

# AN EXTENSIVE SIMPLEX METHOD FOR MAPPING GLOBAL FEASIBILITY

LIANG ZHU<sup>a</sup> and DAVID KAZMER<sup>b,\*</sup>

<sup>a</sup>Department of Mechanical and Industrial Engineering, University of Massachusetts, Amherst, MA, USA; <sup>b</sup>Department of Plastics Engineering, University of Massachusetts, Lowell, MA 01854, USA

(Received 27 June 2002)

Understanding the global feasibility of engineering decision-making problems is fundamental to the synthesis of rational engineering decisions. An Extensive Simplex Method is presented to solve the global feasibility for a linear decision model relating multiple decision variables to multiple performance measures, and constrained by corresponding limits. The developed algorithm effectively traverses all extreme points in the feasible space and establishes the graph structure reflecting the active constraints and their connectivity. The algorithm demarcates basic and nonbasic variables at each extreme point, which is exploited to traverse the active constraints and merge the degenerate extreme points. Finally, a random model generator is presented with the capability to control the matrix sparseness and the model degeneracy for an arbitrary number of decision variables and performance measures. The results indicate that all these model properties are significant factors which affect the total number of extreme points, their connected graph, and the global feasibility.

*Keywords:* Decision making; Multi-attribute design; Global feasible space; Extensive Simplex Method; Extreme points

## 1 INTRODUCTION

Consider a generic engineering decision making problem with the decision variable  $x_i$  to satisfy the specifications of the performance measures  $y_j$  as follows:

$$LSL_j \leq y_j \leq USL_j \quad (1)$$

such that

$$y_j = f_j(x_1, x_2, \dots, x_n) \quad j = 1, 2, \dots, m$$

$$LCL_i \leq x_i \leq UCL_i, \quad i = 1, 2, \dots, n$$

where  $LCL_i$  and  $UCL_i$  are respectively the allowable lower and upper control limits of the decision variable  $x_i$ ,  $LSL_j$  and  $USL_j$  are the lower and upper specification limits of the performance measure  $y_j$ . Without the loss of generality, single-sided decision variables and performance measures can be formulated in Eq. (1) with  $-\infty$  or  $+\infty$  limits.

\* Corresponding author. E-mail: david\_kazmer@uml.edu

A goal of engineering design is to choose a design alternative, which not only satisfies all specifications but also provides the desired overall performance. With one explicit objective function, such a design solution can be readily acquired with common optimization methods. However, design problems generally have multiple performance measures that are coupled and competitive with each other, *e.g.* time, cost, and reliability. It is not trivial to combine these performance measures together. One approach is to define a preference to multiple performance measures and assign them different weighting factors. Due to the difficulty of unambiguous preferences on the performance measures, the conventional practice of multiplying or adding the weighted attributes rarely gives the satisfied solution in engineering design. As such, it is valuable to first explore the feasible space constituted by Eq. (1) without any preference assumptions. Once the practitioner acquires the efficient frontier for multiple attributes, the preferred decisions can be selected for compromising these attributes.

An engineering decision can be viewed as an irrevocable allocation of resources and a choice taken from a set of options [1]. Keeney and Raiffa presented their multiplicative utility function to assess the overall utility values of specific decision options [2]. Saaty's Analytic Hierarchy Process (AHP) method compares homogeneous elements to obtain dominance priorities with respect to a common criterion or attribute [3]. However, in terms of engineering design, it is our belief that the decision based design should not only consider the final evaluation of design options, but also utilize and provide a continuous stream of information for synthesis of these design options. This research has aimed to investigate the global feasibility of Eq. (1) and continuously assist decision synthesis throughout the design process.

This paper presents an algorithm to solve the global feasibility of linear models. The global feasibility of linear models is related to the research of Multiple Objective Linear Programming (MOLP) [4–6]. For instance, Eq. (1) can be formulated in a linear form with  $m_1$ ,  $m_2$ , and  $m_3$  inequality and equality constraints:

$$\begin{aligned} a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n &\leq b_i \\ a_{j1}x_1 + a_{j2}x_2 + \cdots + a_{jn}x_n &\geq b_j \\ a_{k1}x_1 + a_{k2}x_2 + \cdots + a_{kn}x_n &= b_k \end{aligned} \quad (2)$$

where  $m_1 + m_2 + m_3 = m$ . However, the feasibility mapping explores the entire topological structure of the global feasible space to assist the interactive engineering design process, while the MOLP looks for the optimal solutions for specified attributes. As such, the algorithms are designed and implemented differently.

## 2 EXTENSIVE SIMPLEX METHOD

Consider a simple example with two decision variables and three performance measures:

$$\begin{aligned} y_1 &\leq 3, y_2 \leq 6, y_3 \leq 12 \\ y_1 &= x_1 - 2x_2 \\ y_2 &= x_1 + x_2 \\ y_3 &= x_1 + 4x_2 \\ x_1 &\geq 0, x_2 \geq 0 \end{aligned} \quad (3)$$

Figure 1 shows that each constraint of Eq. (3) divides the two-dimensional Euclidean space  $E^2$  into two separate regions. The feasible design space is bounded by all active constraints.

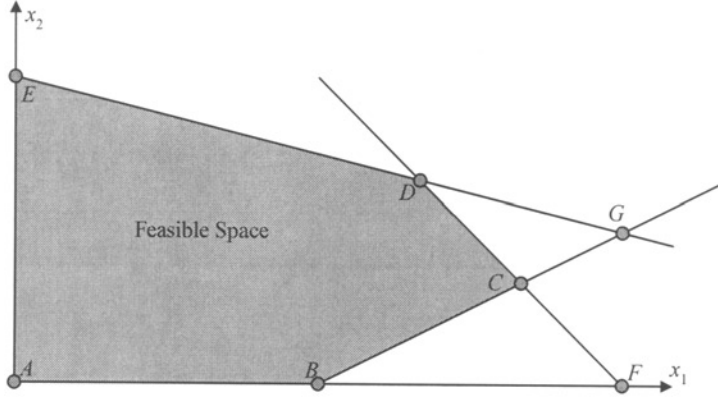


FIGURE 1 Example of feasible space.

Denote the halfspace  $H_i$  as the collection of points that satisfy the constraint,  $H_i = \{(x_1, x_2) \mid a_{i1}x_1 + a_{i2}x_2 \leq b_i\}$ . The intersection of all halfspaces composes the feasible design space  $H$ . Given any two points  $A(x_1, x_2) \in H_i$  and  $A'(x'_1, x'_2) \in H_i$ , for any  $\lambda \in [0, 1]$ ,  $\lambda A + (1 - \lambda)A' = \lambda(a_{i1}x_1 + a_{i2}x_2) + (1 - \lambda)(a_{i1}x'_1 + a_{i2}x'_2) \leq \lambda b_i + (1 - \lambda)b_i \leq b_i$ , i.e.,  $\lambda A + (1 - \lambda)A' \in H_i$ . Therefore  $H$  is convex. As shown Figure 1, the feasible space of Eq. (3) is the polygon defined by the points  $A, B, C, D$ , and  $E$ .

Similarly, each constraint in Eq. (2) can be represented by one halfspace in the  $n$ -dimensional Euclidean space  $E^n$ ,  $H_i = \{(x_1, x_2, \dots, x_n) \mid a_{i1}x_1 + a_{i2}x_2 + \dots + a_{in}x_{in} \leq b_i \text{ or } \geq b_j\}$ .  $H_i$  is the collection of points bounded by the hyperplane  $a_{i1}x_1 + a_{i2}x_2 + \dots + a_{in}x_{in} = b_i$ , the generalized notion of a straight line in  $E^2$  and a plane in  $E^3$ . The feasible space  $H$  is a convex polyhedron intersected by all halfspaces. The feasibility mapping in  $E^n$  is defined by the extreme points of the feasible space and their connectivity. Each extreme point is an intersection point of  $n$  hyperplanes. However, all intersection points are not extreme points. For instance, the intersection points  $F$  and  $G$  in Figure 1 are not inside all halfspaces and these two points are not extreme points. To map the global feasibility, one approach is to solve all intersection points and test their feasibility [7]. However, considering that the set of all extreme points is only a subset of all intersection points, an algorithm which searches only the extreme points is much more efficient.

Supposing all decision variables are nonnegative variables, Eq. (2) can be transformed into equality forms by adding slack variables  $x_{n+1}, \dots, x_{n+m}$ :

$$\begin{aligned} a_{i1}x_1 + a_{i2}x_2 + \dots + a_{in}x_n + x_{n+i} &= b_i \\ a_{j1}x_1 + a_{j2}x_2 + \dots + a_{jn}x_n - x_{n+m+1+j} &= b_j \\ a_{k1}x_1 + a_{k2}x_2 + \dots + a_{kn}x_n + x_{n+m+1+m2+k} &= b_k \end{aligned} \quad (4)$$

where,  $x_1, \dots, x_{n+m} \geq 0$ , and  $b_1, \dots, b_m \geq 0$ . The transformation can be generalized to any real variable  $x_i$  with some normalization procedures (e.g.  $x_i = x_a - x_b$ , where  $x_a, x_b \geq 0$ ).

The first extreme point can be identified in the canonical form [8]:

$$\begin{aligned} x_1 + a'_{11}x_{m+1} + a'_{12}x_{m+2} + \dots + a'_{1n}x_{m+n} &= b'_1 \\ x_2 + a'_{21}x_{m+1} + a'_{22}x_{m+2} + \dots + a'_{2n}x_{m+n} &= b'_2 \\ &\vdots \\ x_m + a'_{m1}x_{m+1} + a'_{m2}x_{m+2} + \dots + a'_{mn}x_{m+n} &= b'_m \end{aligned} \quad (5)$$

Or in the matrix format,

$$\mathbf{Ax} = \mathbf{b} \quad (6)$$

where,

$$\mathbf{A} = \begin{Bmatrix} 1 & 0 & \cdots & 0 & a'_{11} & a'_{12} & \cdots & a'_{1n} \\ 0 & 1 & \cdots & 0 & a'_{21} & a'_{22} & \cdots & a'_{2n} \\ & & \vdots & & & & \vdots & \\ 0 & 0 & \cdots & 1 & a'_{m1} & a'_{m2} & \cdots & a'_{mn} \end{Bmatrix}, \quad \mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_m \\ x_{m+1} \\ x_{m+2} \\ \vdots \\ x_{m+n} \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} b'_1 \\ b'_2 \\ \vdots \\ b'_m \end{pmatrix}$$

and,  $\mathbf{x} \geq 0$ , and  $\mathbf{b} \geq 0$ .

The variables with the unit coefficient column in the matrix format are declared as the basic variables  $\mathbf{xb}$ , and the others as the nonbasic variables  $\mathbf{xnb}$ .  $\mathbf{xb}$  and  $\mathbf{xnb}$  are two exclusive sets which combine to  $\mathbf{x}$ , *i.e.*  $\mathbf{xb} \cap \mathbf{xnb} = \emptyset$ ,  $\mathbf{xb} \cup \mathbf{xnb} = \mathbf{x}$ . One extreme point can trivially be acquired by setting  $\mathbf{xnb} = 0$  and  $\mathbf{xb} = \mathbf{b}$ . In Eq. (6),  $\mathbf{xb}$  is the vector  $\mathbf{x}(x_1, \dots, x_m)$  and  $\mathbf{xnb}$  is the vector  $\mathbf{x}(x_{m+1}, \dots, x_{m+n})$ . So  $\mathbf{x}(b'_1, b'_2, \dots, b'_m, 0, \dots, 0)$  is the first extreme point which is intersected by  $m$  constraints and yet satisfies all specifications. In fact, every extreme point can be associated with a canonical form like Eq. (6). The extreme points are different from each other by the distinctive set of basic variables and nonbasic variables that compose the solution. Considering that Eq. (6) maintains equivalence with row operations, other extreme points can be obtained with a pivot operation which swaps one nonbasic variable and one basic variable [9].

For the sake of argument, assume the nonbasic variable  $x_{m+j}$  and the basic variable  $x_i$  are to be swapped. After unifying the  $i$ th row and eliminating  $x_{m+j}$  in other rows, Eq. (5) is converted to:

$$\begin{aligned} & x_1 + \cdots - \frac{a'_{1j}}{a'_{ij}}x_i + \cdots + \left( a'_{11} - a'_{1j} \frac{a'_{i1}}{a'_{ij}} \right) x_{m+1} + \cdots + 0 + \cdots + \left( a'_{1n} - a'_{1j} \frac{a'_{in}}{a'_{ij}} \right) x_{m+n} \\ & = b'_1 - a'_{1j} \frac{b'_i}{a'_{ij}} \\ & \quad \vdots \\ & \quad \frac{1}{a'_{ij}}x_i + \cdots + \frac{a'_{i1}}{a'_{ij}}x_{m+1} + \cdots + x_{m+j} + \cdots + \frac{a'_{in}}{a'_{ij}}x_{m+n} = \frac{b'_i}{a'_{ij}} \\ & \quad \vdots \\ & - \frac{a'_{mj}}{a'_{ij}}x_i + \cdots + x_m + \left( a'_{m1} - a'_{mj} \frac{a'_{i1}}{a'_{ij}} \right) x_{m+1} + \cdots + 0 + \cdots + \left( a'_{mn} - a'_{mj} \frac{a'_{in}}{a'_{ij}} \right) x_{m+n} \\ & = b'_m - a'_{mj} \frac{b'_i}{a'_{ij}} \end{aligned} \quad (7)$$

In order to maintain all variables non-negative,

$$\begin{aligned}
 b'_1 &= b'_1 - a'_{1j} \frac{b'_i}{a'_{ij}} \geq 0 \\
 &\vdots \\
 b'_i &= \frac{b'_i}{a'_{ij}} \geq 0 \\
 &\vdots \\
 b'_m &= b'_m - a'_{mj} \frac{b'_i}{a'_{ij}} \geq 0
 \end{aligned}$$

Therefore, for any nonbasic variable  $x_{m+j}$ , the correspondent basic variable  $x_k$  of the pivot operation is decided by:

$$k = \text{Minimum} \left\{ \frac{b'_i}{a'_{ij}} : a_{ij} > 0 \right\} \quad (8)$$

Geometrically, the values  $b_i/a_{ij}$  can be viewed as the blocking distance of different constraints on the path of the nonbasic variable  $x_{m+j}$ . In order to traverse the extreme point inside the feasible space, the minimum non-negative blocking distance  $b_k/a_{kj}$  is required.

Suppose that the feasible space  $H$  is represented by a graph  $G = (V, E)$  with the vertices  $V$  and the edges  $E$ . Each vertex represents an extreme point. A pivot operation is an edge in the graph, connecting two extreme points together. Given  $k$  nonbasic variables in Eqs. (5) and (6), a vertex can have at most  $k$  adjacent vertices. If the pivot operations are appropriately set up, all extreme points can effectively be connected together to establish the feasible space.

The pseudo-code of the Extensive Simplex Method is presented in Table I. The algorithm uses the adjacency list representation. All vertices are stored in the list  $Gv$ . Each vertex  $x^e$  in  $Gv$  has an adjacent list  $x^e.Adj$ . Moreover, the data section of the vertex also includes

TABLE I Pseudo-code of the Extensive Simplex Method.

---

	ExtSimplex0 ( $x^s$ )
1	Initial $Gv$ with $x^s$
2	$x^e = x^s$
3	While $x^e \neq \emptyset$
4	For each $x_j \in x^e.xnb$
5	$k = \text{Min}\{b_i/a_{ij} : a_{ij} > 0\}$
6	$xnb(j) = x^e.xb(k)$
7	PIVOT = TRUE
8	For each $x^f \in x^e.Adj$
9	If $xnb = x^f.xnb$ then
10	PIVOT = FALSE
11	If PIVOT then
12	$x^f \leftarrow$ new extreme point pivoting around $a_{kj}$
13	$x^f.xnb \leftarrow xnb$
14	$x^f.xb \leftarrow x^e.xb$ , but $x^f.xb(k) \leftarrow x^e.xnb(j)$
15	$x^f.p \leftarrow x^e$
16	Add $x^e$ in $x^f.Adj$ ,
17	Add $x^f$ in $x^e.Adj$
18	Add $x^f$ to the end of $Gv$
19	$x^e = x^e \leftarrow$ next in $Gv$
20	End

---

TABLE II Pivot Operation.

	$x_1$	$x_2$	$x_3$	$x_4$	$x_5$	$b$	$b_i/a_{i1}$	$b_i/a_{i2}$
$y_1$	<b>I</b>	-2	1	0	0	3	<b>3</b>	-1.5
$y_2$	1	1	0	1	0	6	6	6
$y_3$	1	<b>4</b>	0	0	1	12	12	<b>3</b>
$x^A$	0	0	3	6	12			
	1	-2	1	0	0	3		
	0	3	-1	1	0	3		
	0	6	-1	0	1	9		
$x^B$	3	0	0	3	9			
	1.5	0	1	0	0.5	9		
	0.75	0	0	1	-0.25	3		
	0.25	1	0	0	0.25	3		
$x^E$	0	3	9	3	0			

the extreme point value  $x^e.x(x_1, \dots, x_{n+m})$ , the basic variable set  $x^e.xb(x_1, \dots, x_m)$ , and the nonbasic variable set  $x^e.xnb(x_{m+1}, \dots, x_{n+m})$ . The vertex list  $Gv$  is initialized with the first extreme point  $x^s$  with its basic and nonbasic variable set in the line 1. The loop of the lines 3–19 continues exploring new extreme points until the adjacency list is empty. The lines 4–18 follow each nonbasic variable  $x_j$  of  $x^e.xnb$  to solve other extreme points in the graph  $G$ . After finding the  $k$ th constraint that blocks  $x_j$ , the lines 5–6 locate the proper basic variable  $x^e.xb(k)$  (NOT  $x_k!$ ). The lines 7–10 examine whether the new nonbasic variable set has been included in the graph or not. If not, the new extreme point  $x^f$  with the solution pivoting around  $a_{ij}$  is added to the end of the vertex list  $Gv$  in the lines 11–18.

The algorithm can be demonstrated through the example of Eq. (3). Table II shows the matrix form along with three slack variables  $x_3, x_4$ , and  $x_5$ . The first extreme point is obtained at  $x^A(0, 0, 3, 6, 12)$  with  $x^A.xnb(x_1, x_2)$  and  $x^A.xb(x_3, x_4, x_5)$ . For the first nonbasic variable  $x_1$ , the minimum  $\{b_i/a_{i1}\}$  gives the first basic variable  $x_3$  to be swapped. Therefore, the new extreme point is  $x^B(3, 0, 0, 3, 9)$  with  $x^B.xnb(x_3, x_2)$  and  $x^B.xb(x_1, x_4, x_5)$ . Similarly, the minimum  $\{b_i/a_{i2}\}$  gives the third basic variable  $x_5$  to be swapped corresponding to the second nonbasic variable  $x_2$ .  $x^E$  has the value  $(0, 3, 9, 3, 0)$ ,  $xnb(x_1, x_5)$  and  $xb(x_3, x_4, x_2)$ . Figure 2 indicates the queue status during the procedures. The final result is shown in Table III with five extreme points, in which a bit sequence is used to represent the variable type at each extreme point. Specifically, this sequence is defined as  $\{x_{m+n}, \dots, x_1\}$  in which a 1 indicates a basic variable, and a 0 represents a nonbasic variable. For instance, the bit sequence for the first extreme point is 11100, which describes  $x_1$  and  $x_2$  as nonbasic variables.

The nonbasic variables can be used to identify the active constraints of the extreme point. Specifically, each slack variable represents the constraint condition of one performance measure's specification. Each primary variable represents the corresponding non-negative constraint. Therefore, the primary constraints  $x_1 \geq 0$  and  $x_2 \geq 0$  are the active constraints at the first extreme point  $x^A(11100)$ . Another example is the fifth extreme point. With  $x_4$  and  $x_5$  labeled as the nonbasic variables, the specifications  $y_2 \leq 6$  and  $y_3 \leq 12$  are active

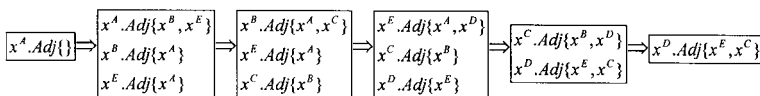


FIGURE 2 Generation of new vertices and edges in the Extensive Simplex Method.

TABLE III Output of Extreme Points.

Name	Variable set	$x_1$	$x_2$	Neighbors	
1	11100	0.00	0.00	2	3
2	11001	3.00	0.00	1	4
3	01110	0.00	3.00	1	5
4	10011	5.00	1.00	2	5
5	00111	4.00	2.00	3	4

at  $\mathbf{x}^D(00111)$ . Given the variable set information at each vertex, it is convenient to trace the hyperplane of performance specification that consists of a loop of extreme points. The pseudo code of this process is shown in Table IV. Supposing that the concerned specification corresponds to the variable  $x_k$  in the pivoting table, the extreme points are qualified if the  $k$ th variable is labeled 0 in *VarSet*. The procedure keeps adding the qualified extreme points into *outLoop* until no more extreme points remain to be processed through the loop of lines 6–10.

The identification of basic and nonbasic variables can also be used to resolve the degeneracy problem. The extreme points have so far been assumed to be the intersection points of  $n$  hyperplanes. However, it is possible to have degenerate extreme points with more than  $n$  intersecting hyperplanes. Theoretically, given a loop of degenerate extreme points, the pivot operations can even be trapped in recursive cycles that result in infinite iteration. To resolve this problem, the simplex method adopted the lexicographic rule to break the cycling in a finite number of steps. However, the lexicographic rule doesn't prevent the generation of redundant extreme points. Given  $n + i$  hyperplanes at an intersection point, there will be  $C(n + i, n)$  extreme points. Each of the extreme points has the same numerical solution and the same number ( $n$ ) of nonbasic variables, but different sets of nonbasic variables. The identification of basic and nonbasic variables provides a representation where the degenerate extreme points are merged into a single vertex with  $n + i$  nonbasic variables.

As such, the Extensive Simplex Method is modified in Table V. Note that the first canonical form and extreme point  $\mathbf{x}^s$  is resolved by the first phase method, which has been commonly adopted in Linear Programming [9]. The problem is concluded to be infeasible if the first phase method doesn't return any extreme point, which indicates no feasible space. The modified code uses a bit sequence *VarSet* to identify the variable type (1 for basic variable or 0 for nonbasic variable). Instead of matching exactly same nonbasic variable set, the code of

TABLE IV Find the Extreme Points Associated with One Performance Specification.

ExtTravLoop( $k$ )	
1	$base = 1 \ll (k - 1)$
2	For each $\mathbf{x}^e$ in $V(G)$
3	If ( $base \& \mathbf{x}^e.VarSet$ ) is 0 then
4	break out
5	$outLoop \leftarrow \{\mathbf{x}^e\}$
6	$\mathbf{x}^f = \mathbf{x}^e.Adj.first$
7	While $\mathbf{x}^f \neq \emptyset$
8	If ( $base \& \mathbf{x}^f.VarSet$ ) is 0 then
9	Add $\mathbf{x}^f$ in <i>outLoop</i>
10	$\mathbf{x}^f = \mathbf{x}^f.Adj.first$
11	$\mathbf{x}^f = \mathbf{x}^f \rightarrow next$
12	End

TABLE V Extensive Simplex Method for Degeneracy and Cycling.

---

```

ExtSimplex( )
1  $x^s \leftarrow$  the first phase method
2 Initial  $G\mathbf{v}$  with  $x^s$ 
3  $x^e = x^s$ 
4 While  $x^e \neq \emptyset$ 
5     For each  $x_j \in x^e.xnb$ 
6         PIVOT = TRUE
7          $k = \text{Min}\{b_i/a_{ij}; a_{ij} > 0\}$ 
8         If  $k = 0$  then
9             Label the unbounded edge
10            PIVOT = FALSE
11        Else
12             $newVarSet = \text{flip the } k\text{-th and the } j\text{-th bit in } x^e.VarSet$ 
13            For each  $x^f \in G\mathbf{v}$ 
14                If  $(x^f.VarSet \& newVarSet) = x^f.VarSet$  then
15                    PIVOT = FALSE
16            If PIVOT then
17                If  $b_k = 0$  then
18                     $x^e.VarSet = x^e.VarSet \& newVarSet$ 
19                Else
20                     $x^f \leftarrow$  new extreme point pivoted around  $a_{kj}$ 
21                     $x^f.VarSet = newVarSet$ 
22                    Add  $x^e$  in  $x^f.Adj$ 
23                    Add  $x^f$  in  $x^e.Adj$ 
24                    Add  $x^f$  to the end of  $G\mathbf{v}$ 
25             $x^e = x^e \leftarrow$  next in  $G\mathbf{v}$ 
26 End

```

---

lines 12–15 passes over the pivot operation when the logic “AND” result of two variable identifications equals the previous variable type, *i.e.* the new nonbasic variable set is included as part of the degenerate extreme point representation. Even if the new variable identification is not included in any existing vertex of the graph, the new extreme point may not be necessary. Recall that the pivot element  $a_{kj}$  is determined by the minimum nonnegative blocking distance of the constraints in Eq. (8). A zero blocking distance means that the pivot operation does not generate a new distinctive extreme point but rather a degenerate point. This is implemented in lines 17–18 of Table V. Without adding any new vertex, the current vertex simply resets its  $k$ th basic variable to the nonbasic variable by combining two available identification sets in the line 18. Otherwise, a normal vertex is added to the end of the vertex list  $G\mathbf{v}$ .

### 3 RANDOM MODEL GENERATOR AND RESULTS

The Extensive Simplex Method utilizes the breadth-first search to establish the graph of the feasibility mapping. The pivot operation of Table V dominates the computational time of the algorithm. Since the pivot is only required for the new extreme point, the overall computation time is asymptotically tight bounded for the number of extreme points. While the average number of extreme points for a class of distributions over convex polytopes has been shown to increase exponentially with problem dimension, this conclusion was made without the consideration of matrix sparseness, model degeneracy, and other characteristics that real engineering decision making models possess [5, 10, 11]. In order to construct problems to test the algorithm, a random model generator was developed and is provided in the

implementation of the Extensive Simplex Method. The random model is configured by specifying the following parameters:

- NMODEL* – number of random models to be generated;
- NX* – number of decision variables;
- NY* – number of performance measures;
- DENSITY* – the ratio of nonzero elements in the coefficient matrix  $\mathbf{A}$ ;
- DEGENERACY* – the ratio of zero elements in the right hand side vector  $\mathbf{b}$ ;
- $[LAL, UAL]$  – lower and upper limits of the coefficients;
- ORIGIN* – the original point of feasible space;
- SEED* – the seed of random number generator, to replicate experiments.

For example, each performance measure has a specification limit and  $NX$  coefficients corresponding to each decision variables.  $NY$  attributes combine to the coefficient matrix  $\mathbf{A}$ , as in Eq. (6). The number of nonzero elements in  $\mathbf{A}$  is specified by *DENSITY*. The positions of non-zero elements are randomly determined with the priority that each row has at least one non-zero element. The non-zero elements themselves are random values between *LAL* and *UAL*.

In order to guarantee the feasibility of the generated models, the right hand side  $\mathbf{b}$  of Eq. (2) is established such that each halfspace of the attribute specification includes the *ORIGIN* point. The distance between the *ORIGIN* point and the boundary of halfspace  $\mathbf{a}_i\mathbf{x} \leq b_i$  is:

$$d_i = \mathbf{a}_i^T \cdot \mathit{ORIGIN} - b_i \quad (9)$$

The distance could be negative, indicating that the *ORIGIN* point is outside the halfspace. To keep the distance positive and the *ORIGIN* point inside all halfspaces, let  $d_i = \mathbf{a}_i^T \alpha_i \mathbf{a}_i$  where  $\alpha_i$  is a nonnegative value between *LAL* and *UAL*. Then:

$$b_i = \mathbf{a}_i^T \cdot (\mathit{ORIGIN} + \alpha_i \mathbf{a}_i) \quad (10)$$

It is important to note that the model becomes degenerate if  $b_i$  is equal to zero in the above model. Suppose that one extreme point can be obtained with the vector  $\mathbf{b}$  as the basic variable set. Since each variable is associated with one constraint on the decision variable or performance measure, having zero elements in the basic variable set is equivalent to having additional primary constraints in the extreme point. Therefore, the degree of degeneracy can be specified by the ratio of zeroed elements in the right hand vector  $\mathbf{b}$ . The smaller *DEGENERACY* is, the less degenerate the model is. The feasible space shrinks solely to the *ORIGIN* point if *DEGENERACY* equals one.

Based on the above rules, the Extensive Simplex Method is applied to the generated random models. Figure 3 shows the distribution of extreme points with three sets of specified parameters. The probability is evaluated relative to the sample of one hundred random models. Depending on the matrix density and model degeneracy, the distribution shifts its peak value from the low to the middle number of the distribution range. This influence is analogous to the effect of the parameters  $\alpha$  and  $\beta$  on a gamma distribution. Although not shown in the figure, the number of decision variables and performance measures does not affect the skewness of the distribution but only increases the number of extreme points. In fact, a fewer number of extreme points relative to the overall distribution is expected with a more degenerate model and/or sparser coefficient matrix. As such, the average case analysis is further pursued.

The average number of extreme points for randomly generated models is shown in Table VI. The comparison of main effects for different model parameters is presented in

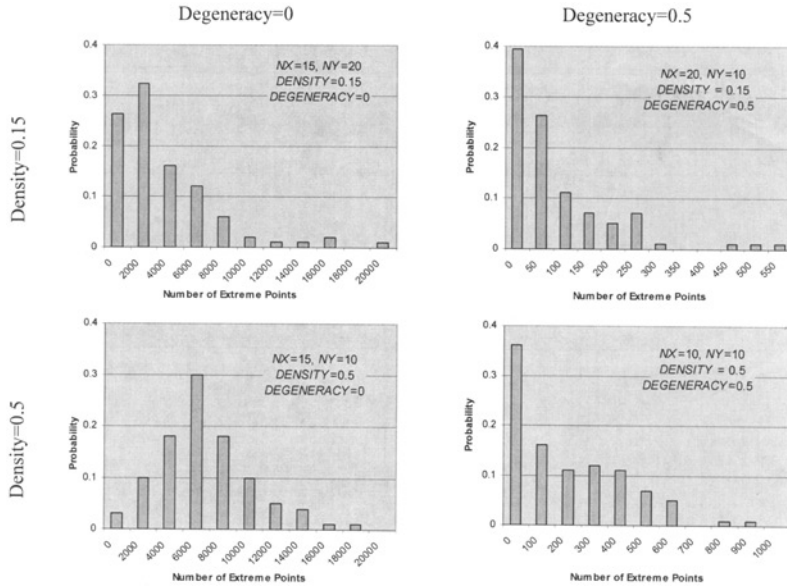


FIGURE 3 Probability distribution of extreme points.

Figure 4 as output from a statistical analysis program (Minitab). As indicated in the table and figure,  $NX$ ,  $NY$ ,  $DENSITY$ , and  $DEGENERACY$  are all significant factors with respect to the expected number of extreme points. The problem size is especially dominated by the decision variables. When  $NX$  increases from 15 to 20, there is a substantial rise in the mean number of extreme points. This result matches well with the theoretical statistical analysis in Ref. [10].

The computation time is closely related to the number of extreme points. When the feasible space includes less than 10,000 extreme points, the feasibility mapping is established very quickly. In the case of a sparse matrix and a highly degenerate model, which is common in real engineering applications, the computation is completed in a few seconds. However, the computation time expands substantially if the extreme points increase. The actual CPU time of the Extensive Simplex Method is shown in Figure 5 for a varying number of extreme points. The result is obtained on a PC with 500 MHz PIII processor and 256M RAM.

TABLE VI The Average Number of Extreme Points at Different Parameters.

	$NX$	$NY$			
		5	10	15	20
(a) $DENSITY = 0.15$ $DEGENERACY = 0.5$	5	1.7	2.4	2.3	2.6
	10	2.8	7.2	9.3	15
	15	3.9	30	92	191
	20	5.8	110	856	4050
(b) $DENSITY = 0.15$ $DEGENERACY = 1$	5	3.7	7.2	12	16
	10	10	42	178	319
	15	23	260	1664	4685
	20	50	1002	11,918	47,728
(c) $DENSITY = 0.5$ $DEGENERACY = 1$	5	14	28	35	39
	10	94	619	1316	2005
	15	444	7589	30,575	67,327
	20	1359	65,285	N/A	N/A

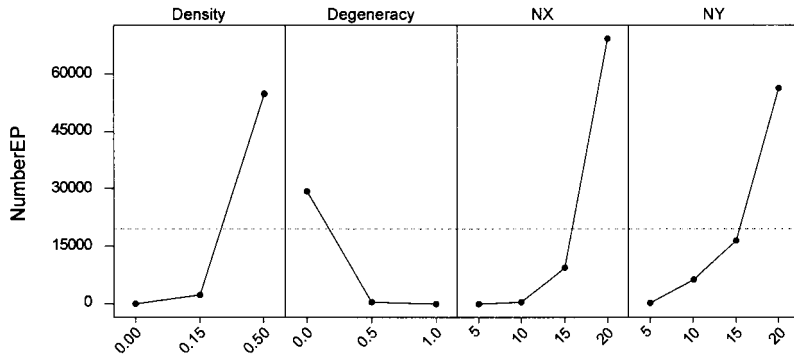


FIGURE 4 Main effects plot regarding the average number of extreme points.

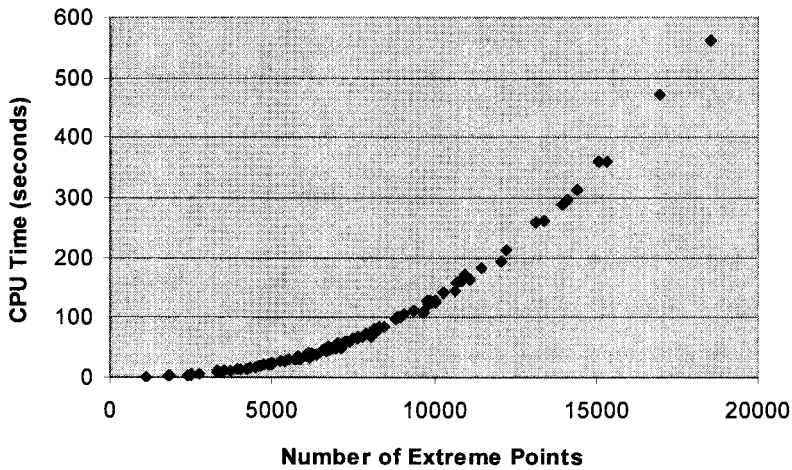


FIGURE 5 The CPU time of the Extensive Simplex Method.

#### 4 CONCLUSION

The global feasibility is important to the rational decision making of engineering design. Based on the established feasibility mapping, the designer can assess the system performance capability, evaluate the feasibility of a design alternative, negotiate the trade-off of system performance measures, and eventually converge to a desirable engineering design [7, 12, 13]. The Extensive Simplex Method was presented, which solves the feasible space and establishes the graph structure of extreme points for a linear decision model. The identification of basic and nonbasic variables at each extreme point enabled effective traversal of the active constraints and representation of the degenerate extreme points. A random model generator was designed to investigate the matrix sparseness and the model degeneracy as well as the number of the decision variables and performance measures. The statistical results reveal that all four model parameters significantly affect the number of extreme points in the feasibility mapping, which is strictly related to the computation time of the algorithm.

### ***Acknowledgement***

This research is supported by National Science Foundation, grant #9702797, through the Division of Design, Manufacture, and Industrial Innovation.

### ***References***

- [1] Hazelrigg, G. A. (1998). A framework for decision-based engineering design. *Journal of Mechanical Design*, **120**, 653–658.
- [2] Keeney, R. L. and Raiffa, H. (1993). *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*, Cambridge University Press.
- [3] Saaty, T. L. (1990). How to make a decision: The analytic hierarchy process. *European Journal of Operational Research*, **48**(1), 9–26.
- [4] Chernikova, N. V. (1965). Algorithm for finding a general formula for the non-negative solutions of a system of linear inequalities. *USSR Computational Mathematics and Mathematical Physics*, **5**(2), 228–233.
- [5] Steuer, R. E. (1994). Random problem generation and the computation of efficient extreme points in multiple objective linear programming. *Computational Optimization and Applications*, **3**, 333–347.
- [6] Steuer, R. E. (2000). *ADBASE: A Multiple Objective Linear Programming Solver for Efficient Extreme Points and Unbounded Efficient Edges*. Terry College of Business, University of Georgia, Athens, Georgia.
- [7] Zhu, L. and Kazmer, D. (2001). A performance-based representation for engineering design. *ASME Journal of Mechanical Design*, **124**(4), 486–493.
- [8] Press, W. H., et al. (1992). *Numerical Recipes in C: the Art of Scientific Computing*, 2nd ed., Cambridge University Press.
- [9] Dantzig, G. B. (1998). *Linear Programming and Extensions*, 16th ed., Princeton University Press.
- [10] Berenguer, S. E. and Smith, R. L. (1986). Expected number of extreme points of a random linear program. *Mathematical Programming*, **35**, 129–134.
- [11] Smale, S. (1983). On the average number of steps of the simplex method of linear programming. *Mathematical Programming*, **27**, 241–262.
- [12] Zhu, L. and Kazmer, D. Synthesis of engineering decisions with an evolving feasible space. Submitted to *Research in Engineering*.
- [13] Zhu, L. (2001). *A Performance-Based Representation for Engineering Design*. In Department of Mechanical and Industrial Engineering. 2001, University of Massachusetts, Amherst, p. 195.



**Taylor & Francis**  
Taylor & Francis Group

Journal .....**Eng. Opt.**

Article ID.... **GENO 031006**

**TO: CORRESPONDING AUTHOR**

**AUTHOR QUERIES - TO BE ANSWERED BY THE AUTHOR**

The following queries have arisen during the typesetting of your manuscript. Please answer the queries.

Q1	Please check setting of Table II.	

Production Editorial Department, Taylor & Francis Ltd.  
4 Park Square, Milton Park, Abingdon OX14 4RN

Telephone: +44 (0) 1235 828600  
Facsimile: +44 (0) 1235 829000