An Efficient Pareto Set Identification Approach for Multi-objective Optimization on Black-box Functions

Songqing Shan G. Gary Wang[1]

1. Abstract

Both multiple objectives and computation-intensive black-box functions often exist simultaneously in engineering design problems. Few of existing multi-objective optimization approaches addresses problems with expensive black-box functions. In this paper, a new method called the Pareto set pursuing (PSP) method is developed. By developing sampling guidance functions based on approximation models, this approach progressively provides a designer with a rich and evenly distributed set of Pareto optimal points. This work describes PSP procedures in detail. From testing and design application, PSP demonstrates considerable promises in efficiency, accuracy, and robustness. Properties of PSP and differences between PSP and other approximation-based methods are also discussed. It is believed that PSP has a great potential to be a practical tool for multi-objective optimization problems.

Key words: Black-box function, multi-objective optimization, Pareto set, Pareto set pursuing, sampling guidance function

2. Introduction

Modern design problems often involve multiple objectives and are thus treated as multi-objective optimization (MOO) problems. In general multi-objective optimization (MOO) methods, excluding methods dealing with uncertainties in MOO, can be roughly categorized into two main classes. The first class of methods converts a MOO problem into a single-objective problem. Such conversion is

[1] Corresponding author, Dept. of Mechanical and Manufacturing Engineering, The University of Manitoba, Winnipeg, MB, Canada R3T 5V6, Tel: 204-474-9463, Fax: 204-275-7507, Email: gary_wang@umanitoba.ca

usually achieved by using either explicit or implicit weights, preferences, utilities, or targets. These methods require *a priori* selection of weights, preferences, or utilities for each of the objective functions [1-2]. It can be very difficult to decide which weighting factors should be used in practice for real problems. In addition, these weights may be inadequate in capturing decision makers' preferences. This class of methods provides information for only one design scenario and lacks of a rigorous method for the weight selection. It is also found that the weighted-sum method is unable to find Pareto points in non-convex regions in the performance space [2]. This class of methods is recently improved by the physical programming method [4-6]. The second class of methods tries to identify a set of discrete points as an approximation of the Pareto optimal frontier for decision makers without pre-assuming their preferences. The most successful and widely used approaches in this category seem to be evolutionary algorithms [7-12]. These methods can generate a large number of Pareto points for decision makers. However, they are usually computationally expensive because a massive number of non-Pareto set points have to be evaluated. A recent detailed survey of MOO methods can be found in Ref. [2].

For MOO problems involving expensive analysis and simulation processes such as finite element analysis (FEA) and computational fluid dynamics (CFD), the use of approximation becomes more attractive. Since FEA and CFD processes are based on complex and numerous simultaneous equations, these processes are often treated as "black-box" functions, for which only inputs and outputs are known. Because the complexity of these expensive processes seems to maintain its pace with computing advances, the challenge brought by these computationally expensive black-box functions on optimization should be addressed. For MOO problems involving expensive black-box functions, the computational burden aggravates. Recent approaches to solve MOO problems with black-box functions are to approximate each single objective function or directly approximate the Pareto optimal frontier [13-15]. The accuracy of the Pareto optimal frontier depends on the accuracy of approximation

models. Wilson *et al.* [13] used the surrogate approximation in lieu of the computationally expensive analyses to explore the multi-objective design space and identify Pareto optimal points, or the Pareto set, from the surrogate. Li *et al.* [13] used a hyper-ellipse surrogate to approximate the Pareto optimal frontier for bi-criteria convex optimization problems. If the approximation is not sufficiently accurate, then the Pareto optimal frontier obtained using the surrogate approximation will not be a good approximation of the actual Pareto optimal frontier. A latest work by Yang *et al.* [15] proposed the first framework (as claimed by the authors) managing approximation models in MOO. In the framework, a GA based method is employed with a sequentially updated approximation model. It differs from [13] by updating the approximation model in the optimization process. The fidelity of the identified frontier solutions, however, is still built upon the accuracy of the approximation model. The work in Ref. [15] also suffers from the problems of the GA-based MOO algorithm, i.e., the algorithm has difficulty in finding frontier points near the extreme points (the minimum obtained by considering only one objective function).

This work proposes a Pareto set identification method by "intelligent" progressive sampling. With no *a priori* preference is needed, this method could automatically sample near the Pareto optimal frontier and rapidly converge to the Pareto set. Before introducing the proposed method, some basic yet important concepts of MOO are reviewed in the next section. The proposed method will be described and its properties will be discussed in Section 4. Section 5 will present numerical studies on a few well-known MOO problems. The proposed method is then applied to an engineering design problem, which will be presented in Section 6. Section 7 will discuss the properties PSP and related issues. Section 8 gives conclusions and future work.

3. Basic Concepts of Multi-Objective Optimization

Multi-objective Optimization

Multi-objective optimization is a vector optimization problem. A general multi-objective optimization problem is to find design variables that optimize a vector objective function subject to a number of constraints and bounds. The multi-objective optimization is often formulized as follows:

$$\begin{aligned} &\text{Minimize } F(x) = \{f_1(x), f_2(x), \ldots, f_i(x), \ldots, f_m(x)\} \\ &\text{Subject to:} \\ &\quad h_j(x) = 0, \qquad j = 1, \ldots, q \\ &\quad g_k(x) \le 0, \qquad k = 1, \ldots, p \\ &\quad x_r^l \le x_r \le x_r^u, \qquad r = 1, \ldots, n \end{aligned} \tag{1}$$

Where the components of the multiple objective function vector, $F(x) = \{f_1(x), f_2(x), \ldots, f_m(x)\}$, are usually in conflict with one another with respect to their own optima. The design variable vector, $x = [x_1, x_2, \ldots, x_n]$, consists of all design variables in the problem bounded in $x_r^l \le x_r \le x_r^u$, $r = 1, \ldots, n$. $h_j(x)$ are equality constraints and $g_k(x)$ are inequality constraints. In this work for the ease of illustration, we define *feasible design space* as the set of all design points represented by design variables that satisfy constraints. Those points within the feasible design space inclusively are called feasible design points or feasible designs. We define *feasible performance space* as a set of all objective function values with respect to every feasible design.

Pareto Set

Pareto set [5] is defined as a set of points of Pareto optimality. A vector of x^* is Pareto optimal if there exists no feasible vector x that would decrease some objective function without causing a simultaneous increase in at least one objective function. Mathematically, the Pareto optimality is defined as follows: a vector of x^* is a Pareto optimum iff, for any x and i,

$$f_j(x) \le f_j(x^*), \qquad j = 1, \ldots, m; \quad j \ne i, \quad \Rightarrow f_i(x) \ge f_i(x^*) \tag{2}$$

In general for MOO, the task is to identify Pareto set points. The question is how to judge if a point is in the Pareto set. To address this question, the fitness function is introduced.

Fitness Function

For a given set of design points, a fitness function is defined as [7]:

$$G_i = [1 - \max_{j \neq i}(\min(f_{s1}^i - f_{s1}^j,\ f_{s2}^i - f_{s2}^j, \cdots, f_{sm}^i - f_{sm}^j))]^l \tag{3}$$

Where, G_i denotes the fitness value of the i^{th} design; f_{sk}^i is the scaled k^{th} objective function value of the i^{th} design, $k = 1,\ldots, m;$ and l is called the frontier exponent. In this work, l is taken as a constant 1. The max in Eq. (3) is over all other designs $j \neq i$ in the set and the min is over all the objectives. The objectives $f_{s1}, f_{s2}, \ldots, f_{sm}$ in the Eq. (3) are scaled to a range [0, 1]. For example for f_{s1}^i,

$$f_{s1}^i = \frac{rawf_{1,i} - rawf_{1,\min}}{rawf_{1,\max} - rawf_{1,\min}} \tag{4}$$

Where, $rawf_{1,i}$ denotes un-scaled value of the first objective for the i^{th} design; $rawf_{1,\max}$ denotes the maximum un-scaled value of the first objective among all designs; and $rawf_{1,\min}$ denotes the minimum un-scaled value of the first objective among all designs. In case that an objective function is a constant, the scaled objective function value f_{s1}^i is taken as 1 in this work.

As the fitness function measures relation between points in the performance space and all the objective function values are scaled to [0, 1], it is not hard to find that for a given set of points following statements hold true:

- Pareto set points should have a fitness function value in the range of [1, 2].
- Non-Pareto set points have a fitness value in [0, 1); and the higher the fitness function value, the closer the point to the Pareto frontier.
- When Pareto set points are closely and evenly distributed, the fitness value of all Pareto set points tends to be 1.

4. Pareto Set Pursing (PSP) Methodology

The goal of PSP is to directly sample many Pareto solutions (points) to approximate the entire Pareto optimal frontier in an economical manner for MOO problems with computationally expensive black-box functions. Assuming at the first iteration we start from random sampling for the unknown problem, it is desirable at the next iteration to sample more points closer to the Pareto frontier than further away. If the trend continues, we can sample right on or very close to the frontier. Figure 1 illustrates such a desired sampling scheme, which is to be achieved by the proposed PSP method.

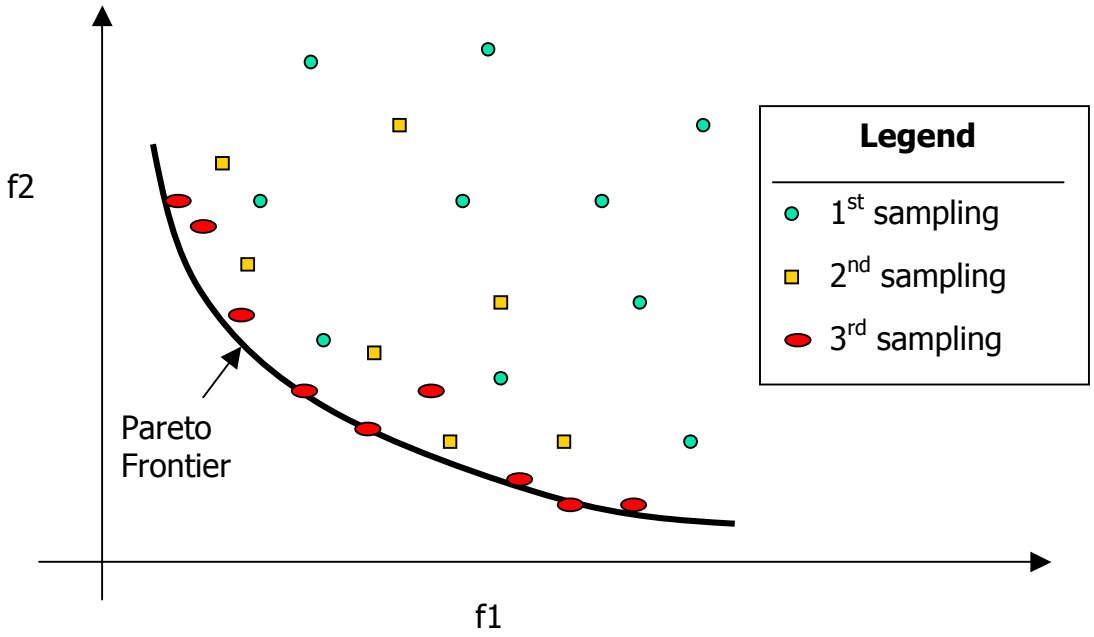

Figure 1 An illustration of the desired sampling scheme for PSP.

Before we introduce details of PSP, the sampling guidance function is defined first. How to construct a sampling guidance function to realize the desired sampling scheme as shown in Figure 1 is a key to PSP.

Sampling Guidance Function

As we know, there exists many methods in Statistics to generate a sample from a given probability density function (PDF) [16]. These methods include inverse transformation, acceptance-rejection

technique, Markov Chain Monte Carlo (MCMC), importance sampling, and so on. A recent work is given by Fu and Wang [17], which can be considered as a Cumulative Density Function (CDF) inverse method. The goal of such sampling is to generate sample points that conform to a given PDF, i.e. more points in the area that has high probability and fewer points in the area that has low probability, as defined by the PDF. Inspired by such sampling, the authors developed a Mode Pursuing Sampling (MPS) method before [18], which used a variation of the objective function to act as a PDF so that more points are generated in areas having lower objective function value and fewer in other areas. In brief, for an expensive black-box objective function, MPS first constructs an approximation model from a few sample points. It then generates a large number of points from the approximation model, sorts the points, and constructs a cumulative function analogous to CDF by adding up all the function values before the current point in the sorted point set. A sample is then drawn from the point set according to this cumulative function. As a result, more new sample points are generated around the current minimum and less in other regions in the design space. MPS uses Fu and Wang's method [17] to generate a sample from a PDF and is an iterative process. It is in essence a discriminative sampling method with approved convergence. To generalize the sampling idea of MPS, this work gives the definition of a sampling guidance function.

A sampling guidance function of a discrete random variable X is described as a function $\hat{z}(x_i) = P[X = x_i]$ which satisfies:

(i) $\hat{z}(x_i) \geq 0$, for each value x_i of X

(ii) representing the nature of the problem

(iii) reflecting one's sampling goal, and

(iv) expressing prior information if used iteratively

Requirement (i) is to ensure all of the guidance function values are larger than 0, similar to a PDF. Apparently, a sampling guidance function should represent the nature of the problem and reflect one's

sampling goal. Thus requirements (ii) and (iii) are then stipulated. If the sampling guidance function is to be used in an iterative process, another desirable feature of a sampling guidance function is to express prior information and be adaptive, which is Requirement (iv). One can note that unlike a PDF, the sampling guidance function doesn't have to satisfy $\sum_{i=1}^{k} \hat{z}(x_i) = 1$. A PDF can thus be considered as a special sampling guidance function. The variation of the objective function in MPS [18] is also a sampling guidance function.

In this work, two types of sampling guidance function are developed. One is for the sampling of "cheap" points from the approximation model of each objective function. The other is for the sampling towards the Pareto frontier. Details about these two types of sampling guidance functions will be given in the description of the PSP procedures.

Procedures of the Pareto Set Pursuing (PSP) Strategy

The following illustrates the procedures of PSP to identify the Pareto set points of an *n*-dimensional MOO problem defined by Eq. (1). Steps of the algorithm are illustrated in Figure 2, along with step-by-step explanations. A well known MOO problem is taken from the literature to facilitate the explanation of PSP [13].

$$
\begin{aligned}
\text{Minimize:} \quad & F(x) = \{f_1(x), f_2(x)\} \\
& f_1(x_1, x_2) = (x_1 + x_2 - 7.5)^2 + (x_2 - x_1 + 3)^2 / 4 \\
& f_2(x_1, x_2) = (x_1 - 1)^2 / 4 + (x_2 - 4)^2 / 2 \\
\text{Subject to:} \quad & g_1(x_1, x_2) = 2.5 - (x_1 - 2)^3 / 2 - x_2 \geq 0 \\
& g_2(x_1, x_2) = 3.85 + 8(x_2 - x_1 + 0.65)^2 - x_2 - x_1 \geq 0 \\
& 0 \leq x_1 \leq 5;\ 0 \leq x_2 \leq 3
\end{aligned}
\tag{5}
$$

This problem involves 2 design variables (n = 2) and 2 objective functions (m = 2) with 2 constraints and variable bounds. Constraints are often handled in many different ways in optimization, e.g.,

penalty or Lagrange methods. They can also be treated as special objective functions as in [19]. A detailed survey of constraint handling is in [20]. For the MOO problem defined in Eq. (1), if there is any computation-intensive constraint, we assume that it can be either treated as another objective function, or can be integrated to existing objective functions as a penalty or Lagrange term. To focus on PSP but not losing generality, we concentrate on problems in which all of the objective functions are computationally expensive black-box functions, while constraints are inexpensive functions.

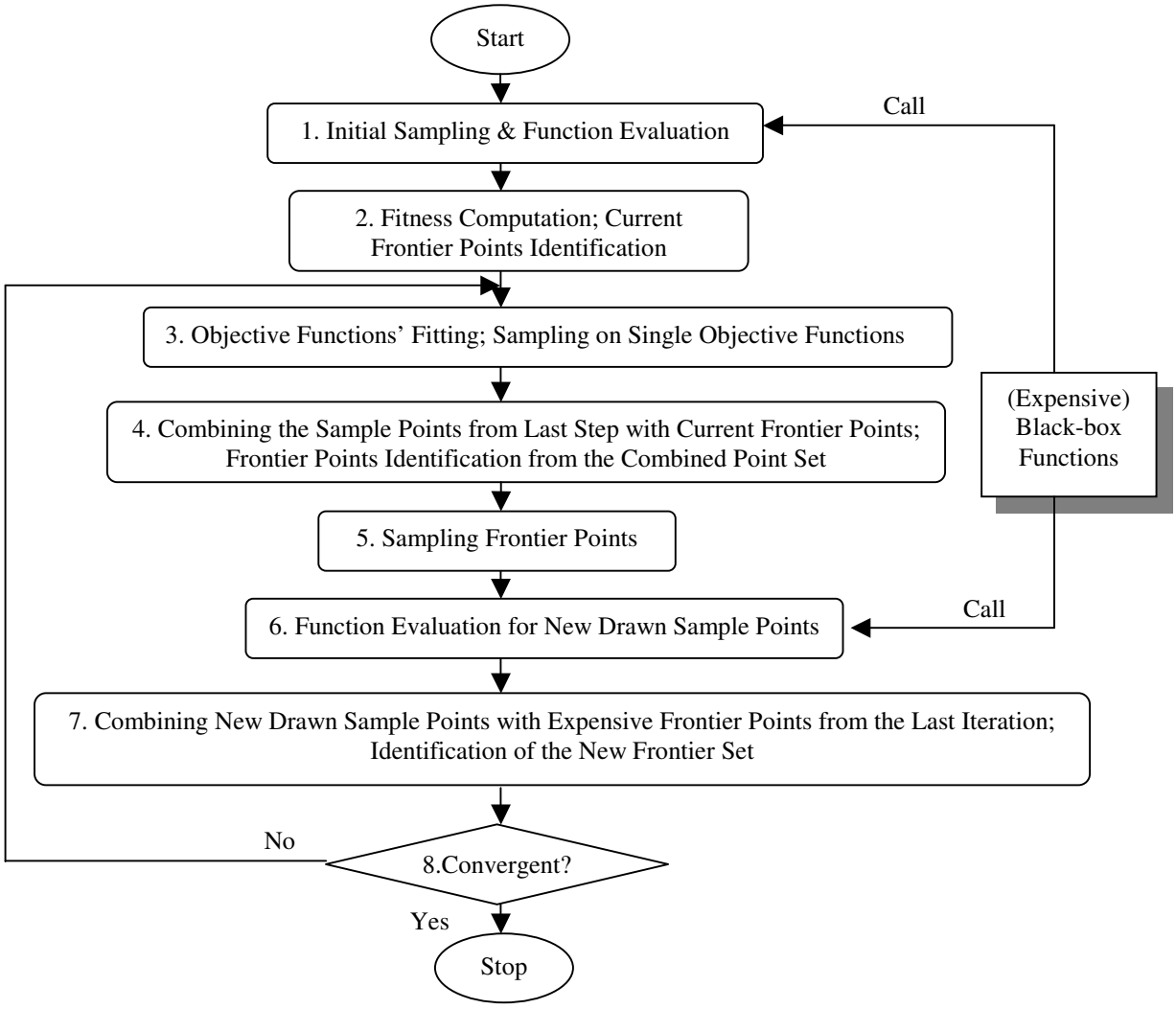

Figure 2 Flowchart of the Pareto-Set-Pursuing Approach

Step 1: Initial Random Sampling and Expensive Function Evaluation.

Initial sample points are used to build an approximation model for each objective function. This work employs both quadratic polynomial fitting (QPF) and radial basis function (RBF) fitting. QPF is the

most widely used response surface model [21]. RBF demonstrates great promise in approximation [22]. Both QPF and RBF are very simple, intuitive, and easy to construct. RBF fitting passes all the expensive points, prevents unnecessary "curvatures" added to the unknown surface, and preserves the minimum among the expensive points, i.e., $\min \hat{f}(x) = \min\{f(x^{(i)}), i = 1,\cdots,k\}$ because its linearity nature. This feature ensures that more expensive points are generated around the current minimum of $f(x)$, rather than being biased because of the approximation model $\hat{f}(x)$. Comparatively QPF fitting smoothes the sample data by a quadratic model; it is thus not absolutely loyal to the sample data and does introduce "curvature" to the data. These two sampling methods are automatically alternated during the sampling procedure based on a criterion that will be explained later. It is to be noted that PSP does not dictate the exclusive use of the two models. Moreover, the accuracy of the approximation model, as will be discussed later in Section 7, is less important in PSP than its common use in conventional approximation-based optimization.

An n-D quadratic polynomial model [21] is usually expressed by Eq. (6).

$$\hat{f}(x) = \beta_0 + \sum_{i=1}^{n} \beta_i x_i + \sum_{i=1}^{n} \beta_{ii} x_i^{\,2} + \sum_{i<j} \sum_{j=1}^{n} \beta_{ij} x_i x_j \tag{6}$$

Where β_i, β_{ii}, and β_{ij} represent regression coefficients, $x_i, (i = 1 \cdots n)$ are design variables, and $\hat{f}(x)$ is the response. RBF has many forms. This work uses its simplest form, i.e., using k function values $f(x^{(1)}), f(x^{(2)}),\ldots, f(x^{(k)})$ to fit a linear spline function

$$\hat{f}(x) = \sum_{i=1}^{k} \alpha_i \| x - x^{(i)} \|, \tag{7}$$

such that $\hat{f}(x^{(i)}) = f(x^{(i)})$, $i = 1, 2,\ldots, k$. Where α_i denotes the regression coefficients and $x^{(i)}$ represents the coordinates of the i^{th} sample point. For both QPF and RBF models, the number of initial random sample points is chosen as the minimum number needed to build a full quadratic approximation model, $(n+1)*(n+2)/2$, where n denotes the number of variables. After random

sampling $(n+1)*(n+2)/2$ number of points, expensive black-box functions are called to evaluate these sample points. The number of the so-far evaluated sample points *np* at the end of the first step thus equals to $(n+1)*(n+2)/2$. For the example problem, 6 points for objective functions f_1 and f_2 are generated in random in the design space ($0 \le x_1 \le 5$ and $0 \le x_2 \le 3$). The points are evaluated expensive points and are plotted in the performance space in Figure 3.

Step 2: Fitness Computation and Current Frontier Points Identification

By using Eq. (3), the fitness value of the evaluated initial sample points are computed and the current frontier points can be identified according to their fitness function values. The average fitness value of all the current frontier points is also computed as it is used as part of the convergence criterion. It is to be noted that the current frontier is to be updated as the procedure continues. In Figure 3 P_1, P_2, and P_3 are identified as the current frontier points. Approximation models for both f_1 and f_2 are then generated by using Eqs. (6) or (7).

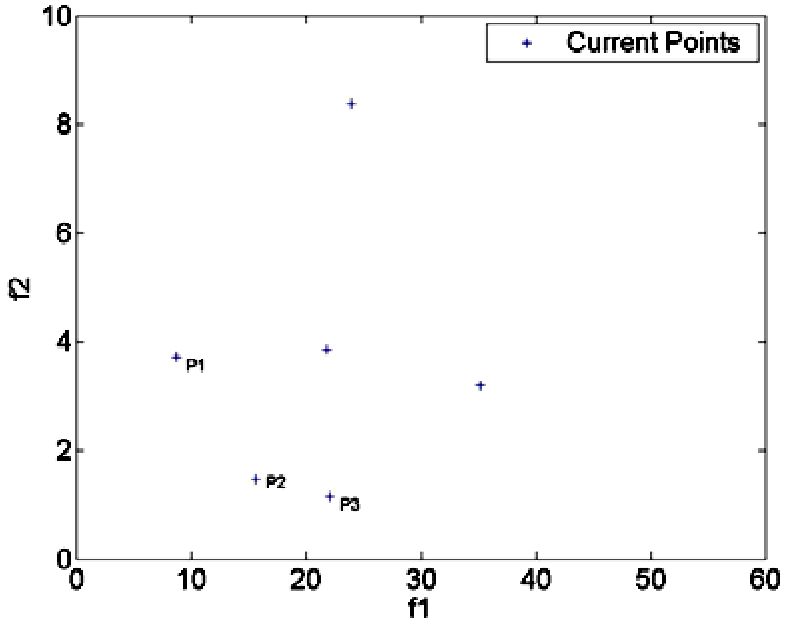

Figure 3 Initial points generated for the example problem with identified current frontier points.

Step 3: Objective Functions' Fitting and Sampling on Single Objective Functions

The goal of this step is to generate a large number of "cheap" points from each approximation model independently, in preparation for the sampling of the frontier points in the performance space. First we use the *np* evaluated function values $f_i(x^{(1)}), f_i(x^{(2)}),\ldots, f_i(x^{(np)})$ (i =1, 2, …, m) to gain the coefficients of the approximation model $\hat{f}_i(x)$, a n-D quadratic polynomial model Eq. (6) or a linear spline function Eq. (7). We could perform random sampling to get the desired large number of "cheap" points. Alternatively, we could put a greater emphasis on points having smaller function values for each objective in order to identify end points on the Pareto frontier. Therefore, we then construct a sampling guidance function for the i^{th} objective function, $\hat{z}_i(x) = c_0 - \hat{f}_i(x)$, $c_0 \geq \hat{f}(x)$. It is easy to see that $\hat{z}_i(x)$ is always positive; it is a linear transformation of its approximation function $\hat{f}_i(x)$; it reflects the goal to sample more points in regions that $\hat{f}_i(x)$ has smaller values; and as $\hat{f}_i(x)$ is built from so-far evaluated expensive points, $\hat{z}_i(x)$ can thus adapt to increasingly richer information. Therefore, $\hat{z}_i(x)$ satisfy all of the four requirements for the sampling guidance function; and it is the first type of sampling guidance function used in this work. The other type is for the sampling of frontier set points which will be discussed later.

An equal number of sample points, e.g., 100, are drawn independently according to each objective function's sampling guidance function $\hat{z}_i(x)$. As a result, we will have $m \times 100$ number of cheap points. It is to be noted although there are more sample points drawn from function f_i having small function values of f_i, these points might have large function values of f_j, $i \neq j$, and vice versa. If the point leading to min(f_i) is identified, this point will be at one of the vertices of the Pareto frontier in the performance space and is usually called an extreme point. Therefore, by independently pursuing the minimum of each objective function, we have a mixture of $m \times 100$ points potentially covering all of the

extreme points on the Pareto frontier. By doing so, the difficulties of GA-based algorithms in identifying extreme points as found in [15] can be overcome.

Step 4: Combining the Sample Points from Last Step with Current Frontier Points; Frontier Points Identification from the Combined Point Set

This step is to prepare for sampling new frontier points. First sample points drawn at Step 3 are combined with current frontier points for the recalculation of the fitness value of all points in the combined set. It is to be noted that current frontier points are evaluated expensive points, their real function values are used in the fitness computation. Sample points drawn from Step 3 are not evaluated from expensive black-box functions; their respective $\hat{f}_i(x)$ function values will be used instead for the fitness value computation. Sample points drawn from Step 3 are used to enrich the information for the construction of a sampling guidance function for the next step, though their function values are only a prediction from the approximation model. Then the fitness function values are computed by using Eq. (3). For the combined point set, points having a fitness value larger than or equal to 1 will be used for the sampling in the next step. Other points are discarded as these points are likely non-frontier points.

Step 5: Sampling Frontier Points

The frontier points obtained from Step 4 are the "best" points among all of the existing points in terms of the possibility of becoming Pareto set points. Frontier points are different from Pareto set points as the former may turn to be non-frontier points if new points (designs) are added as the sampling process iterates. Converged frontier points are deemed Pareto set points in this work. For the frontier points obtained in Step 4 whose fitness value is larger than or equal to 1, define $\hat{z}(x) = G_i - 1 = [1 - \max_{j \neq i}(\min(f_{s1}^i - f_{s1}^j,\ f_{s2}^i - f_{s2}^j, \cdots, f_{sm}^i - f_{sm}^j))]^l - 1$. $\hat{z}(x)$ is nonnegative and represent the nature of the problem because it is compounded from $f_i(x)$ and $\hat{f}_i(x)$. It is also used to sample more points in regions at which the fitness value is higher, because the higher the fitness value,

the more likely Pareto set points exist. Thus this $\hat{z}(x)$ function reflects our sampling goal. Since it is built on $f_i(x)$ and $\hat{f}_i(x)$, it adapts to increasingly richer information as more sample points are evaluated. Therefore $\hat{z}(x)$ satisfies all four requirements and it is in fact the second type of sampling guidance function in this work. Assuming L frontier points are obtained from Step 4, a random sample $x^{(np+1)}$, $x^{(np+2)}$, …, *etc.* from the L frontier points are drawn, guided by the $\hat{z}(x)$ function. To determine the number of to-be-drawn points, a simple heuristics is used in this work. If the ratio of L and the number of the current frontier points in the last iteration is less than 2, L points are drawn. If the ratio is between 2 and 4, the same number of the current frontier points is drawn. Otherwise, a double number of the frontier points in the last iteration are drawn. All the new sample points form a new sample set. If the drawn points have been evaluated before or repetitive, they are discarded from the new sample set to avoid re-evaluation.

Step 6: Function Evaluation for New Sample Points

The points in the new sample set obtained from Step 5 are evaluated by calling expensive black-box functions.

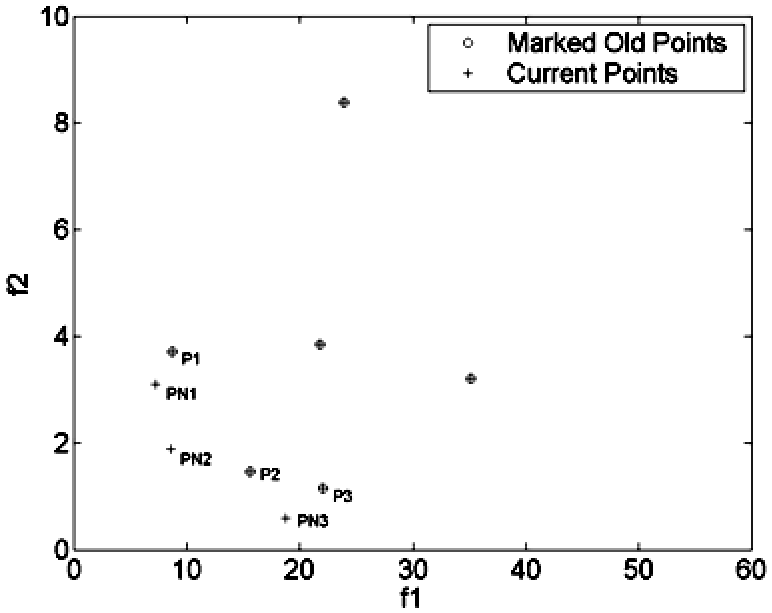

Figure 4 Sample points at the end of the 1st iteration for the example problem.

For the example problem, in total 200 cheap points are generated independently from each approximation model at Step 3. These 200 points are also evaluated by all the approximated objective models. They are then combined with points P_1, P_2, and P_3 to calculate fitness function values. Among all these points, points having fitness values larger than or equal to 1 are chosen as a new point set. New samples are drawn from this point set as shown in Figure 4 by performing Step 5. The '+' symbol indicates locations of the new sample points PN1, PN2 and PN3, after expensive function evaluations.

Step 7: Combining New Sample Points with Expensive Frontier Points from the Last Iteration and Identification of the New Frontier Set

At this step, the newly evaluated sample points PN1, PN2 and PN3 are then combined with formerly evaluated expensive frontier points P1, P2, and P3 shown in Figure 3. Fitness values of this combined point set are then calculated and the frontier points are identified as the final frontier points of this iteration. It is easy to see that PN1, PN2 and PN3 will replace the previous P1, P2, and P3 as the new frontier points. All of the final frontier points are evaluated expensive points. All points evaluated by expensive function evaluations, in this case PN1, PN2, PN3 and the initial 6 points, will participate the modeling of each objective function at the 2^{nd} iteration.

Step 8: Checking Convergence

If the convergence criteria are met, the procedure terminates; otherwise, back to Step 3. Figure 5 shows all of the evaluated sample points and converged frontier points with respect to the feasible performance space for the example problem. As one can see that the converged frontier points override the real Pareto frontier, even at the singular point at the left upper corner. These points also distribute closely and evenly across the entire curve. The ratio of Pareto points obtained over the total number of sample points is high. Since the proposed method calls random processes, 10 independent runs have

been carried out. Similar accuracy and efficiency have been achieved in all of the 10 runs. Detailed results on this test function will be reported in Section 5.

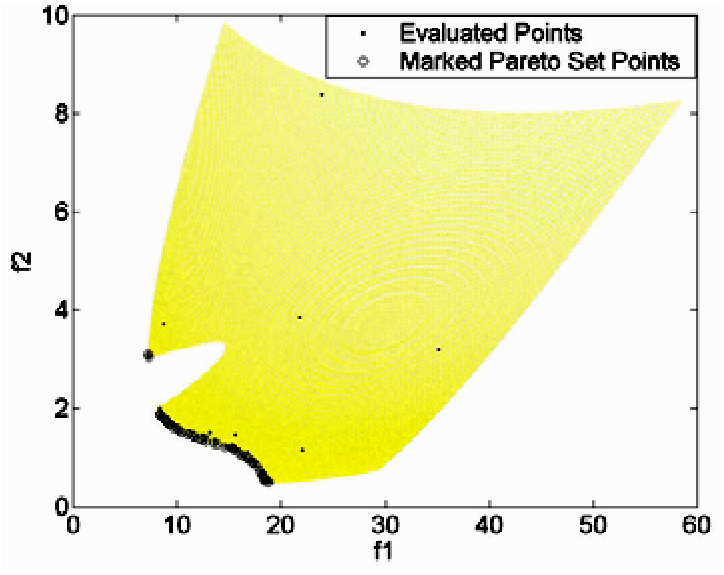

Figure 5 Performance space, evaluated points, and Pareto set points for the example problem.

Convergence criteria

Because no property of a black-box function is available, it is unlikely to develop a rigorous convergence criterion such as Kuhn-Tucker conditions. Many direct search algorithms use the maximum number of iterations or allowable budget as the criterion, e.g., [23]. This work applies two convergence criteria. The first can be thought of as a criterion along the vertical direction, which measures the progress of iterations. It is similar to the one used in [15], i.e., the difference between frontier points after two consecutive iterations is sufficiently small. Assuming $(PS)_k$ is the frontier point set at the k-th iteration, $n(PS)_{k,k+1}$ is the total number of points in $(PS)_k$ that also exist in $(PS)_{k+1}$, and $n(PS)_{k+1}$ is the total number of points in $(PS)_{k+1}$. Then $\rho = \frac{\mathrm{n(PS)}_{k,\,k+1}}{n(PS)_{k+1}} \geq 0.95$ is used as the first convergence criterion, which means most frontier points identified in the last iteration are also in the current frontier and therefore only few new frontier points are found. The second criterion can be

considered as a convergence criterion in the horizontal direction, which measures the closeness and distribution of Pareto points on the frontier at the last iteration. Referring to the fitness function defined in Eq. (3) and its properties, when Pareto points are closely and evenly distributed, the fitness value of all Pareto set points tends to be 1. Therefore, in this work we set the average fitness value of all the current frontier points, $\overline{G} = \frac{1}{L}\sum_{i=1}^{L} G_i$, close to 1 as an additional convergence criterion, where L is the number of frontier points in the last PSP iteration. In practice, we use $1 \le \overline{G} \le 1.02$ (or 1.001). This criterion ensures the Pareto points spread over the frontier, a desired property supported by many researchers [2, 15, 24].

Approximation Model Alternation

At Step 3, QPF and RBF models are automatically alternated to take the advantages of both and make the algorithm more efficient. There are two criteria for alternation. In general QPF is used in the earlier stages because QPF emphasizes more on predicting the overall trend than linear RBF. RBF is used in the later stages because it is more loyal to sample points. The earlier and later stages are marked by evaluating the average fitness value of current frontier points, $\overline{G}_k$, as compared against the desired convergence criterion for $\overline{G}_d$, e. g. 1.001, plus a constant deviation δ (=0.05 in this work). If $\overline{G}_k > \overline{G}_d + \delta$, which indicates that the current frontier is far from convergence so that it is regarded as an earlier stage, QPF is used. Otherwise, RBF is used. The second criterion is based on the number of new sample points. Models are alternated to speed up if the number of new sample points generated after an iteration is less than a given small constant.

5. Numerical Studies

The proposed PSP method is tested with a number of well-known MOO problems taken from the literature. Three test problems are chosen. The first is a convex, bi-objective MOO problem. The second, as described by Eq. (5), has a non-smooth Pareto frontier featured by a singular Pareto point. The third is a highly non-linear problem with 3 objectives. PSP is then applied to a real engineering design problem. The formulas of the test problems are as below.

- Problem 1 [13].

$$\begin{aligned} \text{Minimize:} \quad & f_1(x_1, x_2) = (x_1 - 2)^2 + (x_2 - 1)^2 \\ & f_2(x_1, x_2) = x_1^{\,2} + (x_2 - 6)^2 \\ \text{Subject to :} \quad & g_1(x_1, x_2) = x_1 - 1.6 \le 0 \\ & g_2(x_1, x_2) = 0.4 - x_1 \le 0 \\ & g_3(x_1, x_2) = x_2 - 5 \le 0 \\ & g_4(x_1, x_2) = 2 - x_2 \le 0 \end{aligned} \tag{8}$$

- Problem 2 is the example problem defined by Eq. (5).

- Problem 3 [4]:

$$\begin{aligned} \text{Minimize} \quad & F(x) = \{f_1(x), f_2(x), f_3(x)\} \\ \text{Subject to:} \quad & g_1(x) = 12 - x_1^2 - x_2^2 - x_3^2 \ge 0 \\ & 0 \le x_i \le 5, \ i = 1, 2, 3. \end{aligned} \tag{9}$$

where the objective functions are given by

$$f_1 = 25 - (x_1^3 + x_1^2(1 + x_2 + x_3) + x_2^3 + x_3^3)/10$$
$$f_2 = 35 - (x_1^3 + 2x_2^3 + x_2^2(2 + x_1 + x_3) + x_3^3)/10$$
$$f_3 = 50 - (x_1^3 + x_2^3 + 3x_3^3 + x_3^2(3 + x_1 + x_2))/10$$

Since random processes are used in PSP, 10 runs have been carried out for each problem. The test results are presented below. For each test problem, the number of iterations, total number of evaluated points (expensive), number of converged frontier points, as well as the ratio of the number of converged frontier points to the total number of evaluated points are recorded for 10 different runs. The median and variation range for all integer numbers are recorded. For the point ratio, the mean value and its standard deviation (S.D.) are used instead. The results are listed in Table 1, along with

the result for the panel design problem which will be described in the next section. The total number of expensive function evaluations for each problem equals to the number of evaluated points times the number of objective functions, *m*.

Table 1 Results of testing and application of PSP.

	# of iterations		# of evaluated points		# of converged frontier points		# of converged frontier points / # of evaluated points	
	Median	Range	Median	Range	Median	Range	Mean	S.D.
Problem 1	5	[4 5]	71	[60 86]	64	[52 80]	0.91	0.02
Problem 2	11.5	[9 20]	38.5	[31 55]	24.5	[19 30]	0.61	0.06
Problem 3	31.5	[26 48]	299.5	[272 346]	204.5	[192 249]	0.69	0.03
Panel Design	22	[9 45]	86.5	[50 131]	27	[22 34]	0.32	0.07

As one can see from the table, the total number of expensive points is small for all problems. Considering the small number of iterations, parallel computation could be potentially implemented. Among the modest number of sample points, the ratio of converged frontier points to the total number of sample points are more than 60% for all the test problems. The low ratio of the panel design problem will be discussed later. PSP is also found robust with small variations on the point ratio between different runs. To better observe the accuracy of converged frontier points, all of the evaluated points are plotted together with the feasible performance space, as shown in Figures 5~7. It is clear from the plots that the converged frontier points override exactly or are very close to the real Pareto frontier.

6. Multi-objective Optimization for Fuel Cell Component Design

The multiple function panel is a key component of a patented Proton Exchange Membrane (PEM) fuel stack [26]. The panel consists of a copper fin sandwiched between two square flat copper sheets. Figure 8 illustrates a section of the panel.

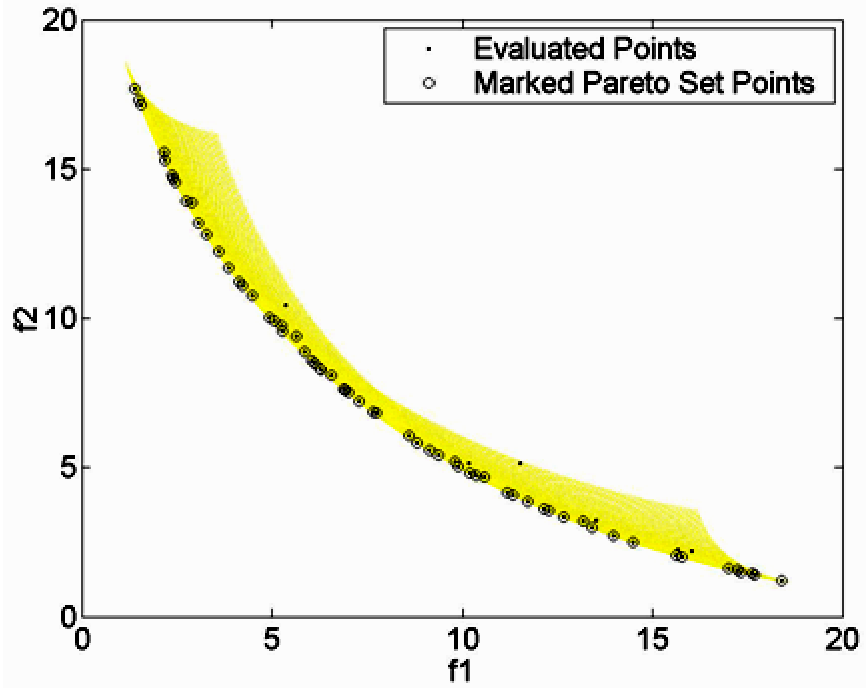

Figure 6 Performance space, evaluated points, and Pareto set points for Problem 1.

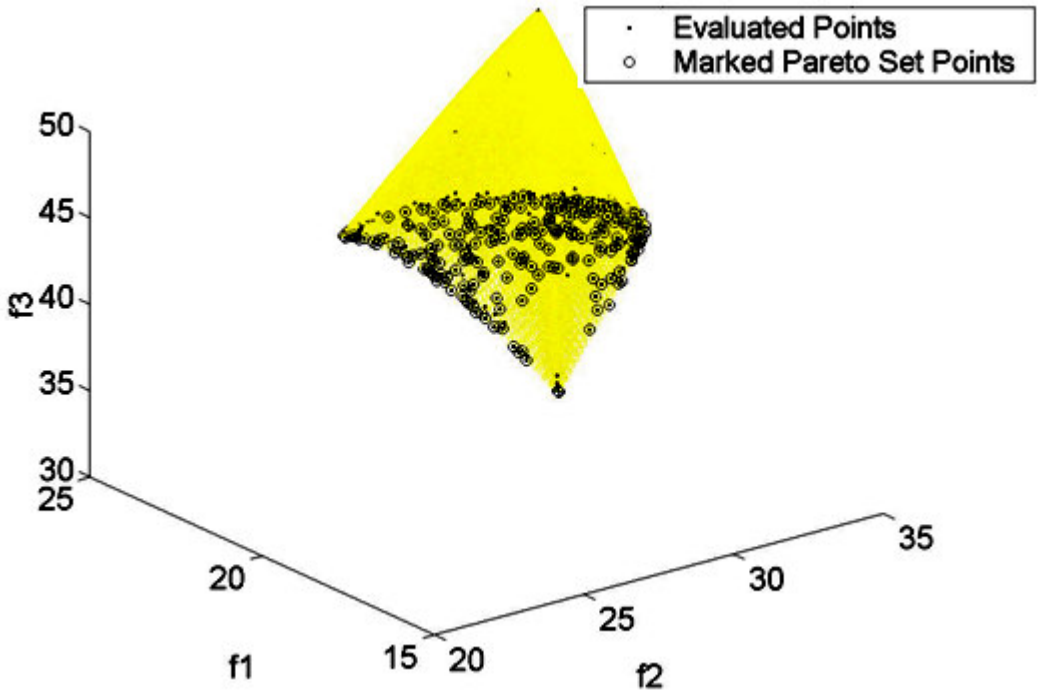

Figure 7 Performance space, evaluated points, and Pareto set points for Problem 3.

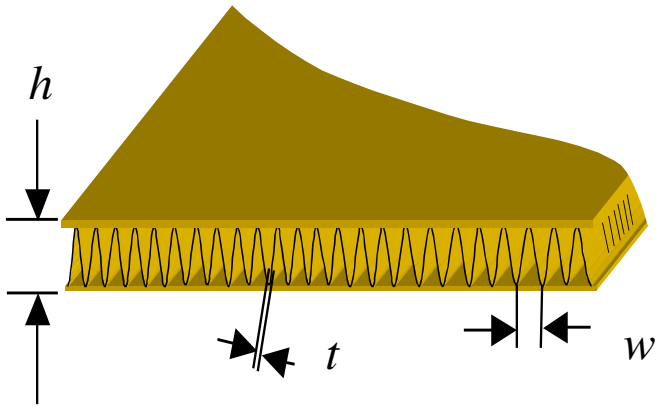

Figure 8 A section of the multiple function panel and design variables.

The panel has three functions: 1) to function as a distributed spring system to compensate the enormous amount of hydro and thermal expansions during the fuel cell operation, 2) to function as a radiator to get rid of the heat generated during the operation, and 3) to function as a conductor to collect electrons for a stack of many fuel cells. The panel is to be made of copper coated with graphite to satisfy the third function requirement, and in the mean time to avoid contamination of metal particles to PEM fuel cells. Design of the panel for the rest two functions is formulated as a MOO problem as follows, with *h*, *w*, and *t* as the 3 design variables.

Minimize: $f_1(h,w,t) = \left| h - \sqrt{(\frac{h}{\sin\theta} - u)^2 - \frac{1}{4}w^2} - 0.034 \right| / 0.034$

Maximize: $f_2(h,w,t) = findht(h,w,t)$ (10)

Subject to:

$$f_2(h,w,t) \geq 340\,; 4 \leq h \leq 9\,;\ 4 \leq w \leq 9\,;\ .06 \leq t \leq 0.6$$

Where $tg\theta = \frac{2h}{w}$, $u = \frac{Pwh}{2Et\sin^2\theta}$. f_1 denotes the ratio of the difference between the panel's real deformation and the ideal deformation over the ideal deformation under a given pressure, *P*. E is the Young's modulus of copper. f_2 is the exit air temperature from the panel, which is to be maximize for the radiation purpose. *findht* is a program which finds the exit air temperature, which can be

considered as a black-box function. Details of the program can be found in Ref. [26]. Results obtained by the Pareto-Set-Pursuing method are listed in Table 1.

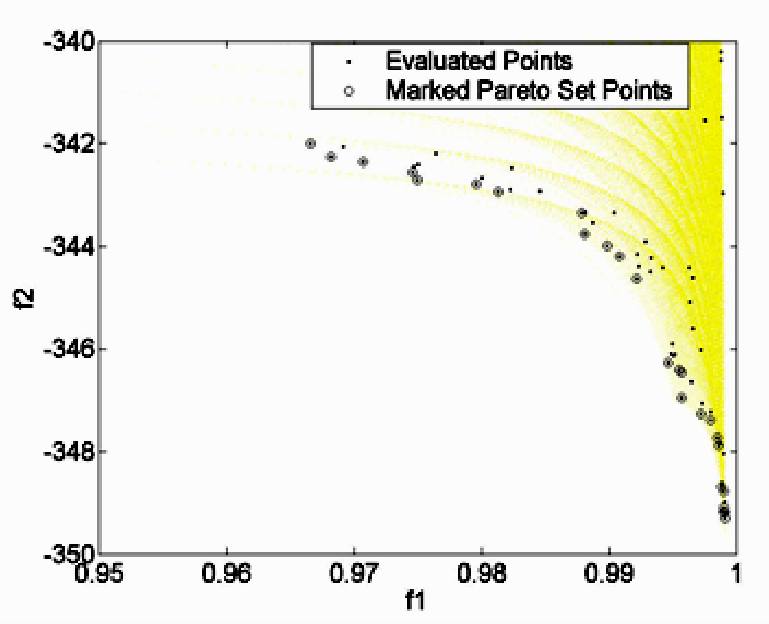

Figure 9 Performance space, evaluated points, Pareto set points for the panel design.

Figure 9 illustrates the results. The unit for f_2 is Kelvin. Values of f_2 are listed as negative as the optimization process minimizes the negative of f_2 for maximization. One can see from Table 1 that the ratio of Pareto set points is not very high for this problem. It is because the performance space is discontinuous with long narrow "tails". It took extra expenses to generate sample points in the upper left region of the Pareto set.

7. Discussions

Features of PSP

One can possibly observe from the PSP procedure that PSP has following two properties: 1) as PSP always samples in the entire design space, every Pareto set point has a probability to be drawn and this probability is much higher than that of non-Pareto set points, according to the sampling guidance function; 2) as the iteration continues, points in $(PS)_k$ will be closer to the real Pareto frontier than those

in $(PS)_{k-1}$, and thus PSP converges to the real Pareto frontier if sampling continues to infinity. From test results and the design application, the proposed PSP method demonstrates great efficiency for MOO problems. By reflecting the PSP process and especially its sampling strategy, it is found that a reduction of search space is inherent by utilizing the nature of the Pareto set. Let's still use the example in Eq. (5) for explanation. Assume at Iteration 1 we randomly sample two design points *a* and *b* in the design space, as shown in Figure 10. Points *a* and *b* are then evaluated by calling the expensive black-box functions and thus their objective function values are obtained. As a result, points A and B in the performance space corresponding respectively to points *a* and *b* are shown in Figure 11.

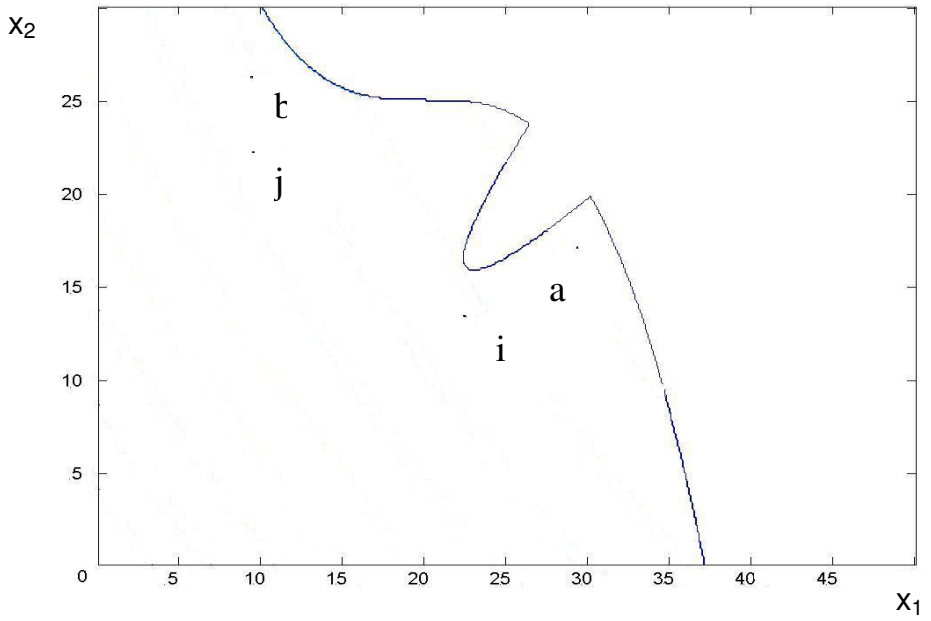

Figure 10 Sample points in the design space at the 1st iteration.

Assume the coordinates of A and B in the performance space are (f_{1A}, f_{2A}) and (f_{1B}, f_{2B}). Currently these two sampling points A and B are frontier points in the performance space. According to the definition of Pareto set points, any point I falls in the region defined by $f_{1I}>f_{1A}$ and $f_{2I}>f_{2A}$ is dominated by Point A. Similarly, any point J falls in the region defined by $f_{1J}>f_{1B}$, $f_{2J}>f_{2B}$ is dominated by Point B. Only points in the shaded area in Figure 11 have potential to dominate Point A or B. Points i and j corresponding to I and J respectively are also plotted in the design space, as shown in Figure 10. As

PSP only draws sample points from current frontier points (though they are a mix of real points and predicted points), it implies that roughly only the shaded regions in Figure 11 are explored further in the entire performance space. The rest of the regions are excluded. In this way the search space from the perspective of performance shrinks a great deal by using only two points. By continuing sampling in the shaded area in Figure 11, new frontier points can be identified and the shaded area will be further shrunk. Therefore, in brief, because of the characteristics of the Pareto set, the PSP sampling discards dominated regions in the performance space and thus converges rapidly.

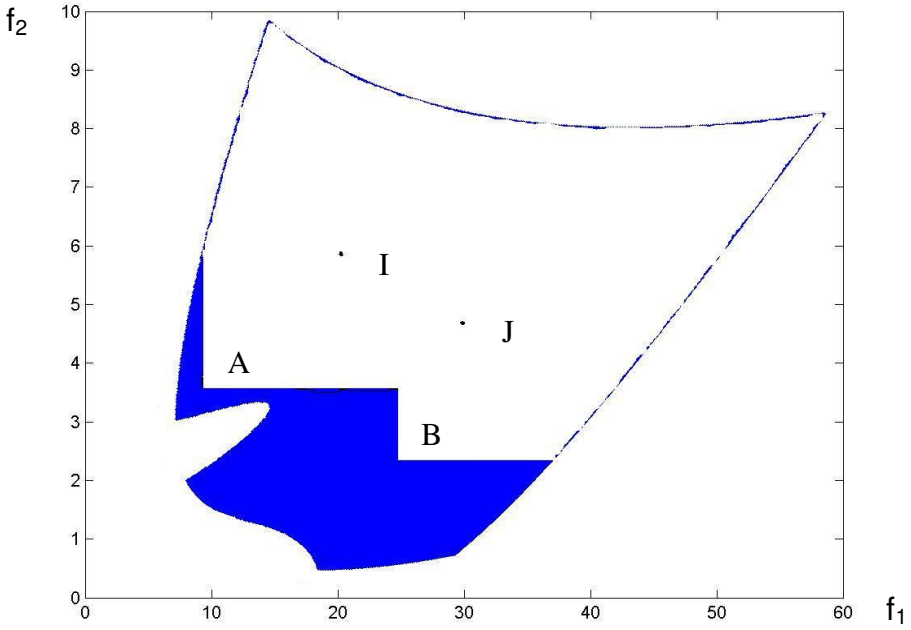

Figure 11 Sample points in the performance space at the 1st iteration

In the process of PSP development, an alternative method to generate initial sample points is explored. That is, we run optimization on each single objective function regardless of other objectives. The optimum of each objective function is used as initial sample points for PSP. In theory these optima are in the Pareto set. By searching for these optima, it is expected that PSP can be sped up. It is found through testing that there is no obvious improvement on the convergence speed of PSP. Furthermore, the optimization processes used to find initial points consume many expensive function evaluations. Overall, this idea requires more function evaluations with no apparent benefits.

Comparison of PSP with Other Approximation-based Methods

Current state-of-art approximation-based methods strive to build an accurate approximation model [13, 15], on which the Pareto frontier is searched. Using the method in Ref. [13] as an example of these methods, its procedure is illustrated in Figure 12a, which is adapted from Ref. [13]. In contrast, the procedure of PSP is simply illustrated in Figure 12b. As one can see that PSP 1) does not need to validate the approximation model as the model is only used to guide the sampling; 2) no third-party optimization algorithm such as GA is called in PSP; and 3) no verification of the frontier points is required since all the identified frontier points in PSP have been evaluated in the process. The build-accurate-model-first approach as in Ref. [13] could be efficient when the black-box function is simple and of low dimension. However, the cost to identify and validate the model could be high, especially when the model is complex. Ref. [23] stated that it was almost impossible to inexpensively construct an accurate approximation model for high-dimensional problems, thus the reliance on the accuracy of approximation model raises concerns. PSP provides an alternative in which the approximation accuracy is not critical but the Pareto frontier points can still be efficiently obtained.

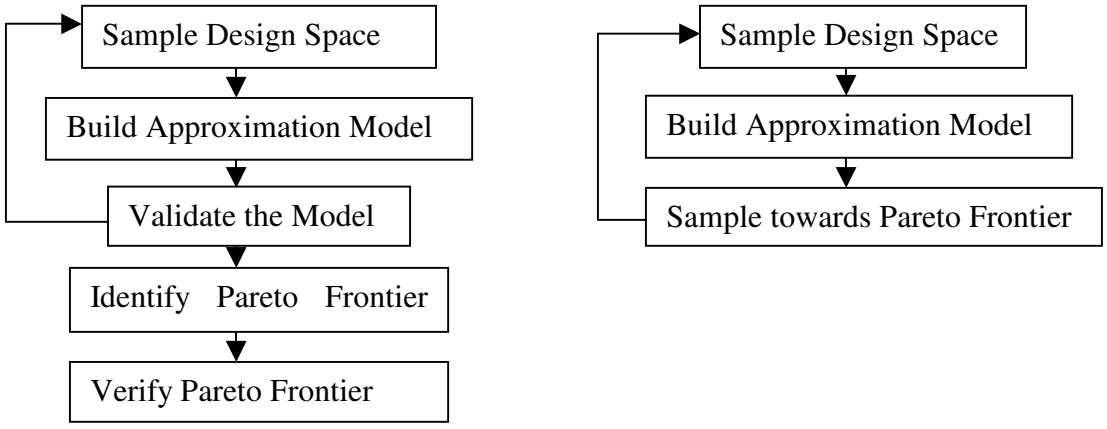

Figure 12 Comparison of PSP with the method in Wilson *et al.* [13]: (a) the procedure of the method in Ref. [13]; (b) a simplified illustration of PSP procedure.

Comparison of PSP with other Direct Search Methods for MOO Problems

The PSP method can also be used as an independent direct search method for all MOO problems. It would be ideal if PSP can be compared with other direct search methods such as GA-based methods by examining their converged Pareto sets. There are a number of difficulties, however. First, the obtained Pareto sets are usually not published. If those data are available, we can see if two sets dominate each other. If all the points in one set are dominated by the points in the other, we can safely conclude that the latter is a better solution set. But the more likely is that only some points in one set are dominated by points in the other. In this case, it is almost impossible to establish an objective criterion to compare the quality of the two [27]. The same is true if both sets are real Pareto sets and therefore no point is dominated by any other point in either set. Given such difficulties, our result is thus only compared visually with the real frontier generated by evaluating a fine grid of points as shown in Figures 5-7, 9.

8. Conclusion

This work presents a new multi-objective optimization (MOO) method, the Pareto Set Pursuing (PSP) method. PSP is especially suitable for design problems involving expensive black-box functions, though it could also be used as a general-purpose MOO method. This approach provides decision makers with a set of Pareto set for choices without any *a priori* knowledge of the objective functions or preferences. Solutions obtained by this method can reflect the entire Pareto optimal frontier, even when the frontier surface is highly non-linear, e.g., non-convex or concave. This approach also automatically captures the Pareto optimal frontier without calling any formal optimization process. It uses approximation to guide the sampling process only and does not demand an accurate approximation model. Through tests and applications, PSP is found robust and efficient, and the Pareto set points found by PSP are real or close-to-real Pareto set points and spread closely and evenly over the entire Pareto optimal frontier. The defined sampling guidance function may be useful for developing other discriminative sampling schemes for various purposes.

Future research will examine closely the applicability of PSP to high-dimensional problems with many design objectives, how to compare PSP with other direct search MOO methods, and how PSP can be improved by replacing the random sampling with more controlled sampling methods such as Latin Hypercube sampling, and so on.

Acknowledgements

Funding from Natural Science and Engineering Research Council (NSERC) of Canada is gratefully appreciated. The authors would also like to thank anonymous reviewers for their valuable comments that resulted in a much improved manuscript.

References

1. Keeney, R.L. and Raifa, H. (1976). Decisions with multiple objective: preferences and value trade-off. John Wiley and Sons, New York.
2. Marler, R. T. and Arora, J. S. (2004) Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization,* 26: pp. 369-395.
3. Chen, W., Wiecek, M.M. and Zhang, J. (1999). Quality utility -- a compromise programming approach to robust design. *Journal of Mechanical Design, Transactions of the ASME*, 121: pp. 179-187.
4. Messac, A. (1996). Physical programming: effective optimization for computational design. *AIAA Journal*, 34(1): pp. 149-158.
5. Tappeta, R.V. and Renaud, J.E. (1999). Interactive multiobjective optimization procedure. *AIAA Journal*, 37(7): pp. 881-889.
6. Tappeta, R.V., Renaud, J.E., Messac, A. and Sundararaj, G. (2000). Interactive physical programming: tradeoff analysis and decision making in multicriteria optimization. *AIAA Journal*, 38(5): pp. 917-926.
7. Schaumann, E.J., Balling, R.J. and Day, K. (1998). Genetic algorithms with multiple objectives. *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, St. Louis, MO, AIAA Vol. 3, Sept. 2-4, 1998, pp. 2114-2123, Paper No. AIAA-98-4974.
8. Deb, K. (1999). Evolutionary algorithms for multi-criterion optimization in engineering design. *Proceedings of evolutionary algorithms in engineering & computer science*, Eurogen-99.
9. Deb, K., Mohan, M. and Mishra, S. (2003). *A fast multi-objective evolutionary algorithm for finding well-spread pareto-optimal solutions*. Report No. 2003002, Indian Institute of Technology Kanpur, Kanpur.
10. Srinivas, N. and Deb, K. (1995). Multiobjective optimization using nondominated sorting in genetic algorithms. *Journal of Evolutionary Computation*, 2(3): pp. 221-248.

11. Nain, P.K.S. and Deb, K. (2002). *A computationally effective multi-objective search and optimization technique using coarse-to-fine grain modeling*. Report No. 2002005, Indian Institute of Technology Kanpur, Kanpur.
12. Luh, G.-C., Chueh, C.-H. and Liu, W.-W. (2003). MOIA: multi-objective immune algorithm. *Journal of Engineering Optimization*, 35(2): pp. 143-164.
13. Wilson, B., Cappelleri, D.J., Simpson, T.W. and Frecker, M.I. (2001). Efficient Pareto frontier exploration using surrogate approximations. *Optimization and Engineering,* 2: pp. 31-50.
14. Li, Y., Fadel, G.M. and Wiecek, M.M. (1998). Approximating Pareto curves using the hyper-ellipse. *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, St. Louis, Paper No. AIAA-98-4961.
15. Yang, B. S., Yeun, Y.-S., and Ruy, W.-S. (2003). Managing Approximation Models in Multiobjective Optimization. *Structural and Multidisciplinary Optimization*, 24: pp. 141-156.
16. Ross, S. M. (2002). *Simulation*, 3rd edition, Academic Press, San Diego, CA.
17. Fu, J.C. and Wang, L. (2002). A random-discretization based Monte Carlo sampling method and its applications. *Methodology and Computing in Applied Probability*, 4: pp. 5-25.
18. Wang, L., Shan, S. and Wang, G.G. (2004). Mode-Pursuing sampling method for global optimization on expensive black-box functions. *Journal of Engineering Optimization*, 36(4): pp. 419-438.
19. Audet, C. and Dennis, J. E. (2004). A Pattern Search Filter Method for Nonlinear Programming Without Derivatives. *SIMA Journal on Optimization*, 14(4): pp. 980-1010.
20. Michalewics, Z. (1995). A Survey of Constraint Handling Techniques in Evolutionary Computation Methods. *Proceedings of the Fourth Annual Conference on Evolutionary Programming*, J. McDonnell, R. Reynolds, and D. Fogel (ed.), MIT Press, Cambridge, MA: pp. 135-155.
21. Myers, R. H. and Montgomery, D. C. (1995). Response surface methodology, process and product optimization using designed experiments. John Wiley & Sons, Inc.
22. Jin, R., Chen, W. and Simpson, T. W. (2001). Comparative Studies of Metamodeling Techniques under Multiple Modeling Criteria. *Journal of Structural and Multidisciplinary Optimization,* 23(1): pp. 1-13.
23. Ong, Y. S., Nair, P. B., and Keane, A. J. (2003). Evolutionary Optimization of Computationally Expensive Problems via Surrogate Modeling. *AIAA Journal*, 41(4): pp. 687-696.
24. Ray, T., and Tsai, H. M. (2004). Swarm Algorithm for Single- and Multiobjective Airfoil Design Optimization. *AIAA Journal*, 42(2): pp. 366-373.
25. Dong, Z. (1999). Fuel cell stack assembly, *Patent Cooperation Treaty Patent WO*, AU, CA, CN, IN, JP, KR, SG, US, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE). Patent No. 99/57781, November 11, 1999.
26. Wang, G.G. (1999). *A quantitative concurrent engineering design method using virtual prototyping-based global optimization and its application in transportation fuel cells*. Ph.D. Dissertation Thesis, University of Victoria, Victoria, Canada.
27. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., and da Fonseca, V. G. (2003). Performance Assessment of Multiobjective Optimizers: An Analysis and Review. *IEEE Transactions on Evolutionary Computation*, 7(2): pp. 117-132.

Nomenclature

c_0	A constant
np	Number of so-far evaluated sample points
l	The exponent for fitness value computation
L	Number of points having fitness value larger than or equal to 1
m	Number of design objectives
n	Number of design variables
k	Number of points; an index of iteration
p	Number of inequality constraints
P	Pressure on the multi-function panel within a fuel cell stack
q	Number of equality constraints
t	Thickness of the fuel cell panel
X	The design variable vector, $x = [x_1, x_2, \ldots, x_n]$
$x^{(i)}$	The i-th design point
x^*	A Pareto optimum
x_r^l	Lower bound of the r^{th} variable
x_r^u	Upper bound of the r^{th} variable
$h(x)$	Equality constraint functions
$g(x)$	Inequality constraint functions
G_i	Pareto fitness value of the i^{th} design
$\overline{G}$	Average fitness value of frontier points
f_{sk}^i	Scaled k^{th} objective function value of the i^{th} design
$rawf_{1,i}$	Un-scaled value of the first objective for the i^{th} design
$rawf_{1,\min}$	Minimum un-scaled value of the first objective among all designs
$rawf_{1,\max}$	Maximum un-scaled value of the first objective among all designs
$\hat{z}$	Sampling guidance function
$f(x)$	A general objective function
$\hat{f}(x)$	The approximation model of $f(x)$
$F(x)$	Vector of all of the objectives
$f_i(x)$	The i-th objective function, $i = 1,\ldots,m$
$\hat{f}_i(x)$	Approximation model of the i-th objective function $f_i(x)$
$(PS)_k$	The frontier point set at the k-th iteration
α_i	Regression coefficients of the RBF model
β_i	Regression coefficient of the quadratic polynomial function model
δ	A small constant for model alternation
ρ	A ratio between number of points for PSP convergence