \quad (79)$$

On the other hand by the Integral probability theorem, its cumulative probability distribution function is shown in Equation 80

$$F(x_2 | x_1) = \int_0^{x_2} \frac{1}{x_1} ds = \frac{1}{x_1} \int_0^{x_2} ds = \frac{x_2}{x_1} = u \quad (80)$$

where $u \sim U(0,1)$ has a zero one uniform distribution

Solving for $x_2$ in Equation 81 the second value $x_2$ is obtained.

$$x_2 = x_1 \cdot u \quad (81)$$
Finally the formula to generate the finite sequence of uniform conditionally decreasing values is shown in Equation 82.

\[ 0 < x_2 < x_1 < 1 \]

\[ x_k = x_{k-1} \cdot u_k \quad \text{for } k = 1, 2 \quad \text{(82)} \]

The pseudo code for the algorithm is as follows:

1. Generate a random value \( x_1 = \text{rand}() \) in the interval \((0, 1)\)
2. Generate a second random value \( u_2 = \text{rand}() \) in the interval \((0, 1)\)
3. Calculate \( x_2 = x_1 \cdot u_2 \)
4. Repeat steps 1, 2 and 3 until obtaining a desired finite sequence of values.

In the next table a finite collection of the sequence is shown.

<table>
<thead>
<tr>
<th>( x_1 )</th>
<th>( x_1^2 )</th>
<th>( x_2^2 )</th>
<th>( x_3^2 )</th>
<th>( x_4^2 )</th>
<th>( x_5^2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>0.744693</td>
<td>0.686775</td>
<td>0.368485</td>
<td>0.780227</td>
<td>0.929386</td>
</tr>
<tr>
<td>( u_2 )</td>
<td>0.188955</td>
<td>0.183511</td>
<td>0.625619</td>
<td>0.081126</td>
<td>0.775713</td>
</tr>
<tr>
<td>( x_2 = x_1 \cdot u_2 )</td>
<td>0.140713</td>
<td>0.126031</td>
<td>0.230531</td>
<td>0.063297</td>
<td>0.720936</td>
</tr>
</tbody>
</table>

### 6.3 Conditional weight generator method: Generation of three random but ranked weights.

This section presents the case to generate three numbers randomly and ordered as

\[ 0 < x_3 < x_2 < x_1 < 1 \]. Every random variable \( x_i \) is conditionally uniformly distributed in the interval \((0, x_{i-1})\). Indeed let \( x_1 \) be a uniformly distributed random variable over \((0, 1)\), and let \( x_2 \) be a uniformly distributed random variable over \((0, x_1)\); and \( x_3 \) uniformly distributed in \((0, x_2)\); the joint density of \( x_1 \) and \( x_2, x_3 \) is given by Equation 83
The conditional density is shown in Equation 84
\[
f(x_3|x_1, x_2) = \frac{f(x_1,x_2,x_3)}{f(x_1,x_2)} = \frac{1}{x_1 x_2} = \begin{cases} \frac{1}{x_2}, & 0 < x_3 < x_2 < x_1 < 1 \\ 0, & otherwise \end{cases} \quad (84)
\]

By the Integral probability theorem its cumulative probability distribution function is shown in Equation 85
\[
F(x_3, |x_1, x_2) = \int_0^{x_3} \frac{1}{x_2} ds = \frac{1}{x_2} \int_0^{x_3} ds = \frac{x_3}{x_2} = u
\quad (85)
\]

where \(u\sim U(0,1)\) has a zero one uniform distribution.

Solving for \(x_3\) from Equation 84 the third value is
\[
x_3 = x_2, u
\]

Therefore by Table 21 and substituting the value of \(x_2\) in the last equation, the third value is shown in Equation 86
\[
x_3 = x_2 x_1, u \quad (86)
\]

Finally the formula to generate the finite sequence of conditionally uniform decreasing values is shown in Equation 87
\[
0 < x_3 < x_2 < x_1 < 1
\]
\[
x_k = x_{k-1} \cdot u_k \quad for \ k = 1,2,3 \quad (87)
\]

Summarizing the above procedure the pseudo code for the algorithm is:

1. Generate a random value \(x_1 = rand()\) in the interval (0, 1)
2. Generate a second random value \(x_2 = rand()\) in the interval (0, 1)
3. Calculate \( x_2 = x_1 \cdot u_2 \)

4. Generate a third random value \( u_3 = rand() \) in the interval \((0, 1)\)

5. Calculate \( x_3 = x_2 \cdot u_3 \)

6. Repeat steps 1 to 5 until obtaining a desired finite sequence of values.

In next table a finite collection of the sequence is shown.

<table>
<thead>
<tr>
<th>( x_1 )</th>
<th>( x_1^3 )</th>
<th>( x_2^3 )</th>
<th>( x_3^3 )</th>
<th>( x_4^3 )</th>
<th>( x_5^3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.933726</td>
<td>0.756749</td>
<td>0.987975</td>
<td>0.454742</td>
<td>0.882838</td>
<td></td>
</tr>
<tr>
<td>0.81095</td>
<td>0.417047</td>
<td>0.864148</td>
<td>0.246687</td>
<td>0.913712</td>
<td></td>
</tr>
<tr>
<td>0.757205</td>
<td>0.3156</td>
<td>0.853756</td>
<td>0.112179</td>
<td>0.806659</td>
<td></td>
</tr>
<tr>
<td>0.484548</td>
<td>0.971786</td>
<td>0.388884</td>
<td>0.784423</td>
<td>0.558285</td>
<td></td>
</tr>
<tr>
<td>0.366902</td>
<td>0.306696</td>
<td>0.332012</td>
<td>0.087996</td>
<td>0.450346</td>
<td></td>
</tr>
</tbody>
</table>

After showing the pattern for the general algorithm for cases two and three the general case will be demonstrated in next section.

**6.4 Development of the method for any number of weights**

General method development: from the last two sections a joint probability function was the starting point for developing the weight generator algorithm and for the general case cannot be different.

Given the joint probability density function

\[
f(x_1, \ldots, x_n) = \begin{cases} \frac{1}{\prod_{i=1}^{n-1} x_i}, & 0 < x_n < x_{n-1} < \cdots < x_2 < x_1 < 1 \\ 0, & \text{Otherwise} \end{cases}
\]  

(88)

a finite decreasing sequence of conditionally uniform random values \( \{x_i\}_{i=1}^n \) such that

\[0 < x_n < \cdots < x_2 < x_1 < 1\]

can be generated.
It can be shown that the above function is a probability density function for every \( n \) value.

\[
\int_{0}^{1} \int_{0}^{x_{2}} \ldots \int_{0}^{x_{n-2}} \int_{0}^{x_{n-1}} \frac{1}{\prod_{i=1}^{n-1} x_i} \, dx_n \, dx_{n-1} \ldots \, dx_2 \, dx_1 = 1
\]

A collection of conditionally probability density functions and its conditionally cumulative distributions will be used to generate the decreasing sequence of values in the open interval \((0,1)\) as is shown below.

For \( n = 1 \) the first p.d.f

\[
f(x_1) = \begin{cases} 
1, & 0 < x_1 < 1 \\
0, & \text{otherwise} 
\end{cases}
\]  

(89)

Is a uniform zero one random distribution to generate first random value \( x_1 \).

To obtain the second value \( x_2 \) the marginal p.d.f

\[
f(x_1, x_2) = \begin{cases} 
\frac{1}{x_1}, & 0 < x_2 < x_1 < 1 \\
0, & \text{otherwise} 
\end{cases}
\]

, and Equation 89 are used to obtain the conditional probability density function as is shown in Equation 90

\[
f(x_2, x_1) = \frac{f(x_1, x_2)}{f(x_1)} = \begin{cases} 
\frac{1}{x_1}, & 0 < x_2 < x_1 < 1 \\
0, & \text{otherwise} 
\end{cases}
\]  

(90)

To obtain \( x_2 \) the Integral probability theorem is applied to the cumulative probability distribution function shown in Equation 91

\[
F(x_2, x_1) = \int_{0}^{x_2} \frac{1}{x_1} \, ds = \frac{1}{x_1} \int_{0}^{x_2} ds = \frac{x_2}{x_1} = u
\]

(91)

where \( u \sim U(0,1) \) has a zero one uniform distribution.
Solving for $x_2$ the second value is obtained as $x_2 = x_1 \cdot u$ \hfill (92)

In a similar form is obtained the third value $x_3$ starting with the joint probability density function shown in Equation 93

$$f(x_1, x_2, x_3) = \begin{cases} 
\frac{1}{x_1 x_2}, & 0 < x_3 < x_2 < x_1 < 1 \\
0, & \text{otherwise}
\end{cases}$$ \hfill (93)

, and Equation 94

$$f(x_1, x_2) = \begin{cases} 
\frac{1}{x_1}, & 0 < x_2 < x_1 < 1 \\
0, & \text{otherwise}
\end{cases}$$ \hfill (94)

is used to calculate the conditional probability density function for $x_3$ given that $x_1, x_2$ are previously known in Equation 95.

$$f(x_3 | x_1, x_2) = \frac{f(x_1, x_2, x_3)}{f(x_1, x_2)} = \begin{cases} 
\frac{1}{x_2}, & 0 < x_3 < x_2 < x_1 < 1 \\
0, & \text{otherwise}
\end{cases}$$ \hfill (95)

The next step is to calculate the cumulative distribution function of Equation 94 and apply the integral probability theorem to solve for $x_3$.

$$F(x_3 | x_1, x_2) = \int_0^{x_3} \frac{1}{x_2} ds = \frac{1}{x_2} \int_0^{x_3} ds = \frac{x_3}{x_2} = u$$ \hfill (96)

$u \sim U(0,1)$, has a zero one uniform distribution

Solving for $x_3$ the third value is obtained as $x_3 = x_2 \cdot u$ \hfill (97)

Since this is an iterative procedure the $n$-th conditional p.d.f. is shown in Equation 98
\[
f(x_n, | x_1, x_2, ..., x_{n-1}) = \frac{f(x_1, x_2, ..., x_n)}{f(x_1, x_2)}
\]
\[
= \begin{cases} 
\frac{1}{x_{n-1}}, & 0 < x_n < x_{n-1} < ... < x_1 < 1 \\
0, & \text{otherwise}
\end{cases}
\]

, and its cumulative distribution function is Equation 99

\[
F(x_n, | x_1, x_2, ..., x_{n-1}) = \int_{0}^{x_n} \frac{1}{x_{n-1}} ds
\]
\[
= \frac{1}{x_{n-1}} \int_{0}^{x_n} ds
\]
\[
= \frac{x_n}{x_{n-1}} = u
\]

, where \( u \sim U(0,1) \) has a zero one uniform distribution. Solving for \( x_n \) in Equation 100 the last value as is shown in Equation 100

\[
x_n = x_{n-1} \cdot u
\]

Finally the formula to generate the finite sequence of uniform conditionally decreasing values is

\[
x_k = x_{k-1} \cdot u_k
\]

, for \( k=1, ..., n \)

The pseudo code for the algorithm is:

1. Obtain the first value generating a random value \( x_1 = rand() \) in the interval \((0, 1)\)
2. Generate a second uniform zero one random value \( u_2 = rand() \) in the interval \((0, 1)\)
3. Calculate \( x_2 = x_1 \cdot u_2 \)
4. Generate a third uniform zero one random value \( u_3 = rand() \) in the interval \((0, 1)\)
5. Calculate \( x_3 = x_2 \cdot u_3 \)
6. Repeat steps 1-6 up to obtain \( x_n = x_{n-1} \cdot u_n \) and stop.
At the end of the process, a finite decreasing sequence of conditionally uniform random values 
\( \{x_i\}_{i=1}^n \) arranged as 
\[ 0 < x_n < \cdots < x_2 < x_1 < 1 \]
is generated.

**Example 6.1**

The conditional-uniform weight generator was applied to prune the Pareto front obtained when solving a redundancy allocation multi-objective optimization by Taboada et al., (2007) The set consist of 75 optimal Pareto front points generated with the evolutionary algorithm NSGA-II.

The problem solved in that paper is a well-known problem called redundancy allocation problem (RAP). The RAP refers to a system of \( s \) subsystems in series. For each subsystem, there are \( m_i \) functionally equivalent components, with different levels of cost, weight, reliability and other characteristics, which may be selected. There is an unlimited supply of each of the \( m_i \) choices.

The objective of the problem is to find how many components to set in parallel in each subsystem and of which supplier in order to optimize three different objects. For this specific case the objectives considered were: reliability cost and weight. Table 23 shows the Pareto-set of solutions obtained in. The Pareto sets consists of 75 solutions.

<table>
<thead>
<tr>
<th>Solution</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reliability</td>
<td>0.682</td>
<td>0.72</td>
<td>0.757</td>
<td>0.789</td>
<td>0.843</td>
<td>0.86</td>
<td>0.86</td>
<td>0.864</td>
<td>0.875</td>
<td>0.877</td>
<td>0.902</td>
<td>0.883</td>
<td>0.888</td>
<td>0.893</td>
<td>0.901</td>
</tr>
<tr>
<td>Cost</td>
<td>13</td>
<td>16</td>
<td>19</td>
<td>21</td>
<td>21</td>
<td>21</td>
<td>23</td>
<td>23</td>
<td>31</td>
<td>25</td>
<td>27</td>
<td>29</td>
<td>31</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>Weight</td>
<td>19</td>
<td>24</td>
<td>23</td>
<td>25</td>
<td>28</td>
<td>31</td>
<td>36</td>
<td>34</td>
<td>20</td>
<td>39</td>
<td>42</td>
<td>55</td>
<td>58</td>
<td>24</td>
<td>24</td>
</tr>
<tr>
<td>Solution</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
</tr>
<tr>
<td>Reliability</td>
<td>0.902</td>
<td>0.916</td>
<td>0.91</td>
<td>0.92</td>
<td>0.926</td>
<td>0.934</td>
<td>0.936</td>
<td>0.943</td>
<td>0.945</td>
<td>0.945</td>
<td>0.962</td>
<td>0.962</td>
<td>0.964</td>
<td>0.968</td>
<td>0.97</td>
</tr>
<tr>
<td>Cost</td>
<td>35</td>
<td>33</td>
<td>34</td>
<td>36</td>
<td>37</td>
<td>35</td>
<td>36</td>
<td>36</td>
<td>38</td>
<td>39</td>
<td>38</td>
<td>39</td>
<td>39</td>
<td>40</td>
<td>41</td>
</tr>
<tr>
<td>Weight</td>
<td>22</td>
<td>28</td>
<td>27</td>
<td>28</td>
<td>26</td>
<td>32</td>
<td>31</td>
<td>32</td>
<td>29</td>
<td>30</td>
<td>36</td>
<td>34</td>
<td>35</td>
<td>40</td>
<td>39</td>
</tr>
<tr>
<td>Solution</td>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
<td>40</td>
<td>41</td>
<td>42</td>
<td>43</td>
<td>44</td>
<td>45</td>
</tr>
<tr>
<td>Reliability</td>
<td>0.973</td>
<td>0.973</td>
<td>0.975</td>
<td>0.98</td>
<td>0.981</td>
<td>0.982</td>
<td>0.982</td>
<td>0.983</td>
<td>0.983</td>
<td>0.983</td>
<td>0.984</td>
<td>0.985</td>
<td>0.986</td>
<td>0.986</td>
<td>0.986</td>
</tr>
<tr>
<td>Cost</td>
<td>42</td>
<td>42</td>
<td>42</td>
<td>44</td>
<td>43</td>
<td>52</td>
<td>45</td>
<td>53</td>
<td>54</td>
<td>46</td>
<td>50</td>
<td>48</td>
<td>54</td>
<td>55</td>
<td>56</td>
</tr>
<tr>
<td>Weight</td>
<td>54</td>
<td>32</td>
<td>41</td>
<td>38</td>
<td>47</td>
<td>35</td>
<td>45</td>
<td>35</td>
<td>39</td>
<td>46</td>
<td>42</td>
<td>44</td>
<td>43</td>
<td>41</td>
<td>37</td>
</tr>
<tr>
<td>Solution</td>
<td>46</td>
<td>47</td>
<td>48</td>
<td>49</td>
<td>50</td>
<td>51</td>
<td>52</td>
<td>53</td>
<td>54</td>
<td>55</td>
<td>56</td>
<td>57</td>
<td>58</td>
<td>59</td>
<td>60</td>
</tr>
<tr>
<td>Reliability</td>
<td>0.991</td>
<td>0.992</td>
<td>0.992</td>
<td>0.994</td>
<td>0.995</td>
<td>0.998</td>
<td>0.998</td>
<td>0.998</td>
<td>0.998</td>
<td>0.99</td>
<td>0.999</td>
<td>0.999</td>
<td>0.999</td>
<td>0.999</td>
<td>0.999</td>
</tr>
<tr>
<td>Cost</td>
<td>58</td>
<td>59</td>
<td>57</td>
<td>58</td>
<td>60</td>
<td>68</td>
<td>69</td>
<td>73</td>
<td>75</td>
<td>72</td>
<td>77</td>
<td>79</td>
<td>80</td>
<td>82</td>
<td>62</td>
</tr>
<tr>
<td>Weight</td>
<td>41</td>
<td>37</td>
<td>50</td>
<td>45</td>
<td>49</td>
<td>45</td>
<td>43</td>
<td>55</td>
<td>47</td>
<td>67</td>
<td>51</td>
<td>69</td>
<td>69</td>
<td>73</td>
<td>82</td>
</tr>
<tr>
<td>Solution</td>
<td>61</td>
<td>62</td>
<td>63</td>
<td>64</td>
<td>65</td>
<td>66</td>
<td>67</td>
<td>68</td>
<td>69</td>
<td>70</td>
<td>71</td>
<td>72</td>
<td>73</td>
<td>74</td>
<td>75</td>
</tr>
<tr>
<td>Reliability</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Cost</td>
<td>81</td>
<td>64</td>
<td>70</td>
<td>74</td>
<td>68</td>
<td>76</td>
<td>85</td>
<td>87</td>
<td>88</td>
<td>90</td>
<td>94</td>
<td>119</td>
<td>121</td>
<td>125</td>
<td>143</td>
</tr>
<tr>
<td>Weight</td>
<td>77</td>
<td>85</td>
<td>89</td>
<td>91</td>
<td>92</td>
<td>94</td>
<td>81</td>
<td>85</td>
<td>85</td>
<td>89</td>
<td>105</td>
<td>110</td>
<td>113</td>
<td>121</td>
<td>120</td>
</tr>
</tbody>
</table>
The algorithm applied to prune the data above in Table 23 once the weights were generated is as follows: The linear combination $f_{\text{composite}} = w_1f_1 + w_2f_2 + w_3f_3$ was calculated for $n=750$, 7500, 75,000 and 750,000 weights triplets and computed the frequency and percentage of each one of the vectors $(f_1, f_2, f_3)$ for every one of the run above mentioned, where $f_1$ stands for cost, $f_2$ stands for weight and $f_3$ stands for reliability and $1 > w_1 > w_2 > w_3 > 0$, obtaining the following results:

### Table 24 Results for n=750 run

<table>
<thead>
<tr>
<th>Point</th>
<th>Frequency</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>241</td>
<td>32.1</td>
</tr>
<tr>
<td>47</td>
<td>198</td>
<td>26.4</td>
</tr>
<tr>
<td>52</td>
<td>148</td>
<td>19.7</td>
</tr>
<tr>
<td>38</td>
<td>105</td>
<td>14.0</td>
</tr>
<tr>
<td>45</td>
<td>39</td>
<td>5.2</td>
</tr>
<tr>
<td>56</td>
<td>16</td>
<td>2.1</td>
</tr>
<tr>
<td>70</td>
<td>2</td>
<td>0.3</td>
</tr>
<tr>
<td>67</td>
<td>1</td>
<td>0.1</td>
</tr>
<tr>
<td>Total</td>
<td>750</td>
<td>100</td>
</tr>
</tbody>
</table>

Figure 31 Graph of pruned values in table 24

Only the points with a ratio grater or equal than 2% were marked with an arrow.

### Table 25 Results for n=7,500 run

<table>
<thead>
<tr>
<th>Point</th>
<th>Frequency</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>25010</td>
<td>33.347</td>
</tr>
<tr>
<td>47</td>
<td>18749</td>
<td>24.999</td>
</tr>
<tr>
<td>52</td>
<td>12907</td>
<td>17.209</td>
</tr>
<tr>
<td>38</td>
<td>11441</td>
<td>15.255</td>
</tr>
<tr>
<td>45</td>
<td>4616</td>
<td>6.155</td>
</tr>
<tr>
<td>56</td>
<td>1664</td>
<td>2.219</td>
</tr>
<tr>
<td>67</td>
<td>249</td>
<td>0.332</td>
</tr>
<tr>
<td>69</td>
<td>194</td>
<td>0.259</td>
</tr>
<tr>
<td>70</td>
<td>83</td>
<td>0.111</td>
</tr>
<tr>
<td>68</td>
<td>52</td>
<td>0.069</td>
</tr>
<tr>
<td>73</td>
<td>15</td>
<td>0.020</td>
</tr>
<tr>
<td>62</td>
<td>12</td>
<td>0.016</td>
</tr>
<tr>
<td>75</td>
<td>7</td>
<td>0.009</td>
</tr>
<tr>
<td>74</td>
<td>1</td>
<td>0.001</td>
</tr>
<tr>
<td>Total</td>
<td>75000</td>
<td>100</td>
</tr>
</tbody>
</table>

Figure 32 Graph of pruned values in table 25
Table 26 Results for \( n=75,000 \) run

<table>
<thead>
<tr>
<th>Point</th>
<th>Frequency</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>25010</td>
<td>33.347</td>
</tr>
<tr>
<td>47</td>
<td>18749</td>
<td>24.999</td>
</tr>
<tr>
<td>52</td>
<td>12907</td>
<td>17.209</td>
</tr>
<tr>
<td>38</td>
<td>11441</td>
<td>15.255</td>
</tr>
<tr>
<td>45</td>
<td>4616</td>
<td>6.155</td>
</tr>
<tr>
<td>56</td>
<td>1664</td>
<td>2.219</td>
</tr>
<tr>
<td>67</td>
<td>249</td>
<td>0.332</td>
</tr>
<tr>
<td>69</td>
<td>194</td>
<td>0.259</td>
</tr>
<tr>
<td>70</td>
<td>83</td>
<td>0.111</td>
</tr>
<tr>
<td>68</td>
<td>52</td>
<td>0.069</td>
</tr>
<tr>
<td>73</td>
<td>15</td>
<td>0.020</td>
</tr>
<tr>
<td>62</td>
<td>12</td>
<td>0.016</td>
</tr>
<tr>
<td>75</td>
<td>7</td>
<td>0.009</td>
</tr>
<tr>
<td>74</td>
<td>1</td>
<td>0.001</td>
</tr>
<tr>
<td>Total</td>
<td>75000</td>
<td>100</td>
</tr>
</tbody>
</table>

Figure 33 Graph of pruned values in table 26

Table 27 Results for \( n=750,000 \) run

<table>
<thead>
<tr>
<th>Point</th>
<th>Frequency</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>252269</td>
<td>33.636</td>
</tr>
<tr>
<td>47</td>
<td>188302</td>
<td>25.107</td>
</tr>
<tr>
<td>52</td>
<td>130459</td>
<td>17.395</td>
</tr>
<tr>
<td>38</td>
<td>111742</td>
<td>14.899</td>
</tr>
<tr>
<td>45</td>
<td>45611</td>
<td>6.081</td>
</tr>
<tr>
<td>56</td>
<td>15532</td>
<td>2.071</td>
</tr>
<tr>
<td>67</td>
<td>2476</td>
<td>0.330</td>
</tr>
<tr>
<td>69</td>
<td>1960</td>
<td>0.261</td>
</tr>
<tr>
<td>70</td>
<td>736</td>
<td>0.098</td>
</tr>
<tr>
<td>68</td>
<td>588</td>
<td>0.078</td>
</tr>
<tr>
<td>62</td>
<td>124</td>
<td>0.017</td>
</tr>
<tr>
<td>73</td>
<td>103</td>
<td>0.014</td>
</tr>
<tr>
<td>75</td>
<td>45</td>
<td>0.006</td>
</tr>
<tr>
<td>74</td>
<td>40</td>
<td>0.005</td>
</tr>
<tr>
<td>71</td>
<td>13</td>
<td>0.002</td>
</tr>
<tr>
<td>Total</td>
<td>7500000</td>
<td>100.000</td>
</tr>
</tbody>
</table>

Figure 34 Graph of pruned values in table 27

Table 28 Tabulation for \( n=750,000 \) run

<table>
<thead>
<tr>
<th>Point</th>
<th>Frequency</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>252269</td>
<td>33.636</td>
</tr>
<tr>
<td>47</td>
<td>188302</td>
<td>25.107</td>
</tr>
<tr>
<td>52</td>
<td>130459</td>
<td>17.395</td>
</tr>
<tr>
<td>38</td>
<td>111742</td>
<td>14.899</td>
</tr>
<tr>
<td>45</td>
<td>45611</td>
<td>6.081</td>
</tr>
<tr>
<td>56</td>
<td>15532</td>
<td>2.071</td>
</tr>
<tr>
<td>67</td>
<td>2476</td>
<td>0.330</td>
</tr>
<tr>
<td>69</td>
<td>1960</td>
<td>0.261</td>
</tr>
<tr>
<td>70</td>
<td>736</td>
<td>0.098</td>
</tr>
<tr>
<td>68</td>
<td>588</td>
<td>0.078</td>
</tr>
<tr>
<td>62</td>
<td>124</td>
<td>0.017</td>
</tr>
<tr>
<td>73</td>
<td>103</td>
<td>0.014</td>
</tr>
<tr>
<td>75</td>
<td>45</td>
<td>0.006</td>
</tr>
<tr>
<td>74</td>
<td>40</td>
<td>0.005</td>
</tr>
<tr>
<td>71</td>
<td>13</td>
<td>0.002</td>
</tr>
<tr>
<td>Total</td>
<td>7500000</td>
<td>100.000</td>
</tr>
</tbody>
</table>
In all of the runs the same pattern was observed. Based on the above tables, the most likely points to be selected are Table 29.

**Table 29** Six most likely optimal points when minimizing the composite function

<table>
<thead>
<tr>
<th>Solution</th>
<th>f1=Cost</th>
<th>f2=Weight</th>
<th>f3=Reliability</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>43</td>
<td>32</td>
<td>0.973429</td>
<td>33.636</td>
</tr>
<tr>
<td>47</td>
<td>59</td>
<td>37</td>
<td>0.992115</td>
<td>25.107</td>
</tr>
<tr>
<td>52</td>
<td>69</td>
<td>43</td>
<td>0.998059</td>
<td>17.395</td>
</tr>
<tr>
<td>38</td>
<td>53</td>
<td>33</td>
<td>0.982575</td>
<td>14.899</td>
</tr>
<tr>
<td>45</td>
<td>56</td>
<td>37</td>
<td>0.990477</td>
<td>6.081</td>
</tr>
<tr>
<td>56</td>
<td>77</td>
<td>51</td>
<td>0.999036</td>
<td>2.071</td>
</tr>
</tbody>
</table>

To end this chapter, Table 30 is added to show and compare the results obtained by both approaches.

**Table 30** Seven most likely optimal points when minimizing the composite function

| Taboada & Coit (2007) |

<table>
<thead>
<tr>
<th>Solution</th>
<th>f1=Cost</th>
<th>f2=Weight</th>
<th>f3=Reliability</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>43</td>
<td>32</td>
<td>0.973429</td>
</tr>
<tr>
<td>34</td>
<td>44</td>
<td>38</td>
<td>0.979653</td>
</tr>
<tr>
<td>37</td>
<td>45</td>
<td>45</td>
<td>0.982441</td>
</tr>
<tr>
<td>42</td>
<td>48</td>
<td>44</td>
<td>0.985093</td>
</tr>
<tr>
<td>49</td>
<td>58</td>
<td>45</td>
<td>0.994349</td>
</tr>
<tr>
<td>60</td>
<td>62</td>
<td>82</td>
<td>0.999363</td>
</tr>
<tr>
<td>62</td>
<td>64</td>
<td>85</td>
<td>0.999632</td>
</tr>
<tr>
<td>65</td>
<td>68</td>
<td>92</td>
<td>0.999783</td>
</tr>
<tr>
<td>70</td>
<td>90</td>
<td>89</td>
<td>0.999962</td>
</tr>
</tbody>
</table>

Summary: Only one point is common to both methods. Both approaches use the non-numerical preferences method with different weight generators. Table 30 was obtained minimizing the composite function and calculated the most likely minimum values. Despite the intersection of both pruned sets is a single point, both subsets share very close properties as optimal points. The reason for such differences is due to the techniques used for pruning points. While in the Taboada & Coit, (2007) method included the constraint $\sum_{i=1}^{n} w_i = 1$, for the conditional
uniform weight generator it is not considered. On the other hand, an advantage of the conditional uniform weight generator is its simplicity to generate any number of weights for any multi-objective optimization problem size.

In next chapter a technique for pruning Pareto fronts will be developed with the help of one of the three weight generators.
Chapter 7
A Sweeping Cones Technique for Post Pareto Analysis

One of the tools needed for calculating a weighting function $f_{\text{composite}} = w_1 f_1 + w_2 f_2 + w_3 f_3$ such that $0 < x_1 < x_2 < \cdots < x_n < 1$ is a weight generator. In chapters 5 and 6, two weight generators were developed to solve that need. This approach consists in the generation of sweeping cones to prune the Pareto front. The main idea behind this procedure lies in the normalizing of the Pareto front using a Euclidean norm as well as the vector weights generated to be used with the non-numerical ranking preferences method. Afterwards, a collection of sweeping cones must be generated to capture those optimal points that best satisfy the ranking preferences.

Geometrically the sweeping cones proposal allocates all of the function values and the weights on a sphere of radius one centered in the origin of coordinates. Afterward the sweeping cones collect optimal points onto the sphere.

7.1 Sweeping cones method

This method requires normalizing the vectors $F = (f_1, f_2, f_3)$ of the Pareto front and the weights obtained to be used in the non-numerical ranking preferences method calculating $\frac{F}{||F||}$ and $\frac{w}{||w||}$ Remark: These transformations send all of $F$ and $w$ vectors over the unitary sphere surface.

Sweeping cones pseudo-code

a) Select a lower boundary $\alpha = 0.742 \approx \cos 42^\circ$

"An initial $42^\circ$ angle is chosen for the first sweeping cone".
b) Randomly generate a collection of weights

\[ 0 < w_1 < w_2 < w_3 < 1, \text{ such that } \sum_{i=1}^{n} w = 1 \]

“The weight vector \( W = (w_1, w_2, w_3) \) is the axis of the cone”

c) Compute the linear combination \( f_{\text{composite}} = w_1 f_1 + w_2 f_2 + w_3 f_3 = f \cdot W \)

\[ = ||f|| \cdot ||W|| \cdot \cos(\theta) = \cos(\theta) \]

d) Choose \( F = (f_1, f_2, f_3) \) such that satisfy \( f_{\text{composite}} > \alpha \)

e) Increase gradually the threshold value \( \alpha \) (reduce the cone angle)

and repeat steps a, b, c and d, until there is no \( F = (f_1, f_2, f_3) \) that satisfy \( f_{\text{composite}} > \alpha \).

7.2 Geometric interpretation of the sweeping cones technique: Start with a 42 degrees cone around each one of the weight vectors is used to seize those of the \( F = (f_1, f_2, f_3) \) optimal values on the unitary sphere with a sweeping search as with a lantern and gradually diminishing the cone opening to separate clustered points and reducing the size of the Pareto front. To reduce the cone opening the \( \alpha \) threshold value must be increased to be hold in the relation \( \alpha < f_{\text{composite}} \) until no points are trapped by the cones as shown in Figures 35, 36, 37, 38 to end the process.
7.3 Example. The performance of the method was tested over a Pareto set obtained in the work presented by Taboada & Coit (2007). The problem solved in such work is known as a multi-objective redundancy allocation problem (MORAP). The RAP refers to a system of $s$ subsystems in series. For each subsystem, there are $m_i$ functionally equivalent components, with different levels of cost, weight, reliability and other characteristics, which may be selected. There is an unlimited supply of each of the $m_i$ choices. The objective of the problem is to find how many components to set in parallel in each subsystem and of which supplier in order to optimize three different objectives in this specific case the objectives considered are: reliability, cost and weight.

Table 31 shows the Pareto-set of solutions obtained. The Pareto sets consists of 75 solutions.

<table>
<thead>
<tr>
<th>Table 31 Non-dominated Solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solution</td>
</tr>
<tr>
<td>Reliability</td>
</tr>
<tr>
<td>Cost</td>
</tr>
<tr>
<td>Weight</td>
</tr>
<tr>
<td>Reliability</td>
</tr>
<tr>
<td>Cost</td>
</tr>
<tr>
<td>Weight</td>
</tr>
<tr>
<td>Reliability</td>
</tr>
<tr>
<td>Cost</td>
</tr>
<tr>
<td>Weight</td>
</tr>
<tr>
<td>Reliability</td>
</tr>
<tr>
<td>Cost</td>
</tr>
<tr>
<td>Weight</td>
</tr>
<tr>
<td>Reliability</td>
</tr>
<tr>
<td>Cost</td>
</tr>
<tr>
<td>Weight</td>
</tr>
</tbody>
</table>

**Figure 37** Three random sweeping cones

**Figure 38** Three random sweeping cones

**Figure 39** Pareto front 75 points
After having applied the sweeping cones proposal to the 75 Pareto front in the redundancy allocation optimization problem, the preliminary results in Table 21 were obtained after a pruning process for several alpha threshold values which is equal to the cosine of the theta angle shown in Table 32: i.e. $\cos(\theta) = \alpha$

Table 32 Pruned values with the sweeping cones approach

Afterwards a second filtering was applied over the first 7 pruned subsets calculating the minimum composite value

$$f_{\text{composite}} = w_1f_1 + w_2f_2 + \cdots + w_nf_n.$$ The results obtained are in Table 33 as well as figures of each of the corresponding minimum values
Table 33 Minimum points in green for each subset

<table>
<thead>
<tr>
<th>Subsets</th>
<th>Point 1</th>
<th>Point 2</th>
<th>Point 3</th>
<th>Point 4</th>
<th>Point 5</th>
<th>Point 6</th>
<th>Point 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>14</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>3</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>47</td>
<td>47</td>
<td>0.750</td>
</tr>
<tr>
<td>5</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>0.748</td>
</tr>
<tr>
<td>6</td>
<td>19</td>
<td>20</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
</tr>
<tr>
<td>7</td>
<td>20</td>
<td>24</td>
<td>38</td>
<td>38</td>
<td>38</td>
<td>38</td>
<td>38</td>
</tr>
<tr>
<td>8</td>
<td>24</td>
<td>25</td>
<td>45</td>
<td>45</td>
<td>45</td>
<td>45</td>
<td>45</td>
</tr>
<tr>
<td>9</td>
<td>25</td>
<td>32</td>
<td>47</td>
<td>47</td>
<td>47</td>
<td>47</td>
<td>47</td>
</tr>
<tr>
<td>10</td>
<td>32</td>
<td>36</td>
<td>51</td>
<td>51</td>
<td>51</td>
<td>51</td>
<td>51</td>
</tr>
<tr>
<td>11</td>
<td>36</td>
<td>38</td>
<td>52</td>
<td>52</td>
<td>52</td>
<td>52</td>
<td>52</td>
</tr>
<tr>
<td>12</td>
<td>38</td>
<td>39</td>
<td>54</td>
<td>54</td>
<td>54</td>
<td>54</td>
<td>54</td>
</tr>
<tr>
<td>13</td>
<td>39</td>
<td>45</td>
<td>56</td>
<td>56</td>
<td>56</td>
<td>56</td>
<td>56</td>
</tr>
<tr>
<td>14</td>
<td>44</td>
<td>46</td>
<td>57</td>
<td>57</td>
<td>57</td>
<td>57</td>
<td>57</td>
</tr>
<tr>
<td>15</td>
<td>45</td>
<td>47</td>
<td>58</td>
<td>58</td>
<td>58</td>
<td>58</td>
<td>58</td>
</tr>
<tr>
<td>16</td>
<td>46</td>
<td>51</td>
<td>59</td>
<td>59</td>
<td>59</td>
<td>59</td>
<td>59</td>
</tr>
<tr>
<td>17</td>
<td>47</td>
<td>52</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>18</td>
<td>51</td>
<td>54</td>
<td>61</td>
<td>61</td>
<td>61</td>
<td>61</td>
<td>61</td>
</tr>
<tr>
<td>19</td>
<td>52</td>
<td>56</td>
<td>62</td>
<td>62</td>
<td>62</td>
<td>62</td>
<td>62</td>
</tr>
<tr>
<td>20</td>
<td>54</td>
<td>57</td>
<td>63</td>
<td>63</td>
<td>63</td>
<td>63</td>
<td>63</td>
</tr>
<tr>
<td>21</td>
<td>56</td>
<td>58</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
</tr>
<tr>
<td>22</td>
<td>58</td>
<td>60</td>
<td>65</td>
<td>65</td>
<td>65</td>
<td>65</td>
<td>65</td>
</tr>
</tbody>
</table>

Figure 40 Minimum point obtained from subset 1

Figure 41 Minimum point obtained from subset 2

Figure 42 Minimum point obtained from subset 3

Figure 43 Minimum point obtained from subset 4

Figure 44 Minimum point obtained from subset 5
Finally the RAP Pareto front was reduced to a subset of seven optimal points after applying a double selection. In table 34 are described the features for each one of the final optimal points.

Table 34 Minimum point per pruned subset

<table>
<thead>
<tr>
<th>α</th>
<th>Subset</th>
<th>Minimum Point value</th>
<th>Cost</th>
<th>Weight</th>
<th>Reliability</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.74</td>
<td>1</td>
<td>18</td>
<td>34</td>
<td>27</td>
<td>0.917232</td>
</tr>
<tr>
<td>0.742</td>
<td>2</td>
<td>45</td>
<td>56</td>
<td>37</td>
<td>0.990477</td>
</tr>
<tr>
<td>0.744</td>
<td>3</td>
<td>56</td>
<td>77</td>
<td>51</td>
<td>0.995036</td>
</tr>
<tr>
<td>0.746</td>
<td>4</td>
<td>14</td>
<td>33</td>
<td>24</td>
<td>0.893196</td>
</tr>
<tr>
<td>0.748</td>
<td>5</td>
<td>47</td>
<td>59</td>
<td>37</td>
<td>0.992115</td>
</tr>
<tr>
<td>0.750</td>
<td>6</td>
<td>38</td>
<td>53</td>
<td>33</td>
<td>0.982575</td>
</tr>
<tr>
<td>0.752</td>
<td>7</td>
<td>16</td>
<td>35</td>
<td>22</td>
<td>0.901588</td>
</tr>
</tbody>
</table>

Number of runs=1,000,000  Average Running time = 44 sec

7.2. Summary

The Sweeping cone method is a double filtering procedure, a sort of “min-max” screening that uses in the second stage the non-numerical preferences method. One filtering is for pruning with lower boundaries the whole Pareto front, and the second one for shrinking
the previously pruned subsets. The method reduced in a 99.91% the 75 points RAP Pareto set in an acceptable execution time of one minute per subset, considering that each run was of one million of iterations. Future work has to be done to evaluate the quality of the solutions obtained.
Chapter 8

Orthogonal search methodology to prune Pareto fronts

In section 3.1 a method called compromise programming was shown as some of the post Pareto procedures for reducing the size of the Pareto front. Compromise method picks a solution which is located at a minimum distance from a reference or ideal point "Z" previously selected, Yu(1973), Zeleny(1973). Usually the Ideal point selected is the vector of the best single objective functions values. Indeed let such vector be written as $Z = (f_1^*, f_2^*, ..., f_n^*)$ where $f_i^* = \min\{f_i\}$ $i = 1, ..., n$. Some metrics to calculate distances from $Z$ to $f = (f_1, f_2, ..., f_n)$ are:

$$d(f, Z) = (\sum_{i=1}^{n}(f_i - f_i^*)^2)^{1/2}$$ (102)

$$d(f, Z) = (\sum_{i=1}^{n}|f_i - f_i^*|^p)^{1/p} p = 1, 2, 3, ...$$ (103)

$$d(f, Z) = \max_{1 \leq i \leq n} \frac{f_i - f_i^*}{\max_{1 \leq j \leq n}|f_j - f_j^*|}$$ (104)

Black dots on Figure 43 are optimum points.

![Figure 47 Ideal point and selected optimal solution](image)

The method developed in this chapter, is based in the compromise programming method and an orthogonal search of the closest trade off points to an “Ideal direction”. Orthogonal search generates a decreasing nested sequence of subsets such that each one of the subsets contains the
optimal point closest to the ideal point. This allows the decision maker, select optimal solutions from the smaller subsets in order to decide from a handful the most suitable.

The orthogonal search prunes the Pareto front starting from the fact that when a Pareto optimal set is normalized with the min-max procedure, the new Pareto normalized set have the zero vector as its optimization ideal point for a multi-objective minimization problem. Indeed the min-max normalization transforms all of the optimal point into numbers in the zero one interval as is shown next. Let \( Z = (f_1^*, f_2^*, ..., f_n^*) \) the vector of minimum values for each one of the single objective functions in a multi-objective optimization problem as defined in chapter 2:

A Multi-objective optimization minimization problem is defined as shown in Equation 1, Ch1

\[
\text{Minimize } f(x) = (f_1(x), ..., f_n(x)) \\
\text{s.t. } g_j(x) \leq 0; j = 1, ..., m \\
x \in R^k
\]

Where: \( n \) is the number of objective functions \( f_n(x) \), \( m \) is the number of inequality constraints \( g_j(x) \), \( x \in R^k \) is a vector of design variables, and \( f_n(x) \in R^k \) is a vector of \( n \) objective functions, where \( f_i: R^n \rightarrow R \text{ for } i = 1, ..., k \). The feasible or decision space is defined as the set of all the decision vectors that satisfy all the constraints of the problem

\[
D = \{x \in R^n | g_j(x) \leq 0; j = 1, ..., m\}.
\]

The objective space is defined as the image \( f(D) \) of the feasible decision space \( X \), i.e.

\[
f(D) = \{f(x) | x \in X\}.
\]

The min-max normalized points are obtained scaling the objective function values using the transformation Equation 105

\[
f_i^* = \frac{f_i - f_i^{\min}}{f_i^{\max} - f_i^{\min}} \tag{105}
\]

Where \( f_i^{\min} = \min\{f_i\} \), \( f_i^{\max} = \max\{f_i\} \) for \( i = 1, 2, ..., n \) and
Thus the minimum value of each one of the scaled values is $\min_{1 \leq i \leq n} f_i^* = 0 \ for \ i = 1,2, \ldots, n$ and $\max_{1 \leq i \leq n} f_i^* = 1 \ for \ i = 1,2, \ldots, n$.

Therefore the Ideal point for the scaled trade off points is the n dimensional vector

$$\vec{0} = (0,0, \ldots, 0)$$

And the search for the best optimal points can be done through the n dimensional direction

$$\vec{1} = (1,1, \ldots, 1)$$

which will named an" Ideal direction". Even though the non-convexity properties of the Pareto front,( according to the optimization methodology for minimizing a multi-objective optimization problem ) one of the ideal directions to follow to attain the minimum of the problem is the ones vector $-\vec{1}$. Based in this simple idea the algorithm developed to prune any Pareto front to a manageable size starts calculating all of the non-dominated points orthogonal to the Ideal direction $\vec{1}$; afterwards with the help of an appropriate alpha $\alpha$ threshold in the interval $(0,1)$ are selected those orthogonal vector closer to the ideal direction by less than $\alpha$. In geometric terms we are constructing a cylinder of radius $\alpha$ and axis to "trap" all of the non-dominated points around the ideal direction $\vec{1}$. Once the $\alpha$ pruned Pareto front subset is obtained, the point from that subset closer to the Ideal point zero is calculated to obtain a single candidate point for the minimum for all of the objective functions that compose the multi-objective optimization problem, Figure 48.

If the size of the pruned Pareto front is yet large for a decision maker needs, decreasing the size of the $\alpha$ threshold will reduce the extent of the Pareto front subset up to a comfortable one. However reducing arbitrarily the threshold size can remove some of the “good points” to be considered as is shown in an example below at the end of the report for a bimodal optimization
Pareto front. In such case is recommended to consider a larger pruned subset and apply any other Pareto pruning method as the Sweeping cones method or any other that fit the needs of the decision maker.

Orthogonal search captures those Pareto front points that are in cylindrical neighborhoods around the Ideal direction \((1,1,1)\). Diameters are marked by double headed arrows (DHA). The first pruned subset is inside the longest DHA, the smaller subset inside the second DHA and so on until the last subset.

8.1 Orthogonal search method:

Denoting the Pareto front values as \(\vec{f} = (f_1, f_2, ..., f_n)\) and a \(1 > \alpha > 0\) threshold selected value:

1. Obtain the orthogonal components to the ideal direction \(\vec{I} = (1,1,1)\) calculating the vector \(\vec{O} = \vec{f} - \text{mean}(\vec{f}) \times \vec{I}\) for each one of the vectors \(\vec{f} = (f_1, f_2, ..., f_n)\).

2. Select those orthogonal components to the ideal direction \(\vec{I}\) that for a selected \(\alpha\) threshold value satisfy \(\|\vec{O}\| < \alpha\) to obtain the Pareto front pruned subset \(P_{\alpha} = \{\vec{f} \in P: \|\vec{O}\| < \alpha\}\)

3. Reduce the size of \(\alpha\) and repeat steps A, B, C until \(P_{\alpha}\) size is sufficiently small.
If the decision maker does not want reducing that much the Pareto-front, then a second filtering technique like the Sweeping cones developed in chapter 7 can be applied to reduce the size of the pruned $P_{\alpha}$ subset

The method was applied to the Pareto fronts DTLZ1, DTLZ7 from the sample Pareto fronts in the Coello’s web page http://delta.cs.cinvestav.mx/~ccoello/EMOO/testfuncs/ to prove its efficiency.

**Example1**: Application of the mesh method to the Pareto front DTLZ1 obtained by Deb *et al*, (2001) when solved the Multi-objective optimization problem:

$$\begin{align*}
\text{Minimize } & F = (f_1(X), f_2(X), \ldots, f_5(X)) \\
\text{s.t. } & f_1(X) = \frac{1}{2} x_1 x_2 (1 + g(X)) \\
& f_2(X) = \frac{1}{2} (1 - x_2) x_1 (1 + g(X)) \\
& f_3(X) = \frac{1}{2} (1 - x_1) (1 + g(X)) \\
& g(X) = 100 \left[ 10 + \sum_{i=1}^{12} (x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5)) \right] \\
& 0 \leq x_i \leq 1 \quad i = 1, \ldots, 12 \\
& X = (x_1, x_2, \ldots, x_{12})
\end{align*}$$

DTLZ1 was obtained from the sample Pareto fronts in the Coello’s web page http://delta.cs.cinvestav.mx/~ccoello/EMOO/testfuncs/ to prove the efficiency of the method.

The original 2500 data Pareto front of example1 is shown in Figure 49 for alpha=1
A second and third run was done to prune the Pareto front using cylinder radio of 0.2 and 0.1 respectively.

Figure 51 and Figure 52 show both pruned subsets which are the intersection of the triangle and a cylinder of radius 0.2 and 0.1 respectively. Figure 51 represents a 89.24% reduction of the original triangular Pareto front, and Figure 52 is a 97.32% reduction. The point closest to the origin of coordinates is brushed in red. The distance from the red dot to the origin of coordinates is equal to 0.581381 and corresponds to the data case 1625.
An immediate conclusion after reducing the size of the cylinder radius is that: the smaller the size of the radius, the smaller the number of points pruned from the Pareto front, and not necessarily the closest point to the Ideal point is kept in the pruned subsets. Therefore it is suggested to stop the pruning process at the decision maker convenience and apply other method over the subset selected for choosing the best points according to the decision maker needs.

The orthogonal search was applied also to a Pareto front generated when solved the bimodal MOP, as is shown in the following

**Example2:** The Bimodal optimization problem proposed by Deb (1999) is:

\[
\text{Minimize } F = (f_1(X), f_2(X))
\]

\[
s.t. \quad f_1(X) = x_1
\]

\[
f_2(X) = \frac{1}{2} (1 - x_2) x_1 (1 + g(X))
\]

\[
f_3(X) = \frac{g(X)}{x_1}
\]

\[
g(X) = 2.0 - e^{\left(\frac{x_2 - 0.2}{0.04}\right)^2} - 0.8 e^{\left(\frac{x_2 - 0.6}{0.4}\right)^2}
\]

\[0 \leq x_i \leq 1 \quad i = 1, 2\]

\[X = (x_1, x_2)\]

For alpha threshold \(\alpha=1\) the scatter plot of the 922 Pareto front points of the bimodal multi-objective optimization problem is shown in Figure 55. However a10% ratio reduction of the
threshold or radius $\alpha=0.1$ gave an almost same ratio reduction of the Pareto front from 922 to 159 points as is shown in Figure 56, which is a 82.75% reduction of the non-dominated points.

For the next run using an alpha threshold $\alpha=.05$ the pruned subset obtained consisted of 77 points which is a 91.64% pruning of the original Pareto front. The associated scatter plot is in Figure 57.

After applying the orthogonal search for an alpha threshold $\alpha=.05$, the Pareto front was reduced to 77 points out from 922 non-dominated points Figure 53. This represents a 91.64% pruning of
the original Pareto front. When the cylinder radius was reduced to alpha=0.01, 16 points remained as noted in Figure 58. The Pareto front pruning for this level was of 16 points out of 922, which is a 98.26% reduction.

![Figure 58](image1.png) 16 points = 98.26% Pareto-front reduction

![Figure 59](image2.png) Seven points = 99.24% Pareto-front reduction

To finish for an alpha threshold $\alpha=0.05$, the Pareto front was reduced to 7 points out of 922 non-dominated points. This represents a 99.24% pruning of the original Pareto front Figure 59.

**Example 3:** Finally the method was applied to the 2401 points Pareto front of the problem DTLZ7 proposed by Deb et al 2001, as the minimization problem:

\[
\text{Minimize } F = (f_1(X), f_2(X), f_3(X))
\]

\[
s.t. \quad f_1(X) = x_1
\]

\[
f_2(X) = x_2
\]

\[
f_3(X) = (1 + g(X))h \left( f_1, f_2, g(X) \right),
\]

and

\[
g(X) = 1 + \frac{9}{22} \sum_{i=1}^{n} x_i
\]

\[
h \left( f_1, f_2, g(X) \right) = 3 - \sum_{i=1}^{2} \left\{ \frac{f_i}{1 + g(1 + \sin(3\pi) f_i)} \right\}
\]

\[
n = 22 \quad 0 \leq x_i \leq 1 \quad i = 1, ..., 12
\]
DTLZ7 was obtained from the sample Pareto fronts in the Cello’s web page http://delta.cs.cinvestav.mx/~ccoello/EMOO/testfuncs/ to prove the efficiency of the method. The original 2401 points of Pareto front is shown in Figures 60 and Figure 61 from top to bottom, in order to see how the Pareto is reduced when applying several radiuses to the cylinder that contains the pruned Pareto subset.

![Figure 60 DTLZ7 Pareto front, 2401 points](image)

![Figure 61 DTLZ7 Pareto front from top](image)

![Figure 62 Pareto reduced to 794 points](image)

![Figure 63 794 points from top to bottom view](image)

In Table 35 is described how the Pareto front is reduced for several of the cylinder radiuses corresponding to the plots of figures from 60 to 67; besides is shown the closest point to the origin of coordinates or Ideal point and its distance to it.
It can be noted that despite the rapid reduction of the Pareto front for a threshold of 0.4 the closest point to the origin of coordinates changed to case 1151 which is farther from the Ideal point compared to case 1051. Moreover for the radius 0.3 the closest point to the Ideal point is getting away from it. This has to do with the Pareto front shape. In this case is suggested to stop pruning at radius 0.5 and use other pruning procedures, to select the best points according to the needs of a decision maker.

For figures 64-67 the point at minimum distance from the origin of coordinates has been removed from mesh by the orthogonal search.
Summary: The orthogonal search and cylinder bounding over a couple of different Pareto fronts showed that for any Pareto front size, it can be pruned in three or four steps to optimal point subsets that can be easily analyzed by a decision maker, and always is possible to get the optimal point closest to the ideal point. Unless what matters is the selection of a handful of acceptable solutions instead of just a single point, the Pareto front size reduction is up to the decision maker. As noted in the last example, the shape of the Pareto front is a “barrier” to continue reducing arbitrarily the radius of the cylinder that contains the pruned points without losing the closest non-dominated to the Ideal point; this is a disadvantage of the orthogonal search procedure that has to be taken into account when the method be used.
Chapter 9

A hybrid technique for pruning Pareto fronts

In this Chapter is presented a combination of the orthogonal search method developed in Chapter 8; the non-numerical ranking preferences method presented in Chapter 3 section 3.4 and the conditional uniform weighting method in Chapter 6.

This hybrid method is applied to reduce the size of the Pareto front obtained by Dingzhou Cao, Alper Murat, Ratna, Babu Chinnam in 2013 “Efficient exact optimization of multi-objective redundancy allocation problems in series-parallel systems”.

The redundancy allocation problem (RAP) is a technique used to increase the reliability of a system by allocating redundant components to its subsystems. A disadvantage of this procedure is that can enlarge the design cost, system weight and volume. The objective of RAP is to determine optimal system designs that maximize system reliability as in transportation, electrical power, and telecommunications among many others. Dingzhou et al, (2013) proposed to solve the redundancy allocation problem with a decomposition-based approach consisting in generating Pareto optimal subsets using a modification of the “Enhanced adaptive $\epsilon$-constraint method” for each subsystem of the multi-objective problem previously decomposed, to finally reunite all the subsets to form the whole Pareto front. In Figure 64 is shown in blue and red, the superimposed Pareto front obtained with the Dingzhou et al, (2013) method, where the Pareto front part obtained with the NSGA-II evolutionary algorithm is depicted with red symbols. The decomposition-based approach method increased the size of the Pareto front compared with the meta-heuristic approach mentioned above. The Pareto front obtained in red and blue by Dingzhou et al, (2013) consists of 6112 non-dominated points compared to the 1263 red points
when the RAP problem was solved with the Evolutionary algorithm NSGA-II (Deb K et al, 2002).

\[ f_i^* = \frac{f_i - f_i^{\text{min}}}{f_i^{\text{max}} - f_i^{\text{min}}} \]  

Figure 68 Pareto-optimal front (red NSGA2)

9.1 Orthogonal search implementation

To apply the hybrid approach, the Pareto front points in Figure 68 are normalized according to the procedure shown below in Equation 105 and the original Pareto optimal points stated as a minimization problem result. The min-max normalization is performed using the transformation

\[ f_i^{\text{min}} = \min\{f_i\}, f_i^{\text{max}} = \max\{f_i\} \text{ for } i = 1,2,\ldots,n \text{ and} \]

\[ \text{Thus } \min_{1 \leq i \leq n} f_i^* = 0 \text{ for } i = 1,2,\ldots,n \text{ and } \max_{1 \leq i \leq n} f_i^* = 1 \text{ for } i = 1,2,\ldots,n. \]

Next the Pareto front is reduced using the orthogonal search starting in Figure 69.
The orthogonal search is iteratively applied reducing the size of the threshold value as in shown in next figures, up to a radius value size such that the closest point (red) to the Ideal point is visible in the mesh generated by the funneling effect created by the orthogonal search.

Figure 69 Normalized Pareto-front alpha= 1

Figure 70 Pruned Pareto-front for an alpha=0.5

Figure 71 Pruned 4401 points for an alpha=0.3

Figure 72 Pruned 4283 points for alpha=0.2
As can be seen the orthogonal search method reduces efficiently the size of the Pareto front following the Ideal direction

$$\bar{1} = (1, 1, ..., 1)$$

To get closer to the Ideal point

$$\bar{0} = (0, 0, ..., 0)$$

Since the Pareto front is approximately convex shaped, the orthogonal search collects points near to the knee:” Knee is a solution of the Pareto-optimal front where a small improvement in one
objective would lead to a large deterioration in at least one other objective”, Rachmawati & Srinivasan, (2006).

Whether the Pareto optimal set be or not approximately convex, the orthogonal search method reduces in a drastic manner the size of any set just applying a decreasing sequence of threshold values to reduce the mesh size. Once an acceptable subset has been selected by the decision maker, the second approach called non-numerical ranking preferences ranking is applied to find out after applying tabulation the most likeable sub-collection of points to be selected. At this point the Hybrid approach process terminates and some of pruned data are shown in the following tables:

**Table 36** Subsets pruned and shared points, where point 360 is the closest point to Ideal point

<table>
<thead>
<tr>
<th>31 ordered points</th>
<th>Pruned for radius = .05</th>
<th>points pruned for radius = .04</th>
<th>points pruned for radius = .03</th>
<th>points pruned for radius = .02</th>
</tr>
</thead>
<tbody>
<tr>
<td>360</td>
<td>360</td>
<td>360</td>
<td>360</td>
<td>360</td>
</tr>
<tr>
<td>307</td>
<td>307</td>
<td>307</td>
<td>307</td>
<td>307</td>
</tr>
<tr>
<td>334</td>
<td>334</td>
<td>334</td>
<td>334</td>
<td>334</td>
</tr>
<tr>
<td>371</td>
<td>371</td>
<td>371</td>
<td>371</td>
<td>371</td>
</tr>
<tr>
<td>421</td>
<td>421</td>
<td>418</td>
<td>308</td>
<td>338</td>
</tr>
<tr>
<td>418</td>
<td>418</td>
<td>308</td>
<td>285</td>
<td>281</td>
</tr>
<tr>
<td>308</td>
<td>308</td>
<td>361</td>
<td>306</td>
<td>338</td>
</tr>
<tr>
<td>393</td>
<td>393</td>
<td>490</td>
<td>258</td>
<td>281</td>
</tr>
<tr>
<td>485</td>
<td>485</td>
<td>258</td>
<td>281</td>
<td></td>
</tr>
<tr>
<td>361</td>
<td>361</td>
<td>306</td>
<td>338</td>
<td></td>
</tr>
<tr>
<td>490</td>
<td>490</td>
<td>258</td>
<td>281</td>
<td></td>
</tr>
<tr>
<td>285</td>
<td>285</td>
<td>345</td>
<td></td>
<td></td>
</tr>
<tr>
<td>365</td>
<td>365</td>
<td>338</td>
<td></td>
<td></td>
</tr>
<tr>
<td>366</td>
<td>366</td>
<td>281</td>
<td></td>
<td></td>
</tr>
<tr>
<td>523</td>
<td>523</td>
<td>284</td>
<td></td>
<td></td>
</tr>
<tr>
<td>258</td>
<td>258</td>
<td>257</td>
<td></td>
<td></td>
</tr>
<tr>
<td>345</td>
<td>345</td>
<td>320</td>
<td></td>
<td></td>
</tr>
<tr>
<td>338</td>
<td>338</td>
<td>234</td>
<td></td>
<td></td>
</tr>
<tr>
<td>575</td>
<td>575</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>561</td>
<td>561</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>281</td>
<td>281</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>563</td>
<td>563</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>340</td>
<td>340</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>491</td>
<td>491</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>603</td>
<td>603</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>311</td>
<td>311</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>284</td>
<td>284</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>257</td>
<td>257</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>320</td>
<td>320</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>234</td>
<td>234</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In Table 36 are allocated 31 points from the 6112 points of the Pareto optimal set. Point 360 is the closet point to the Ideal point $\vec{0} = (0,0,0)$, and the rest are allocated according to its distance.
to $\vec{0}$ in increasing order i.e. $\text{distance}(\vec{0}, \text{point 360}) \leq \text{distance}(\vec{0}, \text{point 307}) \leq \ldots \leq \text{distance}(\vec{0}, \text{point 234}) = \text{maximum distance}$. Table 37 was obtained in the following way: firstly a subset of 41 points for a radius of .05 was obtained, secondly an ordered subset of size 41 from the original Pareto front of 6112 points was selected and ordered as it was explained above to compare the number of sharing points with subsets of radius 0.05, 0.04, 0.03 and 0.02. Thirdly the common ordered points according to their distance to the ideal point where allocated in table 37. In Table 37 is shown the percentage of points shared among all the pruned subsets and the 41 ordered points from the Pareto front.

<table>
<thead>
<tr>
<th></th>
<th>Number of points generated per subset</th>
<th>Number of points shared with the 41 ordered points subset</th>
<th>Percentage of shared points with the 41 ordered points subset</th>
</tr>
</thead>
<tbody>
<tr>
<td>41 ordered points</td>
<td>41</td>
<td>41</td>
<td>100%</td>
</tr>
<tr>
<td>Pruned for radius = .05</td>
<td>41</td>
<td>30</td>
<td>73%</td>
</tr>
<tr>
<td>points pruned for radius = .04</td>
<td>24</td>
<td>18</td>
<td>73.50%</td>
</tr>
<tr>
<td>points pruned for radius = .03</td>
<td>14</td>
<td>11</td>
<td>78.60%</td>
</tr>
<tr>
<td>points pruned for radius = .02</td>
<td>6</td>
<td>6</td>
<td>100%</td>
</tr>
</tbody>
</table>

This graphical presentation shows that the orthogonal search method gives to the decision maker good candidates of trade-off points not far from the minimum point. In next paragraph are shown in Table 38 points that are closer and farthest from the Ideal point for some of the pruned subsets with the orthogonal search.
Summary of the results obtained for several threshold values

Table 38 Subsets pruned sizes, and distances from the closest and farthest points to the Ideal point (0,0,0)

<table>
<thead>
<tr>
<th>alpha radius</th>
<th>pruned subset size</th>
<th>Percentage of the Pareto set</th>
<th>minimum point</th>
<th>Distance to Ideal point</th>
<th>maximum point</th>
<th>Distance to Ideal point</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>3992</td>
<td>65.3%</td>
<td>360</td>
<td>0.1744</td>
<td>5133</td>
<td>0.8648</td>
</tr>
<tr>
<td>0.3</td>
<td>1437</td>
<td>23.5%</td>
<td>360</td>
<td>0.1744</td>
<td>2954</td>
<td>0.6184</td>
</tr>
<tr>
<td>0.2</td>
<td>640</td>
<td>10.5%</td>
<td>360</td>
<td>0.1744</td>
<td>45</td>
<td>0.4692</td>
</tr>
<tr>
<td>0.1</td>
<td>178</td>
<td>2.9%</td>
<td>360</td>
<td>0.1744</td>
<td>125</td>
<td>0.2564</td>
</tr>
<tr>
<td>0.05</td>
<td>41</td>
<td>0.7%</td>
<td>360</td>
<td>0.1744</td>
<td>208</td>
<td>0.2076</td>
</tr>
<tr>
<td>0.04</td>
<td>24</td>
<td>0.4%</td>
<td>360</td>
<td>0.1744</td>
<td>386</td>
<td>0.2058</td>
</tr>
<tr>
<td>0.03</td>
<td>14</td>
<td>0.2%</td>
<td>360</td>
<td>0.1744</td>
<td>287</td>
<td>0.1991</td>
</tr>
<tr>
<td>0.02</td>
<td>6</td>
<td>0.1%</td>
<td>360</td>
<td>0.1744</td>
<td>281</td>
<td>0.3262</td>
</tr>
</tbody>
</table>

9.2 Non-numerical ranking preferences application

The non-numerical ranking preferences method is applied over a pruned subset selected of 178 points and the conditional uniform weighting generator for the pruning stage; The constraint $\sum_{i=1}^{n} w_i$ and $0 < w_i < 1$ for all i : sum of weights equals one and each one of the weights are in descent order $1 > w_1 > w_2 > w_3 > 0$ was assumed. To end the process tabulation was applied for each one of the weighting sums $f_{composite}(x) = w_1f_1(x) + w_2f_2(x) + w_3f_3(x)$ to find out that only four points remained. The four points are presented in Tables 39 and 40 for runs of 17800 and 178000 respectively. An important result is that regardless the number of runs the same subset of optimal points is obtained with more likeable point 216.

Table 39 Pruned subset for a 17,800 run

<table>
<thead>
<tr>
<th>Point</th>
<th>count</th>
<th>percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>125</td>
<td>2381</td>
<td>13.38</td>
</tr>
<tr>
<td>193</td>
<td>680</td>
<td>3.82</td>
</tr>
<tr>
<td>216</td>
<td>11913</td>
<td>66.93</td>
</tr>
<tr>
<td>366</td>
<td>2826</td>
<td>15.88</td>
</tr>
<tr>
<td>Totals</td>
<td>17800</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 40 Pruned subset for a 178,000 run

<table>
<thead>
<tr>
<th>Point</th>
<th>count</th>
<th>percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>125</td>
<td>24111</td>
<td>13.55</td>
</tr>
<tr>
<td>193</td>
<td>6854</td>
<td>3.85</td>
</tr>
<tr>
<td>216</td>
<td>119245</td>
<td>66.99</td>
</tr>
<tr>
<td>366</td>
<td>27790</td>
<td>15.61</td>
</tr>
<tr>
<td>Totals</td>
<td>178000</td>
<td>100</td>
</tr>
</tbody>
</table>
In Table 41 are shown the corresponding values for cost, weight and reliability for each one of the optimal points pruned with the hybrid approach and compared their features with point 360 the closest to the Ideal point (0,0,0). Here is shown that getting just the closest point to the Ideal point is not enough for a decision maker. A subset of points and values near to the optimum point helps to make sensitivity analysis comparing features with other possible trade-off vectors.

<table>
<thead>
<tr>
<th>Point</th>
<th>Cost</th>
<th>Weight</th>
<th>Reliability</th>
</tr>
</thead>
<tbody>
<tr>
<td>125</td>
<td>20</td>
<td>29</td>
<td>0.863722</td>
</tr>
<tr>
<td>193</td>
<td>23</td>
<td>23</td>
<td>0.865298</td>
</tr>
<tr>
<td>216</td>
<td>24</td>
<td>21</td>
<td>0.865993</td>
</tr>
<tr>
<td>366</td>
<td>30</td>
<td>19</td>
<td>0.869719</td>
</tr>
<tr>
<td>360</td>
<td>30</td>
<td>23</td>
<td>0.940125</td>
</tr>
</tbody>
</table>

Note that point 360 has higher reliability compared to the other points obtained with the non-numerical preferences method, but costly compared to the four most likeable obtained by tabulation. This selection assumed for the random weights generated that

\[ w_1 = w_{\text{reliability}} > w_2 = w_{\text{weight}} > w_3 = w_{\text{cost}} \]

for the composite function

\[ f_{\text{composite}}(x) = w_1 f_1(x) + w_2 f_2(x) + w_3 f_3(x) \]

**Conclusions:** The combination of several strategies for reducing the size of a Pareto optimal set is a good alternative for reducing the size of any Pareto front regardless its shape. However if the Optimal set is approximately convex, the method works better. For sets with multiple knees an extension of the method can be used as is proposed for future research in next chapter of conclusions for the dissertation.
Chapter 10
Future research

For the reduction of a Pareto front using the non-numerical preference's method now can be utilized at least three algorithms for the generation of ordered weights in the construction of the composite function required by such procedure. These weights generators were demonstrated in the early chapters of this dissertation and subsequently tested with various approaches and techniques for the reduction of the Pareto front. The novelty of these weight generators is that can be used for any multi-objective optimization problem size or any number of objective functions. Besides of its applicability for the construction of composite functions, these weight generators were applied to randomly generate axis for cones in the sweeping cones technique (SWECT). The SWECT approach is used to un-cluster those points with a desired feature of the optimal solutions and subsequently the same weight generator was used over each one of the subsets obtained to select the most like optimum vector according to its frequency of appearance. The penultimate chapter was devoted to develop the “mesh” technique which searches optimal points from the Pareto front following the path of an Ideal direction which points to an Ideal point taken as a light house. Even though the mesh technique works better if the optimal point set is approximately convex, this last method can be extended to the search of points in the neighborhood of knees Figure 77 of any non-convex Pareto front; provided knee representative points have been obtained with some of the recently developed methods (Rachmawati, Srinivasan 2006-2009)
Finally in last chapter a hybrid method combining some of the techniques previously developed was created for reducing a Pareto front which was obtained for solving a redundancy allocation design problem with a direct non-evolutionary algorithm (Dingzhou Cao, Alper Murat, Ratna Babu Chinnam, 2013), unlike all of the data sets used to test the pruning methods developed in this research.
References


34. Rachmawati, Srinivasan, 2009. Multiobjective Evolutionary Algorithm with Controllable Focus on the Knees of the Pareto Front. IEEE transactions on evolutionary computation, vol. 13, no. 4, August.


43. Zitzler E., Marco Laumanns, and Lothar Thiele “SPEA2: Improving the Strength Pareto Evolutionary Algorithm” Computer Engineering and Networks Laboratory (TIK) Department of Electrical Engineering Swiss Federal Institute of Technology (ETH) Zurich

44. Zitzler, E., Thiele, L. An evolutionary algorithm for multi-objective optimization. The strenght Pareto approach. Technical Report 43, Zurich, Switzerland Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH)

Curriculum Vita

Victor Carrillo was born in Lerdo, Durango, Mexico. The first of six offspring of Victor M. Carrillo and Maria Elena Saucedo. He received his bachelor degree in the Mexico City is National Polytechnic Institute in the Fall of 1977. Then he started his professional career as a lecturer in the Metropolitan Autonomous University in Mexico City from 1977 to 1992. He has been a professor in the Autonomous University of Juarez city Chihuahua since 1993. He enrolled at the the University of Texas at El Paso to pursue a master’s degree in Statistics. In May 1990 he graduate and obtained his MS degree in Statistics. In the summer of 1990, he worked as a lecturer at the El Paso Community College for the summer term in 1990. In August 2009, he returned to UTEP to pursue a PhD degree in Computational Science. Victor’s dissertation work focuses on post-Pareto optimality analysis. During his PhD studies he worked under the direction of Dr. Heidi Taboada. In fall 2012, he received the Best MS thesis award from the Computational Science program. During his PhD studies, he had the chance to present his work at different conferences. For instance, he presented his work titled “Applications and performance of the non-numerical ranking preferences method for post-Pareto optimality” at the Complex Adaptive Systems conference in Chicago, IL in 2011. He attended the 2012 INFORMS conference held at Phoenix, AZ and had the opportunity to present a talk about his research in post-Pareto Analysis. In November 2012, he presented the works titled "General Iterative procedure for the Non-numerical ranking preferences method in Multi-objective Optimization" and "A Post-Pareto Approach for Multi-Objective Decision Making Using a Non-Uniform Weight Generator Method", in the Complex Adaptive Systems conference in Washington D.C. In May 2013 he attended the 2013 Industrial and Systems Engineering Research Conference in
San Juan, Puerto Rico, where he presented his work titled “A sweeping technique for post Pareto analysis”.

Permanent address: Solar de Gardenias 2440

Juarez, Chihuahua Mexico

This dissertation was typed by Victor M Carrillo