AIAA JOURNAL
Vol. 39, No. 5, May 2001

Mathematical and Pragmatic Perspectives of Physical Programming

Achille Messac* and C. P. Sukam†
Rensselaer Polytechnic Institute, Troy, New York 12180-3590
and
Emanuel Melachrinoudis‡
Northeastern University, Boston, Massachusetts 02115

Physical programming (PP) is an emerging multiobjective and design optimization method that has been applied successfully in diverse areas of engineering and operations research. The application of PP calls for the designer to express preferences by defining ranges of differing degrees of desirability for each design metric. Although this approach works well in practice, it has never been shown that the resulting optimal solution is not unduly sensitive to these numerical range definitions. PP is shown to be numerically well conditioned, and its sensitivity to designer input (with respect to optimal solution) is compared with that of other popular methods. The important proof is provided that all solutions obtained through PP are Pareto optimal and the notion of Pareto optimality is extended to one of pragmatic implication. The important notion of P dominance that extends the concept of Pareto optimality beyond the cases minimize and maximize is introduced. P dominance is shown to lead to the important concept of generalized Pareto optimality. Numerical results are provided that illustrate the favorable numerical properties of physical programming.

I. Introduction

MULTICRITERIA optimization is a useful and challenging activity with application to numerous disciplines. It provides decision makers with the tools for making good decisions and, hopefully, saving time in the decision making process. Most multicriteria optimization strategies require a complete and detailed formulation of the problem; that is, the aggregate objective function, the constraints, and the bounds on the decision variable constraints need to be clearly defined in mathematical terms. This task can be time consuming and a potential source of serious errors. Moreover, this formulation process can be beyond the expertise of many decision makers because they do not usually think in terms of such mathematical models. We believe that this situation severely reduces the potential popularity of optimization. Because decision making is an activity that pervades most of professional life, a more user-friendly form of optimization could have truly broad appeal. This new formal quantitative optimization-based decision making process would replace more conventional trial and error methods based on tentative judgement, intuition, and experience.

For optimization models to be appealing, it must also be possible to construct a formulation that faithfully reflects the actual situation, in terms of both the behavior of the physical system and the decision maker's preference. The optimization problem formulation often fails to include important constraints; most multicriteria optimization approaches are implemented in a black box environment in terms of the physical meaning of the designer-set parameters. Understanding how to set a weight parameter in an optimization model is generally a difficult trial-and-error process. It is not clear how to set these user-defined parameters effectively. More critically, we do not generally know how sensitive the final solution is to these user parameters. The ability to effectively exploit available physical information is usually weak, and the ability to exploit information generated during previous optimization runs is similarly not inherent to the models.

In addition to these concerns are important and fundamental questions:

1) Is the optimal solution generated by a given method always Pareto optimal?

2) Given a Pareto point, can it always be captured by a given method?

3) Is the ratio of changes in optimal solution to changes of model input parameters a reasonably small and finite number?

The second question is addressed in Ref. 1 within the context of generic objective functions. Ongoing research is examining this question within the context of physical programming. Physical programming[2] is a method that requests physically motivated information from the designer and produces a problem statement that reflects the realistic texture of the designer's preferences. It defines a lexicon that significantly impacts the computational burden by entirely eliminating the uncertain, time consuming, and often frustrating process of weight tweaking. This paper examines questions one and three in the case of physical programming and provides positive answers to both. This examination is conducted in conjunction with other popular methods. The issues are thoroughly discussed in Refs. 1–14.

The remainder of this paper is organized as follows. Section II presents a brief outline of physical programming. In Sec. III, the definition of Pareto optimality with respect to the physical programming classification of preferences is introduced, and the proof that physical programming solutions are Pareto optimal is given. In the process of this development, we introduce the important pragmatic concepts of P dominance and generalized Pareto optimality. Section IV provides mathematical and physical insight into the numerical stability of several popular methods regarding the sensitivity of optimal solutions to user input parameters. The role of the curvature of the aggregate objective function hypersurface is examined within the context of numerical examples. In Sec. V, the issues of Sec. IV are again addressed, this time in the case of physical programming. In the latter case the movement of optimal solutions on the Pareto frontier, caused by changes in user-defined preference input parameters, is examined. Concluding remarks are presented in Sec. VI.

II. Physical Programming Synopsis

This section provides a synopsis of the physical programming (PP) method. For a comprehensive presentation, see Ref. 2. PP is intended to be a simple and user-friendly optimization method that required negligible knowledge of optimization. The application of

Received 23 February 2000; revision received 30 May 2000; accepted for publication 9 June 2000.

*Associate Professor, Department of Mechanical Engineering, Aeronautical Engineering and Mechanics; messac@rpi.edu. Associate Fellow AIAA.

†Graduate Student, Department of Mechanical Engineering, Aeronautical Engineering and Mechanics. Student Member AIAA.

‡Associate Professor, Department of Mechanical Engineering, Multidisciplinary Design Laboratory.

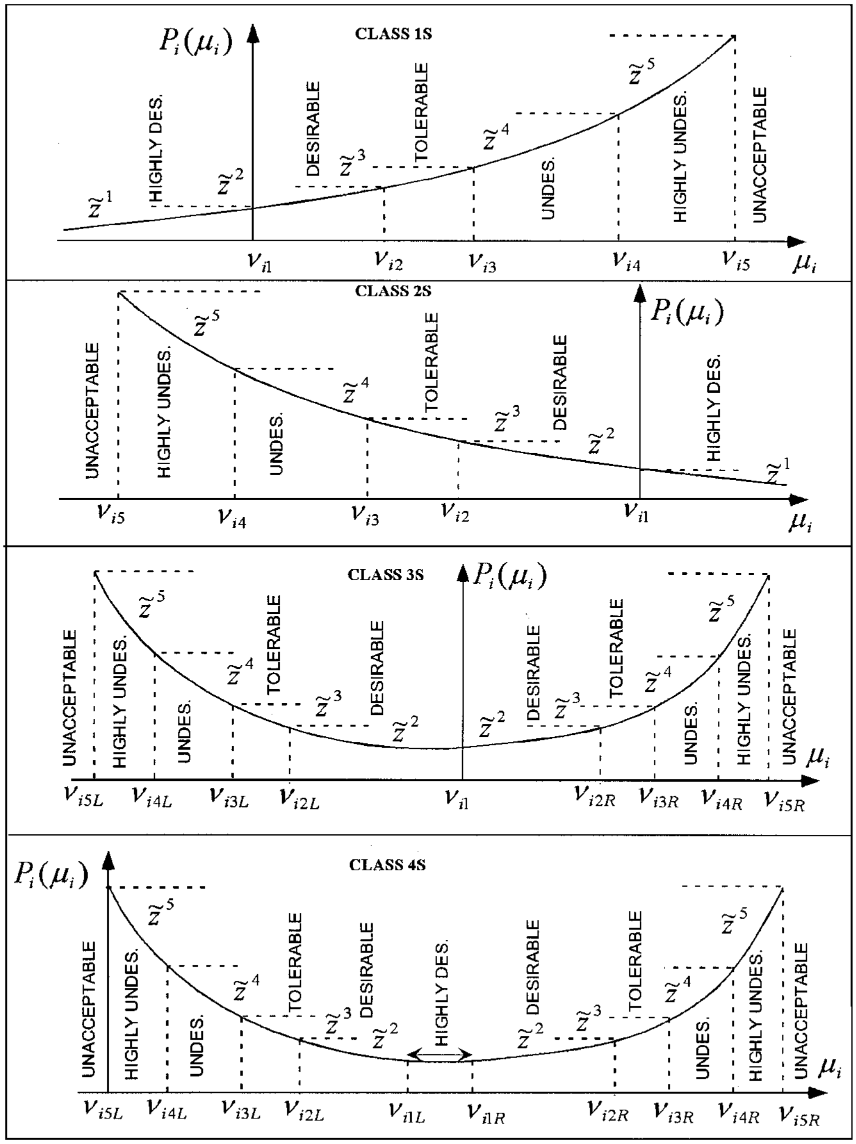


Fig. 1 Class function ranges for *i*th generic design metric.

PP employs a flexible and more natural problem formulation framework. In PP, the designer does not need to specify optimization weights in the problem formulation phase; rather, the designer specifies ranges of different degrees of desirability for each design objective.

PP also addresses the inherent multiobjective nature of design problems, where multiple conflicting objectives govern the search for the best solution. PP provides a flexible and more deterministic approach to obtaining a solution that satisfies the typically complex texture of a designer's preferences. PP is implemented in the software package entitled PhysPro,[15] which is MATLAB® based. Some of the important concepts used in the PP method are described as follows.

Design Metrics

The problem formulation involves identifying the characteristics of the system, or design, that allow the designer to judge the effectiveness of the outcome. Those characteristics, or design metrics, are denoted by μ_i, which are components of the vector $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_m)$. The elements μ_i represent behavior. Design metrics may be quantities that the designer wishes to minimize, maximize, take on a certain value (goal), fall within a particular range, or to be less than, greater than, or equal to particular values. The designer defines preference with respect to each design metric by providing certain numerical values. The design metric may become part of an aggregate objective function that will be minimized or may instead be treated as inequality or equality constraints that subjugate the aggregate objective function.

Classification of Objectives and Class Functions

Within the PP procedure, the engineer expresses objectives with respect to each design metric using four different classes. Each class comprises two cases, hard and soft, referring to the sharpness of the preference. All soft class functions will become constituent components of the aggregate objective function. Figure 1 shows the qualitative meaning of each soft class. The value of the design metric under consideration, μ_i, is on the horizontal axis, and the function that will be minimized for that objective, $P_i(\mu_i)$, called the class function, is on the vertical axis.

The desired behavior of a generic design metric is described by one of eight subclasses, four soft and four hard. These classes are characterized as follows. Soft consists of class 1S, smaller is better, that is, minimization; class 2S, larger is better, that is, maximization; class 3S, value is better; and class 4S, range is better. Hard consists of class 1H, must be smaller, that is, $\mu_i \leq \mu_{i,\max}$; class 2H, must be larger, that is, $\mu_i \geq \mu_{i,\min}$; class 3H, must be equal, that is, $\mu_i = \mu_{i,\text{val}}$; and class 4H, must be in range, that is, $\mu_{i,\min} \leq \mu_i \leq \mu_{i,\max}$.

For each of these classes, we form a class function (Fig. 1). The class functions provide the means for a designer to express the spectrum of preferences for a given design metric. The class functions provide information that is deliberately imprecise. By design, the utopian value of the class functions is zero. Next, we explain how quantitative specifications are associated with each design metric.

PP Lexicon

PP defines a lexicon that provides the means to express preference in a flexible way. This lexicon comprises terms that characterize the degree of desirability of 6 ranges for each generic design metric for classes 1S and 2S, 10 ranges for class 3S, and 11 for class 4S. To illustrate, consider the case of class 1S. The ranges are defined as follows, in order of decreasing preference: highly desirable range ($\mu_i \leq \nu_{i1}$), desirable range ($\nu_{i1} \leq \mu_i \leq \nu_{i2}$), tolerable range ($\nu_{i2} \leq \mu_i \leq \nu_{i3}$), undesirable range ($\nu_{i3} \leq \mu_i \leq \nu_{i4}$), highly undesirable range ($\nu_{i4} \leq \mu_i \leq \nu_{i5}$), and unacceptable range ($\mu_i \geq \nu_{i5}$).

The parameters ν_{i1}–ν_{i5} are physically meaningful constants that are specified by the designer to quantify the preference associated with the ith design metric. These parameters delineate the ranges for each design metric.

The class functions map design metrics into nondimensional, strictly positive real numbers. This mapping, in effect, transforms design metrics with disparate units and physical meaning onto a dimensionless scale through a unimodal function. Figure 1 shows the mathematical nature of the class functions and shows how they allow a designer to express the ranges of differing goodness, or preferences, for a given design metric. Consider the first curve of Fig. 1, the class function for class 1S design metrics. Six ranges are defined. The parameters ν_{i1}–ν_{i5} are specified by the designer. When the value of the objective μ_i is less than ν_{i1} (highly desirable range), the value of the class function is small. This calls for little further minimization of the class function. When, on the other hand, the value of the objective μ_i is between ν_{i4} and ν_{i5} (highly undesirable range), the value of the class function is large. This calls for significant minimization of the class function. The behavior of the other class functions is indicated in Fig. 1. Preferences regarding each design metric are treated independently, which allows the inherent multiobjective nature of the problem to be preserved. The preceding discussion describes the basic process of PP: The value of the class function for each design metric governs the optimization path in objective space.

In the next section, after defining the concept of Pareto optimality and its generalized extension for application in PP and other methods, the Pareto optimality of solutions obtained using PP is established.

III. PP Pareto Optimality, P Dominance, Generalized Pareto Optimality, and Fundamental Theorem

This section begins by examining the generic definition of Pareto optimality within the context of a general multiobjective problem statement, that is, one where the design metrics will not simply be minimized and maximized. This development then leads us to the definition and proof of the fundamental theorem addressing the Pareto optimality of PP optimal solutions.

We begin with the multiobjective problem statement and associated definitions. Various problems in practical applications of conventional optimization, particularly in engineering design, can be recast in the form of a nonlinear multiobjective problem.

Problem P:

$$\min_{x\in D} \boldsymbol{\mu}(\boldsymbol{x}) = \begin{pmatrix} \mu_1(\boldsymbol{x}) \\ \mu_2(\boldsymbol{x}) \\ \vdots \\ \mu_m(\boldsymbol{x}) \end{pmatrix} \tag{1}$$

where $D = \{\boldsymbol{x} \in R^n \mid \boldsymbol{h}(\boldsymbol{x}) = 0,\ \boldsymbol{g}(\boldsymbol{x}) \leq 0,\ \boldsymbol{x}_{\min} \leq \boldsymbol{x} \leq \boldsymbol{x}_{\max}\}$, with $\boldsymbol{h}: R^n \to R^{n_e}$, $\boldsymbol{g}: R^n \to R^{n_i}$, $\boldsymbol{x}_{\min} \in (R \cup \{-\infty\})^n$, and $\boldsymbol{x}_{\max} \in (R \cup \{+\infty\})^n$; m is the number of objectives, $m \geq 2$; n_e and n_i are the numbers of equality and inequality constraints, respectively; and $\boldsymbol{\mu}_i$ is at least twice differentiable for $i \in \{1, \ldots, m\}$. For any decision vector $\boldsymbol{x} = (x_1, \ldots, x_n)$, a criterion vector $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_m)$ is defined according to the function $\boldsymbol{\mu}: R^n \to R^m$. Let $Z = \{\boldsymbol{z} \in R^m \mid \boldsymbol{z} = \boldsymbol{\mu}(x), \boldsymbol{x} \in D\}$ be the set of images of all points in D, where D is the feasible region in the decision space and Z the feasible region in the objective space; $[\mu_1(\boldsymbol{x}), \ldots, \mu_m(\boldsymbol{x})]$ are the coordinates of the image of $\boldsymbol{x}$ in the objective space.

It is highly improbable that a single $\boldsymbol{x}^* \in P$ will minimize all μ_i simultaneously; therefore, we seek Pareto optimal solutions.[16] A solution to a multiobjective programming problem is said to be Pareto optimal (noninferior, nondominated) if there exists no other feasible solution that will yield an improvement in one design metric without causing a degradation in at least one other design metric. Thus, $\boldsymbol{x} \in D$ is Pareto optimal if there does not exist $\boldsymbol{y} \in D$, whose design metric vector, $\boldsymbol{q} = \boldsymbol{\mu}(\boldsymbol{y})$, dominates the design metric vector of $\boldsymbol{x}$, $\boldsymbol{p} = \boldsymbol{\mu}(\boldsymbol{x})$; that is, $\boldsymbol{q} \leq \boldsymbol{p}$ (componentwise) with $\boldsymbol{p} \neq \boldsymbol{q}$. In other words, there does not exist $\boldsymbol{y} \in D$ satisfying $\mu_i(\boldsymbol{y}) \leq \mu_i(\boldsymbol{x})$ for all $i \in \{1, \ldots, m\}$ and $\mu_j(\boldsymbol{y}) < \mu_j(\boldsymbol{x})$ for some $j \in \{1, \ldots, m\}$. A point $\boldsymbol{x} \in D$ is said to be locally Pareto optimal if there is a neighborhood of that point in D where it is Pareto optimal. Thus, a point, which is Pareto optimal, that is, globally, is also locally Pareto optimal. The definition of Pareto optimality is extended to design metric vectors as well. Therefore, in the preceding definition, $\boldsymbol{p} \in Z$ is Pareto optimal (respectively, locally Pareto optimal) in the objective space.

In conventional mathematical formalism, all soft design metrics are generally minimized or maximized. For minimization, a smaller value of the objective is always preferred to a larger value thereof, and for maximization, the contrary applies. As stated, PP recognizes the limitations of such a problem formulation framework by employing a new expansive and flexible lexicon. As we recall, in classes 3 and 4, we define the optimization of a design metric in terms of reaching a given value or a given range. Note that in the cases of classes 3 and 4, which are very common in practice, the preceding definition of Pareto optimality practically fails. In these important cases, it is not possible to tell whether a smaller or a larger value of the design metric is better or worse, thereby striking at the heart of the conventional definition of Pareto optimality. Therefore, it is helpful to extend the definition of Pareto optimality in a way that accounts for these cases. Next, we define the concept of Pareto optimality with respect to the PP classification of preferences (and other methods). We assume that the design metrics are $\mu_1(\boldsymbol{x}), \ldots, \mu_{n_{sc}}(\boldsymbol{x}), \ldots, \mu_m(\boldsymbol{x})$, where n_{sc} is the number of soft design metrics. The PP[2] problem model takes the form referred to here as problem (P1). Specifically, we have the following.

Problem (P1):

$$\min_x P(\mu(x)) = \frac{1}{n_{sc}} \sum_{i=1}^{n_{sc}} P_i(\mu_i(x)) \tag{2}$$

subject to

$$\mu_i(\boldsymbol{x}) \leq \nu_{i5} \tag{2a}$$

for class 1S metrics,

$$\mu_i(\boldsymbol{x}) \geq \nu_{i5} \tag{2b}$$

for class 2S metrics,

$$\nu_{i5L} \leq \mu_i(\boldsymbol{x}) \leq \nu_{i5R} \tag{2c}$$

for class 3S metrics,

$$\nu_{i5L} \leq \mu_i(\boldsymbol{x}) \leq \nu_{i5R} \tag{2d}$$

for class 4S metrics,

$$\mu_i(\boldsymbol{x}) \leq \nu_{i,\max} \tag{2e}$$

for class 1H metrics,

$$\mu_i(\boldsymbol{x}) \geq \nu_{i,\min} \tag{2f}$$

for class 2H metrics,

$$\mu_i(\boldsymbol{x}) = \nu_{i,\mathrm{val}} \tag{2g}$$

for class 3H metrics,

$$\nu_{i,\min} \leq \mu_i(\boldsymbol{x}) \leq \nu_{i,\max} \tag{2h}$$

for class 4H metrics,

$$x_{i,\min} \leq x_i \leq x_{i,\max} \tag{2i}$$

for decision variable constraints, where $\nu_{i,\min}$, $\nu_{i,\max}$, $\nu_{i,\mathrm{val}}$, $x_{i,\min}$, and $x_{i,\max}$ represent prescribed values. The aggregate objective function comprises class functions associated with soft design metrics. The hard design metrics are treated as constraints. The set of decision vectors satisfying Eq. (2a–2i) define the feasible region D.

To extend the concept of Pareto optimality to the common case of classes 3 and 4, we begin by introducing pertinent terminology. Consider the design parameter vectors $\boldsymbol{x}, \boldsymbol{y} \in D$, and a soft design metric μ_i, $i \in \{1, \ldots, n_{\mathrm{sc}}\}$. The vector $\boldsymbol{x}$ is said to be P dominated by $\boldsymbol{y}$ with respect to μ_i if $P_i[\mu_i(\boldsymbol{y})] < P_i[\mu_i(\boldsymbol{x})]$. We emphasize here an important distinction. The notion of P dominance with respect to a particular design metric is distinct from the conventional notion of dominance in the context of several design metrics. The similarities and distinctions of these two definitions are discussed in the following.

The preceding definition implies that $\mu_i(\boldsymbol{y}) < \mu_i(\boldsymbol{x})$ for μ_i belonging to class 1S (minimization), and $\mu_i(\boldsymbol{x}) < \mu_i(\boldsymbol{y})$ for μ_i belonging to class 2S (maximization). In the case of classes 3S and 4S, we are not able to predict the relationship between $\mu_i(\boldsymbol{x})$ and $\mu_i(\boldsymbol{y})$ even when we know that one is preferred over the other (see Fig. 1, classes 3S and 4S). To see this, consider the following situation. Let $\mu_i(\boldsymbol{x})$ belong to the left desirable region (class 4, Fig. 1). Let it be known that the designer prefers $\mu_i(\boldsymbol{y})$ over $\mu_i(\boldsymbol{x})$. Therefore, we know that $P_i[\mu_i(\boldsymbol{y})] < P_i[\mu_i(\boldsymbol{x})]$. However, we have no way to tell which is the smaller of $\mu_i(\boldsymbol{y})$ and $\mu_i(\boldsymbol{x})$ because $\mu_i(\boldsymbol{y})$ may be on the right or left of $\mu_i(\boldsymbol{y})$ in one of the, for example, tolerable regions. For example, $\nu_{i2L} \leq \mu_i(\boldsymbol{y}) \leq \nu_{i1L}$ and $\{\nu_{i3L} \leq \mu_i(\boldsymbol{x}) \leq \nu_{i2L}$ or $\nu_{i2R} \leq \mu_i(\boldsymbol{x}) \leq \nu_{i3R}\}$. Therefore, we see that any definition of Pareto optimality that relies on the relative size of the design metrics will encounter difficulties in the common case of classes 3 and 4.

We now address the notion of P dominance in the important context of m design metrics. The design parameter $\boldsymbol{x}$ is said to be P dominated by $\boldsymbol{y}$ (or equivalently, $\boldsymbol{y}$ P dominates $\boldsymbol{x}$) if there exists at least one $k \in \{1, \ldots, n_{\mathrm{sc}}\}$ such that $\boldsymbol{x}$ is P dominated by $\boldsymbol{y}$ with respect to μ_k; for each $j \in \{1, \ldots, n_{\mathrm{sc}}\}$, $j \neq k$, $\boldsymbol{y}$ is not dominated by $\boldsymbol{x}$ with respect to μ_j. This means that $P_k[\mu_k(\boldsymbol{y})] < P_k[\mu_k(\boldsymbol{x})]$ and $P_j[\mu_j(\boldsymbol{y})] \leq P_j[\mu_j(\boldsymbol{x})]$ for $j \in \{1, \ldots, n_{\mathrm{sc}}\}$, $j \neq k$.

We observe that the definition of P dominance is the natural extension of the definition of dominance in the parlance of conventional optimization. Thus, $\boldsymbol{x}$ is a P-nondominated vector if any further improvement in any one of the design metric values $\mu_k(\boldsymbol{x})$ requires a worsening of at least one other design metric value. The earlier definition leads us to the important concept of generalized Pareto optimality, which is defined as follows. A point $\boldsymbol{x} \in D$ is said to be generalized Pareto optimal (GPO) if there does not exist a point $\boldsymbol{y}$ that P dominates $\boldsymbol{x}$. In addition, a point $\boldsymbol{x} \in D$ is said to be locally GPO if there is a neighborhood of the point $\boldsymbol{x}$ in D, where $\boldsymbol{x}$ is GPO.

These definitions formally extend the well-known concepts of Pareto optimality and dominance. We note, first, that although the preceding discussion took place in the context of classes 3 and 4 of PP-defined preferences, the definitions are directly applicable and helpful in the context of any other multiobjective optimization approach. In fact, these definitions allow these approaches to seemlessly extend their applications from minimization (class 1S) and maximization (class 2S) to the important and common cases of classes 3 and 4. We note, second, that the discussion of these new concepts is in no way limited to the way the class functions are defined. That is, we could very well have defined a higher value of the preference function to be preferred over a lower value thereof (see Fig. 1).

We conclude the discussion of these extended concepts by making some insightful and useful observations. As discussed in the preceding section, one property commonly considered necessary for any candidate solution of a multiobjective optimization problem is that the solution be Pareto optimal. Pareto optimality allows the designer to restrict attention to a single solution or a very small subset of solutions among the much larger set of feasible solutions. Pareto optimality is always defined within the context of designer preference, for example, minimize the design metrics. Unfortunately, conventional definitions of Pareto optimality are generally formulated in terms of the relative values of these design metrics.

As we see from the preceding discussion, in the case of classes 3 and 4, these definitions degenerate. These definitions are defined in design metric space. In light of this realization, we see that a more appropriate space for the definition of Pareto optimality is the preference space. In other words, in Fig. 1, instead of using the design metrics (horizontal axes) to define Pareto optimality, we should use the preference measures (vertical axes). In preference space, the traditional definition of Pareto optimality applies, where the word design metric is replaced by preference measure (in the PP case, the vertical axes in Fig. 1). This realization naturally leads to the notion of P dominance, P for preference (or PP, as a special case). This concept of P dominance naturally leads to that of GPO.

We now have the requisite tools to directly address the chief topic of this section, namely, the proof that all optimal solutions generated using PP are Pareto optimal, or, more specifically, GPO. This development follows, in the form of a theorem and proof.

PP GPO

Theorem: Let $\boldsymbol{a} \in D$; if $\boldsymbol{a}$ is the optimal vector in problem (P1), then $\boldsymbol{a}$ is GPO.

Proof: Suppose that $\boldsymbol{a}$ is not GPO; then, there exists a point $\boldsymbol{c} \in D$ such that $\boldsymbol{a}$ is P dominated by $\boldsymbol{c}$; thus, there exists at least one index k, with $k \in \{1, \ldots, n_{\mathrm{sc}}\}$, such that $P_k[\mu_k(\boldsymbol{c})] < P_k[\mu_k(\boldsymbol{a})]$ and $P_j[\mu_j(\boldsymbol{c})] \leq P_j[\mu_j(\boldsymbol{a})]$ for all $j \in \{1, \ldots, n_{\mathrm{sc}}\}$, $j \neq k$. Because $\boldsymbol{a}$ is the optimal vector in problem (P1), then $P[\boldsymbol{\mu}(\boldsymbol{a})] = \min_x \boldsymbol{P}[\boldsymbol{\mu}(\boldsymbol{x})]$. We have

$$P[\boldsymbol{\mu}(\boldsymbol{c})] - P[\boldsymbol{\mu}(\boldsymbol{a})] = \frac{1}{n_{\mathrm{sc}}} \sum_{i=1}^{n_{\mathrm{sc}}} P_i[\mu_i(\boldsymbol{c})] - \frac{1}{n_{\mathrm{sc}}} \sum_{i=1}^{n_{\mathrm{sc}}} P_i[\mu_i(\boldsymbol{a})]$$

$$= \frac{1}{n_{\mathrm{sc}}} \sum_{i=1}^{n_{\mathrm{sc}}} \{P_i[\mu_i(\boldsymbol{c})] - P_i[\mu_i(\boldsymbol{a})]\} \tag{3}$$

It follows from Eq. (3) that $P[(\boldsymbol{\mu}(\boldsymbol{c})] - P[\boldsymbol{\mu}(\boldsymbol{a})]$ is negative; therefore, $P[\boldsymbol{\mu}(\boldsymbol{c})] < P[\boldsymbol{\mu}(\boldsymbol{a})]$. This contradicts that $P[\boldsymbol{\mu}(\boldsymbol{a})] = \min_x P[\boldsymbol{\mu}(\boldsymbol{x})]$. Thus, $\boldsymbol{a}$ is not P dominated by $\boldsymbol{c}$. Hence, $\boldsymbol{a}$ is GPO.

From a pragmatic perspective, we now make some comments that relate theory to practice. The most common computational codes for nonlinear optimization are gradient based and can at best guarantee local Pareto optimality of the obtained solution. Consequently, if the code used for the implementation of the PP problem (P1) gives only local minima, then the solution to problem (P1) is locally GPO. Therefore, to guarantee (global) generalized Pareto optimality, we simply need to use a global optimization code as the PP optimization engine. With the completion of the developments of P dominance, generalized Pareto optimality, and proof of the generalized Pareto optimality of PP's solutions, we address the next topic, namely, the sensitivity of PP's solution to changes in user parameter preference input.

IV. Sensitivity of Optimal Solutions to Changes in User Parameters: Other Methods

All practical multiobjective optimization/design methods involve the participation of the designer in several ways. One of the most important aspects of this participation involves the designer's specification of input parameters that guide the eventual optimal solution. As the designer changes these input parameters, the optimal solution location moves on the Pareto frontier. Some forms of these input parameters are commonly called weights. Others, such as the exponent of a design metric, are not generally referred to as weights because they are not primary tools used for navigating in the objective space. These parameters primarily affect the curvature of the aggregate objective function. The PP method, as mentioned earlier, uses instead physically meaningful preference parameters, which play in essence the same role as weights in other methods. Some methods explicitly seek to generate a comprehensive representation of the Pareto frontier. When the simplicity of the problem allows, a closed form of the Pareto frontier is derived.[17] More recently, attempts have been made to approximate the entire frontier.[4,5,14,18,19] Our pervasive use of input parameters to guide our eventual solution gives rise to some important and fundamental questions.

1) Does the designer have reliable guidance regarding how to change these input parameters to converge to the desired optimal solution?

2) What is the impact of the changes in input parameters on the shape of the aggregate objective function hypersurface? More important, can these shape changes result in the arbitrary ability to capture all Pareto points?

3) Do the aggregate objective function shape changes result in the arbitrary ability to obtain all Pareto points? More important, does a small change in input parameters yield to a correspondingly small change in the optimal solution?

First Question: Guidance in Changing Input Parameters

These questions deserve our careful examination. The first is thoroughly examined in Ref. 2, which concludes that the typical weight-tweaking process is a problematic area of multiobjective optimization and proposes the PP method as an effective alternative. The second and third questions are examined in the following, within the context of PP and other methods.

Second Question: Hypersurface Changes

The answer to the second question has profound implications. In Ref. 1, it is shown that the very ability of an aggregate objective function (AOF) to yield expected Pareto solutions depends on the curvature of the hypersurface of the AOF at the optimum point, that is, at the point where the AOF supports the Pareto frontier. Therefore, we should expect these input parameters to give us the arbitrary ability to change the curvature of the AOF. In fact, such ability is a prerequisite if we wish to be able to generate any and all Pareto points. We should have the ability to obtain a vanishing radius of curvature of the AOF as we appropriately alter the input parameters.

To further examine this question and gain insight into the behavior of different generic AOF structures, we study the behavior of three common AOFs namely, the weighted sum method, the weighted-square-sum method, and the compromise programming method. The case of PP is examined later in more depth.

Weighted sum (WS):

$$J = w_1\mu_1 + w_2\mu_2 \tag{4a}$$

or

$$J = \mu_1 + w_r\mu_2, \qquad w_r = w_2/w_1 \tag{4b}$$

Weighted square sum (WSS):

$$J = w_1\mu_1^2 + w_2\mu_2^2 \tag{5}$$

Compromise programming (CP):

$$J = w_1\mu_1^s + w_2\mu_2^s \tag{6}$$

For each of these cases, we study the behavior of the shapes of the AOFs. The first column of Fig. 2 shows pertinent graphs. To represent the changes in input parameters, we let the weights w_1 and w_2 change according to the information provided in Table 1. When the weights are varied according to Table 1, we see that the change in AOF's hypersurface is, in fact, very favorable because this change pattern of the weights makes it possible to cover the whole spectrum of possibilities. In this way, we are taking a conservative approach in our eventual conclusions. In reality, we usually change weights more in line with Eq. (4b), for which the equivalent weight w_r would be highly uneven. If we implement an even rate of change in w_r, we will only cover a small portion of the intended spectrum of weights. These unsettling observations provide useful insights into some of the sources of deficiencies of weight-based methods. In Table 1, from case 2 to case 1, w_r jumps from 9 to ∞.

Table 1 Input parameter variation

Case	w_1, γ_1	w_2, γ_2	w_r
1	0	1	∞
2	0.1	0.9	9
3	0.2	0.8	4
4	0.3	0.7	2.3
5	0.4	0.6	1.5
6	0.5	0.5	1
7	0.6	0.4	0.6
8	0.7	0.3	0.42
9	0.8	0.2	0.25
10	0.9	0.1	0.11
11	1	0	0

First, we consider the case of the WS, the simplest and most popular method used in practice and arguably the most deficient.[1] We note that the WS method represents a hyperplane whose slope changes as the input parameters (weights) vary. Whereas this slope change allows for the WS AOF to yield a series of Pareto points along the Pareto frontier, the structure of the WS method fatally fails to permit any change in AOF curvature. As a result, the WS method cannot yield any desired Pareto solution located on concave Pareto frontiers. In Fig. 2a, we represent the effect of the weight changes [Eq. (4a)] by plotting the corresponding slopes of the AOF. We note that the slopes change evenly, but the slope change that would result from an even change of w_r would not be at all even.

Second, we consider the case of the WSS, shown in Eq. (5). As explained in Ref. 1, this AOF structure is significantly more flexible than that of the WS. Figure 2c shows the contour lines of the AOF for $J = 1$, for different values of the weights according to Table 1. We note that the shape of the AOF changes as the weights change and so does its curvature. These are desirable properties. However, the weights alone are not able to arbitrarily alter the curvature. To do so, we can use the CP method, discussed next.

The CP AOF is represented in Eq. (6). Figure 2e shows the impact of changing the power s in Eq. (6). These contour lines (for $J = 1$) were developed for case 2 (Table 1); and for $s = 2$, 3, 4, 5, and 12. Note that by increasing the power s we are able to arbitrarily increase curvature. This ability makes the CP method able to capture, in theory, all Pareto points.

We have established the respective abilities of three popular AOFs to change the shape of their hypersurface as their input parameters are varied. Next, we examine the influence of these changes in input parameters on the optimal solutions obtained.

Third Question: Hypersurface Shape Changes, Numerical Conditioning

We are interested in two important related questions. First, do these changes in AOF (resulting from weights' changes) provide the arbitrary ability to obtain all Pareto points? Second, does a small change in input parameters result in a correspondingly small change in the solution?

To answer the first question, the developments of Ref. 1 are helpful. Here, we only examine this question within the context of the following five optimization examples:

$$\min \boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} \tag{7a}$$

subject to

$$0 \le \mu_1 \le 1 \tag{7b}$$

$$0 \le \mu_2 \le 1 \tag{7c}$$

and for example 1 only,

$$(\mu_1 - 1)^4 + (\mu_2 - 1)^4 \le 1 \tag{8}$$

for example 2 only,

$$(\mu_1 - 1)^2 + (\mu_2 - 1)^2 \le 1 \tag{9}$$

for example 3 only,

$$\mu_1 + \mu_2 \ge 1 \tag{10}$$

for example 4 only,

$$\mu_1^2 + \mu_2^2 \ge 1 \tag{11}$$

and for example 5 only,

$$\mu_1^4 + \mu_2^4 \ge 1 \tag{12}$$

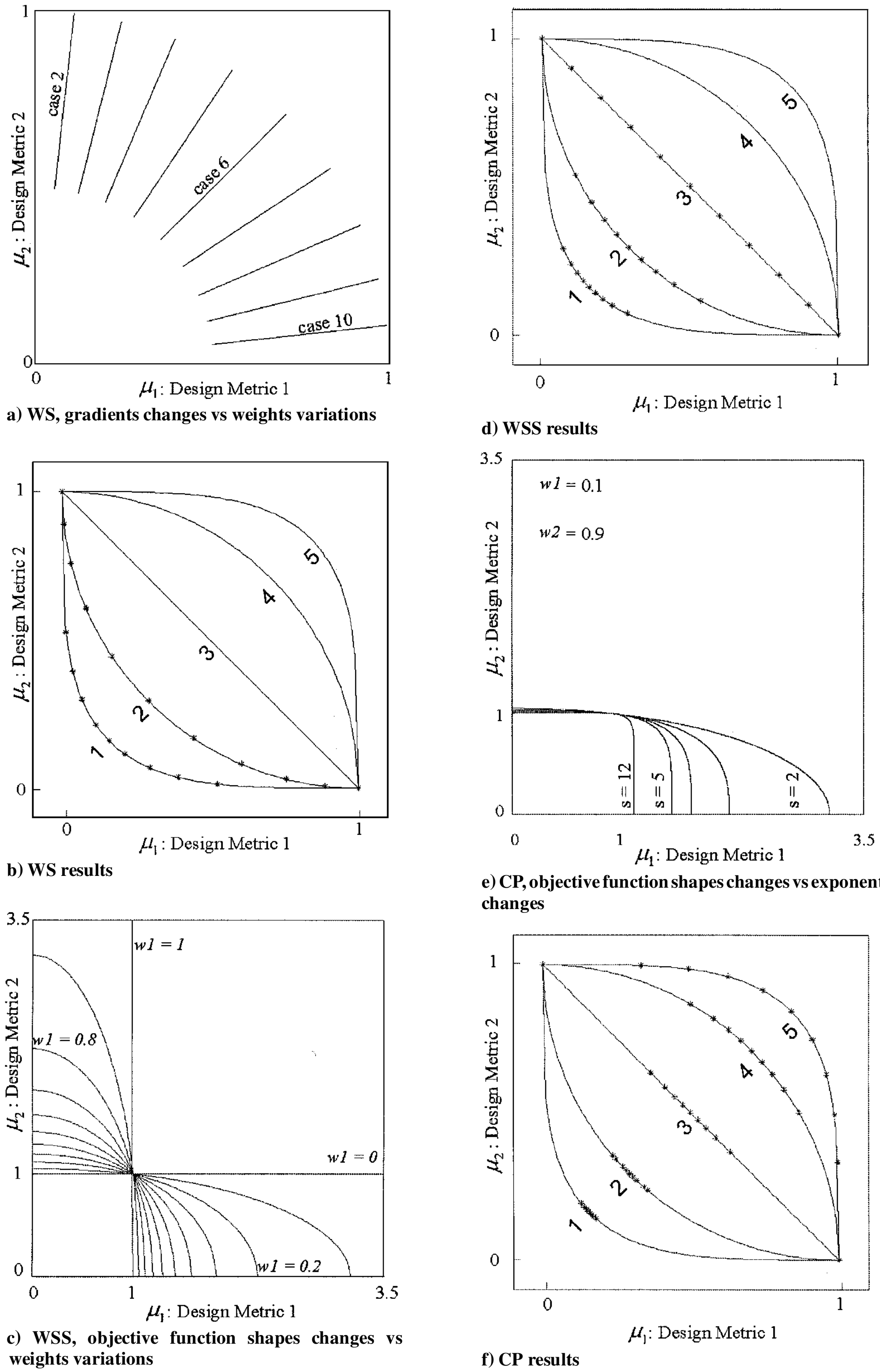


Fig. 2 Sensitivity of optimal results to input parameter variation.

These examples were designed to provide relatively extreme cases of AOF curvature: negative, zero, and positive.

For the WS AOF, the results of the optimization examples are reported in Fig. 2b. As we see, for example 1 (the leftmost curve) the optimal solutions reflecting the weights of Table 1 are fairly well distributed except near the endpoints where the distance between solutions is significantly larger. For example 2, the distribution of the solutions is improved, as the Pareto frontier is more benign. In the cases of examples 3–5, however, the deficiencies of the WS method exhibit themselves. For example 3, the situation is degenerate, and we can only reliably generate the endpoints. For examples 4 and 5, only the endpoints can be generated at all. The WS cannot yield solutions on concave Pareto frontiers.

We now consider the WSS AOF, whose optimization results are presented in Fig. 2d. In this case, the distribution of the solutions is not favorable for examples 1 and 2, but fairly even for example 3. However, for examples 4 and 5, no solution could be obtained, except for the endpoints. The WSS AOF could not yield the required curvature (see Ref. 1).

The CP AOF resulted in a mixed assessment (see Fig. 2f). On the positive side, it could capture solutions in all regions of all examples. On the negative side, however, the distribution of the solutions was not particularly even. That is, for an even distribution of changes in input parameters, we did not obtain an even distribution of solutions. To capture the points in example 5, we used the exponent $s = 6$ in Eq. (6) ($s = 2$ would be equivalent to the WSS).

In summary, the WS AOF yielded a good distribution of solutions, but did so only for convex Pareto frontiers. The distribution was better for more benign Pareto frontiers (ones with relatively even curvatures). The WSS AOF worked well in the case of a straight line Pareto frontier but displayed a poorer solution distribution and did not work for Pareto frontiers of high curvatures. The CP AOF is the most flexible, but may also display the worst distribution of solutions. Therefore, this strong flexibility in the CP method to capture Pareto points is not easily exploitable. We cannot rely on hope that even changes in input parameters will allow us to navigate in the objective space in an even/predictable way. We now examine the issues addressed in the case of PP. This latter case is sufficiently different from the former to warrant its own section.

V. Sensitivity of Optimal Solutions to Changes in User Parameters: PP

In this section, we examine the behavior of the optimal solution, that is, its movement on the Pareto frontier, when the user input parameters are varied. We parenthetically note that the issues at hand are different from those of sensitivity of performance to design parameter uncertainties.[20] The preceding section addressed this issue for three popular methods: WS, WSS, and CP. This examination is significantly different in the case of PP. Whereas in the preceding cases the user parameters were physically meaningless but relatively few, the number of PP user input parameters is larger. Fortunately, the PP user inputs are physically meaningful preference parameters pertinent to each design metric. To study the sensitivity issues, we need to devise a way to examine the effects of these preference parameter changes. In addition, we need this way to be an analog of how we changed the weights in Sec. IV, that is, Table 1. To perform the examination of this section, we proceed as follows. 1) We decide how to change the user preference inputs. 2) We examine the consequence of these changes within the context of the examples defined in Sec. IV.

Preference-Change Method

To represent a designer's preference of one design metric relative to another design metric (let us consider class 1S for simplicity), we shift the preferences leftward or rightward uniformly. That is, we are keeping the lengths of same ranges in Fig. 1 equal. For example, all of the desirable ranges have equal length; the same applies to tolerable, undesirable, and highly undesirable ranges (for classes 1 and 2, the highly desirable range has infinite length). When we shift the preferences to the right, we place increased importance on minimizing the respective design metric; the converse applies to a shift to the left. In this way, we can simulate an analog of the weight changes of Table 1 by similarly shifting the preferences. We will then observe the corresponding changes in the optimal solutions. Let us examine what happens to the AOF hypersurface when we change the preferences in this fashion.

AOF Shape Change with Respect to Preference Changes

We have decided to change the preferences such that the lengths of the ranges are constant. Furthermore, according to Ref. 2, the class function values at the regions' intersections are also constant. Specifically (see Fig. 1), we have

$$P_i(v_{ik}) = z^k \qquad \forall i, k = 1, \ldots, 5 \tag{13}$$

for the ith design metric. Therefore, the change that takes place as one travels across the kth region is also constant and given by

$$\tilde{z}^k \equiv z^k - z^{k-1}, \qquad k = 2, 3, 4, 5, \qquad z^1 \equiv \tilde{z}^1 = 0.1 \tag{14}$$

We have

$$\tilde{z}^k = \lambda n_{\text{sc}} \tilde{z}^{k-1}, \qquad \lambda > 1, \qquad k = 2, 3, 4, 5 \tag{15}$$

where n_{sc} is the number of soft design metrics. Equation (15) reminds us (see Ref. 2) that the vertical excursions across any same regions are equal. Similarly, by assumption, we have taken the region lengths to be constant (in fact, equal). That is,

$$\bar{v}_{ik} = v_{ik} - v_{i(k-1)} \equiv \boldsymbol{a}_i/4, \qquad k = 2, 3, 4, 5 \tag{16}$$

At this point, we are ready to state that the AOF hypersurface will not change its shape when we alter the preferences in the manner discussed. This is because the surface shape only depends on $\bar{v}_{ik}$ and $\tilde{z}^k$ (see Ref. 2 for details).

Determining the Shifting Vector

Note here that some detailed aspects of the strategy for generating the PP solutions are not germane to the topic of the paper and are, therefore, presented concisely. A complete presentation of the generation of evenly spaced Pareto solutions using PP is presented in Ref. 21. The interested reader is advised to refer to that publication. The first step in the procedure is to determine the extrema of each design metric. The design metrics are minimized individually, and the minimum value of the ith design metric, $F_{\min,i}$, and the corresponding values of the other objectives are used to form a row vector $\boldsymbol{R}_i$ of n_{sc} elements. This vector corresponds to a point in the efficient set where the value of the ith design metric is a minimum. Similarly, the vector $\boldsymbol{R}_i$ of all of the n_{sc} soft design metrics are calculated, and the matrix F is formed. The structure of F is as follows:

$$F = \begin{bmatrix} F_{\min,1} & F_{1,2} & \cdot & F_{1,n_{\text{sc}}} \\ F_{2,1} & F_{\min,2} & \cdot & F_{2,n_{\text{sc}}} \\ \cdot & \cdot & \cdot & \cdot \\ F_{n_{\text{sc}},1} & F_{n_{\text{sc}},2} & \cdot & F_{\min,n_{\text{sc}}} \end{bmatrix} \tag{17}$$

The minimum values of the design metrics form the diagonal elements of F. We see that the maximum value the ith design metric can take in the efficient set is the maximum value of the ith column of the F matrix, given by $F_{\max,i}$. We define the preference vectors as follows. Define the quantity that will represent the translation across the feasible space for the ith design metric as

$$S_i^t = \gamma_i F_{\min,i} + (1 - \gamma_i) F_{\max,i}, \qquad 0 \le \gamma_i \le 1 \tag{18}$$

In our case, we let γ_i take the values in Table 1, for $i = 1$ and 2. Next we define the three vectors

$$S_i^p = -(n_p - 1)(a_i/4) \tag{19}$$

for $n_p = 1, 2, 3, 4$, or 5,

$$S_i^f = \alpha_i d_i, \qquad -1 \le \alpha_i \le 1 \tag{20}$$

$$\boldsymbol{P}_i^o = \left\{ 0 \quad \tfrac{1}{4} \quad \tfrac{1}{2} \quad \tfrac{3}{4} \quad 1 \right\} a_i \tag{21}$$

where

$$a_i = d_i / n_d, \qquad d_i = F_{\max,i} - F_{\min,i} \tag{22}$$

We note that it is useful to have

$$\left| S_i^f + S_i^p \right| \le d_i/2 \tag{23}$$

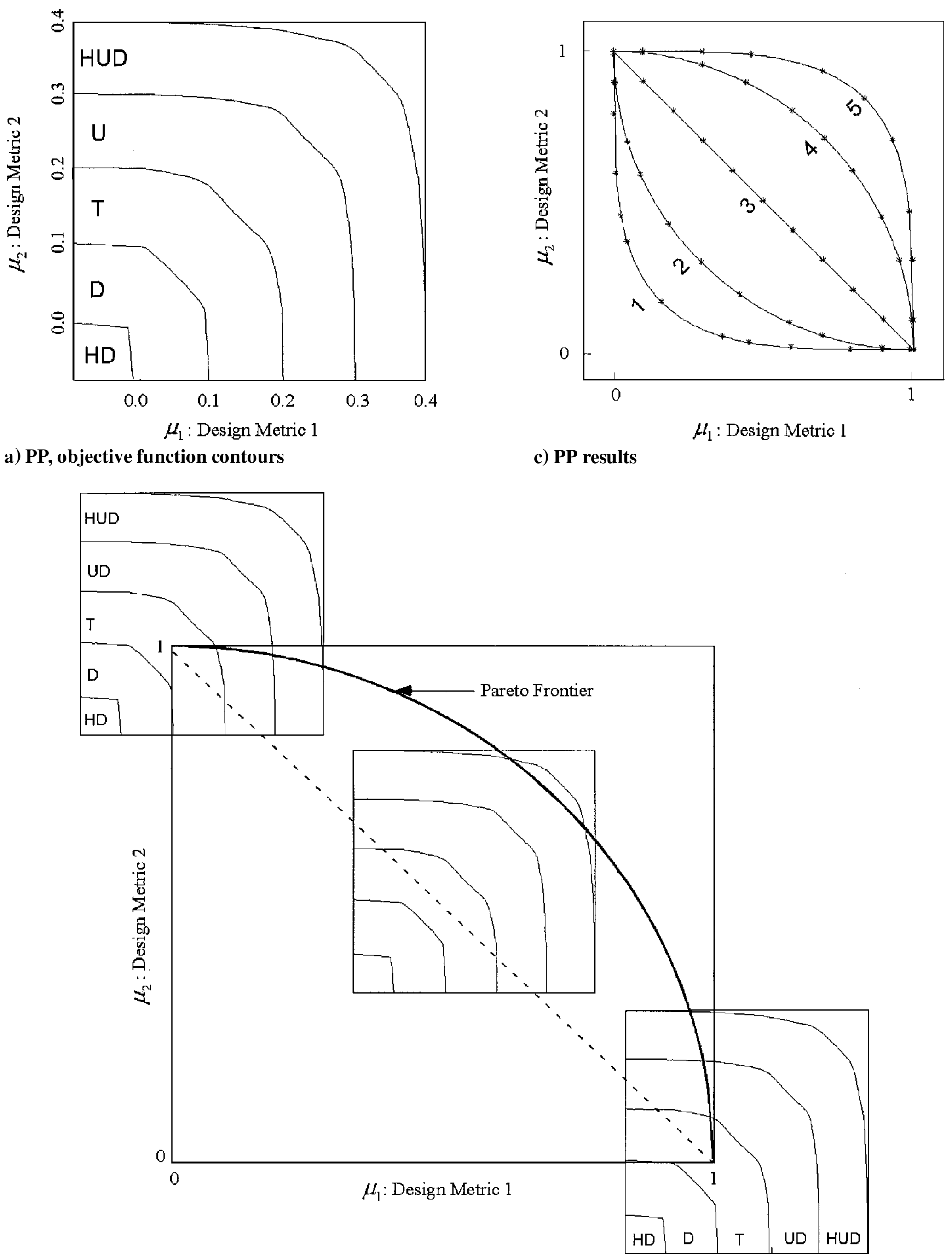
which leads to the requirement

$$-\tfrac{1}{2} + (n_p - 1)/4n_d \le \alpha_i \le \tfrac{1}{2} + (n_p - 1)/4n_d \tag{24}$$

for the preference to impact the solution effectively. With the preceding development, we obtain the preference vector for the ith generic design metric as (see Fig. 1, class 1S)

$$\boldsymbol{P}_i = \begin{Bmatrix} v_{i1} \\ v_{i2} \\ v_{i3} \\ v_{i4} \\ v_{i5} \end{Bmatrix} = \left\{ S_i^t + S_i^f + S_i^p \right\} \boldsymbol{E} + \boldsymbol{P}_i^o \tag{25}$$

where $\boldsymbol{E} = \{1, 1, 1, 1, 1\}$.



b) PP, AOF changes resulting from input preference parameter changes

Fig. 3 Stability of PP solutions with respect to input preference parameter changes.

With the particular value $n_d = 2$, we can ensure that we cover all of the Pareto hypersurface. As stated, Ref. 21 provides a comprehensive presentation and set of applications of the use of the approach to generate evenly spaced Pareto points for different optimization approaches. Only the minimum required information is presented here for the sake of brevity. We now proceed with the numerical examples.

In line with the preceding development, we let $n_d = 2.5$ and $d_i = 1$. For the sake of variety, we let $n_p = 3$ for examples 1–3, and $n_p = 1$ for examples 4 and 5. We also let $\alpha_1 = \alpha_2 = 0$. We note that the value of n_p has some impact on the distribution of the solutions.

Figure 3 shows the solutions obtained using PP. We immediately observe that the distribution of solutions is highly favorable. In Fig. 3a, we see the generic shape of the PP hypersurface, which offers virtually arbitrary flexibility in terms of curvature. This property affords it the ability to capture all Pareto points. Figure 3b shows how this hypersurface translates across the feasible space to capture an evenly distributed set of Pareto points. Finally, in Fig. 3c we observe that, indeed, the points are very well distributed for all of the examples, for both convex flat and concave Pareto frontiers. The practical meaning of this observation is the knowledge that the PP method behaves well with respect to designer input preference parameters. A small change in input will generally yield a correspondingly small change in optimal solution.

VI. Conclusions

This paper presented a theoretical examination of the Pareto optimality of solutions obtained using PP, the result of which is that all solutions obtained using PP are indeed Pareto optimal. In the process, the important concept of P dominance was presented and developed within the context of multiobjective optimization methods. This notion of P dominance led to the associated concept of generalized Pareto optimality, which must be invoked any time we are interested in doing more than minimizing and maximizing design metrics (actually, a common occurrence). Next, this paper examined the important issue of stability of optimal solutions with respect to user input parameters to express preferences. This question was examined within the context of PP and other popular methods, culminating a useful comparison of these methods. The conclusion is that PP compares favorably to the other methods and yields an evenly distributed set of solutions for an evenly distributed set of preference input parameters. Work in this area is ongoing and will hopefully allow for more comprehensive conclusions.

Acknowledgments

Support of the National Science Foundation through Grant DMI 97-02248 for Achille Messac is gratefully acknowledged.

References

[1]Messac, A., Sundararaj, G. J., Tappeta, R. V., and Renaud, J. E., "The Ability of Objective Functions to Generate Non-Convex Pareto Frontiers," *AIAA Journal*, Vol. 38, No. 3, 2000, pp. 1084–1091; also AIAA Paper 99-1211, April 1999.

[2]Messac, A., "Physical Programming: Effective Optimization for Computational Design," *AIAA Journal*, Vol. 34, No. 1, 1996, pp. 149–158.

[3]Chen, W., Wiecek, M. M., and Zhang, J., "Quality Utility—A Compromise Programming Approach to Robust Design," DETC 98, American Society of Mechanical Engineers, Design Engineering Technical Conf., Paper DAC5601, Sept. 1998.

[4]Das, I., and Dennis, J. E., "A Closer Look at Drawbacks of Minimizing Weighted-Sums of Objectives for Pareto Set Generation in Multicriteria Optimization Problems," *Structural Optimization*, Vol. 14, No. 1, 1997, pp. 63–69.

[5]Das, I., and Dennis, J. E., "Normal-Boundary Intersection: A New Method for Generating the Pareto Surface in Nonlinear Multicriteria Optimization Problems," *SIAM Journal on Optimization*, Vol. 8, No. 3, 1998, pp. 631–657.

[6]Koski, J., "Defectiveness of Weighted Method in Multicriterion Optimization of Structures," *Communications in Applied Numerical Methods*, Vol. 1, No. 6, 1985, pp. 333–337.

[7]Messac, A., and Hattis, P., "Physical Programming Design Optimization for High Speed Civil Transport (HSCT)," *Journal of Aircraft*, Vol. 33, No. 2, 1996, pp. 446–449.

[8]Messac, A., Gupta, S., and Akbulut, B., "Linear Physical Programming: Effective Optimization for Complex Linear Systems," *Transactions on Operational Research*, Vol. 8, Oct. 1996, pp. 39–59.

[9]Messac, A., and Wilson, B., "Physical Programming for Computational Control," *AIAA Journal*, Vol. 36, No. 2, 1998, pp. 219–226.

[10]Messac, A., "Control–Structure Integrated Design with Closed-Form Design Metrics Using Physical Programming," *AIAA Journal*, Vol. 36, No. 5, 1998, pp. 855–864.

[11]Tappeta, R. V., Renaud, J. E., Messac, A., and Sundararaj, G. J., "Interactive Physical Programming: Tradeoff Analysis and Decision Making in Multicriteria Optimization," *AIAA Journal*, Vol. 38, No. 5, 2000, pp. 917–926; also AIAA Paper 99-1209, April 1999.

[12]Athan, T. W., and Papalambros, P. Y., "A Note on Weighted Criteria Methods for Compromise Solutions on Multi-Objective Optimization," *Engineering Optimization*, Vol. 27, No. 2, 1996, pp. 155–176.

[13]Osyczka, A., *Multicriterion Optimization in Engineering with Fortran Programs*, Ellis Horwood Series in Engineering, Halsted, New York, 1984, pp. 43–47.

[14]Rakowska, J., Haftka, R. T., and Watson, L. T., "Tracing the Efficient Curve for Multi-Objective Control–Structure Optimization," *Computer Systems Engineering*, Vol. 2, No. 6, 1991, pp. 461–471.

[15]Messac, A., "PhysPro: A MATLAB-Based Physical Programming Software," Optimal Systems, Lexington, MA, 1994.

[16]Cohon, J. L., *Multiobjective Programming and Planning, Mathematics in Science and Engineering*, Vol. 140, Academic Press, San Diego, CA, 1978, pp. 219, 220.

[17]Lin, J. G., "Three Methods for Determining Pareto-Optimal Solutions of Multiple-Objective Problems," *Directions in Large-Scale Systems*, edited by Y. C. Ho and S. K. Mitter, Plenum, New York, 1975, pp. 117, 118.

[18]Lundlerg, B. N., and Poore, A. B., "Bifurcations and Sensitivity in Parametric Programming," *Proceedings of the Third Air Force/NASA Symposium on Recent Advances in Multidisciplinary Analysis and Optimization*, pp. 50–55.

[19]Rao, J. R., and Papamlabros, P. Y., "A Non-Linear Programming Continuation Strategy for One Parameter Design Optimization Problems," *Proceedings of ASME Design Automation Conference*, American Society of Mechanical Engineers, Fairfield, NJ, 1989, pp. 77–89.

[20]Messac, A., and Sundararaj, G. J., "A Robust Design Approach Using Physical Programming," AIAA Paper 2000-0562, Jan. 2000.

[21]Messac, A., and Sundararaj, G. J., "Physical Programming's Ability to Generate a Well-Distributed Set of Pareto Points," AIAA Paper 2000-1666, April 2000.

A. Chattopadhyay
Associate Editor