2004-01-0240

Design Space Reduction for Multi-objective Optimization and Robust Design Optimization Problems

G. Gary Wang* & Songqing Shan
Dept. of Mechanical and Industrial Engineering
The University of Manitoba
Winnipeg, MB, Canada R3T 5V6
Tel: 204-474-9463 Fax: 204-275-7507
Email: gary_wang@umanitoba.ca

ABSTRACT

Modern engineering design often involves computation-intensive simulation processes and multiple objectives. Engineers prefer an efficient optimization method that can provide them insights into the problem, yield multiple good or optimal design solutions, and assist decision-making. This work proposed a rough-set based method that can systematically identify regions (or subspaces) from the original design space for multiple objectives. In the smaller regions, any design solution (point) very likely satisfies multiple design goals. Engineers can pick many design solutions from or continue to search in those regions. Robust design optimization (RDO) problems can be formulated as a bi-objective optimization problem and thus in this work RDO is considered a special case of multi-objective optimization (MOO). Examples show that the regions can be efficiently identified. Pareto-optimal frontiers generated from the regions are identical with those generated from the original design space, which indicates that important design information can be captured by only these regions (subspaces). Advantages and limitations of the proposed method are discussed.

Keywords: Rough Set, multi-objective optimization, robust design optimization, space reduction, engineering design

INTRODUCTION

In modern design, it is common to deal with optimization problems with multiple objectives. For example, concurrent engineering design requires the optimization from many product life-cycle aspects such as performance, manufacturing, assembly, maintenance, and recycling. Multi-objective optimization (MOO) has been an intensively studied topic [1]. Normally, multiple objectives are aggregated into one objective either by the weighted-sum method, deviation sum method, preference function, or utility function. Other multi-objective optimization methods include the constrain-oriented method and the mini-max formulation strategy. The constraint-oriented method treats all but one objective as constraints. By controlling the upper bounds of the objectives, the Pareto set can be obtained. The mini-max strategy minimizes the maximum relative deviation of the objective function from its minimum objective function value. In addition to the above-mentioned deterministic approaches, genetic algorithms (GA) have been successfully applied in solving multiple objective functions. Robust design optimization (RDO) looks for a minimum of the objective function that is insensitive to the variation of design variables. It is found that the RDO can be normally formulated as a bi-objective robust design (BORD) problem [2] by minimizing simultaneously the mean and variance of the objective function with respect to design variables. Therefore, in this work, RDO is considered as one of the special cases of the MOO problems.

In industry, computer aided design and analysis (CAD/CAE) tools are extensively used for design evaluation and simulation. The process of using these tools, such as finite element analysis (FEA) and computational fluid dynamics (CFD) tools, are often computation-intensive. Because such processes could provide engineers very accurate prediction of product behavior, they are expected to be integrated with optimization to search for the design optimum. The problem is that these processes are not transparent to the optimizer; there is no explicit and clean objective function. Gradient information of such function is too expensive to obtain or is unreliable [3]. Also, the objective / constraint functions as well as design variables can be both discontinuous and continuous. The field of optimization provides us many quantitative and systematic search strategies that can help tackle the concurrent design problem [4]. However, there are several limitations of classic optimization methods that

* Corresponding author

hinder the direct application of these methods in modern engineering design. First, classic optimization methods are based on explicitly formulated and / or cheap-to-compute models, while engineering design involves implicit and expensive-to-compute models. Second, classic methods only provide a single solution, while engineers prefer multiple design alternatives. Third, the classic optimization process is sequential, non-transparent, and provides nearly no sights to engineers. Lastly, to apply the optimization methods, high-level expertise on optimization is also required for engineers. Therefore, there is a gap between the capability of classic optimization and the demand of modern engineering design. Furthermore, from a design engineer's perspective, multiple good solutions are always preferred than a single optimum solution since in the real practice the obtained optimum might not be feasible. Also, engineers would prefer a decision support tool based on optimization and no high-level optimization skills is required to use such a tool. Such a decision support tool should be able to give the engineers more insights to the design problem, and "explain" intuitively why the suggested solutions are good (or optimal). Ideally, the suggested solution should be robust, reliable, and globally optimal. Current metamodeling-based optimization approaches aim to approximate the computation-intensive processes with explicit and simple models. The major difficulty is the so-called "curse of size", i.e., with the increase of the number of design variables, the number of sample points needed to construct a good approximation model increases exponentially [5]. The traditional method is to reduce the dimensionality of the problem [6]. Recent trend seems to aim at reducing the size of the search space by searching for feasible or attractive regions [7-11].

To address the need for multiple solutions and the efficiency for optimization, this research proposes a design space reduction method for multi-objective optimization (MOO) and robust design optimization problems (RDO). The authors' previous research applied the rough set for space exploration and global optimization [12]. Such a method established a mapping from a given function value to the attractive design space, i.e., regions (or subspaces) in the original design space can be identified so that points in the subspaces lead to function values equal to or lower than the given function value. Based on the previous research, goals of objectives are used first to identify attractive design regions for each individual objective so that its goal can be achieved. If there is no common design subspaces among all attractive subspaces, it is most likely that the goals could not be simultaneously satisfied and goals are to be adjusted. Otherwise, further search can concentrate on the common subspace to increase the optimization efficiency. Following sections will briefly review the rough set approach for the identification of attractive spaces. Then the strategy for space reduction on MOO problems will be described. RDO, as a special case of MOO, will be briefly discussed. Test examples and results are then explained and discussed.

APPLICATION OF ROUGH SET FOR ATTRACTIVE SPACE IDENTIFICATION

Our previous work [12] documented in detail the related theory of rough set and how it was applied to identify attractive design spaces. In this work, a brief review is given in order to introduce some associated new concepts for the ease of understanding this work. Assume that we have a function to be optimized as shown in Equation 1. This function is in fact well known as the six-hump camel back problem as it has six local optima in the given design space [13]. For simplicity, this problem is referred as SC problem in later sections.

$$f_{sc}(x) = 4x_1^2 - 2.1x_1^4 - \tfrac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4, x_{1,2} \in [-2, 2] \quad \text{Equation 1}$$

First a decision system is constructed as in Table 1. Where a_1 and a_2 refer to x_1 and x_2 respectively; u's are sample points (or objects); d is the decision value according to a given criteria d_t (for this case, if the function value of the point u_i is larger than d_t = -0.5, it is coded as '1', otherwise the decision is '0'.).

Table 1 Example of an information (decision) system.

S	a_1	a_2	d
u_1	1.158705	-1.372335	1
u_2	-0.225895	-0.763139	0
u_3	0.414520	-0.553701	1
u_4	-1.080601	1.612387	1
u_5	1.831858	-0.093501	1
u_6	-0.876749	-1.934921	1
u_7	-1.972033	-1.043081	1
u_8	0.198583	1.895447	1
u_9	0.661393	-0.973502	1
u_{10}	-0.225895	-1.572636	1
u_{11}	-0.061620	0.873131	0

Table 2 a set of cuts for the decision system defined in Table 1

A	Cuts
a_1	-0.5505
	0.0685
a_2	-1.472

The rough set tool can automatically find some values of a_1 and a_2, which are called cuts that can discern the points according to their decision values. The results for the SC problem are shown in Table 2 and Figure 1; detail procedures for obtaining the results are in Ref. [12]. As shown in Figure 1, the circle dots indicate the two points whose decision is '0'. The generated cuts separate these two points from the rest of points whose decision is '1'. Then the rough set approach will name each partition of a_1 and a_2 based on the cuts with zero and integer numbers. For example, all the points falling in the upper right rectangle in Figure

1 will be represented by a single object u_1 with a_1=2 and a_2 =1. Thus Table 1 is simplified to Table 3.

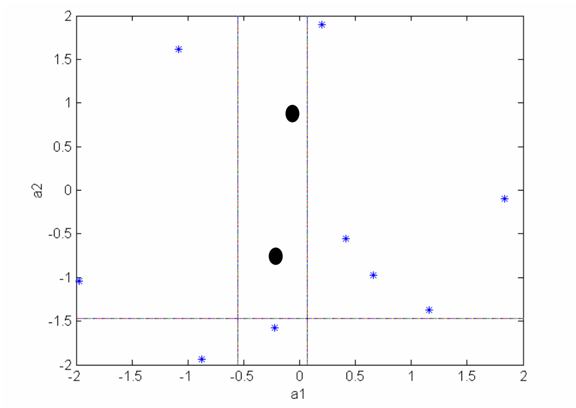


Figure 1 A geometrical representation of data partition and cuts.

Table 3 A simplified decision system.

S*	a1	a2	d
u_1	2	1	1
u_2	1	1	0
u_3	0	1	1
u_4	0	0	1
u_5	1	0	1

By sorting the simplified decision table, the rough set method can yield Table 4, the rules of the decision system. For the SC problem, the rule (a_1= 1) & (a_2 = 1) ⇒ (d = 0) leads to a reduced design space a_1=[-0.5505, 0.0685] and a_2=[-1.472, 2] after translating the rules back to its original decision system. We know that in the reduced space all the so-far obtained sample points are attractive, i.e., their function values are equal to or lower than the threshold d_t.

Table 4 Rules for the decision system.

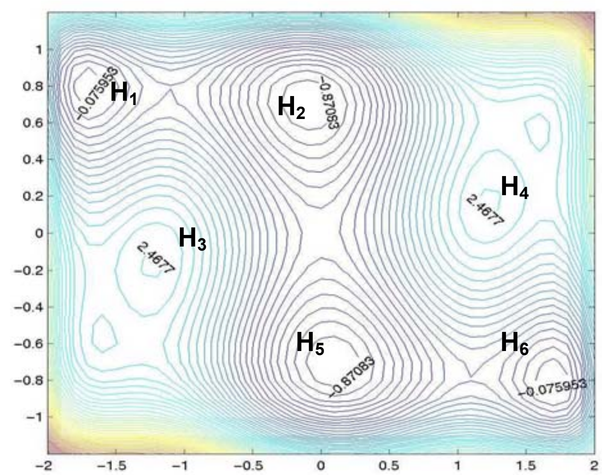
(a_1 = 2) ⇒ (d = 1)
(a_1 = 0) ⇒ (d = 1)
(a_1= 1) & (a_2 = 1) ⇒ (d = 0)
(a_2 = 0) ⇒ (d = 1)

In the authors' previous work [12], sample points are generated iteratively in the original design space until an overlapping coefficient, C, is larger than 0.65. The final obtained design space is considered the attractive space. The definitions of attractive space and C are given below.

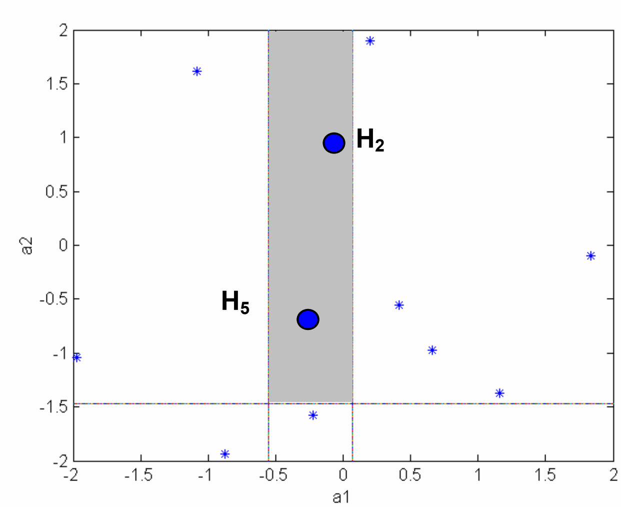
Given the design variable x (x_1, x_2, …, x_m) and its value range X, the objective function f(x) (f_1(x), f_2(x), …, f_n(x)) and its local optima f_{min}(x), the constraint function g(x) (g_1(x), g_2(x), …, g_k(x)), and the decision threshold d_t, we introduce a number of definitions. A design space is defined as $S_d = (x, f(x)) = \{X \in R \mid \forall x \in X, f(x) \in R\}$. An attractive space is $S_a = (x, f(x), g(x), d_t) = \{X \in R \mid \forall x \in X, f(x) \le d_t \text{ and } g(x) \le 0\}$ within which any x makes the value of the objective function f(x) equal to or less than a given decision threshold d_t and satisfies the constraints $g(x) \le 0$.

Assuming that $S_{a_i} = \{s_{i_1} \cup s_{i_2} \cup ... \cup s_{i_m}\}$ and $S_{a_{(i+1)}} = \{s_{(i+1)_1} \cup s_{(i+1)_2} \cup ... \cup s_{(i+1)_n}\}$ are the attractive design spaces obtained through the i-th and ($i+1$)-th samplings respectively, we define an overlap coefficient C as following:

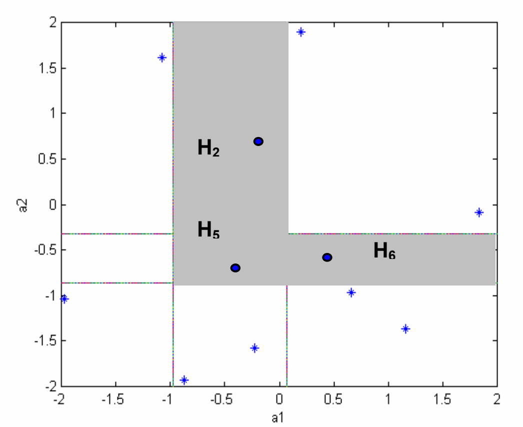
$$C = \frac{Vol\ (S_{a_i} \cap S_{a_{(i+1)}})}{Vol\ (S_{a_i} \cup S_{a_{(i+1)}})} \qquad \text{Equation 2}$$



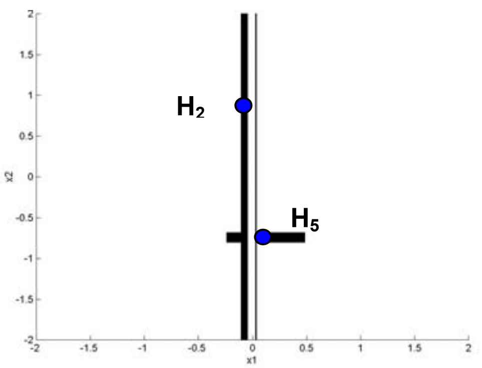
a. Contour of the original SC function



b. A representation of space partition when d_t = -0.5



c. A representation of space partition when $d_t = 0$



d. The final converged space with d_t = -0.5

Figure 2 Comparison of space partitions with the contour of the SC function.

Where, the *Vol*( ) function means the volume of the space, which is defined as the product of ranges along each *x* component direction. When the number of samples or objects increases to the infinity, the two attractive design spaces S_{a_i} and $S_{a_{(i+1)}}$ will overlap each other, that is, we have

$$C = \lim_{i \to \infty} \frac{Vol\,(S_{a_i} \cap S_{a_{(i+1)}})}{Vol\,(S_{a_i} \cup S_{a_{(i+1)}})} = 1 \qquad \text{Equation 3}$$

Assuming that $S_{a_i} = \{s_{i_1} \cup s_{i_2} \cup \ldots \cup s_{i_m}\}$ and $S_{a_{(i+1)}} = \{s_{(i+1)_1} \cup s_{(i+1)_2} \cup \ldots \cup s_{(i+1)_n}\}$ are the attractive design spaces obtained through the *i-th* and (i+1)-th samplings respectively, we define an overlap coefficient *C* as following:

$$C = \frac{Vol\,(S_{a_i} \cap S_{a_{(i+1)}})}{Vol\,(S_{a_i} \cup S_{a_{(i+1)}})} \qquad \text{Equation 2}$$

Figure 2 d shows the final converged space by using the iterative sampling along with the stopping criterion $C \geq 0.65$ with d_t = -0.5.

SPACE REDUCTION FOR MULTI-OBJECTIVE OPTIMIZATION

For engineering design, engineers often have enough insights to know the least achievable goal for each objective. Or, goals are sometimes given by the management to the engineers, for example, cost of the product, performance ranges, and so on. The competitor's product specifications can also function as goals for a new product development. Let's assume a goal for each objective is known in advance. For consistency with the rough set theory, we denote the goal for the objective fi as $d_t^{f_i}$, i = 1, …, m. By applying the aforementioned method, for $\forall\, i^{th}$ objective function, the attractive space $S_a^{f_i}$ can be identified with a given goal $d_t^{f_i}$. Define the intersection of all the attractive spaces $S_a^{f_a}$ as

$$S_a^{f_a} = S_a^{f_1} \cap S_a^{f_2} \cap \cdots \cap S_a^{f_m} \qquad \text{Equation 4}$$

Lemma: For a multi-objective optimization problem with objective functions f_1, f_2, … f_m and their corresponding goals $d_t^{f_1}, d_t^{f_2}, \cdots, d_t^{f_m}$, if $S_a^{f_a} \neq 0$, then $S_a^{f_a}$ is the attractive space for the given multi-objective optimization problem. Otherwise, the given multi-objective optimization problem is not solvable unless goals $d_t^{f_1}, d_t^{f_2}, \cdots, d_t^{f_m}$ are revised.

The attractive space $S_a^{f_a}$ can then be used in lieu of the original design space for further exploration and optimization. Because the computation of each $S_a^{f_i}$ needs only a limited number of function evaluations [12], the overall expenses for computing $S_a^{f_a}$ is the sum of the cost for all of the $S_a^{f_i}$, i = 1, …, m.

ALGORITHM Often $S_a^{f_i}$ obtained by using the rough set tool consists of more than one region in the original design space. These regions could be overlapping with each other. In addition, $S_a^{f_i}$, i =1, …, m overlaps with each other if $S_a^{f_a} = S_a^{f_1} \cap S_a^{f_2} \cap \cdots \cap S_a^{f_m}$ is not empty. To calculate the common spaces, the following algorithm is applied for any two $S_a^{f_1}$ and $S_a^{f_2}$ among $S_a^{f_i}$, i =1, …, m.

<u>Common space search algorithm</u>

Input: $S_a^{f_1}$ and $S_a^{f_2}$ (each is expressed as a matrix with many possibly overlapping subspaces; every two columns represent a subspace).

Output: The common spaces of $S_a^{f_1}$ and $S_a^{f_2}$, $Vol\,(S_a^{f_1} \cap S_a^{f_2})$, $Vol\,(S_a^{f_1})$, and $Vol\,(S_a^{f_2})$

BEGIN:

1) Sort out all the cuts on x_j, *j* = 1, …, n in $S_a^{f_1}$ and $S_a^{f_2}$ from low to high. *n* is the number of design

variables; every two neighboring cuts define an interval of x_j.

2) Set $Vol(S_a^{f_1} \cap S_a^{f_2}) = Vol(S_a^{f_1}) = Vol(S_a^{f_2}) = 0$

3) For (the first interval of x_1, the second interval of x_1, … , the last interval of x_1)

For (the first interval of x_2, the second interval x_2, … , the last interval of x_2)

…

For (the first interval of x_n, the second interval x_n, … , the last interval of x_n)

Obtain S_c, which is the sub-space defined by the intervals of each x component

If $S_c \subset S_a^{f_1}$, then

$Vol(S_a^{f_1}) = Vol(S_a^{f_1})$ + Vol(Sc)

If $S_c \subset S_a^{f_2}$, then

$Vol(S_a^{f_2}) = Vol(S_a^{f_2})$ + Vol(Sc)

If $S_c \subset S_a^{f_1}$ AND $S_c \subset S_a^{f_2}$, then $Vol(S_a^{f_1} \cap S_a^{f_2}) = Vol(S_a^{f_1} \cap S_a^{f_2})$ + Vol(Sc)

Output S_c

Else continue

End

…

End

End

END

This algorithm identifies the common spaces and calculates the volume of the common spaces. It can also be used to calculate the overlapping coefficient C between subspaces, defined by Eq. 3.

ROBUST DESIGN OPTIMIZATION FORMULATION

A standard engineering optimization problem is normally formulated as the following:

$$\begin{aligned} \min \quad & f(x) \\ \text{S.T.} \quad & g_j(x) \le 0, \ j = 1, \cdots, J \\ & x_L \le x \le x_U \end{aligned} \qquad \text{Equation 5}$$

Where *f(x)* is the objective function and $g_j(x)$ is the *j*-th constraint function; x, x_L and x_U are vectors of design variables, their lower bounds and upper bounds, respectively. If the design variable *x* follows a statistical distribution, a robust design problem can be stated as a bi-objective robust design (BORD) problem as the following [2] (the worst case scenario for constraints):

$$\begin{aligned} \min \quad & [\mu_f, \ \sigma_f] \\ \text{S.T.} \quad & g_j(x) + k_j \sum_{i=1}^{n} \left| \frac{\partial g_j}{\partial x_i} \right| \Delta x_i \le 0, \ j = 1, 2, \cdots, J \\ & x_L + \Delta x \le x \le x_U - \Delta x \end{aligned} \qquad \text{Equation 6}$$

Where μ_f and σ_f are the mean and standard deviation of the objective function *f(x)*, respectively. Their values can be obtained through Monte Carlo simulation or the first order Taylor expansion if the design deviation of x_i is small. When using Taylor expansions, μ_f and σ_f can be represented by the following equations [14]:

$$\begin{aligned} \mu_f &= f(x) \\ \sigma_f^2 &= \sum_{i=1}^{n} \left(\frac{\partial f}{\partial x_i} \right)^2 \sigma_{x_i}^2 \end{aligned} \qquad \text{Equation 7}$$

Where σ_{x_i} is the standard deviation of the i-th x component.

From Eq. 6 and 7, the robust design problem becomes a special multi-objective optimization problem. The following section will use two examples to further illustrate the proposed method.

TEST EXAMPLE I

The first test problem is taken from Ref. [2]. The original mathematical problem is formulated below:

$$\begin{aligned} \min \quad & f(x) = (x_1 - 4.0)^3 + (x_1 - 3.0)^4 + (x_2 - 5.0)^2 + 10.0 \\ \text{S.T.} \quad & g(x) = -x_1 - x_2 + 6.45 \le 0 \\ & 1 \le x_1 \le 10 \\ & 1 \le x_2 \le 10 \end{aligned} \qquad \text{Equation 8}$$

The associated BORD problem can be then formulated using Eq. 6.

$$\begin{aligned} \min \quad & \left[\frac{\mu_f}{\mu_f^*}, \frac{\sigma_f}{\sigma_f^*} \right] \\ \text{S.T.} \quad & g(x) = -x_1 - x_2 + 6.45 + 2k\Delta x \le 0 \\ & 1 + \Delta x \le x_1 \le 10 - \Delta x \\ & 1 + \Delta x \le x_2 \le 10 - \Delta x \end{aligned} \qquad \text{Equation 9}$$

Where the size of variation Δx =1.0 ($\Delta x_1 = \Delta x_2 = 1$) and the penalty factor k is taken as 1.0. The ideal solutions are obtained as

$x^*_{\mu_f} = (2.00000 \quad 6.45074)$ for $\mu^*_f = 5.10464$ and $x^*_{\sigma_f} = (3.50559 \quad 4.99187)$ for $\sigma^*_f = 0.416796$.

By considering the standard deviation for both *x* as Δx */3* and using Eq. 7, we have

$$\mu_f(x) = (x_1 - 4.0)^3 + (x_1 - 3.0)^4 + (x_2 - 5.0)^2 + 100$$

$$\sigma_f(x) = \frac{\Delta x}{3}\sqrt{(3.0(x_1 - 4.0)^2 + 4.0(x_1 - 3.0)^3)^2 + (2.0(x_2 - 5.0))^2} \quad \text{Equation 10}$$

The Pareto-optimal frontier for the BORD problem is illustrated by Figure 3, which is identical to that depicted by Chen et al. [2]. The design points that correspond to the Pareto set points are found as shown in Figure 4. Figure 3 and Figure 4 are generated from the original space. New plots will be created in the reduced space to evaluate the impact of the space reduction to potential design information loss.

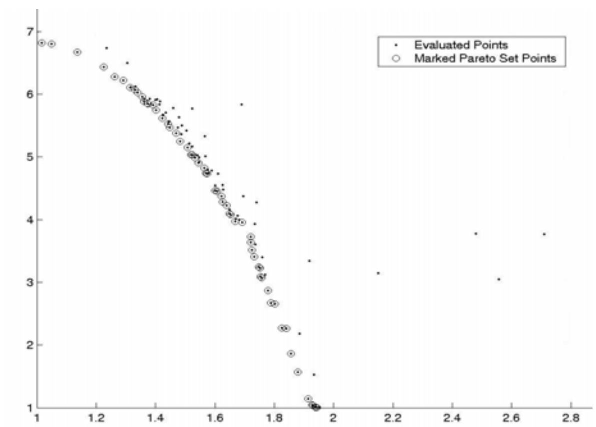


Figure 3 The Pareto-optimal frontier for Problem I (the circles indicate points in the Pareto set.)

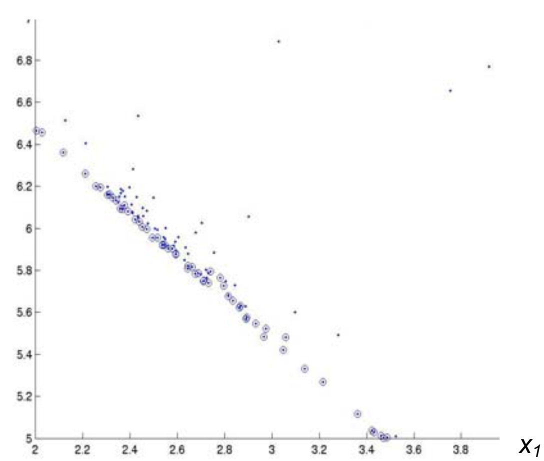


Figure 4 The Pareto design points enclosed by circles.

MINIMIZATION OF FUNCTION MEAN The goal was pre-set as μ_f / μ^*_f =10. The history of space reduction by using the proposed method is recorded in Table 5. As one can see, the process terminates after three iterations with the reduced space $x_1 \in [2 \ 5.3660]$ and $x_2 \in [2 \ 9]$. The total number of function evaluations is 18.

Table 5 the record of the space reduction for the mean function when d_t = 10.

Iteration	Reduced Space	Cumulative # of Function Evaluations	Overlapping Coefficient C
1	$x_1 \in [2.9305 \ 5.2580]$ $x_2 \in [2 \ 9]$	6	N/A
2	$x_1 \in [2.6730 \ 5.5515]$ $x_2 \in [2 \ 9]$	12	0.8086
3	$x_1 \in [2 \ 5.3660]$ $x_2 \in [2 \ 9]$	18	0.6623

MINIMIZATION OF VARIANCE The goal was also set as σ_f / σ^*_f =10. The history of space reduction is recorded in Table 6. For the variance minimization, two regions *A* and *B* are generated with 32 function evaluations after 4 iterations, as shown in the last row of Table 6. The two regions are defined by Space *A*: $x_1 \in [2.8980 \quad 4.2430]$ & $x_2 \in [3.9695 \quad 5.8280]$ and Space *B*: $x_1 \in [2 \quad 4.2430]$ & $x_2 \in [5.8280 \quad 9]$.

Table 6 the record of the space reduction for the variance function when d_t = 10.

Iteration	Reduced Space	Cumulative # of Function Evaluations	Overlapping Coefficient C
1	$x_1 \in [2 \ 4.749]$ $x_2 \in [2.870 \ 9]$	8	N/A
2	a. $x_1 \in [4.0365 \ 4.3120]$ $x_2 \in [2 \ 9]$ b. $x_1 \in [2 \ 4.0365]$ $x_2 \in [5.3285 \ 9]$	14	0.6322
3	a. $x_1 \in [3.5365 \ 4.4375]$ $x_2 \in [2 \ 9]$ b. $x_1 \in [2 \ 3.5265]$ $x_2 \in [5.3390 \ 9]$	23	0.7858
4	a. $x_1 \in [2.8980 \ 4.2430]$ $x_2 \in [3.9695 \ 5.8280]$ b. $x_1 \in [2 \ 4.2430]$ $x_2 \in [5.8280 \ 9]$	32	0.6825

COMMON SPACE The intersection of the space $x_1 \in [2 \quad 5.3660]$ and $x_2 \in [2 \quad 9]$ from the minimization of the function mean and the two spaces from the minimization of variance includes two regions, Space *A*: $x_1 \in [2.8980 \quad 4.2430]$ & $x_2 \in [3.9695 \quad 5.8280]$, and Space *B*: $x_1 \in [2 \quad 4.2430]$ & $x_2 \in [5.8280 \quad 9]$. Space *A* and *B* are depicted in Figure 5. The volume of the reduced space is 19.62% of the original design space.

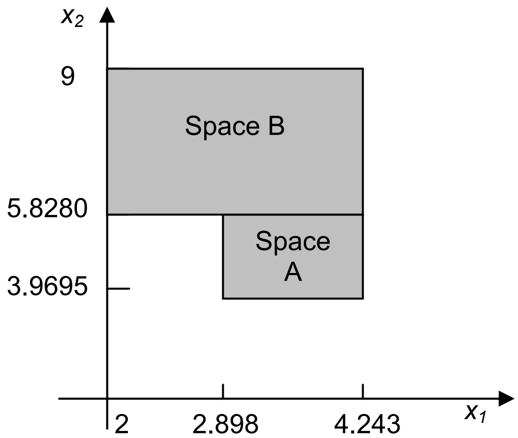


Figure 5 The reduced design space for Problem 1.

RESULT EVALUATION For the reduced space, we need to evaluate if the reduced space fail to capture most, if not all, of the Pareto-optimal points. This is examined from the performance space as well as the design space. If the new spaces are used to search for the Pareto points, we obtain Figure 6 and Figure 7. Comparing Figure 6 and Figure 7 with Figure 3, one can see that the space reduction didn't loose potential Pareto-optimal points from the performance space. Design points in Space *A* can lead to a part of the Pareto-optimal frontier; and design points in Space *B* can lead to the rest part of the Pareto-optimal frontier. In other word, design points in Spaces *A* and *B* can lead to a Pareto-optimal frontier which is identical with Figure 3. Moreover, the method clearly points out that both Space *A* and Space *B* can lead to Pareto solutions. The space reduction was accomplished with only 40 function evaluations in total for both functions. From the design solution perspective, one can see that all of the Pareto points in Figure 4 are included within the reduced design space shown in Figure 5.

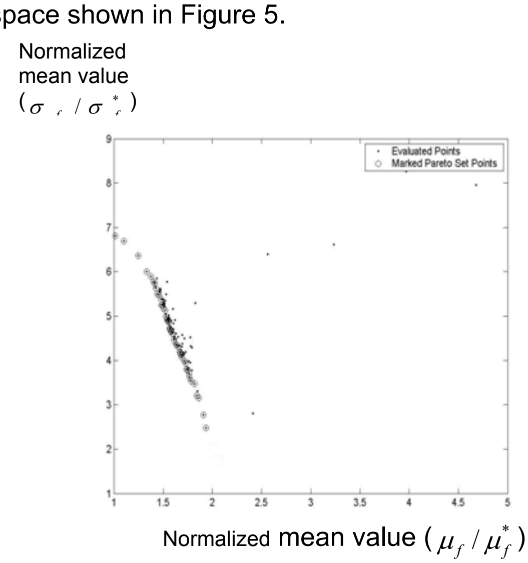


Figure 6 The Pareto-optimal frontier searched within Space A (circled points).

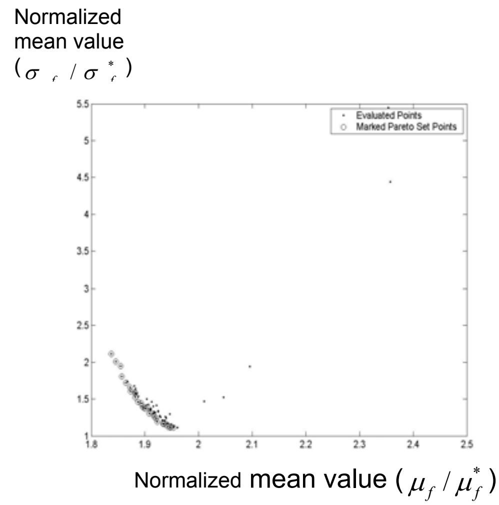


Figure 7 The Pareto-optimal frontier searched within Space B (circled points).

TEST EXAMPLE II

The second example is a simple beam design problem shown in Figure 8. Taken from Ref. [15], this problem is to minimize the vertical deflection of an I-beam under given loads as well as the cross-section area, while simultaneously satisfying the stress constraint. Various parameter values for the problem are:

- Allowable bending stress of the beam = 6 kN/cm^2.
- Young's Modulus of Elasticity (E) = 2x104 kN/cm^2.
- Maximal bending forces P = 600 kN and Q = 50 kN.
- Length of the beam (L) = 200 cm.

The optimization problem can be formulized as below:

Minimize the Vertical Deflection

$$f_1(x) = \frac{5000}{\frac{1}{12} x_3 (x_1 - 2x_4)^3 + \frac{1}{6} x_2 x_4^3 + 2 x_2 x_4 (\frac{x_1 - x_4}{2})^2} \qquad \text{Equation 11}$$

Minimize the Cross Section Area

$$f_2(x) = 2x_2 x_4 + x_3 (x_1 - 2x_4)$$

Subject to:

Stress Constraint

$$g_1(x) = \frac{180000 x_1}{x_3 (x_1 - 2x_4)^3 + 2x_2 x_4 [4x_4^2 + 3x_1 (x_1 - 2x_4)]} + \frac{15000 x_2}{(x_1 - 2x_4) x_3^3 + 2x_4 x_2^3} \le 16$$

Initial Design Space

$$10 \le x_1 \le 80, \quad 10 \le x_2 \le 50, \quad 0.9 \le x_3 \le 5, \quad 0.9 \le x_4 \le 5$$

Now let us assume the objective functions are computation-intensive and thus the reduction of the number of function evaluations is our concern.

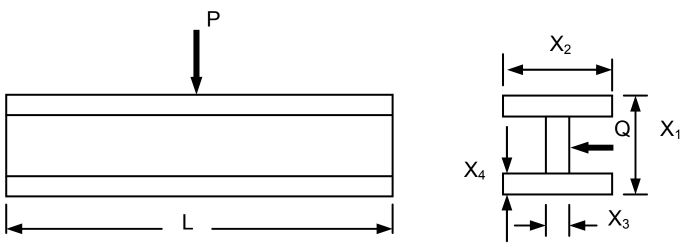


Figure 8 The Beam Design Problem

MINIMIZATION OF THE DEFLECTION With the goal being set as f_1 = 0.02, the final space was found after 6 iterations with 112 function evaluations. For simplicity, the iteration history is omitted here. The sub-spaces found are listed below:

Space *A*:
x_1 [73.5385 80], x_2 [36.0035 50], x_3 [0.9 5], x_4 [0.9 5]
Space *B*:
x_1 [60.027 65.509], x_2 [29.0380 36.0035], x_3 [0.9 5], x_4 [0.9 5]
Space *C*:
x_1 [65.5090 73.5385], x_2 [36.0035 50], x_3 [0.9 5], x_4 [0.9 5]
Space *D*:
x_1 [54.6700 60.0270], x_2 [36.0035 50], x_3 [2.7665 5], x_4 [0.9 5]
Space *E*:
x_1 [65.5090 80], x_2 [29.0380 36.0035], x_3 [2.7665 5], x_4 [0.9 5]
Space *F*:
x_1 [54.6700 65.5090], x_2 [36.0035 50], x_3 [0.9 5], x_4 [3.0510 5]
Space *G*:
x_1 [73.5385 80], x_2 [10 50], x_3 [0.9 5], x_4 [3.0510 5]

MINIMIZATION OF THE CROSS-SECTION AREA The goal for the cross section area is set as f_2 = 320. The sampling process converges after 4 iterations only with 23 function evaluations. The obtained sub-spaces are:

Space *I*:
x_1 [36.9375 80], x_2 [23.0050 39.5395], x_3 [0.9 5], x_4 [0.9 3.1140]
Space *II*:
x_1 [36.9375 80], x_2 [1108825 23.0050], x_3 [0.9 5], x_4 [3.1140 5]
Space *III*:
x_1 [10 80], x_2 [39.5395 50], x_3 [0.9 5], x_4 [0.9 3.1140]

COMMON SPACES The intersection between Spaces *A~G* and Space *I~III* is calculated by using the algorithm described in Section Space Reduction for Multi-objective Optimization. As a result, 41 sub-spaces are generated. The total volume of these subspaces is only 8.55% of the original design space. Engineers can pick design solutions from these subspaces, which very likely satisfy the goals for all the objectives. If further optimization were to be carried out, the number of spaces (and thus the optimization processes) would be too large. Ideally the optimal number of subspaces may be identified to achieve the best trade-off between the amount of space reduction and the number of subspaces, and thus the number of optimization processes. In this work for simplicity, the minimum hyper-box that encloses all of the 41 subspaces is chosen as the recommended reduced space for further optimization. For the beam problem, the space is x_1: [54.6700 80], x_2 [11.8825 50], x_3 [0.9 5] and x_4 [0.9 5.]. The volume of this space is 34.48% of the original design space.

RESULT EVALUATION We would like to see if the reduced space could still lead to the Pareto-optimal frontier, and if not, what is lost. The Pareto-optimal frontier is then identified from both the original design space and the recommended reduced space. The generated frontiers are plotted in Figure 9. As one can see from Figure 9, the two frontiers almost coincide with each other. If we apply the goals for the two functions f_1=0.02 and f_2=320 to Figure 9, we can obtain the so-called attractive Pareto-optimal frontier. That is to say, the points in the attractive Pareto-optimal frontier satisfy the goals for all the objectives. As one can see from Figure 9, the two frontiers are almost identical in the attractive Pareto-optimal frontier. Therefore, for the beam problem, we can conclude that the space reduction does not result in the loss of Pareto-optimal points.

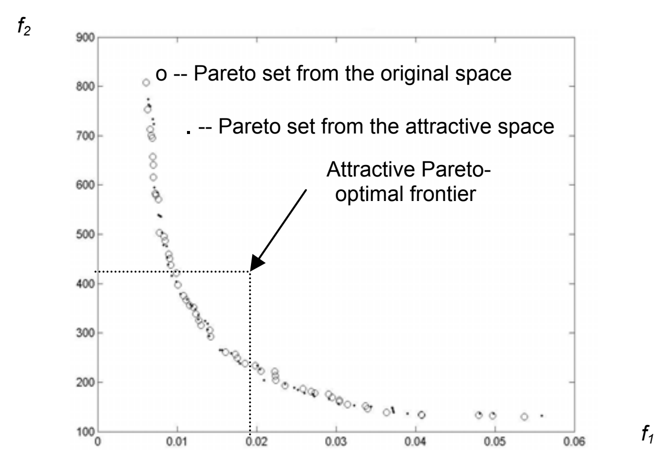


Figure 9 Comparison of Pareto-optimal frontiers generated from the original space and the reduced space.

DISCUSSION

From the test problems, one can see that the original design space for multi-objective optimization problems can be reduced with a limited number of function evaluations by using the proposed method. Moreover, the reduced space can still capture all of the Pareto points, i.e., the space reduction can be adequate without the risk of loosing the Pareto design points. It is also found that if goals are too tough, it might be hard to sample points satisfying the goals. Often more sample points are required to reach a reasonably accurate subspace. Otherwise, the probability of missing attractive spaces is high. On the other hand, if goals are too easy to satisfy, the space reduction effect is not significant. In real design, goals could potentially be appropriately determined by experienced engineers or competitors. One advantage of the proposed method is that when the goals are too high, there could be no overlaps between subspaces. The method can thus send early feedback to the decision maker about the goals before plunging into the expensive optimization process.

CONCLUSION

This work presents a new method that can help reduce the design search space for multi-objective optimization (MOO) problems and robust design optimization problems, if they are formulated as a special case of MOO. The proposed method should have following advantages: 1) if the product design goals are not achievable, this method can efficiently identify this situation without wasting time running expensive optimizations; 2) assuming all the goals are realistic, this method can help design engineers focus on a smaller design space for further optimization; 3) in the reduced space, likely all the design solutions satisfy the design goals and further optimization may not be necessary; engineers thus have many good (optimal) design solutions; and 4) this method supports simultaneous computation because it samples several points simultaneously. Though it is not demonstrated by the examples, this method should work for both continuous and discontinuous functions with either discrete or continuous variables because the method itself does not dictate the need for continuous functions or variables. The proposed method, however, bear a few limitations. First, the number of subspaces generated is large; these subspaces may be combined to reach a small number of subspaces without significantly increasing the size of the final space. Second, the sampling method, which is inherited Latin Hypercube Sampling method [16], might not be economical for high-dimensional design problems. Thirdly, the constraints are assumed inexpensive to compute. Future research may address these limitations.

ACKNOWLEDGMENTS

Research funding from Natural Science and Engineering Research Council (NSERC) of Canada is highly appreciated.

REFERENCES

1. Eschenauer, H., Koski, J. and Osyczka, A. (1990). Multicriteria Design Optimization. Springer-Verlag Berlin, Heidelberg.
2. 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.
3. Haftka, R.T., Scott, E.P. and Cruz, J.R. (1998). Optimization and Experiments: A Survey. *Applied Mechanics Review*, 51(7): pp. 435-448.
4. Arora, J.S. (1989). Introduction to Optimum Design. McGraw-Hill Higher Education, New York.
5. Koch, P.N., Simpson, T.W., Allen, J.K. and Mistree, F. (1999). Statistical Approximations for Multidisciplinary Desgin Optimization: The Problem of Size. *Journal of Aircraft*, 36(1): pp. 275-286.
6. Montgomery, D. (1991). Design and Analysis of Experiments. John Wiley and Sons, New York.
7. Hacker, K., Eddy, J. and Lewis, K. (2001). Tuning a Hybrid Optimization Algorithm by Determining the Modality of the Design Space. *ASME 2001 Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Pittsburgh, PA.
8. Gelsey, A., Schwabacher, M. and Smith, D. (1998). Using Modeling Knowledge to Guide Design Space Search. *Artificial Intelligence*, 100(1-0): pp. 1-27.
9. Sasena, M., Papalambros, P. and Goovaerts, P. (2002). Global Optimization of Problems with Disconnected Feasible Regions Via Surrogate Modeling. *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Atlanta, Georgia.
10. Zhu, L. and Kazmer, D. (2003). An Extensive Simplex Method for Mapping Global Feasibility. *Journal of Engineering Optimization*, 35(2): pp. 165-176.
11. Frits, A.P. and Mavris, D.N. (2002). A Screening Method For Customizating Designs Around Non-Convergent Region of Design Spaces. *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Atlanta, Georgia.
12. Shan, S. and Wang, G.G. (2003). Introducing rough set for design space exploration and optimization. *ASME 2003 Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Chicago, Illinois, USA.
13. Branin, F.H. and Hoo, S.K. (1972). A method for finding multiple extrema of a function of n variables. In: F. Lootsma (Editor), *Numerical methods for non-linear optimization*. Academic Press, New York, pp. 231-237.
14. Chen, W., Fu, W., Biggers, S.B. and Latour, R.A. (2000). An Affordable Approach for Robust Design of Thick Laminated Composite Structure. *Journal of Optimization and Engineering*, 1(3): pp. 305-322.

15. Gold, S. and Krishnamurty, S. (1997). Trade-offs in Robust Engineering Design. The 1997 ASME Design Engineering Technical Conference and Computers and Information in Engineering Conference, Saramento, California, USA.
16. Wang, G.G. (2003). Adaptive Response Surface Method Using Inherited Latin Hypercube Design Points. *Transactions of ASME, Journal of Mechanical Design*, 125: pp. 210-220.