An Approach to Facilitate Decision Tradeoffs in Pareto Solution Sets

Edward M. Kasprzak¹
Department of Mechanical and Aerospace Engineering
University at Buffalo
Buffalo, New York 14260 USA

Kemper E. Lewis²
Department of Mechanical and Aerospace Engineering
University at Buffalo
Buffalo, New York 14260 USA

ABSTRACT

An approach to constructing Pareto sets and negotiating the decision tradeoffs within the Pareto set is presented. Points in the Pareto sets are generated by sampling the design space, and then a polynomial curve is developed to fit the points in order to approximate the Pareto set. A new method, called the scaling method, is used to automatically determine the appropriate weights for objectives of the problem based upon a selected design from the Pareto set. The Pareto set is then mapped to the design space to help designers make effective tradeoffs and decisions concerning the Pareto solution set. A vehicle dynamics design problem which has been developed with an industrial partner is used to demonstrate the usefulness of the approach.

1 INTRODUCTION

The principal tenet of Decision-Based Design is that a designer's primary role is to make decisions. Many of these decisions are associated with tradeoffs among competing performance, cost, and/or quality objectives. As one objective is sacrificed another is usually rewarded. When designing complex, multidisciplinary systems, a designer needs decision support tools to help perform simulation and optimization, and to help explore these design tradeoffs. Performing these tradeoffs between objectives is usually accomplished using a weighted sum objective function. However, this is where the difficulties can begin. Determination of the corresponding weights of this overall objective function is, many times, an art in itself. More importantly, the
weights may change over time as preferences shift, or may change when the same system is being used in multiple environments.

For instance, consider the design of a race car. The car must perform well on turns of various different radii which will be taken at various different speeds. While the car can be "fine-tuned" for a particular size corner, the basic configuration and conceptual layout must remain the same as most race tracks contain corners with several different radii. This mimics a multiobjective design problem, where the radius of the corner is a given objective. That is, for two different radii corners there would be two objective functions: one to minimize the time to navigate the smaller radius and one to minimize the time on the larger radius. The optimal car configuration is unlikely to be the same for each corner due to differences in speed, aerodynamic forces and tire performance. Race teams want their car to perform well on all corners to give the best aggregate lap time. Therefore, a compromise decision must be made between the optimal designs for each radius corner.

Various approaches to multiobjective design have been developed. In Goal Programming [1] one objective is chosen and the remaining are made constraints. However, which objective to select and what threshold levels to set for the remaining constraints is a difficult task. Physical Programming avoids setting weights for an overall objective by allowing a designer to express preference with respect to each given design metric by using the classifications: highly-desirable, desirable, tolerable, undesirable, highly-undesirable, and unacceptable [2]. For every design metric, the designer prescribes numerical values to define each of these ranges. This method, in the same spirit as this paper, avoids the tedious task of assigning weights and is much more sophisticated than selecting weights arbitrarily, but requires knowledge of the numerical values of each preference range.

Another approach to handle multiple objectives that avoids having to use objective weights is the concept of Pareto optimality, where a set of Pareto-optimal solutions that are generated. The designer can then choose from this set of solutions according to the relative satisfaction and preference of each objective. However, there are many difficulties in Pareto set optimization as
well. Generating the Pareto set is a difficult task, especially in nonconvex problems [3]. Then, once the Pareto set is generated there are a number of additional challenges for a designer. These include 1) determining the best solution, or determining what preference weightings generate a chosen solution, and 2) mapping a chosen Pareto solution back into the design space in order to determine the corresponding values of the design variables.

In this paper, we present an approach to generate the Pareto set, and more importantly, once the set has been generated to predict the optimal combination of objective weights and corresponding design variables of a preferred solution. In the next section, we present the concepts underlying this approach and then we present a race car dynamics design problem and illustrate the usefulness of the approach. We then provide some commentary on the methods presented and outline directions for further development and implementation.

2 GENERATION AND EXPLORATION OF THE PARETO SET

In this section, we present the approach to generating the Pareto set, exploring the Pareto set to determine the best solution(s), and mapping the Pareto set back into the design space in order to facilitate effective tradeoff decision making.

2.1 Generation of the Pareto Set

Consider the multiobjective optimization problem formulated as

\[
\begin{align*}
\text{minimize} & \quad \mathbf{F}(\mathbf{x}) \\
\text{subject to} & \quad \mathbf{x} \in \mathbf{X} \subseteq \mathbb{R}^n,
\end{align*}
\]

(1)

where \( \mathbf{F}(\mathbf{x}) = [f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})] \) and \( f_i(\mathbf{x}), i = 1, \ldots, m \), are real-valued continuous functions defined in \( \mathbb{R}^n \). Let \( \mathbf{X} \) denote the design space that is formed by both the design constraints and the range of design variables \( \mathbf{x} \), and \( \mathbf{Y} = \mathbf{F}(\mathbf{X}) \subseteq \mathbb{R}^m \) be the objective space where there are \( m \) objectives. If the objective functions remain in conflict with each other over the design space, then it is impossible
to find a point at which they would assume their minimum values simultaneously, and consequently, the classical concept of a common optimal solution does not apply. In this situation, the concept of Pareto solutions is exercised. A point $x^0$ is called a Pareto solution of the problem if there is no other feasible point $x$ such that $f_i(x) \leq f_i(x^0)$, $i=1, ..., m$, with strict inequality for at least one index $i$. If the objective functions are in conflict, then in general, there may be infinitely many Pareto solutions. Minimizing each objective function individually over the design space we obtain

$$f_i^{\text{min}} = \text{minimum} \{f_i(x), x \in X\}, \ i=1, ..., m,$$

which yields a utopia (ideal) point $u \in \mathbb{R}^m$ defined as

$$u_i = f_i^{\text{min}}, \ i=1, ..., m. \quad (2)$$

Due to the conflict between the objective functions, the utopia point is never in $Y$.

The first challenge to solving a multiobjective problem and obtaining valuable solutions is to populate the Pareto set. A common practice for finding Pareto solutions has been the weighted sum method that performs a minimization of the linear combination of the objective functions. The corresponding weighted-sum problem is

$$\text{minimize} \quad \sum_{i=1}^{m} w_i f_i(x)$$

subject to \quad $x \in X \subset \mathbb{R}^n$, \quad (3)

where $w_i \geq 0$, $i=1, ..., m$, and $\sum_{i=1}^{m} w_i f_i = 1$. Scalars $w_i$ are referred to as the weights assigned to the objective $f_i$, $i=1, ..., m$, and determine the importance of each objective. However, not every Pareto solution can be found by solving the weighted sum problem. That is, there may not exist a weight $w$ such that a given Pareto point may be found by solving the weighted sum problem. Koski [4] explores this disadvantage in optimal structural design. More recently, Das [5]
performed numerical experiments and additionally concluded that for convex multiobjective optimization problems, an even spread of weights \( w \) does not produce an even spread of points in the efficient set. Zeleny [6], Yu and Leitmann [7], and others developed Compromise Programming (CP), an approach based on a procedure that finds an efficient point closest to the utopia point and avoids the pitfalls of the weighted approach in nonconvex problems. Chen has utilized CP to perform multiobjective robust design in order to model robustness tradeoffs [3].

As an alternative to the methods mentioned above, we generate a sampling of the Pareto set without using a weighted objective function. We simply take a grid of the design space and evaluate the values of the individual objective functions at each point. These points are plotted in the performance space. This is illustrated conceptually in Figure 1 where the values of the objectives at each design point are plotted in the performance space. An important point to make here is equal objective weights are used implicitly in plotting Figure 1. If we had plotted \( f_2 \) vs. \((2 \times f_1)\), instead of \( f_2 \) vs. \( f_1 \), the figure would look quite different. From the plot of the performance space the Pareto set is determined, thereby avoiding the pitfalls of using a weighted sum to generate the Pareto set. We then fit a polynomial equation through the Pareto points, approximating the entire Pareto set. This method, given a fine enough discretization, is guaranteed to find the convex and nonconvex portions of the Pareto Set.
The use of a polynomial to describe the Pareto set facilitates a tradeoff analysis of the Pareto set which would be more difficult otherwise. Specifically, the objective weights needed to cause any member of the Pareto set to become the optimal solution can be predicted. In the next section, we describe the exploration of the tradeoffs in the Pareto set.

2.2 Exploration of the Pareto Set

Once the Pareto set is found, the next challenge is to determine the 'best' solution of the set. There are four general classes of methods for determining the 'best' solution in a Pareto set: no-preference, a posteriori, a priori, and interactive methods [8]. No-preference approaches do not include the preferences of the decision maker. A posteriori methods involve the decision maker after the analysis (when a designer chooses from among a set of possible solutions), while a priori methods involve the designer before the analysis (by determining the weights or preferences before the analysis). Interactive approaches involve the decision maker during the process to generate preferences progressively. In this work, an interactive approach is taken to determine the best solution. However, a definition of optimality from a no-preference approach is used to aid decision makers in determining the appropriate weights given a chosen solution.

In no-preference approaches, the best solution is defined by geometric relationships only [9]. A common approach is to use the L2 norm, where the best solution is the one that minimizes:

\[
\left( \sum_{i=1}^{m} (f_i(x) - f_i^{\min})^2 \right)^{1/2}
\]

This norm is adapted here, but is used in an interactive approach to determine the best Pareto solution. The use of this norm is crucial to the method described here for scaling the performance space. Given a Pareto set, the L2 norm predicts the best solution for a particular set of relative objective weights. These relative weights are implicit in the Pareto plot and are determined by the axes of the plot. For instance, in Figure 1, the implicit weights are each 1, as the axes are \((1 \times f_2)\) and \((1 \times f_1)\). As the relative objective weights change, so does the shape of
the Pareto set and the best solution predicted by the L_2 norm. In the method presented here, the polynomial describing the Pareto set is determined from a grid search of the design space and by initially constructing the Pareto set on a plot of \((1 \times f_2)\) vs. \((1 \times f_1)\). However, this equal importance is not likely to be the preference scheme of interest to a designer and, when used with the L_2 norm, will not predict the most desirable solution. Therefore, a method has been developed to scale the performance space so that the most desirable solution corresponds to the ‘best’ solution as defined by the L_2 norm definition.

Consider Figure 2 where an original Pareto set, identified by a grid search, is shown in 2a). In the end, a designer may identify solution A as the preferred solution because of the preferences and satisfaction of the objectives. However, Figure 2a) shows solution B to be, according to the L_2 norm, the ‘best’ solution. Therefore, in order to make A the ‘best’ solution the space must be ‘stretched’ or ‘compressed’, as shown in Figure 2b) in accordance with the designers specific objective weights. Now the preferred solution, A, is also the best solution. This scaling of the design space, in effect, gives a designer the proper relationship between the preferences of the objectives. In other words, it gives a designer the ratio of the weights of the objectives that produce the solution at point A.

![Diagram showing rescaling of Pareto set](image-url)

**Figure 2. The Rescaling of the Pareto Set**
The scaling method used to move from Figure 2a) to 2b) is described in detail in [10]. It is based on a considerable theoretical foundation. The core tenet of the theory says that an interior point, P, on the Pareto set cannot be the optimal point unless the slope of the Pareto set at P is perpendicular to a line connecting the utopia to point P. A mathematical relationship is derived which predicts an amount n which the x-axis \((f_1)\) would need to be rescaled for point P to become optimal. This value, n, represents a change in the weighting of the objectives from the baseline weighting scheme which produced the original Pareto set. Figure 2b shows a value of \(n > 1\), meaning the x-axis \((n \times f_1)\) has increased in importance relative to the y-axis \((f_2)\). Note that as \(n \rightarrow \infty\), the x-axis will be stretched to the right and the 'best' solution, as defined by the \(L_2\) norm, will tend to move towards \(f_{1\text{min}}\).

The scaling method is useful in two decision-making scenarios. The first, allows a designer to automatically determine the best Pareto solution given a set of relative objective preferences. This involves calculating the weighting ratio, n, over the entire domain of the Pareto set. The second use, as illustrated above, allows a designer to predict what weighting of the objectives will result in a particular member of the Pareto set becoming the optimal solution. We explore each of these uses in Section 3 using our case study.

2.3 Mapping of the Pareto Set to the Design Space

One final challenge remains. Once the desired solution is found in the Pareto set, the value of the design variables that produce this solution must be found. Therefore, the performance space must be mapped to the design space, as shown in Figure 3. This mapping is more difficult than mapping from the design space to the performance space (Figure 1), as it may be a one-to-many mapping. In other words, for a given performance, there may be multiple design variable combinations that produce the desired performance. Therefore, this mapping is non-trivial.
One method which can be used to map from the performance space to the design space involves finding the intersection of constant objective contours [11]. This method is referred to as the contour method in this work. For instance, suppose a designer wanted to determine the design variables which produce point A in Figure 2. In the performance space, point A has a value of $f_1$ and $f_2$ on the x- and y-axes, respectively. These values represent the level of performance for those objectives. For illustrative purposes, suppose there are two design variables: $X_1$ and $X_2$. Figure 4 shows how these performance levels appear as contours in the design space. The point where these contours contact is the design which produced the performance at point A in Figure 2.
By repeating this process for a sequence of points along the Pareto set a trace of the Pareto set in the design space can be constructed. Conveniently, contours of points in the Pareto Set never intersect. They approach and touch at only one point, giving a one-to-one mapping from the performance space to the design space. Therefore, the contour method is a useful technique in determining the relationship between the set of Pareto solutions in the performance space and the design space.

In the next section, we demonstrate this approach on the design of a race vehicle that has been developed with an industrial partner, Milliken Research Associates (a consultant to multiple automotive companies and auto racing teams).

3 AN EXAMPLE: THE DESIGN OF A RACE CAR

The example presented is from the field of vehicle dynamics. A race car is represented by a computer model and then run in a computer simulated "race". The objective is to tune the vehicle by way of the available design variables for optimal performance, or minimal elapsed time in the race. Fractions of a second are all that distinguish race cars many times. Therefore, tuning the vehicle to optimize its performance is extremely important.

3.1 Vehicle Model

The vehicle model used is a model developed by Milliken Research Associates and simulates the design of a simplified race care. The key components of the model are weights, aerodynamic drag and downforce, roll stiffness, and tire dynamics. The tire data was made available by Milliken Research Associates, Inc. This tire data was gathered on a tire testing machine and gives tire lateral force as a non-linear function of the load carried by the tire and the tire's slip angle for both the race car front and rear tires.

Tire load is calculated from the vehicle's weight, normalized longitudinal center of gravity (CG) location, aerodynamic drag, aerodynamic downforce (from wings), height of the center of
gravity, vehicle lateral acceleration, and normalized roll stiffness distribution [12]. Of these, the longitudinal CG location and normalized roll stiffness distribution are the design variables available to tune the vehicle. Normalized Longitudinal CG location, \( a' \), is the location of the CG expressed as a fraction of the wheelbase aft of the front axle. Normalized roll stiffness distribution, \( K' \), is expressed as a fraction of the roll stiffness provided by the front axle relative to the total roll stiffness. Figure 5 depicts the design variables in this vehicle model.

\[
K' = \frac{K_F}{K_F + K_R}
\]

\[
a' = \frac{a}{\ell}
\]

**Figure 5. Configuration of the Race Car**

For a given set of design variables the fastest possible vehicle speed for a given track radius is iteratively determined through an optimization routine using the Golden Section Method and Successive Quadratic Approximation Method [13] within the constraints imposed by steady-state cornering. Once the vehicle speed on each radius is known, calculation of the lap time is elementary.

**3.2 Race Definition**

The race is conducted on two separate circles which represent the two objectives of the problem: one with a 100' radius and one with a 400' radius. Because the radius is constant the vehicle corners at steady state conditions for a constant forward velocity. With the vehicle at
maximum steady-state speed on the small circle the time required for the race car to complete \( x \) laps is taken. The value \( x \) is the number of laps the race will contain on the 100' circle. The vehicle is next raced on the 400' radius circle and the time to complete \( y \) laps at maximum steady-state speed is taken. The overall time for the race is then calculated from:

\[
\text{Race Time} = x \times (\text{Time for 1 lap on 100' radius circle}) + y \times (\text{Time for 1 lap on 400' radius circle})
\]  

(5)

where \( x \) and \( y \) represent the number of laps on each radius and therefore, the importance or weighting of each objective.

This race is the sum of performance at two different steady-state conditions. The choice of optimal design variables for each circle differ, so a compromise solution is needed.

### 3.3 Pareto Set Generation

A grid of 625 evenly distributed points was selected from the design space to determine the performance for a particular race and vehicle configuration. This race was defined as one lap on each radius--that is, \((x, y) = (1, 1)\) in Eq. 5. The results were plotted in terms of the lap times on each radius, referred to as the performance space, as shown in Figure 6. Each point represents a single \((a', K')\) pair and has as its coordinates the resulting lap times. The generation of each point included an iterative optimization scheme to determine the fastest possible speed given a vehicle configuration and track radius. Each point required between 5 and 8 seconds on a 200Mhz Pentium PC. Thus, the 625 points needed to determine each lap time on both radii required approximately 2.5 hours. Figure 7 shows the Pareto set derived from Figure 6 along with an eighth order polynomial fit using least-squares criteria (axes are the square roots of the lap times for better visualization).
With the utopia point defined as (0,0), the optimal performance in this particular race is found at (lap time on 100' radius, lap time on 400' radius) = (8.931, 9.101) seconds, giving a race time of 18.032 seconds. This is the optimal performance for one lap on each radius, (x, y) = (1, 1). As the choice of race definition (number of laps on each radius) changes, the optimal
performance may change. A designer can explore the Pareto set in two ways at this point. In the next section, we illustrate both decision-making approaches.

### 3.4 Pareto Set Scaling

The first way a designer could exercise the Pareto set is to define a race and determine what point of the Pareto set is optimal for this race definition. A designer could decide that a race of \( x \) laps on the 100' radius and \( y \) laps on the 400' radius is more realistic, where \( x \neq y \). The scaling factor is \( n = \frac{x}{y} \). The \( x \)-axis, \( f_1 \), would then effectively be 'stretched' or 'compressed' by \( n \) as shown in Figure 2, and the best solution from the resulting Pareto set is found using the \( L_2 \) norm definition.

For the entire domain of the race car Pareto set, Figure 8 shows the predicted scaling factor, \( n \), required for members of the Pareto set to become optimal as a function of the 100' radius lap time. For instance, a designer may decide that a race of 15 laps on the 100' radius and 1 lap on the 400' radius is more realistic instead of 1 lap on each. In this case, \( n = \frac{15}{1} = 15 \). This race definition would also apply to lap ratios of \( (x, y) = (30,2), (45,3), \) etc. With \( n = 15 \) in Figure 8, the square root of the 100' radius lap time equals 2.9823 seconds. Referring back to Figure 7, the square root of the lap time on the 400' radius at this location on the Pareto set is 3.0652 seconds. The optimal performance for this race definition is \((15 \text{ laps } \times \text{ lap time on 100' radius}, 1 \text{ lap } \times \text{ lap time on 400' radius}) = (15 \times 9.395, 8.894) \) seconds, giving a race time of 149.826 seconds.

The second way a designer could exercise the Pareto set is to select a Pareto solution that is attractive from a performance standpoint and determine the scaling factor (objective importances) for which this solution is optimal. For instance, a designer may prefer a Pareto solution from Figure 7 where \( f_1 = 8.9 \) seconds and \( f_2 = 9.33 \). From Figure 8, the optimal scaling for this design is approximately \( n = 8 \) (at \( \sqrt{8.9} = 2.983 \)), which means that this solution is optimal for a race of 8 laps on the 100' radius and 1 lap on the 400' radius (or equivalent multiples).

A designer can also assess how sensitive the solution in the performance space is to changes in objective weighting. In Figure 8, as \( n \) decreases from 40 to 10, the resulting performance does
not change much (square root of 100' radius time increases from 2.9811 to 2.9828, or 29% of its range). However, when \( n \) varies between 10 and 5, the performance may change dramatically (square root of 100' radius time varies from 2.9828 to 2.9885, or 75% of range). This gives a designer valuable information concerning the sensitivity of the design to changes in race definitions.

![Figure 8. Theoretical Optimal Scaling](image)

For different race conditions, the optimal performance has been determined. Unfortunately, the design variables \((a', K')\) which produce this performance remain unknown. Finding the values of the design variables corresponding to a chosen solution becomes even more important in the cases where multiple solutions may be considered optimal. In Figure 8, for values of \( n \) between 5 and 8, there may be more than one optimal solution. Therefore, in order to make an effective decision, it would be advantageous if a designer knew the values of the design variables corresponding to each solution. In the next section, the contour method is applied to determine the corresponding design variables of a chosen Pareto solution.
3.5 Mapping of the Pareto Set Using the Contour Method

The contour method is now applied to determine the design variables which produce the optimal performance for the two race definitions listed in the previous section. In the performance space, a particular member of the Pareto set has been identified as the optimal performance for a given race definition by 1) defining a race, and finding the best solution for that race, or 2) directly choosing a solution and determining for what race the solution is best. The corresponding design variables of this solution are now sought. More generally, the shadow of the entire Pareto set in the design space is sought.

Two of the race definitions examined in the previous section are: \( n = 1 \) and \( n = 15 \). For the \( n = 1 \) example, the optimal performance is determined to be \((8.931, 9.101)\) seconds on the 100' and 400' radii, respectively. Figure 9 plots these performance levels as contours in the design space. Because the 400' radius performance is nearly identical to the optimal performance on this circle, the contour is very small. Had it been identically the optimal performance the contour would have collapsed to a point in the design space. The performance level on the 100' radius is sub-optimal so a large contour exists. The two contours touch at the design point \((a', K') = (0.614, 0.375)\). This is the optimal design for the \( n = 1 \) race definition \((x = y = 1)\).

![Figure 9. Optimal Lap Contours: n = 1](image-url)
Similarly, Figure 10 presents the contours for the $n = 15$ race definition. In the previous section, the optimal performance is determined to be $(15 \times 8.894, 9.395)$ seconds on the 100' and 400' radii, respectively. Since neither of these performance levels are optimum for either circle, each contour is easily visible. The contours touch at the design point $(a', K') = (0.628, 0.615)$--the optimal design for the $n = 15$ race definition. Note that, compared with Figure 9, the 100' contour is smaller and the 400' contour is much larger. This shows a shift in the design toward the optimum on the 100' circle and away from the 400' circle optimum design.

![Figure 10. Optimal Lap Contours: $n = 15$](image)

Through repeated application of this method over a range of race definitions ($n$ values) a series of points in the design space can be amassed. Figure 11 shows an approximate plot of the Pareto set in the design space. While the endpoints of this line are the optimum designs on the 100' and 400' radii, respectively, the trace of the line between these points is very complex.
Figure 11. Estimated Pareto Set Plotted in the Design Space

By using the Contour Method, the trace of the Pareto set in the design space can be determined. Whether used over a range of n or for a single case of interest, the contour method is a reasonable approach to mapping Pareto sets from the performance space to the design space.

4 COMMENTS ON THE METHODS PRESENTED

When used together, the methods presented in this paper provide an effective tool for designers to examine multi-variable, multi-objective optimization problems. The discretization of the design space to find the Pareto set is an approach which avoids the pitfalls of other methods. However, the importance of having a sufficiently fine grid cannot go unstated. As the density of the grid in the design space increases, the quality of the resulting Pareto set estimate in the performance space increases. The 625 grid points used in this study are able to generate a Pareto set with various inflection points and require 2.5 hours. However, generating these points can be automated. Once they are generated, finding preferred solutions by exploring and negotiating within the Pareto set becomes straightforward and very effective. For more analytically complex problems, alternative methods to generate the Pareto set, discussed in
Section 2.1 can be combined with the methods presented here, namely, the scaling method and contour method.

The polynomial fit to the Pareto set needs to be of sufficiently high order to accurately represent the shape of the Pareto set. An eighth order polynomial was fit in the above example. One reason for the high order is to model the concave and convex portions of the Pareto set. The need for such a high order polynomial gives further importance to the discretization of the design space. The Pareto set needs to be representatively populated to fit the polynomial.

The scaling method makes use of the polynomial representation of the Pareto set to provide information about the relative weighting required to cause a member of the Pareto set to become the optimal solution. This technique is presented in greater detail in [10], but has only been developed for two-objective problems. The underlying mathematics of the scaling method appear to be extensible to three or more dimensions, although this has yet to be undertaken.

Of the methods presented, the contour method is most dependent upon the skill of the person using it to obtain good results. Being a graphical method, it is not automated and requires the user to manually determine the point where the contours touch. Again, a fine discretization results in better contour plots which, in turn, result in improved determination of the design variables. Extension of this method to three or more design variables poses a particular challenge: the contours become n-dimensional surfaces and the point where the surfaces intersect in three- or more-dimensional space would need to be determined. This is a difficult problem which needs further attention to make the contour method viable for more than two design variables. The key appears to be to represent the contours mathematically, not graphically, to determine their common point. That said, the contour method is an effective method to return to the design space from the performance space in a two design variable problem.
5 SUMMARY

In this paper, we present a set of methods to help designers make tradeoff decision in multiobjective design problems. Given a set of objectives, we generate a polynomial approximation to the Pareto set from a grid search of the design space. The optimal ratio or weighting of the objective functions for a preferred design is predicted using the scaling method. Finally, we map the Pareto set from the performance space into the design space by means of the contour method to provide further tradeoff information for the designers. These methods alleviate the need to choose weights in a weighted sum objective function, allow a designer to determine the desired performance and preferred design, and allow for mapping of the desired performance to the corresponding design variables.

ACKNOWLEDGMENTS

We would like to thank Douglas Milliken of Milliken Research Associates for providing the tire data for the race car example and for providing valuable feedback and support during the progress of this work. Computer simulations and costs were provided by Milliken Research Associates, and by the University at Buffalo.

REFERENCES


