Multi-Objective Optimization

Carlos A. Coello Coello

ccoello@cs.cinvestav.mx
CINVESTAV-IPN
Evolutionary Computation Group (EVOCINV)
Computer Science Department
Av. IPN No. 2508, Col. San Pedro Zacatenco
México, D.F. 07360, MEXICO

Patna, India, December, 2016

LECTURE 1

Motivation

Most problems in nature have several (possibly conflicting) objectives to be satisfied (e.g., design a bridge for which want to minimize its weight and cost while maximizing its safety). Many of these problems are frequently treated as single-objective optimization problems by transforming all but one objective into constraints.

Formal Definition

Find the vector $\vec{x}^* = \left[x_1^*, x_2^*, \ldots, x_n^*\right]^T$ which will satisfy the m inequality constraints:

$$g_i(\vec{x}) \leq 0 \quad i = 1, 2, \ldots, m \tag{1}$$

the p equality constraints

$$h_i(\vec{x}) = 0 \quad i = 1, 2, \ldots, p \tag{2}$$

and will optimize the vector function

$$\vec{f}(\vec{x}) = \left[f_1(\vec{x}), f_2(\vec{x}), \ldots, f_k(\vec{x})\right]^T \tag{3}$$

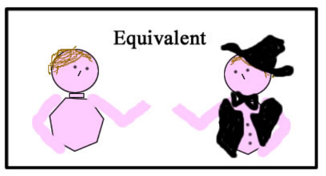
In order to know how "good" a certain solution is, it is necessary to have some criteria to evaluate it. These criteria are expressed as computable functions of the decision variables, that are called **objective functions.**

In real-world problems, some of these objective functions are in *conflict* with others, and some have to be minimized while others are maximized. These objective functions may be **commensurable** (measured in the same units) or **non-commensurable** (measured in different units).

In Operations Research, it is a common practice to differentiate among attributes, criteria, objectives and goals.

Attributes are often thought of as differentiating aspects, properties or characteristics of alternatives or consequences. **Criteria** generally denote evaluative measures, dimensions or scales against which alternatives may be gauged in a value or worth sense. **Objectives** are sometimes viewed in the same way, but may also denote specific desired levels of attainment or vague ideals. **Goals** usually indicate either of the latter notions. A distinction commonly made in Operations Research is to use the term **goal** to designate potentially attainable levels, and **objective** to designate unattainable ideals.

Basic Concepts



Several researchers use the terms **objective**, **criteria**, and **attribute** interchangeably to represent an MOP's goals or objectives (i.e., distinct mathematical functions) to be achieved. The terms **objective space** or **objective function space** are also used to denote the coordinate space within which vectors resulting from evaluating an MOP are plotted.

Ideal Objective Vector

It is an objective vector minimizing each of the objective functions. The components z_i^* of the **ideal objective vector** $\mathbf{z}^* \in \mathbb{R}^k$ are obtained by minimizing each of the objective functions individually, subject to the constraints. That is, it is obtained by solving:

$$\text{minimize } f_i(\vec{x}) \tag{4}$$

$$\text{subject to } \vec{x} \in \mathcal{F}, \text{ for } i = 1, \ldots, k \tag{5}$$

Ideal Objective Vector

The ideal vector is unreachable in most cases (except when there is no conflict among the objectives). However, the ideal vector is adopted by some mathematical programming techniques in which normally the idea is to minimize the distance of a solution with respect to such ideal vector.

Generating the ideal vector is not particularly complicated (except when some (or all) of the objective functions, when considered in isolation, presents multimodality). However, its generation has an additional computational cost that is not always affordable. Some multi-objective metaheuristics adopt an approximation of the ideal vector that is updated at each iteration.

Utopian Objective Vector

Some authors (e.g., Miettinen [1999]) also consider the concept of **utopian objective vector.**

The utopian objective vector is defined as: $\mathbf{z}^{**} \in \mathbb{R}^k$ and it is an infeasible objective vector whose components are formed by:

$$z^{**} = z_i^* - \varepsilon_i \tag{6}$$

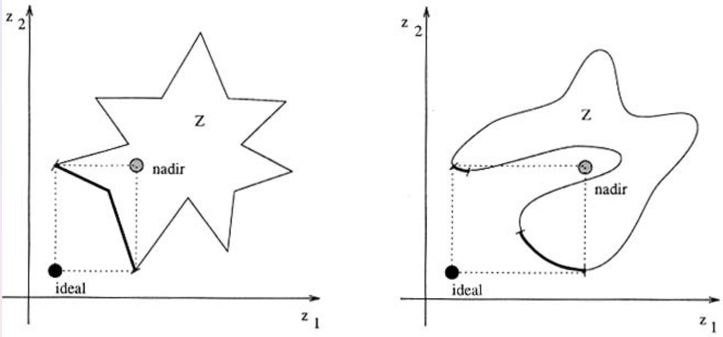
for every $i = 1, \ldots, k$, where z_i^* is a component of the ideal objective vector and $\varepsilon_i > 0$ is a scalar which is relatively small, but computationally significant. Clearly, the utopian objective vector is strictly better (i.e., it strictly dominates) every Pareto optimal solution.

Nadir Objective Vector

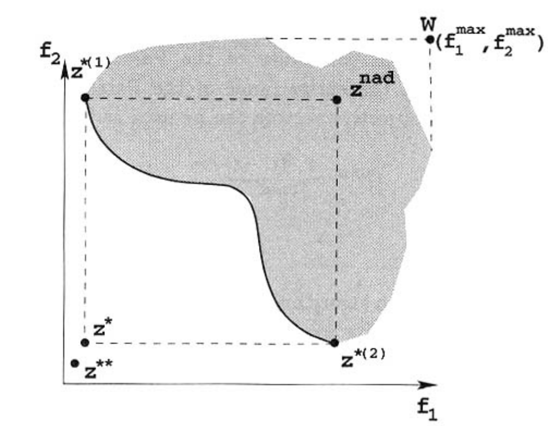
It refers to the upper bounds of the Pareto optimal set. It is normally denoted as $\mathbf{z}^{nad}$ and its components are normally quite difficult to obtain.

When computing the ideal vector, normally a **payoff table** is created using the decision vectors obtained. Row i of the payoff table displays the values of all the objective functions calculated at the point where f_i obtained its minimal value. Hence, z_i^* is at the main diagonal of the table. The maximal value of the column i in the payoff table can be selected as an estimate of the upper bound of the objective f_i for $i = 1, \ldots, k$ over the Pareto optimal set.

Basic Concepts



In this figure, a black circle is used to indicate the Ideal Objective Vectors and a gray circle is use to indicate the Nadir Objective Vectors. It is worth noting that the Nadir Objective Vector may be infeasible.



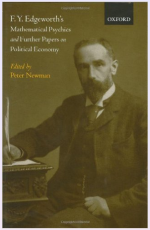
This figure shows the Ideal Objective Vector (z^*), the Utopian Objective Vector (z^{**}) and the Nadir Objective Vector (z^{nad}).

In multiobjective optimization problems, there are three possible situations:

- Minimize all the objective functions
- Maximize all the objective functions
- Minimize some and maximize others

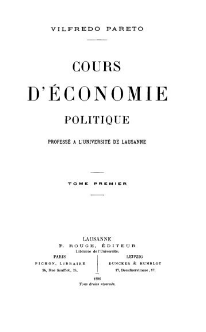
For simplicity reasons, normally all the functions are converted to a maximization or minimization form. For example, the following identity may be used to convert all the functions which are to be maximized into a form which allows their minimization:

$$\max \; f_i(\vec{x}) = \min(-f_i(\vec{x})) \tag{7}$$



Having several objective functions, the notion of "optimum" changes, because in MOPs, the aim is to find good compromises (or "trade-offs") rather than a single solution as in global optimization.

The notion of "optimum" that is most commonly adopted is that originally proposed by Francis Ysidro Edgeworth (in 1881) in his book entitled **Mathematical Psychics.**



VILFREDO PARETO

COURS

D'ÉCONOMIE

POLITIQUE

PROFESSÉ A L'UNIVERSITÉ DE LAUSANNE

TOME PREMIER

LAUSANNE
F. ROUGE, ÉDITEUR
Librairie de l'Université.

PARIS
PICHON, LIBRAIRE
24, Rue Soufflot, 24.

LEIPZIG
DUNCKER & HUMBLOT
17, Dresdnerstrasse, 17.

1896
Tous droits réservés.

This notion was generalized by the italian economist Vilfredo Pareto (in 1896) in his book **Cours d'Economie Politique.** Although some authors call *Edgeworth-Pareto optimum* to this notion (originally called **ophelimity**) it is normally preferred to use the most commonly accepted term: **Pareto optimum.**

Pareto Optimality

We say that a vector of decision variables $\vec{x}^* \in \mathcal{F}$ is **Pareto optimal** if there does not exist another $\vec{x} \in \mathcal{F}$ such that $f_i(\vec{x}) \leq f_i(\vec{x}^*)$ for all $i = 1, \ldots, k$ and $f_j(\vec{x}) < f_j(\vec{x}^*)$ for at least one j (assuming that all the objectives are being minimized).

Other important definitions

In words, this definition says that $\vec{x}^*$ is **Pareto optimal** if there exists no feasible vector of decision variables $\vec{x} \in \mathcal{F}$ which would decrease some criterion without causing a simultaneous increase in at least one other criterion. This concept normally produces a set of solutions called the **Pareto optimal set**. The vectors $\vec{x}^*$ corresponding to the solutions included in the Pareto optimal set are called **nondominated**. The image of the Pareto optimal set is called the **Pareto front**.

Basic Concepts

Pareto Dominance

A vector $\vec{u} = (u_1, \ldots, u_k)$ is said to **dominate** $\vec{v} = (v_1, \ldots, v_k)$ (denoted by $\vec{u} \preceq \vec{v}$) if and only if u is partially less than v, i.e., $\forall i \in \{1, \ldots, k\},\ u_i \leq v_i \wedge \exists i \in \{1, \ldots, k\} : u_i < v_i$.

Pareto Optimal Set

For a given MOP $\vec{f}(x)$, the Pareto optimal set ($\mathcal{P}^*$) is defined as:

$$\mathcal{P}^* := \{x \in \mathcal{F} \mid \neg\exists\ x' \in \mathcal{F}\ \ \vec{f}(x') \preceq \vec{f}(x)\}. \tag{8}$$

Pareto Front

For a given MOP $\vec{f}(x)$ and Pareto optimal set $\mathcal{P}^*$, the Pareto front ($\mathcal{PF}^*$) is defined as:

$$\mathcal{PF}^* := \{\vec{u} = \vec{f} = (f_1(x), \ldots, f_k(x)) \mid x \in \mathcal{P}^*\}. \tag{9}$$

The concept of **Pareto dominance** implies that, for a solution to *dominate* another one, it should not be worse in any objective and must be strictly better in at least one of them.

Consequently, when comparing two solutions **A** and **B**, using Pareto dominance, there are three possible outcomes:

1. **A** dominates **B**.
2. **A** is dominated by **B**.
3. **A** and **B** are not dominated by each other (i.e., they are both non-dominated).

Properties of the dominance relation

Cormen et al. [1990] provide the properties of the dominance relation:

- **Reflection:** The dominance relation **is not reflexive**, because any relation p does not dominate itself.
- **Symmetry:** The dominance relation **is not symmetric** because $p \preceq q$ does not imply $q \preceq p$. In fact, the opposite is true. In other words, if p dominates q, then q does not dominate p. Therefore, the dominance relation is **asymmetric**.
- **Antisymmetry:** Since the dominance relation is not symmetric, it can't be **antisymmetric**.
- **Transitivity:** The dominance relation is **transitive**. This is because if $p \preceq q$ and $q \preceq r$, then $p \preceq r$.

Properties of the dominance relation

Another interesting property of the dominance relation is that if a solution p does not dominate another solution q, this does not imply that q dominates p.

For a binary relation to qualify as an order relation, it must be at least *transitive* [Chankong & Haimes, 1983]. Thus, the dominance relation is an order relation. However, since the dominance relation is not reflexive, it is a **strict partial order**.

Properties of the dominance relation

In general, if a relation is reflexive, antisymmetric and transitive, it is called (in a general sense) a **partial order**. A set in which a partial order is defined is called **partially ordered set.**

However, it is important to emphasize that the dominance relation is not reflexive and is not antisymmetric. Therefore, the dominance relation is not a partial order, but only a strict partial order.

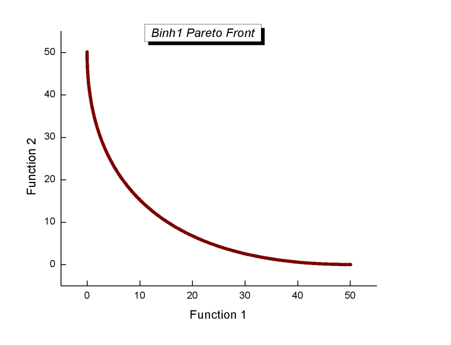
Weak Pareto Optimality

A point $\vec{x}^* \in \mathcal{F}$ is a **weakly Pareto optimal solution** if there is no $\vec{x} \in \mathcal{F}$ such that $f_i(\vec{x}) < f_i(\vec{x}^*)$, for $i = 1, \ldots, k$.

Strong Pareto Optimality

A point $\vec{x}^* \in \mathcal{F}$ is a **strongly Pareto optimal solution** if there is $\vec{x} \in \mathcal{F}$ such that $f_i(\vec{x}) \leq f_i(\vec{x}^*)$, for $i = 1, \ldots, k$.

Basic Concepts



Pareto Front

For a given multi-objective optimization problem $\vec{f}(x)$ and a Pareto optimal set $\mathcal{P}^*$, the Pareto Front ($\mathcal{PF}^*$) is defined as:

$$\mathcal{PF}^* := \{\vec{u} = \vec{f} = (f_1(x), \ldots, f_k(x)) \mid x \in \mathcal{P}^*\}. \tag{10}$$

Pareto Front

In general, it is impossible to find an analytical expression that represents the line or hyper-surface corresponding to the Pareto Optimal Front. This is possible only in very simple (textbook) cases.

The normal procedure for generating (an approximation of) the Pareto optimal front of a problem is to compute all the (or as many as possible) feasible points and to obtain their corresponding objective function values. When we had obtained a sufficient number of such points, it is possible to determine which are the nondominated solutions from them (the Pareto optimal set). As indicated before, the image of the Pareto optimal set is the Pareto front.

Algorithm 1 to Obtain Nondominated Solutions (Simple Method)

Algorithm

Step 1: Set a counter $i = 1$ and create a (empty) set of nondominated solutions P'

Step 2: For a solution $j \in P$, where P is the population ($j \neq i$), check if the solution j dominates solution i. If it dominates it, go to Step 4.

Step 3: If there are more solutions left in P, increase j by one and go to Step 2; otherwise, $P' = P' \cup \{i\}$

Step 4: Increase i by one. If $i \leq N$, go to Step 2; otherwise, stop and declare P as the nondominated set.

Algorithm 2 to Obtain Nondominated Solutions (Continuous Update)

Algorithm

Step 1: Initialize $P' = \{1\}$. Set a counter $i = 2$.

Step 2: Set $j = 1$.

Step 3: Compare solutions i and j from P'.

Step 4: If i dominates j, then delete the j^{th} member of P'. If $j < |P'|$, then increase j by one and go to Step 3. Otherwise, go to Step 5. Alternatively, if the j^{th} member of P' dominates i, increase i by one and go to Step 2.

Step 5: Insert i in P'. If $i < N$, increase i by one and go to Step 2. Otherwise, stop and declare P' as the nondominated set.

Computational Efficiency

Both Algorithm 1 and Algorithm 2 shown before have an algoritmic complexity $O(MN^2)$ in the worst case. In this case M is the number of objectives and N is the number of solutions. However, in practice, Algorithm 2 requires about half of the computational effort required by Algorithm 1.

So the obvious question is: **can we obtain nondominated solutions in a more efficient way (computational speaking)?**

Algorithm 3 to Obtain Nondominated Solutions

Theoretically, the most efficient algorithm that we can have for obtaining nondominated solutions is the one proposed by Kung et al. [1975]. This algorithm requires the set (or population) to be sorted based on the first objective. Then, the population is recursively divided in two halves: (S) superior and (I) inferior. Knowing that the first half is better than the second in terms of the first objective function, the inferior half is checked (in terms of Pareto dominance) with respect to the superior half.

Algorithm 3 to Obtain Nondominated Solutions

The solutions of (I) which are not dominated by any member of (S) are combined with the members of (S) to form a mixed population M. This union and dominance checking starts with the most inner case (in which there is only one member either in S or in I, after performing several recursive divisions of the population) and then the algorithm continues in a bottom up manner.

Algorithm 3 to Obtain Nondominated Solutions

This algorithm has a complexity $O(N(\log M)^{M-2})$ for $M \geq 4$ and $O(N \log N)$ for $M = 2$ y $M = 3$.

So, as we increase the number of objectives, Algorithm 3 also approximates the quadratic complexity of Algorithm 1 and Algorithm 2. However, for 2 or 3 objectives, this algorithm is clearly more efficient.

For more on Kung's Algorithm

H.T. Kung, F. Luccio, and F.P. Preparata, "**On finding the maxima of a set of vectors**", *Journal of the Association for Computing Machinery*, **22**(4):469–476, 1975.

For more information on this topic

- J.L. Bentley, H.T. Kung, M. Schkolnick, and C.D. Thompson, "**On the Average Number of Maxima in a Set of Vectors and Applications**", *Journal of the Association for Computing Machinery*, **25**(4):536–543, October 1978.
- Jon Louis Bentley, "**Multidimensional Divide-and-Conquer**", *Communications of the ACM*, **23**(4):214–229, April 1980.

For more information on this topic

- Jon L. Bentley, Kenneth L. Clarkson, and David B. Levine, **Fast Linear Expected-Time Algorithms for Computing Maxima and Convex Hulls**", *Algorithmica*, 9:168–183, 1993.
- Lixin Ding, Sanyou Zheng, and Lishan Kang, "**A Fast Algorithm on Finding the Non-dominated Set in Multi-objective Optimization**", in *Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003)*, Vol. 4, pp 2565–2571, IEEE Press, Canberra, Australia, December 2003.
- Michael A. Yukish, **Algorithms to Identify Pareto Points in Multi-Dimensional Data Sets**, PhD thesis, College of Engineering, Pennsylvania State University, USA, August 2004.

For more information on this topic

- Parke Godfrey, Ryan Shipley, and Jarek Gryz, "**Maximal Vector Computation in Large Data Sets**", Technical Report CS-2004-06, Department of Computer Science and Engineering, York University, Canada, December 2004.
- K.K. Mishra and Sandeep Harit, "**A Fast Algorithm for Finding the Non Dominated Set in Multiobjective Optimization**", *International Journal of Computer Applications*, **1**(25):35–39, 2010.
- Maxim Buzdalov and Anatoly Shalyto, "**A Provably Asymptotically Fast Version of the Generalized Jensen Algorithm for Non-dominated Sorting**", in Thomas Bartz-Beielstein et al. (Eds), *Parallel Problem Solving from Nature - PPSN XIII, 13th International Conference*, pp. 528–537. Springer. Lecture Notes in Computer Science Vol. 8672, Ljubljana, Slovenia, September 13-17 2014.

Optimality Conditions

Fritz-John's Necessary Condition. A necessary condition for $\mathbf{x}^*$ to be Pareto optimal is that there exist vectors $\lambda \geq 0$ and $\mathbf{u} \geq 0$ (where $\lambda \in \mathbb{R}^M$, $\mathbf{u} \in \mathbb{R}^J$ and $\lambda, \mathbf{u} \neq 0$) such that the following conditions hold:

1. $\sum_{m=1}^{M} \lambda_m \nabla f_m(\mathbf{x}^*) - \sum_{j=1}^{f} u_j \nabla g_j(\mathbf{x}^*) = 0$
2. $u_j g_j(\mathbf{x}^*) = 0$ for every $j = 1, 2, \ldots, J$.

Optimality Conditions

These conditions are very similar to the Kuhn-Tucker conditions of optimality for single-objective problems. The difference lies on the addition (in this case) of the vector of the gradients of the objectives.

For an unconstrained multi-objective optimization problem, the previous theorem requires the following condition:

$$\sum_{m=1}^{M} \lambda_m \nabla f_m(\mathbf{x}^*) = 0 \tag{11}$$

for a solution to be Pareto optimal.

Optimality Conditions

For nonlinear objective functions, it is expected that the partial derivatives are nonlinear. For a given vector λ, it is possible to check the non-existence of a Pareto optimal solution using the previously defined conditions.

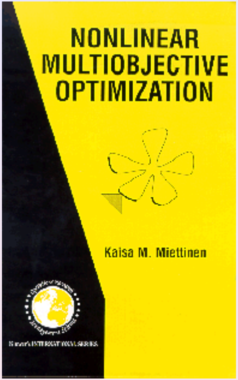
If the necessary conditions are not satisfied, then there does not exist a Pareto optimal solution corresponding to the given vector λ. It is worth noting, however, that since this is a necessary condition, the existence of a solution that is Pareto optimal is not guaranteeed. In other words, a solution that satisfies these conditions is not necessarily Pareto optimal.

Optimality Conditions

Kuhn-Tucker's Sufficiency Conditions for Pareto Optimality: Let's assume that the objective functions are convex and the constraints are non-convex. Let's assume that the objective functions and the constraints are continuously differentiable in a feasible solution $\mathbf{x}^*$. A sufficient condition for $\mathbf{x}^*$ to be Pareto optimal is that there exist vectors $\lambda > 0$ and $u \geq 0$ (where $\lambda \in \mathbb{R}^M$ and $u \in \mathbb{R}^J$) such that the following equations hold:

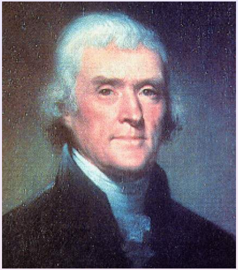
1. $\sum_{m=1}^{M} \lambda_{\mathbf{m}} \nabla f_i(\mathbf{x}^*) - \sum_{j=1}^{J} u_j \nabla g_j(\mathbf{x}^*) = 0$
2. $u_j g_j(\mathbf{x}^*) = 0$ for every $j = 1, 2, \ldots, J$.

Basic Concepts

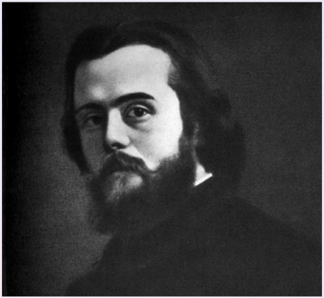


For more information

Kaisa M. Miettinen, **Nonlinear Multiobjective Optimization**, Kluwer Academic Publishers, Boston, Massachusetts, 1999.



Multiobjective optimization is an intrinsic part of economic equilibium theory and, as such, it can be said to have been founded by Adam Smith in his famous treatise entitled **An Inquiry into the Nature and Causes of the Wealth of Nations**, in 1776.



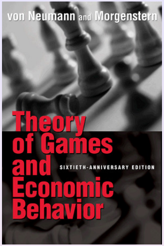
The concept of general economic equilibrium is normally attributed to Léon Walras (1834-1910). Within economic equilibrium theory, the most relevant works (besides those of Walras) are those from Jevons and Menger on utility theory, and the work on welfare theory by Edgeworth and Pareto, spanning the period from 1874 to 1906.

A second area that is considered to be one of the main precursors of multi-objective optimization is the inception of the psycological theory of games and the notion of (game) strategy.

Games of chance have a very ancient history. However, Félix Édouard Justin Émile Borel (1871-1956) is normally considered as the one who started the psicological theory of games and the one who introduced the formal definition of strategies that are based on analyzing the psychology of the opponent.

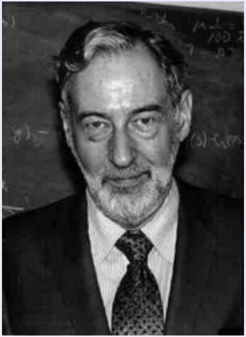
Game Theory

The so-called **game theory** can be traced back to a work by Borel from 1921. However, many historians normally attribute the origins of game theory to a paper from the famous hungarian mathematician John von Neumann which was orally presented in 1926 and published in 1928.



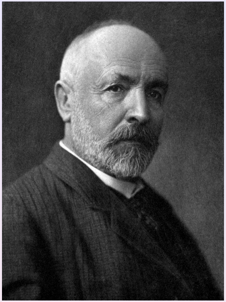
Game Theory

In 1944, John von Neumann and Oskar Morgenstern mentioned (in their famous book on **Game Theory**) that they had found a problem in economics that was a "peculiar and disconcerting mixture of several problems in conflict with each other" which could not be solved with the classical optimization methods known at that time. It remains a mystery why is that von Neummann did not get interested in this peculiar problem.



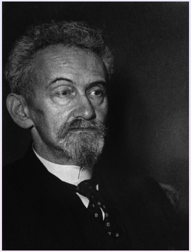
In 1951, Tjalling C. Koopmans edited a book entitled **Activity Analysis of Production and Allocation**, in which the concept of **efficient** vector (which is the same as a **nondominated vector**) was used in a meaningful way for the very first time.

Historical Highlights of Multi-Objective Optimization



Mathematical Foundations

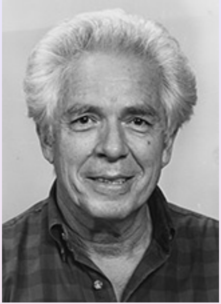
The origins of the mathematical foundations of multi-objective optimization can be traced back to the period from 1895 to 1906 in which Georg Cantor and Felix Hausdorff established the foundations of ordered spaces of infinite dimensions.



Mathematical Foundations

Cantor also introduced equivalent classes and established the first set of sufficiency conditions for the existence of a utility function.

Hausdorff provided the first example of a complete ordering.



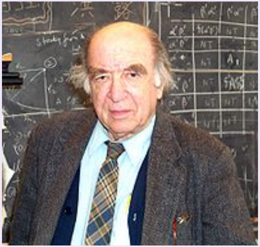
Mathematical Foundations

However, it was the concept of the **maximum vector problem** introduced by Harold W. Kuhn and Albert W. Tucker (1951) which allowed multi-objective optimization to become a mathematical discipline on its own.

Mathematical Foundations

It is well-known that the now famous conditions of optimality commonly attributed to Kuhn and Tucker had been previously stated and proved by W. Karush in an unpublished Masters thesis in 1939.

Kuhn and Tucker gave credit to Karush, which is the reason why many books call them **Karush-Kuhn-Tucker (KKT) Conditions.**



Mathematical Foundations

Nevertheless, the theory of multi-objective optimization remained practically unexplored during the 1950s. It was until the 1960s, in which the mathematical foundations of the area were consolidated when Leonid Hurwicz generalized Kuhn and Tucker's results to topological vector spaces.

Kenneth J. Arrow did some very important pioneering work in the 1950s using the concept of admissible points and stating his famous *impossibility theorem* which relates to multi-criteria decision making.

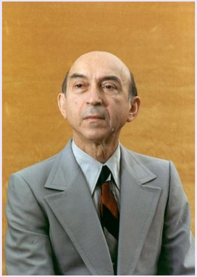
Historical Highlights of Multi-Objective Optimization

Goal Programming

Perhaps the most important outcome from the 1950s was the development of **Goal Programming**, which was originally introduced by Abraham Charnes and William Wager Cooper in 1957. However, Goal Programming became popular in the 1960s.

Applications

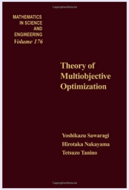
The first application of multi-objective optimization outside economics was done by Koopmans (1951) in production theory. Later on, Marglin (1967) developed the first applications of multi-objective optimization in water resources.



Applications

The first engineering application of multi-objective optimization reported in the literature is a paper published by Lofti Zadeh in the early 1960s (related to automatic control). However, multi-objective optimization applications generalized until the 1970s.

A Taxonomy of Multi-Objective Optimization Techniques



There have been several attempts to classify the many multiobjective optimization techniques currently in use. First of all, it is quite important to distinguish two stages in which the solution of a multiobjective optimization problem can be divided: the optimization of the several objective functions involved and the process of deciding what kind of “trade-offs” are appropriate from the decision maker perspective (the so-called **multicriteria decision making process**).

A Taxonomy of Multi-Objective Optimization Techniques

Cohon and Marks [1975] proposed one of the most popular classifications of techniques within the Operations Research community:

1. Generating techniques (**a posteriori** articulation of preferences).
2. Techniques which rely on prior articulation of preferences (**non-interactive methods**).
3. Techniques which rely on progressive articulation of preferences (**interaction with the decision maker**).

Other classifications are obviously possible (see for example [Duckstein, 1984]). However, the classification proposed by Cohon and Marks [1975] will be adopted here, because it focuses the classification on the way in which each technique handles the two problems of searching and making (multi-criterion) decisions:

1. ***A priori* Preference Articulation**: make decisions *before* searching (decide $\Rightarrow$ search).
2. ***A posteriori* Preference Articulation**: search *before* making decisions (search $\Rightarrow$ decide).
3. **Progressive Preference Articulation**: integrate search and decision making (decide $\Leftrightarrow$ search).

A Priori Preference Articulation

This group of techniques includes those approaches that assume that either a certain desired achievable goals or a certain pre-ordering of the objectives can be performed by the decision maker prior to the search. Some representative algorithms within this class are the following:

- Global Criterion Method
- Goal Programming
- Goal-Attainment Method
- Lexicographic Ordering
- Min-Max Optimization

All of them will be briefly described next.

Global Criterion Method

In this method, the aim is to minimize a function which defines a global criterion which is a measure of how close the decision maker can get to the ideal vector $\vec{f}^0$. The most common form of this function is [Osyczka, 1984]:

$$f(\vec{x}) = \sum_{i=1}^{k} \left(\frac{f_i^0 - f_i(\vec{x})}{f_i^0} \right)^p \tag{12}$$

where k is the number of objectives.

It is normally suggested to use $p = 1$ or $p = 2$, but other values of p can also be used. Obviously, the results differ greatly depending on the value of p chosen. Thus, the selection of the best p is an issue in this method, and it could also be the case that any p could produce an unacceptable solution.

A Priori Preference Articulation

Global Criterion Method

Another possible measure of 'closeness to the ideal solution' is a family of L_p-metrics defined as follows:

$$L_p(f) = \left[\sum_{i=1}^{k} \left| f_i^0 - f_i(x) \right|^p \right]^{1/p}, \quad 1 \leq p \leq \infty \tag{13}$$

In general, relative deviations of the form

$$\frac{f_i^0 - f_i(x)}{f_i^0} \tag{14}$$

are preferred over absolute deviations, because they have a substantive meaning in any context.

Global Criterion Method

The relevant L_p metrics are

$$L_p(f) = \left[\sum_{i=1}^{k} \left| \frac{f_i^0 - f_i(\vec{x})}{f_i^0} \right|^p \right]^{1/p}, \quad 1 \leq p \leq \infty \tag{15}$$

The value of p indicates the type of distance: for $p = 1$, all deviations from f_i^* are taken into account in direct proportion to their magnitudes, which corresponds to 'group utility' [Duckstein, 1980].

Global Criterion Method

Koski [1984] has suggested L_p-metrics with a normalized vector objective function of the form

$$f_i(\vec{x}) = \frac{f_i(\vec{x}) - \min\limits_{x \in F} f_i(\vec{x})}{\max\limits_{x \in F} f_i(\vec{x}) - \min\limits_{x \in F} f_i(\vec{x})} \tag{16}$$

In this case, the values of every normalized function are limited to the range [0,1].

Compromise Programming

Using the global criterion method one non-inferior solution is obtained. If certain parameters w_i are used as weights for the criteria, a required set of non-inferior solutions can be found. Duckstein [1984] calls this method **compromise programming**.

$$L_p(\vec{x}) = \left[\sum_{i=1}^{k} w_i^p \left| \frac{f_i(\vec{x}) - f_i^0}{f_{i\ \max} - f_i^0} \right|^p \right]^{1/p} \tag{17}$$

where w_i are the weights, $f_{i\ \max}$ is the worst value obtainable for criterion i; $f_i(\vec{x})$ is the result of implementing decision $\vec{x}$ with respect to the i^{th} criterion.

Displaced Ideal

The **Displaced Ideal** technique [Zeleny, 1977] which proceeds to define an ideal point, a solution point, another ideal point, etc. is an extension of compromise programming.

Wierzbicki's Method

Another variation of this technique is the method suggested by Wierzbicki [1979, 1980] in which the global function has a form such that it penalizes the deviations from the so-called reference objective. Any reasonable or desirable point in the space of objectives chosen by the decision maker can be considered as the reference objective.

Let $\vec{f}^r = [f_1^r, f_2^r, \ldots, f_k^r]^T$ be a vector which defines this point. Then the function which is minimized has the form

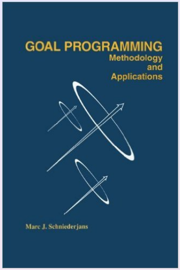
$$P(\vec{x}, \vec{f}^r) = -\sum_{i=1}^{k}(f_i(\vec{x} - f_i^r)^2 + \varrho \sum_{i=1}^{k}(\max(0, f_i(\vec{x} - f_i^r)^2) \quad (18)$$

where $\varrho > 0$ is a penalty coefficient which in this method can be chosen as constant.

By minimizing (18) for the assumed point $\vec{f}^r$, a non-inferior solution which is close to this point can be obtained. If for different points $\vec{f}^r$ the procedure is carried out, some representation of non-inferior solutions can be found.

More information on the Global Criterion Method can be found at the following references:

- A. Osyczka, **Multicriterion Optimization in Engineering with FORTRAN programs**, Ellis Horwood Limited, 1984.
- M. Zeleny, “Compromise Programming”, in J. Cochrane and M. Zeleny, editors, **Multiple Criteria Decision Making**, pp. 262–301. University of South Carolina Press, Columbia, South Carolina, 1973.
- M. Zeleny, **Multiple Criteria Decision Making**, McGraw-Hill Book Company, New York, 1982.



Goal Programming

Charnes and Cooper [1961] and Ijiri [1965] are credited with the development of the goal programming method for a linear model, and played a key role in applying it to industrial problems. This was one of the earliest techniques specifically designed to deal with multiobjective optimization problems.

Goal Programming

In this method, the decision maker (DM) has to assign targets or goals that wishes to achieve for each objective. These values are incorporated into the problem as additional constraints. The objective function then tries to minimize the absolute deviations from the targets to the objectives. The simplest form of this method may be formulated as follows:

$$\min \sum_{i=1}^{k} |f_i(\vec{x}) - T_i| \,, \quad \text{subject to } \vec{x} \in \mathcal{F} \tag{19}$$

where T_i denotes the target or goal set by the decision maker for the i^{th} objective function $f_i(\vec{x})$, and $\mathcal{F}$ represents the feasible region. The criterion, then, is to minimize the sum of the absolute values of the differences between target values and actually achieved values.

Goal Programming

A more general formulation of the goal programming objective function is a weighted sum of the p^{th} power of the deviation $|f_i(\vec{x}) - T_i|$ [Haimes, 1975]. Such a formulation has been called **generalized goal programming** [Ignizio, 1976; Ignizio, 1981]. Looking again to equation (19), the objective function is nonlinear and the simplex method can be applied only after transforming this equation into a linear form, thus reducing goal programming to a special type of linear programming. In this transformation, new variables d_i^+ and d_i^- are defined such that [Charnes, 1961]:

$$d_i^+ = \frac{1}{2}\{|f_i(\vec{x}) - T_i| + [f_i(\vec{x}) - T_i]\}, \tag{20}$$

$$d_i^- = \frac{1}{2}\{|f_i(\vec{x}) - T_i| - [f_i(\vec{x}) - T_i]\}, \tag{21}$$

A Priori Preference Articulation

Goal Programming

Adding and subtracting these equations, the following equivalent linear formulation may be found:

$$\min \ Z_0 = \sum_{i=1}^{k} (d_i^+ + d_i^-), \tag{22}$$

subject to

$$\begin{gathered} \vec{x} \in \mathcal{F} \\ f_i(\vec{x}) - d_i^+ + d_i^- = T_i \\ d_i^+, d_i^- \geq 0, \quad i = 1, \ldots, k \end{gathered} \tag{23}$$

Since it is not possible to have both under- and overachievements of the goal simultaneously, then at least one of the deviational variables must be zero. In other words:

$$d_i^+ \cdot d_i^- = 0 \tag{24}$$

A Priori Preference Articulation

Goal Programming

Fortunately, this constraint is automatically fulfilled by the simplex method because the objective function drives either d_i^+ or d_i^- or both variables simultaneously to zero for all i.

Sometimes it may be desirable to express preference for over- or underachievement of a goal. Thus, it may be more desirable to overachieve a targeted reliability figure than to underachieve it. To express preference for deviations, the DM can assign relative weights w_i^+ and w_i^- to positive and negative deviations, respectively, for each target T_i. If a minimization problem is considered, choosing the w_i^+ to be larger than w_i^- would be expressing preference for underachievement of a goal.

Goal Programming

In addition, goal programming provides the flexibility to deal with cases that have conflicting multiple goals.

Essentially, the goals may be ranked in order of importance to the problem solver. That is, a priority factor, p_i $(i = 1, \ldots, k)$ is assigned to the deviational variables associated with the goals. This variation of goal programming is called “lexicographic ordering” by some authors (e.g., [Miettinen, 1999]).

It is worth noting that these factors p_i are conceptually different from weights.

Goal Programming

The resulting optimization model becomes

$$\min S_0 = \sum_{i=1}^{k} p_i(w_i^+ d_i^+ + w_i^- d_i^-), \tag{25}$$

subject to

$$\begin{gathered} \vec{x} \in \mathcal{F} \\ f_i(\vec{x}) - d_i^+ + d_i^- = T_i \\ d_i^+, d_i^- \geq 0, \quad i = 1, \ldots, k \end{gathered} \tag{26}$$

Note that this technique yields a nondominated solution if the goal point is chosen in the feasible domain [Duckstein, 1984].

Goal Programming

More information on Goal Programming can be found at the following references:

- A. Charnes, W. W. Cooper, R. J. Niehaus, and A. Stedry, "**Static and dynamic assignment models with multiple objectives and some remarks on organization design**", *Management Science*, **15**(8):B365–B375, 1969.
- S. Lee, **Goal Programming for Decision Analysis**, Auerbach, Philadelphia, 1972.
- J. P. Ignizio, **Goal Programming and Extensions**, Heath, Lexington, Massachusetts, 1976.
- D. F. Jones and M. Tamiz, "**Goal Programming in the Period 1990–2000**, in M. Ehrgott and X. Gandibleux, editors, *Multiple Criteria Optimization. State of the Art. Annotated Bibliographic Surveys*, pp. 129–170. Kluwer Academic Publishers, Boston/Dordrecht/London, 2002.

Goal Attainment Method

In this approach, a vector of weights $w_1, w_2, \ldots, w_k$ relating the relative under- or over-attainment of the desired goals must be elicited from the decision maker in addition to the goal vector $b_1, b_2, \ldots, b_k$ for the objective functions $f_1, f_2, \ldots, f_k$. To find the best-compromise solution x^*, the following problem is solved [Gembikci, 1974; Gembicki, 1975]:

$$\text{Minimize} \quad \alpha \tag{27}$$

subject to:

$$\begin{aligned} g_j(\vec{x}) \leq 0; \quad j = 1, 2, \ldots, m \\ b_i + \alpha \cdot w_i \geq f_i(\vec{x}); \quad i = 1, 2, \ldots, k \end{aligned} \tag{28}$$

where α is a scalar variable unrestricted in sign and the weights $w_1, w_2, \ldots, w_k$ are normalized so that

$$\sum_{i=1}^{k} |w_i| = 1 \tag{29}$$

Goal Attainment Method

If some $w_i = 0$ ($i = 1, 2, \ldots, k$), it means that the maximum limit of objectives $f_i(\vec{x})$ is b_i.

It can be easily shown [Chen, 1994] that a set of nondominated solutions can be generated by varying the weights, with $w_i \geq 0$ ($i = 1, 2, \ldots, k$) even for nonconvex problems.

It should be pointed out that the optimum value of α informs the DM of whether the goals are attainable or not. A negative value of α implies that the goal of the decision maker is attainable and an improved solution is then to be obtained. Otherwise, if $\alpha > 0$, then the DM's goal is unattainable.

Goal Attainment Method

For more information on the Goal Attainment Method, see:

- F. W. Gembicki, **Vector Optimization for Control with Performance and Parameter Sensitivity Indices**, PhD thesis, Case Western Reserve University, Cleveland, Ohio, 1974.
- F. W. Gembicki and Y. Y. Haimes, "**Approach to performance and sensitivity multiobjective optimization: the goal attainment method**", *IEEE Transactions on Automatic Control*, AC-15:591–593, 1975.
- Y. L. Chen and C. C. Liu, "**Multiobjective VAR planning using the goal-attainment method**", *IEE Proceedings on Generation, Transmission and Distribution*, **141**(3):227–232, May 1994.

Lexicographic Ordering

This is a peculiar method in which the aggregations performed are not scalar. In this method, the objectives are ranked in order of importance by the decision maker (from best to worst).

The optimum solution $\vec{x}^*$ is then obtained by minimizing the objective functions, starting with the most important one and proceeding according to the order of importance of the objectives.

Lexicographic Ordering

Let the subscripts of the objectives indicate not only the objective function number, but also the priority of the objective. Thus, $f_1(\vec{x})$ and $f_k(\vec{x})$ denote the most and least important objective functions, respectively. Then the first problem is formulated as

$$\text{Minimize} \quad f_1(\vec{x}) \tag{30}$$

subject to

$$g_j(\vec{x}) \leq 0; \quad j = 1, 2, \ldots, m \tag{31}$$

and its solution $\vec{x}_1^*$ and $f_1^* = f(\vec{x}_1^*)$ is obtained. Then, the second problem is formulated as

$$\text{Minimize} \quad f_2(\vec{x}) \tag{32}$$

subject to

$$g_j(\vec{x}) \leq 0; \quad j = 1, 2, \ldots, m \tag{33}$$

$$f_1(\vec{x}) = f_1^* \tag{34}$$

and the solution of this problem is obtained as $\vec{x}_2^*$ and $f_2^* = f_2(\vec{x}_2^*)$.

Lexicographic Ordering

This procedure is repeated until all k objectives have been considered. The i^{th} problem is given by

$$\text{Minimize} \quad f_i(\vec{x}) \tag{35}$$

subject to

$$g_j(\vec{x}) \leq 0; \quad j = 1, 2, \ldots, m \tag{36}$$

$$f_l(\vec{x}) = f_l^*, \quad l = 1, 2, \ldots, i-1 \tag{37}$$

The solution obtained at the end, i.e., $\vec{x}_k^*$ is taken as the desired solution $\vec{x}^*$ of the problem.

Lexicographic Ordering

More information on Lexicographic Ordering can be found at:

- G. V. Sarma, L. Sellami, and K. D. Houam, "**Application of Lexicographic Goal Programming in Production Planning—Two case studies**", *Opsearch*, **30**(2):141–162, 1993.
- S. S. Rao, "**Multiobjective Optimization in Structural Design with Uncertain Parameters and Stochastic Processes**", *AIAA Journal*, **22**(11):1670–1678, November 1984.

A Priori Preference Articulation

Min-Max Optimization

The idea of stating the min-max optimum and applying it to multiobjective optimization problems was taken from game theory, which deals with solving conflicting situations. The min-max approach to a linear model was proposed by Jutler [1967] and Solich [1969].

It was further developed by Osyczka [1978], Rao [1986] and Tseng & Lu [1990].

The **min-max optimum** compares relative deviations from the separately attainable minima.

Min-Max Optimization

Let's consider the i^{th} objective function for which the relative deviation can be calculated from

$$z_i^{'}(\vec{x}) = \frac{|f_i(\vec{x}) - f_i^0|}{|f_i^0|} \tag{38}$$

or from

$$z_i^{''}(\vec{x}) = \frac{|f_i(\vec{x}) - f_i^0)|}{|f_i(\vec{x})|} \tag{39}$$

It should be clear that for equations (38) and (39) it is necessary to assume that for every $i \in I$ ($I = 1, 2, \ldots, k$) and for every $\vec{x} \in \mathcal{F}$, $f_i(\vec{x}) \neq 0$.

Min-Max Optimization

If all the objective functions are going to be minimized, then equation (38) defines function relative increments, whereas if all of them are going to be maximized, it defines relative decrements. Equation (39) works conversely.

Let $\vec{z}(\vec{x}) = [z_1(\vec{x}), \ldots, z_i(\vec{x}), \ldots, z_k(\vec{x})]^T$ be a vector of the relative increments which are defined in $\mathbb{R}^k$.
The components of the vector $z(\vec{x})$ are evaluated from the formula

$$\forall_{i \in I} z_i(\vec{x}) = \max \{z_i^{'}(\vec{x}), z_i^{''}(\vec{x})\} \tag{40}$$

Min-Max Optimization

Now the min-max optimum can be defined as follows [Osyczka, 1984]:
A point $\vec{x}^* \subset \mathcal{F}$ is min-max optimal, if for every $\vec{x} \in \mathcal{F}$ the following recurrence formula is satisfied:
Step 1:

$$v_1(\vec{x}^*) = \min_{x \in \mathcal{F}} \max_{i \in I}\{z_i(\vec{x})\} \tag{41}$$

and then $I_i = \{i_1\}$, where i_1 is the index for which the value of $z_1(\vec{x})$ is maximal.

If there is a set of solutions $X_1 \subset \mathcal{F}$ which satisfies Step 1, then we continue to Step 2.

Min-Max Optimization

Step 2:

$$v_2(\vec{x}^*) = \min_{x \in X_1} \left(\max_{i \in I, i \notin I_1} \{z_i(\vec{x})\} \right) \tag{42}$$

and then $I_2 = \{i_1, i_2\}$, where i_2 is the index for which the value of $z_i(x)$ in this step is maximal.

If there is a set of solutions $X_{r-1} \subset \mathcal{F}$ which satisfies Step $r-1$, then we continue to Step r.

A Priori Preference Articulation

Min-Max Optimization

Step r:

$$v_r(\vec{x}^*) = \min_{x \in X_{r-1}} \left(\max_{i \in I, i \notin I_{r-1}} \{z_i(\vec{x})\} \right) \tag{43}$$

and then $I_r = \{I_{r-1}, i_r\}$, where i_r is the index for which the value of $z_i(\vec{x})$ in the rth step is maximal.

If there is a set of solutions $X_{k-1} \subset \mathcal{F}$ which satisfies Step $k-1$, then we continue to Step k.

A Priori Preference Articulation

Min-Max Optimization

Step k:

$$v_k(\vec{x}^*) = \min_{\vec{x} \in X_{k-1}} z_i(\vec{x}) \text{ for } i \in I \text{ and } i \notin I_{k-1} \tag{44}$$

where $v_1(\vec{x}^*), \ldots, v_k(\vec{x}^*)$ is the set of optimal values of fractional deviations ordered non-increasingly.

This optimum can be described in words as follows. Knowing the extremes of the objective functions which can be obtained by solving the optimization problems for each criterion separately, the desirable solution is the one which gives the smallest values of the relative increments of all the objective functions.

Min-Max Optimization

The point $\vec{x}^* \in \mathcal{F}$ which satisfies the equations of Steps 1 and 2 may be called the best compromise solution considering all the criteria simultaneously and on equal terms of importance.

It should be noticed that even when these equations look quite complicated, in many optimization models, only the first step of this process is necessary to determine the optimum.

A Posteriori Preference Articulation

These techniques do not require prior preference information from the DM. Some of the techniques included in this category are among the oldest multiobjective optimization approaches proposed. The reason is that the main idea of these approaches follows directly from the Kuhn-Tucker conditions for noninferior solutions [Cohon and Marks, 1975].

The two most representative algorithms within this class are the following:

- Linear Combination of Weights
- The ϵ-Constraint Method

Both of them will be briefly described next.

A Posteriori Preference Articulation

Linear Combination of Weights

Zadeh [1963] was the first to show that the third of the Kuhn-Tucker conditions for noninferior solutions implies that these noninferior solutions might be found by solving a scalar optimization problem in which the objective function is a weighted sum of the components of the original vector-valued function.

Linear Combination of Weights

That is, the solution to the following problem is, in general, noninferior:

$$\min \sum_{i=1}^{k} w_i f_i(\vec{x}) \tag{45}$$

subject to:

$$\vec{x} \in \mathcal{F} \tag{46}$$

where $w_i \geq 0$ for all i and is strictly positive for at least one objective.

The noninferior set and the set of noninferior solutions can be generated by parametrically varying the weights w_i in the objective function. This was initially demonstrated by Gass and Saaty [1955] for a two-objective problem.

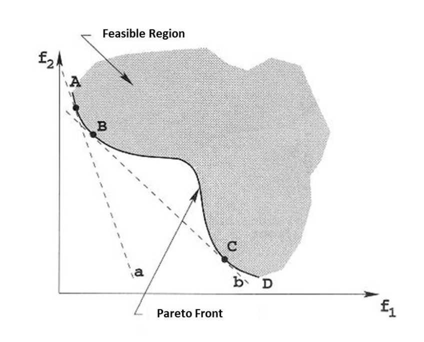
Linear Combination of Weights

Note that the weighting coefficients do not reflect proportionally the relative importance of the objectives, but are only factors which, when varied, locate points in the Pareto optimal set.

For the numerical methods that can be used to seek the minimum of equation (45), this location depends not only on w_i values, but also on the units in which the functions are expressed.

It is easy to prove that the use of a linear combination of weights is unable to generate non-convex portions of a Pareto front, regardless of the weight values that we use.

A Posteriori Preference Articulation



Linear Combination of Weights

This is an example of a case in which the true Pareto front cannot be completely covered using a linear combination of weights.

The ϵ-Constraint Method

This method also follows directly from the Kuhn-Tucker conditions for noninferior solutions. The third Kuhn-Tucker condition for optimality can be rewritten as:

$$w_r \nabla f_r(\vec{x}) + \sum_{l=1,\ l \neq r}^{k} w_l \nabla f_l(\vec{x}) - \sum_{i=1}^{m} \lambda_i \nabla g_i(\vec{x}) = 0 \qquad (47)$$

Since only relative values of the weights are of significance, the r^{th} objective can be selected so that $w_r = 1$. The previous condition defined in equation (47) then becomes:

$$\nabla f_r(\vec{x}) \sum_{l=1,\ l \neq r}^{k} w_l \nabla f(\vec{x}) - \sum_{i=1}^{m} \lambda_i \nabla g_i(\vec{x}) = 0 \qquad (48)$$

The ϵ-Constraint Method

This rewritten condition allows the second term to be interpreted as a weighted sum of the gradients of $k-1$ lower-bound constraints, since there is a plus sign before the summation. This interpretation implies that noninferior solutions can be found by solving:

$$\min f_r(\vec{x}) \tag{49}$$

subject to:

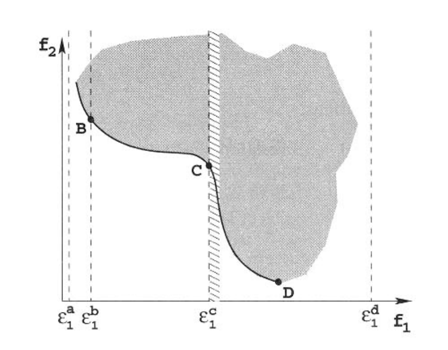
$$f_l(\vec{x}) \leq \epsilon_l \;\text{ for } l = 1, 2, \ldots, k \;\text{ and } l \neq r \tag{50}$$

where ϵ_l are assumed values of the objective functions that must not be exceeded.

The ϵ-Constraint Method

The idea of this method is to minimize one (the most preferred or primary) objective function at a time, considering the other objectives as constraints bound by some allowable levels ϵ_l. By varying these levels ϵ_l, the noninferior solutions of the problem can be obtained.

It is important to be aware of the fact that a preliminary analysis is required to identify proper starting values for ϵ_l. To get adequate ϵ_l values, single-objective optimizations are normally carried out for each objective function in turn by using mathematical programming techniques.



The ϵ-Constraint Method

This figure illustrates the way in which the ϵ-Constraint Method works.

Progressive Preference Articulation

These techniques normally operate in three stages [Cohon and Marks, 1975]:

1. Find a nondominated solution
2. Get the reaction of the DM regarding this nondominated solution, and modify the preferences of the objectives accordingly
3. Repeat the two previous steps until the DM is satisfied or no further improvement is possible.

Next, we will briefly describe 3 methods that adopt progressive preference articulation:

- Probabilistic Trade-Off Development Method
- STEP Method
- Sequential Multiobjective Problem Solving Method

Probabilistic Trade-Off Development Method

The main motivation for the development of this method (also known as PROTRADE) was to be able to handle risk in the development of the objective trade-offs, and at the same time being able to accommodate the preferences of the DM in a progressive manner [Goicochea, 1976].

In this case, it is assumed that our multiobjective optimization problem has a probabilistic objective function and probabilistic constraints [Goicochea, 1977]. According to a 12-step algorithm, an initial solution is found using a surrogate objective function, and then a multiattribute utility function is formed leading to a new surrogate objective function and a new solution. The solution is checked to see if it is satisfactory to the decision maker. The process is repeated until a satisfactory solution is reached, as described in [Goicochea, 1979; Goicochea, 1982].

Probabilistic Trade-Off Development Method

The results of the multiobjective optimization provide not only levels of attainment of the objective function elements (as in the goal attainment method [Gembicki, 1974]), but also the probabilities of reaching those levels. The technique is interactive, which means that the DM formulates a preference function in a progressive manner, after a trial process [Duckstein, 1984].

One interesting aspect of this approach is that the DM actually ranks objectives in order of importance (a multi-attribute utility function is used to assist the DM in the articulation of preferences) at the beginning of the process, and later uses pairwise comparisons to reconcile these preferences with the "real" (observed) behavior of the attributes. This allows not only an interactive participation of the DM, but it also allows to gain knowledge about the trade-offs of the problem.

Probabilistic Trade-Off Development Method

For more information about this method, see:

- A. Goicoechea, L. Duckstein, and M. Fogel, **Multi-objective programming in watershed management: A study of the Charleston watershed**, *Water Resources Research*, **12**(6):1085–1092, December 1976.
- A. Goicoechea, L. Duckstein, and M. Fogel, **Multiple objectives under uncertainty: An illustrative application of PROTRADE**", *Water Resources Research*, **15**(2):203–210, April 1979.
- A. Goicoechea, D. R. Hansen, and L. Duckstein, **Multiobjective Analysis with Engineering and Business Applications**, John Wiley and Sons, New York, 1982.

STEP Method

This method (also known as STEM) is an iterative technique based on the progressive articulation of preferences. The basic idea is to converge toward the 'best' solution in the min-max sense, in no more than k steps, being k the number of objectives. This technique, which is mostly useful for linear problems, starts from an ideal point and proceeds in six steps.

One criticism to this technique is the fact that it assumes that a best-compromise solution does not exist if it is not found after executing all the k steps of its iterative process. This does not give any clue to the DM of what to do [Cohon and Marks, 1975].

STEP Method

Another problem with the STEP Method is that it does not explicitly capture the trade-offs between the objectives. The weights in no way reflect a value judgment on the part of the DM. The weights are artificial quantities, generated by the analyst to reflect deviations from an ideal solution, which is itself an artificial quantity. This definition of the weights serves to obscure rather than capture the normative nature of the multiobjective optimization problems [Cohon and Marks, 1975].

For more details of this technique see:

- R. Benayoun, J. Montgolfier, J. Tergny, and O. Laritchev, "**Linear programming with multiple objective functions: Step Method (STEM)**", *Mathematical Programming*, **1**(3):366–375, 1971.
- J. L. Cohon and D. H. Marks, "**A Review and Evaluation of Multiobjective Programming Techniques**", *Water Resources Research*, **11**(2):208–220, April 1975.

Sequential Multiobjective Problem Solving Method

This method (also known as SEMOPS) was proposed by Monarchi et al. [1973] and it basically involves the DM in an interactive fashion in the search for a satisfactory course of action.

A surrogate objective function is used based on the goal and aspiration levels of the DM. The goal levels are conditions imposed on the DM by external forces, and the aspiration levels are attainment levels of the objectives which the DM personally desires to achieve. One would say, then, that goals do not change once they are stated, but that the aspiration levels may change during the iteration process.

Sequential Multiobjective Problem Solving Method

Operationally, SEMOPS is a three-step procedure involving setup, iteration, and termination. Setup involves structuring a principal problem and a set of auxiliary problems with a surrogate objective function.

The iteration step involves cycling between an optimization phase (by the analyst), and an evaluation phase (by the DM) until a satisfactory solution is reached, if it exists.

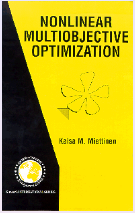
The procedure terminates when either a satisfactory solution is found, or the DM concludes that none of the nondominated solutions obtained are satisfactory and gives up in the search.

Sequential Multiobjective Problem Solving Method

More information on this method can be found in:

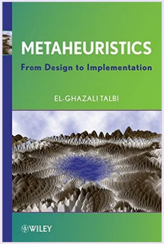
- D. E. Monarchi, C. C. Kisiel, and L. Duckstein, "**Interactive multiobjective programming in water resources: a case study**", *Water Resources Research*, **9**(4):837–850, August 1973.
- A. Goicoechea, D. R. Hansen, and L. Duckstein, **Multiobjective Analysis with Engineering and Business Applications**, John Wiley and Sons, New York, 1982.

Current State



Currently, there are some 30 mathematical programming techniques for nonlinear multi-objective optimization. However, they have several limitations. For example, some of them require that the objectives (and the constraints) are differentiable. Other approaches cannot be applied to disconnected or to non-convex Pareto fronts. Additionally, most of them generate a single solution per algorithmic execution.

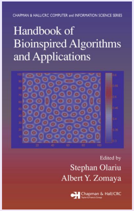
Current State



This has motivated the use of metaheuristics (particularly, bio-inspired metaheuristics).

A **metaheuristic** is a high-level search procedure that applies some form of rule or set of rules based on some source of knowledge, in order to explore the search space in a more efficient way.

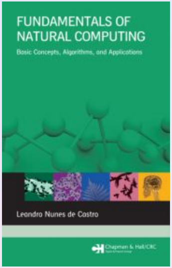
Current State



From the many metaheuristics currently available, one particular class has become very popular in the last 20 years: **bio-inspired metaheuristics**.

Bio-inspired metaheuristics use rules that are inspired on some biological metaphore (e.g., in the case of evolutionary algorithms, the inspiration is Darwin's survival of the fittest principle). Most bio-inspired metheuristics are stochastic search techniques (e.g., evolutionary algorithms, particle swarm optimization, ant colony optimization, etc.).

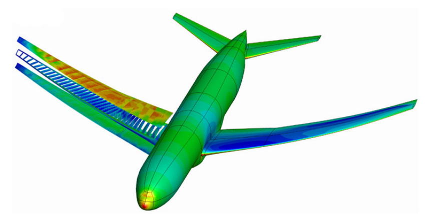
Current State



Most bio-inspired metaheuristics operate on a set of solutions (normally called *population*) at each iteration. A clever use of this population in multi-objective optimization, allows the generation of several elements of the Pareto optimal set in a single algorithmic execution.

Also, bio-inspired metaheuristics require little information about the domain (e.g., they don't require derivatives) and are less susceptible to the shape or continuity of the Pareto front.

Current State



In spite of their several advantages, bio-inspired metaheuristics also have some disadvantages. One of them is that they cannot guarantee convergence to the true Pareto front of a problem in most practical cases. Another one (which is more relevant in practical applications) is that their computational cost is normally significantly higher than that of mathematical programming techniques. This is due to their stochastic nature, which requires sampling several solutions to find an appropriate search direction. This may be unaffordable in some applications (e.g., in aeronautical engineering).