Proceedings of IDETC/CIE 2005
ASME 2005 International Design Engineering Technical Conferences &
Computers and Information in Engineering Conference
September 24-28, 2005, Long Beach, California, USA

DETC2005-84853 (Draft)

An Approach to Feasibility Assessment In Preliminary Design

Scott Ferguson
Ashwin Gurnani
Graduate Research Assistants
Department of Mechanical and Aerospace Engineering
University at Buffalo, Buffalo, NY

Joseph Donndelinger
Staff Research Engineer
Vehicle Development Research Lab
General Motors Research & Development Center
Warren, MI

Kemper Lewis
Associate Professor
Department of Mechanical and Aerospace Engineering
University at Buffalo, Buffalo, NY
Corresponding Author: kelewis@eng.buffalo.edu, (716) 645-2593 x2232

ABSTRACT

In this paper, we present the development and application of a Technical Feasibility Model (TFM) used in preliminary design to determine: whether or not a set of desired product specifications is technically feasible, and the optimality of those specifications with respect to the Pareto frontier. The TFM is developed by integrating the capabilities of a multidisciplinary design framework, a multi-objective design optimization tool, a Pareto set gap analyzer, metamodeling methods, and mathematical methods for feasibility assessment. This tool is then applied to a three objective example problem and to a five objective passenger vehicle design problem by analyzing benchmarking data from 78 late model sedans.

1.0 INTRODUCTION

The goal of this research is to develop analytical tools to support the definition of a balanced and compatible set of product specifications in the preliminary stages of the product development process. Ensuring that product specifications are mutually compatible and feasible from an engineering design perspective is not a trivial task. It is, however, a paramount task in effective execution of the product development process.

The concepts developed in this paper were designed for use with a Multi-Objective Genetic Algorithm (MOGA) in conjunction with a General Motors multidisciplinary optimization problem. One of the driving forces behind preliminary vehicle design is the amount of market share that the proposed design will capture. In this process, initial designs are approached from the customer perspective, defining an initial combination of objective values based upon product specifications. However, a vehicle designed solely from a marketing domain cannot be guaranteed feasibility in the engineering domain. Technical feasibility is extremely important, as it ensures that the proposed design can be realistically designed, developed, and manufactured. Developing a Technical Feasibility Model (TFM) bridges the gap between these domains by providing a tool to ensure feasibility in the engineering domain based upon given product specifications. Rather than just using the system constraints to ensure feasibility, the TFM also provides information about the optimality of a given design in the engineering domain. Therefore, a TFM can be used in the preliminary steps of product development processes to ensure feasibility or to highlight which specifications need to be refined, sacrificed, emphasized, or improved upon.

To construct a TFM, an initial set of Pareto optimal points for various vehicle performance characteristics is first generated. A Gap Analyzer has been developed to identify and populate any gaps that might exist within the initial Pareto frontier. Potential gaps in the frontier are addressed by tuning the MOGA to find designs within those regions. After a thorough representation of the Pareto frontier is developed, a mathematical representation of the Pareto set is then determined using metamodeling techniques. To test the feasibility of given product specifications, a feasibility assessment tool has been developed to determine if a proposed vehicle is feasible, feasible and Pareto optimal, or infeasible.

This information may then be used to identify compatible combinations of performance targets and design parameters.

Having presented the purpose of this paper, Section 2 provides the background that is used as the foundation of this work. Section 3 presents a detailed discussion of the development of the different stages needed in the construction of a Technical Feasibility Model (TFM), building upon the concepts presented in Section 2. Section 4 discusses the development of a TFM for a selected problem and presents the results of the process. The problem presented is a mathematical problem with three objectives to allow for an in-depth explanation of the process and results obtained from the TFM. The final section of this paper covers the conclusions of the results as well as recommendations for future work in this area.

2.0 BACKGROUND

This research occurs at the intersection of two technology areas: the development and application of multidisciplinary design and optimization frameworks and the generation of Pareto frontiers. An overview of the relevant background material from each of these fields follows.

2.1 MULTIDISCIPLINARY DESIGN AND OPTIMIZATION FRAMEWORKS

The arrangement of various engineering disciplines within a common framework to establish protocols and pathways for information flow is a well-established practice within large-scale engineering firms designing complex products. These frameworks are often embodied in software tools that provide for substantial automation of the design process and for the application of Multidisciplinary Design Optimization (MDO) methods. The popularity of these frameworks has given rise to a number of research programs within various sectors of the engineering design community. In the academic sector, the Center for Research in Computation and its Applications (CERCA) has developed the Virtual Airplane Design and Optimization framework (VADOR) [1]. In the government sector, Sandia National Labs have developed the Design and Analysis Kit for Optimization (DAKOTA) [2] and researchers at NASA Langley have developed the Framework for Inter-Disciplinary Optimization (FIDO) [3]. In the private sector, frameworks have been developed both as commercial products and as proprietary applications. Commercial products include the GENESIS / SDRC I-DEAS solution [4] offered by Vanderplaats R & D and the commercial version of the Federated Intelligent Product Environment (FIPER) [5] being developed by Engineous software. Proprietary corporate solutions include the MDO frameworks developed within the General Motors R&D Center [6,7].

These MDO frameworks provide powerful engineering analysis environments with substantial improvements in operational efficiency; however, they are often still very computationally expensive, especially when they include finite element analyses. One means of further improving computational speed is to employ surrogate models in place of the computationally expensive models. Surrogate models have been applied successfully within the automotive industry in robust engineering applications [8] and in MOVE, the Multifunction Optimization Visual Environment [9]. The approach in MOVE is to employ a series of discipline-specific surrogate models with a coordinated schema of design variables and responses. In contrast, the approach outlined in this paper is to exercise an MDO framework to generate a single surrogate model.

After selecting a framework, it is necessary to choose a technique that will effectively study the model to generate the needed solutions. As most models are complex and contain multiple objectives, design variables, and constraints, an approach must be selected that is robust enough to handle a wide range of problems. This topic is addressed in the next section.

2.2 PARETO FRONTIER GENERATION

Multiobjective optimization is a set of formal tools aimed at providing designers with accurate, complete, and rational information to make effective decisions. Fundamental to multiobjective optimization and fundamental to this paper is the concept of Pareto optimality. When multiple competing objectives exist, the optimum is no longer a single design point but an entire set of non-dominated design points. This is commonly known as the Pareto set [10]. The Pareto set is composed of Pareto optimal solutions. In simple terms, a Pareto optimal solution is one for which any improvement in one objective must result in the degradation of at least one other objective. Mathematically, a feasible design variable vector, $\bar{x}'$, is Pareto optimal if and only if there is no feasible design variable vector, $\bar{x}$, with the characteristics shown in Eq. (1).

$$\begin{aligned} F_i(\bar{x}) &\le F_i(\bar{x}') && \text{for all i, i = 1…}n \\ F_i(\bar{x}) &< F_i(\bar{x}') && \text{for at least one i, } 1 \le i \le n \end{aligned} \quad (1)$$

In which n is the number of objectives and the use of the less indicates an improvement in an objective. The Pareto set may be used to generate a Pareto frontier, a continuous mathematical function representing all of the possible Pareto optimal solutions.

Common methods of generating Pareto frontiers employ repeated conversion of multi-objective problems into single objective problems. However, these methods have been proven to perform poorly when attempting to populate Pareto frontiers under many circumstances. To avoid these problems, many researchers have turned to other methods for generating Pareto frontiers. Messac and Sundararaj [11] have applied Physical Programming to generate Pareto frontiers without relying on weights, instead using designer preferences in the form of metric classes in the optimization process. Narayanan and Azarm [12] have developed the Interactive Sequential Hybrid Optimization Technique (I-SHOT), solving a multi-objective problem through the repeated application of a simple genetic algorithm. A discussion of the tradeoffs in multi-objective optimization when using operators with and within genetic algorithms can be found in Azarm, Reynolds, & Narayanan [13]. Fitness and ranking schemes for use in genetic programming are developed in [14-16], while the development

of genetic programming to efficiently generate a thorough spread of points along a Pareto frontier is found in [17].

Given this background, the approach taken in this paper for solving multiobjective problems involves the use of genetic programming. A Multi-Objective Genetic Algorithm (MOGA) can be used to populate the entire Pareto frontier in a single optimization run without repeated conversion from a multi-objective to single objective problem. In a multi-objective problem, there is not a single measure of performance and a simple greater-than/less-than comparison is no longer sufficient. The fitness evaluation in a MOGA incorporates the concept of Pareto optimality. In this way, a design that exhibits dominant performance characteristics is favored above one that does not and is therefore more likely to proliferate.

In addition, MOGAs typically require far fewer function evaluations to converge to a set of solutions than other methods (e.g. grid searches, iterative weighted sums). Another advantage is that the MOGA is very robust to ill-conditioned problems (multi-modal, discontinuous, discrete, etc). Because of this, the MOGA is more likely than these other methods to yield a dense and uniformly populated Pareto frontier. Furthermore, since the MOGA is of zero order, the form of the evaluation function is irrelevant to the workings of the algorithm and the method therefore lends itself well to use with other analysis codes for which there is no analytical evaluation function. Also of importance, a MOGA can be tailored for specific problems with computational complexity and parallel computing issues in mind.

The stopping criteria for the MOGA used in this paper are based upon the maximum number of evaluations allowed. While a MOGA is quite effective at finding the frontier in complex problems, there is no guarantee that all portions of the frontier have been identified. As mentioned in Section 1, metamodeling techniques will be applied to the frontier for use within the TFM. To accurately fit a surface to the frontier requires that there be no major holes, or gaps, in the data set of non-dominated points. To identify and fill these potential holes in the performance space of the frontier, a Gap Analyzer has been created and is described in the next section.

3.0 FRONTIER GENERATION/GAP ANALYZER

While a MOGA is very efficient at populating a Pareto set, it may not always cover the entire performance space. This is most likely to occur when exercising systems in which an evaluation of a design point is either computationally or time expensive. In these cases, most of the Pareto frontier has been identified as seen in Figure 1. However, there are significant gaps in the frontier that may decrease the fidelity of the surface fit to the frontier.

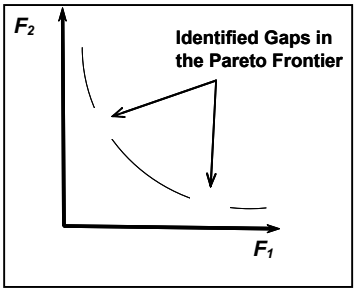

Figure 1. Initial Pareto Frontier from MOGA Results

To solve this problem, a Gap Analyzer has been developed to work with the MOGA in populating these gaps and ensuring the integrity of the surface fit. In order to complete this process, the performance space is discretized into a series of hyperboxes, or n-dimensional boxes, with step sizes specified by the designer. This allows the designer to select meaningful sizes for the gaps in the Pareto frontier that will be identified. For example, consider the objective of 0-60 mph acceleration time. A step size of one-twentieth of a second would be too fine; it would result in identification of a large number of gaps, along with a dramatic increase in either computational or run-time expense, but this increment is practically imperceptible by most customers. Likewise, a step size of two seconds would be too coarse; it would substantially reduce the number of gaps identified and the computational expense associated with filling them, but this increment is far beyond the threshold of customer perception. The selection of step size for an objective therefore involves a tradeoff between gap identification and the time and resources needed to evaluate populations of points.

3.1 GAP ANALYZER METHODOLOGY

Discretization of the performance space into hyperboxes results in populated and non-populated hyperboxes. A populated hyperbox is one that contains at least one or more Pareto optimal points within its boundaries. However, because we are not able to ascertain with certainty that the points in our frontier are actually Pareto optimal points for our system, they will be referred to henceforth as non-dominated designs.

Once the populated hyperboxes in performance space have been identified, all hyperboxes that are dominated are eliminated from further analysis. This elimination will serve to greatly reduce the amount of space that needs to be further explored by the MOGA to capture the true behavior of the frontier. A test for weak domination is determined by

$$F_{B_{\max,i}} \leq F_{C_{\max,i}} \text{ for all i = 1…n} \quad (2)$$

for at least one objective it must hold true that

$$F_{B_{\max,i}} = F_{C_{\max,i}} \quad (3)$$

and for at least one objective it must also hold true that

$$F_{B_{\max,i}} < F_{C_{\max,i}} \quad (4)$$

In which case hyperbox B weakly dominates hyperbox C. For this formulation, the *max* subscript refers to the larger of the two indices value of a hyperbox for each objective. To determine if a hyperbox B strongly dominates hyperbox C, Eq. (5) must hold true.

$$F_{B_{\max,i}} < F_{C_{\max,i}} \text{ for all i = 1…n} \qquad (5)$$

The maximum value of an objective for a given hyperbox is the upper bound, which is determined by the step size initially selected by the designer.

For this approach, all hyperboxes that do not weakly or strongly dominate a populated hyperbox are eliminated, as seen in Figure 2. The regions shaded in gray correspond to the hyperboxes that do not dominate a populated hyperbox. In doing so, a large portion of the performance space can be eliminated from further consideration, greatly increasing the efficiency with which gaps in the frontier can be identified and evaluated.

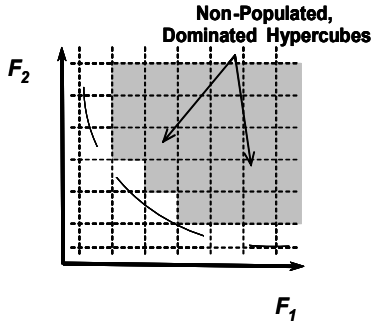

Figure 2. Elimination of Non-Populated, Dominated Hyperboxes

From the remaining set of hyperboxes in the performance space, the non-populated, non-dominated hyperboxes are separated from the non-dominated, populated hyperboxes. Next, the hyperboxes from this set that strongly dominate populated non-dominated hyperboxes are eliminated as they lie in the infeasible performance space for the system. Elimination of these hyperboxes will further reduce the number of generations of the MOGA that need to be executed and the number of evaluations required from the system. This test for dominance is carried out using Eqs. (2) – (4) and is shown graphically in Figure 3.

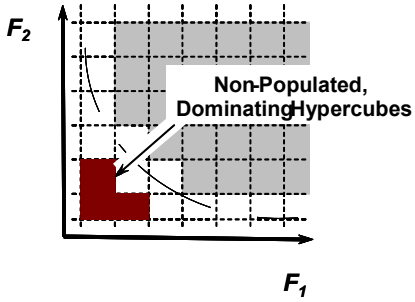

Figure 3. Identification of Non-Populated, Dominating Hyperboxes

In this figure, the areas shaded in red are those that contain hyperboxes that strongly dominate populated non-dominated boxes. Assuming that populated, non-dominated hyperboxes capture the performance bounds of the system, it is impossible to realize a feasible design that is located in the red portion of the performance space.

Remaining hyperboxes in the performance space correspond to gap locations within the frontier. To reduce the number of MOGA instances that must be created, organizing these hyperboxes into appropriate clusters is an important task. The indices of these hyperboxes are compared to identify hyperboxes that are located adjacent to each other. Hyperboxes that are found to be adjacent are combined into a single gap entity, called a gap cluster. Clustering these hyperboxes reduces the number of extra MOGA instances required, while still studying all identified regions of the performance space. Figure 4 shows the identification of gaps in the frontier from the remaining non-populated, non-dominated hyperboxes remaining.

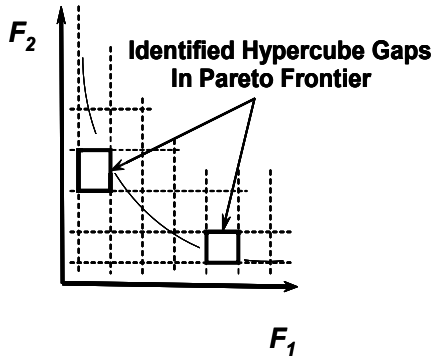

Figure 4. Identification of Gaps in the Pareto Surface

If the Gap Analyzer algorithm identifies at least one gap, one instance of the MOGA is created to further populate each gap. The initial population for each instance will contain points found in both populated, non-dominated hyperboxes surrounding the gap, as well as populated, dominated hyperboxes that are adjacent to the specific gap cluster. Placing constraints on the selectors within the MOGA will drive the process into the gaps to find designs within these regions. These new points serve to fill in the gaps in the frontier, ensuring integrity in surface fitting and feasibility assessment.

Once this process has been completed, it is possible to repeat the process for a different hyperbox step size in an effort to decrease the distance between non-dominated designs located on the frontier.

Note that the figures used in this section to illustrate the Gap Analyzer algorithm are for a 2-D problem. The implementation of the Gap Analyzer, however, is not restricted to 2-D problems. The Gap Analyzer algorithm uses concepts from multi-objective optimization that may be applied for any number of objectives. It is inherently a numerical, iteration-based method that moves through the performance space, incrementing along one objective at a time. Additionally, the computational efficiency of the algorithm is retained in higher

dimensions because the algorithm immediately increments to the next objective and resets the current objective to the initial box on encountering a dominated box when moving along that objective.

Application of the Gap Analyzer to a multiobjective problem ensures that all regions of the surface have been identified. The added computation expense of this algorithm is balanced by the enhanced representation of the surface. This is an important consideration when using metamodeling techniques, as discussed in the next section.

3.2 SURFACE FITTING AND VALIDATION

Once the full set of Pareto optimal points has been created, the next step is to develop a means of determining whether a given test point is feasible, feasible and Pareto optimal, or infeasible. This is accomplished by formulating the Pareto frontier as a continuous mathematical function representing each of the discrete points in the Pareto set. It is known that for any given point in the performance space, a test point is feasible if it lies above this Pareto frontier (assuming minimization of all objectives); it is feasible and Pareto optimal if it lies on the Pareto frontier; and it is infeasible if it lies below the Pareto frontier. The region below the Pareto frontier is said to be infeasible because of the conflicting nature of the objectives. This conflict requires a tradeoff, preventing designs from obtaining optimal values of all objectives at the same time. While these designs may not necessarily violate system constraints, it is not possible to populate this region of the performance space. It is important to note that for any multiobjective problem, the Pareto set is the best possible solution set, as mentioned earlier and no design can exist beyond the region bounded by this set.

Though a *feasible* design might satisfy system constraints, in this work, feasibility is defined with respect to the design's location in the performance space relative to the Pareto set. The utopia point is defined as the point in the performance space that is the best for all objectives. However, this point can rarely be achieved due to the conflicting nature of the objectives. Figure 5 depicts the problem statement in two dimensions.

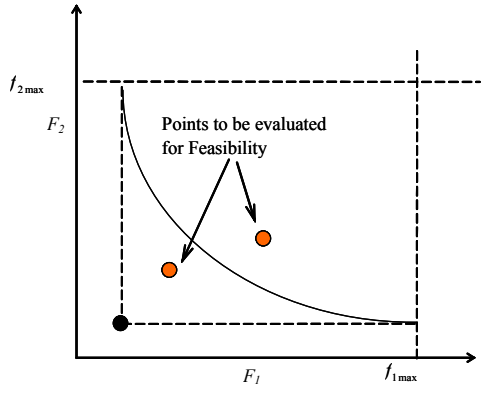

Figure 5. Identification of Potential Points for Feasibility Testing

To generate a mathematical representation of the Pareto surface, different model forms are used as basis for fitting the Pareto set to continuous functions against which feasibility of new designs can be tested. The two models investigated in this work are:

a) Unconstrained quadratic function: A second order equation with no interaction terms is used as the basis function for fitting the Pareto points. The generalized equation for this model is:

$$F_n = a_0 + \sum_{i=1}^{n-1} a_i F_i + \sum_{i=1}^{n-1} b_i F_i^2 \tag{6}$$

in which F_i represents the i^{th} objective in our multi-objective performance space. The coefficients a_o, a_i and b_i are determined using the method of least squares with no side constraints on them.

b) Constrained quadratic function: With no side constraints in place, the representation of the Pareto frontier developed using Eq. (6) may not necessarily exhibit the inverse relationships between objectives inherent in the definition of Pareto optimality. That is, without any restrictions on the coefficients, parts of the quadratic surface could have positive gradients, which contradicts a basic assumption of the behavior of Pareto frontiers (assuming minimization of all objectives). To prevent these from occurring, the coefficients in Eq. (6) are determined using the method of least squares but with additional side constraints imposed on the coefficient values. If the coefficients are constrained such that the surface always has a non-positive gradient, it will necessarily exhibit inverse relationships between objectives and the folding over of the surface will also be prevented. The generalized partial gradient for the i^{th} objective is given in Eq. (7).

$$\frac{\partial F_n}{\partial F_i} = a_i + 2b_i F_i \tag{7}$$

Eq. (8) gives the constraint on the gradient that is necessary to reflect the geometric nature of the Pareto frontier (assuming minimization of all objectives).

$$\frac{\partial F_n}{\partial F_i} = a_i + 2b_i F_i \le 0 \tag{8}$$

Further, let F_i be normalized between 0 and 1, with 0 being the most desired value and 1 being the least desired objective value (since minimization of objectives is sought). The largest (worst case) value that $\frac{\partial F_n}{\partial F_i}$ can take is $a_i + 2b_i$. Hence it is sufficient to ensure that these worst case values are less than or equal to zero.

Therefore, this model may be expressed mathematically as given in Eq. (9).

$$F_n = a_0 + \sum_{i=1}^{n-1} a_i F_i + \sum_{i=1}^{n-1} b_i F_i^2 \qquad a_i + 2b_i \le 0 \qquad i = 1,..,n \tag{9}$$

Once again, the method of least squares is used to determine the value of the coefficients. Henceforth in this paper, the vector of coefficients that are determined for the surface model will be represented as θ. Thus,

$$\theta = \{a_0, a_i, b_i\} \qquad i = 1,...,n-1$$

In the next section, the method to determine the feasibility of a proposed design, given a representation of the Pareto frontier is presented.

3.4 FEASIBILITY ASSESSMENT TOOL

The feasibility of a new design is determined based on the geometric location of the test point relative to the Pareto frontier. The steps of the method, termed as the *Ray Method* are as follows.

1. First, the equation of the line joining the Utopia point and the candidate design point is generated. The vector form of an n-dimensional equation is given in Eq. (10).

$$\bar{r} = \bar{r_0} + t\vec{\eta} \tag{10}$$

where,

$\bar{r}$ - N-dimensional variable vector

$\bar{r_0}$ - N-dimensional point (corresponding to either the Utopia point or the test point)

$\bar{\eta}$ - Slope Vector (determined using the Utopia and test point)

t - variable parameter (representing single degree of freedom)

Consider a design point P_0, represented as,

$$P_0 = (F_{10}, F_{20}, F_{30}, ..., F_{n0})$$

and the Utopia point of the performance space (assuming minimization of all objectives is desired), given as,

$$U = (F_{1\min}, F_{2\min}, F_{3\min}, ..., F_{n\min})$$

Then, the parametric form of Eq. (10) is given in Eq. (11).

$$\begin{aligned} F_1 &= F_{10} + t(F_{1\min} - F_{10}) \\ F_2 &= F_{20} + t(F_{2\min} - F_{20}) \\ F_3 &= F_{30} + t(F_{3\min} - F_{30}) \end{aligned} \tag{11}$$

.

2. Next, the point of intersection of the generated line and the Pareto frontier is determined. To do so, Eq. (11) is substituted into either Eq. (6) or (9) (based on the model used). This yields one equation with the single unknown t.

3. The equation determined in step **2** is solved for t. The determined value of t is substituted back into Eq. (11), producing a single solution that is the point of intersection of the line joining the Utopia point and the test point with the Pareto frontier (Eq. (6) or (9)).

4. Finally, the distances from the Utopia point to the intersection point and from the Utopia point to the test point are evaluated. There are three possible cases that could occur depending upon this distance.

- If the distance from the Utopia point to the point of intersection is *greater than* the distance from the Utopia point to the test point, then the test point is infeasible (below the Pareto frontier). This implies that the test point is closer to the Utopia point than the Pareto frontier and is shown graphically using a two dimensional example in Figure 6.

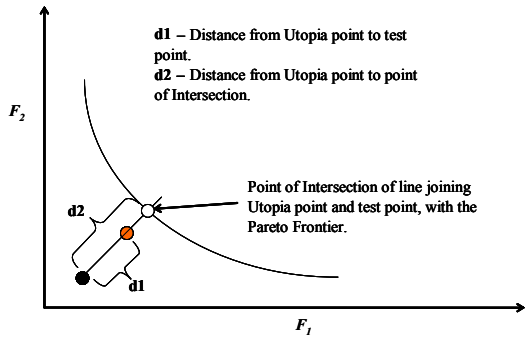

Figure 6. Infeasible Point Identification Using the Ray Method

- If the distance from the Utopia point to the point of intersection is *less than* the distance from the Utopia point to the test point, then the test point is feasible and dominated (above the Pareto frontier). This implies that the test point is indeed feasible, but technically not as superior as it could be. This is shown for a two dimensional example in Figure 7.

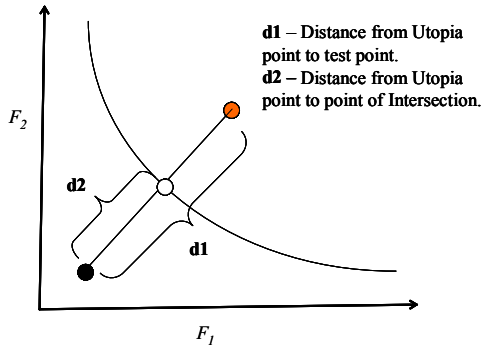

Figure 7. Feasible Point Identification Using the Ray Method

- If the distance from the Utopia point to the point of intersection is *equal to* the distance from the Utopia point to the test point, then the test point is feasible and non-dominated. This implies that the test point is at the current limit of performance and is a potentially superior design.

Thus the steps detailed above may be applied to determine whether or not a point is feasible with respect to the Pareto frontier objectives. In Section 4, this method is demonstrated through an example problem with 3 objectives and 2 design variables.

4.0 FEASIBILITY ASSESSMENT: RESULTS

In this section, we present results from the application of the feasibility assessment methods. Section 4.1 introduces the problem used for this case study. This problem is presented to help explain the results of the TFM, as the GM problem is proprietary and cannot be presented here. The data points used are found using the MOGA and are presumed to be Pareto optimal because they are all non-dominated. Section 4.2 covers the application of the Gap Analyzer to identify the possible gaps in the Pareto set. Once the gaps have been filled and the Pareto set has been sufficiently populated, surfaces are fit through these points using the methods discussed in Section 3.2. These results are discussed in Section 4.3. Finally, Section 4.4 contains a demonstration of the feasibility assessment method.

4.1 RESULTS: PROBLEM STATEMENT

The approach to determine technical feasibility is applied to a multiobjective problem adapted from [18]. This problem consists of three objective functions and two design variables, suiting both the needs of this paper in terms of complexity and in the ability to visualize the results. The problem statement is given in Eq. (12).

$$\begin{aligned} &\textit{Minimize}: \\ &\quad F_1 = x^2 + (y-1)^2 \\ &\quad F_2 = x^2 + (y+1)^2 + 1 \\ &\quad F_3 = (x-1)^2 + y^2 + 2 \\ &\textit{Subject To}: \\ &\quad -2 \le x, y \le 2 \end{aligned} \tag{12}$$

Development of the initial Pareto frontier was completed by running a MOGA for 58 generations, resulting in a total evaluation of 10000 unique designs. These evaluations resulted in a Pareto frontier composed of 6829 non-dominated designs. We chose to evaluate such a large number of designs to allow for an exhaustive, but not entirely complete, representation of the frontier.

Plotting the Pareto frontier in the performance space, as seen in Figure 8, the overall shape can be studied. Examining the location of the Pareto points, they appear to be dispersed over the range of the frontier, and not present only in large clusters. Obviously, the size of the discretization selected for each objective will have a dramatic effect on the number of potential gaps identified in the frontier. For this problem, a discretization size of 0.2 was selected for all objectives. This value was chosen to allow for the identification of gaps that would be of a significant size to warrant further investigation. Also, it is possible to choose a different discretization size for each objective, and selecting the same discretization size essentially creates hypercubes in the performance space instead of hyperboxes.

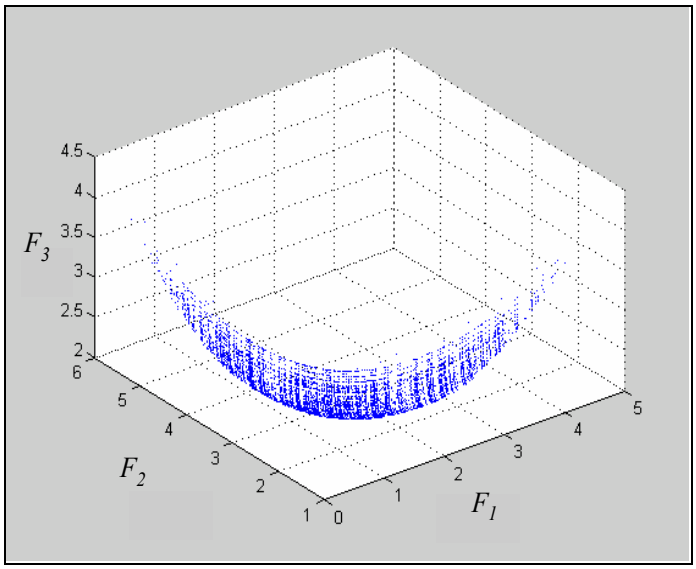

Figure 8. Pareto Frontier from MOGA Results

Table 1 lists the minimum and maximum values on the Pareto frontier for each objective. Using this information along with the discretization size, the performance space is portioned into 4831 unique hyperboxes. Each hyperbox contains an index for its location in each objective, with the value 1 corresponding to the hyperbox index closest to the minimum of that objective. Having completed this, the number of filled hyperboxes must be identified. As stated in Section 3.1, a filled hyperbox is one that has at least one Pareto frontier design located within its boundaries. There were 250 hyperboxes identified in the performance space that contained points from the Pareto frontier. Figure 9 is a plot of the Pareto frontier in the performance space. The axes relate to the index of the hyperbox, and the points correspond to the centroid of each hyperbox. In this frontier, there are currently 250 points composing the frontier. However, there still are 4581 hyperboxes remaining in the performance space for the gap analyzer analysis.

Objective	Minimum Value	Maximum Value	Discretization Size
F1	0.0041	4.0243	0.2
F2	1.0007	5.1950	0.2
F3	2.0001	4.0794	0.2

Table 1. Maximum and Minimum Values on the Pareto Frontier for Each Objective

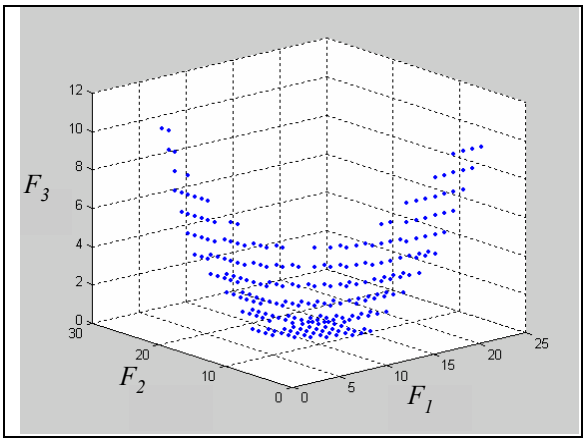

Figure 9. Hyperbox Index Representation of the Pareto Frontier

In the next section, the Gap Analyzer methodology introduced in Section 3.2 is used to study the gaps in this example problem.

4.2 RESULTS: GAP ANALYSIS

Execution of the Gap Analyzer on the Pareto surface began with the identification of hyperboxes that were non-populated and dominated by Pareto points. Of the 4581 hyperboxes to be accounted for in the performance space, it was found that 3237 of the hyperboxes were strongly dominated by Pareto points. Next, the hyperboxes that strongly dominated populated hyperboxes are removed, as they cannot contain points that lie on the Pareto surface. There were 370 hyperboxes identified in the performance space that strongly dominated hyperboxes containing Pareto frontier points. Completing this step, there were 994 remaining hyperboxes in the performance space that could potentially correspond to gaps in the Pareto frontier.

Next, the list containing the indices for the 994 remaining hyperboxes were compared to the indices of the populated hyperboxes of the Pareto frontier. Hyperboxes that could identify as a gap in the frontier were determined to be those that were adjacent to the populated hyperboxes. Completing this analysis left 230 unique hyperboxes in the performance space. These 230 hyperboxes correspond to gaps in the Pareto frontier. With the identification process complete, it was necessary to determine possible gap clusters to be studied by additional instances of the MOGA. The breakdown of the original 4831 into their separate categories is shown in Table 2.

As expected, the majority of hyperboxes in the performance space belong to those in the category of being strongly dominated by filled hyperboxes. The number of hyperboxes needing investigation as potential gaps – in this problem 230 – relate to only 4.6% of the entire performance space, greatly narrowing the region of space that must be investigated.

Hyperbox Type	Number of Hyperboxes	Percentage of Performance Space
Filled (Pareto)	250	5%
Strongly Dominated by Filled Hyperboxes	3237	67%
Strongly Dominating Filled Hyperboxes	370	7.6%
Weakly Dominated Hyperboxes Not Adjacent to Filled Hyperboxes	764	15.8%
Remaining Hyperboxes Adjacent to Filled Hyperboxes (Gaps)	230	4.6%
Total	**4831**	**100%**

Table 2. Analysis of Hyperbox Type and Percentage Composition in the Performance Space

In order to efficiently apply new instances of the MOGA to investigate the potential gaps in the frontier, the hyperboxes must be clustered. Clustering of the gaps was completed by combining hyperboxes that were adjacent to each other in the performance space. For this work, adjacency is defined as having an index number a maximum of one increment away in any dimension. Also implemented in this procedure was the constraint that a cluster was stopped after it began to contain more than ten percent of the total number of hyperboxes identified as potential gaps. This is done in order to constrain the region in the performance space where new instances of the MOGA would be created. Given the definition of adjacency, it is hypothetically possible to combine all hyperboxes identified as gaps into a single cluster. To avoid this from happening, the constraint is set on the number of hyperboxes that can form a gap. Using this constraint, nineteen different clusters were identified in the performance space. The number of hyperboxes found in each cluster is shown in Figure 10.

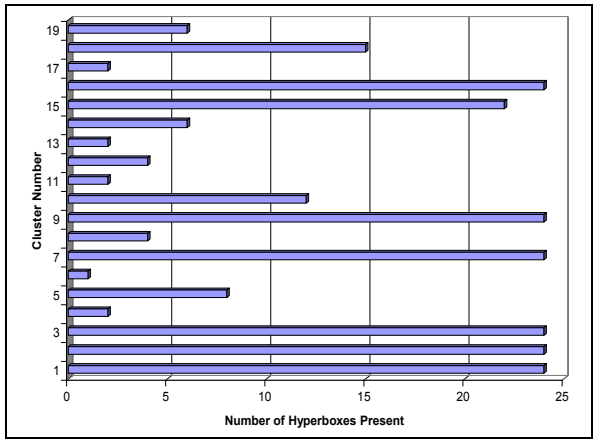

Figure 10. Distribution of Hyperboxes Per Gap Cluster

After putting the potential gap hyperboxes into the 19 different clusters, instances of the MOGA were created to attempt to place designs within the clusters. As this problem is a minimization problem, the upper index of each cluster was used as a constraint for each instance of the MOGA. This ensures

that the MOGA will attempt to populate only the region of interest as defined by each cluster. As an example, two clusters are used in the following discussion. Clusters 1 and 4 have been selected and the upper bound of the indices for each cluster is shown in Table 3.

Cluster	Index for F1	Index for F2	Index for F3
1	2	19	11
4	3	11	5

Table 3. Upper Bound Indices for Selected Clusters

For each cluster studied, another instance of the MOGA was created. The original problem statement was modified by adding constraints to make designs only within the bounds identified by each gap cluster feasible. Each instance of the MOGA was allowed to run for 1000 evaluations. This number was arbitrarily selected, as the purpose of this exercise was not computational efficiency, but instead determining if a selected gap could indeed be filled. Gap 1 consisted of 24 clustered hyperboxes in the performance space, and after evaluating the feasible designs found from the separate MOGA run, 2 of those hyperboxes were filled with designs. Gap 4 was composed of only 2 hyperboxes, and the separate instance of the MOGA was unable to find any designs that would fill those hyperboxes. This data is summarized in Table 4.

Cluster	Hyperboxes in Cluster	Filled Hyperboxes
1	24	2
4	2	0

Table 4. Results of Gap Study

Having filled hyperboxes while studying Gap 1, those hyperboxes need to be made a part of the Pareto frontier. Modifying the graph seen in Figure 9, the two black circles in the performance space shown in Figure 11 represent the location of the two hyperboxes filled when studying Gap 1.

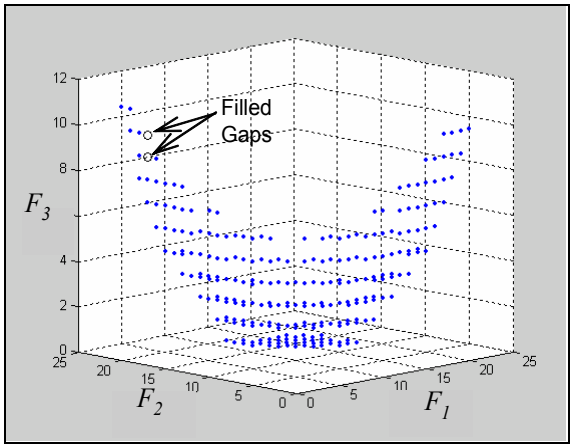

Figure 11. Filled Hyperboxes from Gap 1 Study

Studying each of the gaps in this manner will provide a complete representation of the Pareto frontier. This representation of the frontier can be used for surface fitting, which is discussed in the next section.

4.3 RESULTS: CURVE FITTING

Using the Pareto set generated by the MOGA, the two methods presented in Section 3.3 are applied to determine mathematical expressions of the Pareto surface. For both cases, a least-squares regression was applied to the data to develop the coefficients of the polynomial surface in Matlab. For the constrained surface, the Optimization Toolbox within Matlab was also used to determine the coefficients while handling the constraints placed upon the model. The R^2 values for the two different models are given in Table 5.

Pareto Surface Model	R^2
Unconstrained Quadratic (Eq. (6))	0.891
Constrained Quadratic (Eq. (9))	0.376

Table 5. R^2 values for Pareto Surface Models

As seen in Table 5, the Unconstrained Quadratic model exhibits substantially less fitting error than the Constrained Quadratic model. This result is as expected due to the constraints imposed on the allowable values of the coefficients. All Pareto sets are required to have non-positive gradients since the objective functions forming the Pareto frontier are assumed to be minimized and are competing against each other. Since the Pareto set has a non positive gradient as seen in Figure 8 and the constrained model provides a low fit, only the unconstrained model is used for this case study. Additionally, for problems with Pareto sets known to having non-positive gradients, using a constrained model would be overkill. For example, in the application of the method for the General Motors problem, it was not known if the gradients were non-positive and hence the constrained surface fit was used. This might be the case for all problems encountered in industry since these problems are larger than the one used here.

The coefficients of the unconstrained surface of Eq. (6) are given in Eq. (12).

$$\begin{aligned} a_0 &= 1.47922 & b_1 &= 2.36493 \\ a_1 &= -2.87069 & b_2 &= 2.52139 \\ a_2 &= -2.96181 \end{aligned} \qquad (12)$$

With the given mathematical function of the Pareto surface, the feasibility study is performed next for potential new designs in Section 4.4

4.4 RESULTS: FEASIBILITY TESTING

Once the expressions for the Pareto surfaces are generated, they may be used to evaluate new sets of product specifications for feasibility. Typically, these sets would be desired combinations of specifications. In this paper the process of evaluating feasibility is illustrated using performance values determined from the nature of the performance space. Since the entire performance space is known, data points from the regions that are known to be feasible and infeasible are generated and used

in the code developed to determine feasibility. Note that when generating this data, only the objective function values are provided with no design variable information. Determining the corresponding design variable values once a set of performance measures are determined to be feasible is an area of ongoing work and the methods developed for this are shown in [19]. Using this data set serves two purposes: not only does it illustrate the feasibility assessment process it also provides insight into the validity of the model. The results of applying the Ray Method to test for feasibility for the unconstrained function representation of the Pareto frontier for the case study problem are shown in Table 6.

Number	F_1	F_2	F_3	Feasibility Results
1	4.0	5.0	4.0	Feasible
2	3.0	3.0	3.0	Feasible
3	0.0	0.0	0.0	Out of Bounds
4	1.1059	2.1044	2.5518	Feasible
5	1	2	2.2	Infeasible

Table 6. Feasibility Results for Test Points

Looking closely at the results shown in Table 6, test points 1 and 2 lie above the Pareto set of points in the performance space, thus entitling them *feasible*. The 3rd test point is *out of bounds* since the performance measure values are better than the Utopia point. Test point 4 is said to be *feasible*, though based on the distance measures, the point lies precisely on the Pareto set. The code calls this point feasible as opposed to Pareto optimal due to numerical error. However, since the feasibility assessment tool is used to only determine if a point is feasible from the engineering domain, calling test point 4 feasible is sufficient. Finally, test point 5 is said to be infeasible since it lies below the Pareto set of points. The performance values shown in Table 6 are plotted along with the Pareto set of points and are shown in Figure 12.

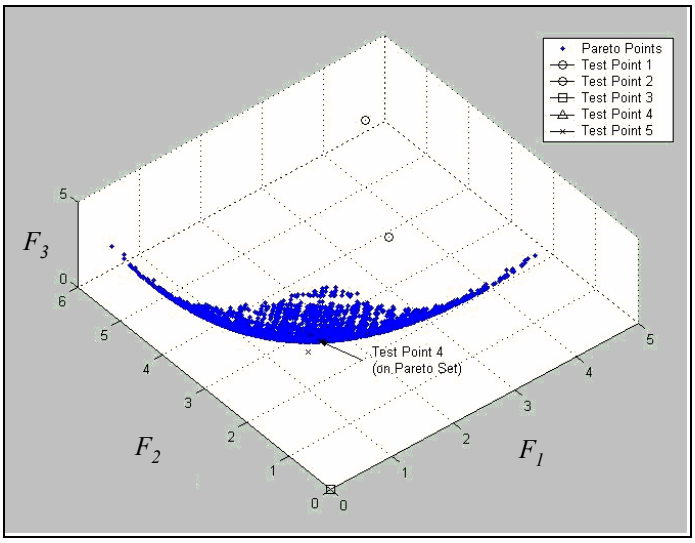

Figure 12. Feasibility Test Points in Performance Space

Thus we see from Figure 12 that the feasibility assessment tool behaves exactly as it is expected to for the 3 objective case study problem.

In three dimensions, the application of the feasibility test is trivial. Dealing with a three dimensional performance space allows for easy visualization and the feasibility of a test point can clearly be seen. However, when dealing with problems of greater than three objectives, using visualization techniques no longer becomes a simple task. Application of the Ray Method allows for quick and efficient testing of numerous test designs with very little computational expense. To demonstrate this for the General Motors problem, feasibility is tested using specifications from 78 late-model sedans. This problem consisted of 5 objective functions, 10 design variables, and 3 constraints. Because of the large number of dimensions in this problem, the surface that was created was not only difficult to understand, but almost impossible to visually use when determining the feasibility of a given test point. Also, in practical application of the feasibility model it is important to quantify how optimal a given test point was with respect to the Pareto surface. It is not enough to simply know whether the test point was feasible; vehicle development engineers need to understand to what extent a given test point challenges the state of the art in vehicle development technology. This information could only be determined via the use of a Pareto frontier.

Of the vehicles failing the feasibility test, 17 of them failed because they has a better value in at least one objective than the best value found in the Pareto set. In each case, this was because the vehicles employed technologies that were outside the scope of the multidisciplinary analysis system used to generate the Pareto surface. This is not only an expected result, but an encouraging one because it demonstrates the consistency of the framework being used: vehicles employing technologies within the scope of the underlying analysis system pass the feasibility test, as expected, while those employing technologies outside the bounds of the underlying analysis system fail the feasibility test, also as expected.

Thus, in this section, the results of the Feasibility assessment are presented and these results not only illustrate but validate the developed methodology of this paper. In the next section, some concluding remarks and sources of future work are presented.

5.0 CONCLUSIONS AND FUTURE WORK

In this work, we have successfully developed an approach to determine whether or not a given set of performance specifications, actual or hypothetical, is feasible based on a multi-objective analytical model. This method can be used to effectively map the performance limits for a set of available technologies and to integrate gap analysis, metamodeling, and feasibility assessment algorithms to assess the feasibility of a vector of performance attributes. Application of the approach using a simple 3 objective case study is presented along with a feasibility test of 5 hypothetical points from different locations of the performance space. This approach was also applied to a five objective vehicle design problem where both the feasibility and optimality of 78 vehicles were tested. These methods and

tools provide engineers with critical information that allows them to define product specifications while maintaining high confidence for successful realization of the product's design.

Future work in this area includes expanding capabilities of the analytical model to include additional attributes as well as the additional degrees of design freedom required to support them. This expansion may require the application of parallel processing techniques and the development of a more sophisticated MOGA to reduce the time necessary for sampling of the design space. As the scope of the performance space expands, the Gap Analyzer will play an increasingly important role in maintaining a high level of efficiency when populating the Pareto set by reducing the overall number of evaluation and generations required by the MOGA. Additional information about the relationships between performance space and design space in the neighborhood of a given test point could also provide additional benefit when considering tradeoffs in this multi-objective preliminary design process.

ACKNOWLEDGMENTS

We would like to thank the General Motors Corporation and the National Science Foundation Grant DMII-9875706 for their support of this research.

REFERENCES

[1] Alzubbi, A., Ndiaye, A., Mahdavi, B., Guibault, F., Ozell, B., and Trepanier, J-Y, 2000, "On the use of JAVA and RMI in the development of a computer framework for MDO", *8th AIAA Symposium on Multidisciplinary Analysis and Optimization,* Long Beach, CA, AIAA-2000-4903.

[2] Giunta, A., Eldred, M., 2000, "Implementation Of A Trust Region Model Management Strategy In The Dakota Optimization Toolkit", *8th AIAA Symposium on Multidisciplinary Analysis and Optimization,* Long Beach, CA, AIAA-2000-4935.

[3] Townsend, J.C., J. A. Samareh, R. P. Weston and W. E. Zorumski, 1998, "Integration Of A CAD System Into An MDO Framework", *NASA Langley Research Center,* technical report NASA/TM-1998-207672.

[4] Kosaka, I., Charpentier, C., and Watson, B., 2000, "An Interface Between SDRC I-DEAS And The Genesis Structural Analysis And Optimization Code", *8th AIAA Symposium on Multidisciplinary Analysis and Optimization,* Long Beach, CA, AIAA-2000-4933.

[5] Röhl, P.J., R. M. Kolonay, R. K. Irani, M. Sobolewski, K. Kao, and M. W. Bailey 2000, "A Federated Intelligent Product Environment", *8th AIAA Symposium on Multidisciplinary Analysis and Optimization,* Long Beach, CA, AIAA-2000-4902.

[6] Bennett, J. A., Botkin, M. E., Koromilas, C., Lust, R. V., Neal, M. O., Wang, J. T., and Zwiers, R. I., 1995, "A Multidisciplinary Framework for Preliminary Vehicle Analysis and Design," in "Multidisciplinary Design Optimization: State of the Art," *Proceedings of the ICASE/NASA Langley Workshop on Multidisciplinary Design Optimization,* SIAM, Hampton, VA, pp. 3-21.

[7] Fenyes, P. A., Donndelinger, J. A., and Bourassa, J.-F., 2002, "A New System For Multidisciplinary Design and Optimization of Vehicle Architectures", *9th AIAA Symposium on Multidisciplinary Analysis and Optimization,* Atlanta, GA, AIAA-2002-5509.

[8] Lee, J., Li, D.J., Liu, X., Soderburg, N., Sudjianto, A., Vora, M., and Wang, S., 2001, "An Approach to Robust Design Employing Computer Experiments", *ASME Design Engineering Technical Conferences, Design Automation Conference*, Pittsburgh, PA, DETC01/DAC-21094.

[9] Longacre, K., Vance, J. M., and DeVries, R., 1996, "A Computer Tool to Facilitate Cross-Attribute Optimization", *6th AIAA Symposium on Multidisciplinary Analysis and Optimization,* Bellevue, WA, AIAA-96-4132-CP.

[10] Pareto, V., 1906, Manuale di Econòmica Polìttica, Società Editrice Libràia, Milan, Italy; translated into English by A. S. Schwier, as Manual of Political Economy, Macmillan, New York, 1971.

[11] Messac, A. and Sundararaj, J. G., 2000, "Physical Programming's Ability to Generate a Well-Distributed Set of Pareto Points," *41st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, Paper No. AIAA 2000-1666.

[12] Narayanan, S., and Azarm, S., 1999, "A Multi-objective Interactive Sequential Hybrid Optimization Technique for Design Decision Making", *Engineering Optimization*, Vol. 32, pp. 485-500.

[13] Azarm S., Reynolds, B. J., and Narayanan, S., 1999, "Comparison of Two Multi-objective Optimization Techniques with and within Genetic Algorithms," *ASME Design Engineering Technical Conferences,* DETC99/DAC-8584.

[14] Balling, R. J., 2000, "Pareto Sets in Decision-Based Design," *Engineering Valuation and Cost Analysis*, Vol. 3, pp. 189-198.Sds

[15] Goldberg, D.E., 1989, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, MA.Sds

[16] Fonseca, C. M. and Fleming, P. J., 1998, "Multi-objective Optimization and Multiple Constraint Handling with Evolutionary Algorithms – Part 1: A Unified Formulation," *IEEE Transactions on Systems, Management, and Cybernetics – Part A: Systems and Humans*, Vol. 28, pp. 26-37.

[17] Lewis, K. and Eddy, J. 2001, "Effective Generation of Pareto Sets Using Genetic Programming", *ASME Design Engineering Technical Conferences, Design Automation Conference*, Pittsburgh, PA, DETC01/DAC-21094.

[18] Viennet, R., Fontiex, C., and Marc, I., 1996, "Multiobjective Optimization Using a Genetic Algorithm for Determining a Pareto Set", *Journal of Systems Science*, Vol. 27(2), pp. 255-260.

[19] Lewis, K., Donndelinger, J., Ferguson, S., and Gurnani, A., 2005, "On the Issues of Convergence and Performance to Design Mapping in Multiobjective Optimization Problems", *ASME Design Engineering Technical Conferences, Computers in Engineering Conference*, Long Beach, CA, DETC2005-CIE (Submitted for review).