

Improving Engineering Design with Fuzzy Sets

Erik K. Antonsson*

Engineering Design Research Laboratory
Division of Engineering and Applied Science
California Institute of Technology

and

Kevin N. Otto[†]

Department of Mechanical Engineering
Massachusetts Institute of Technology

January 10, 1995

Abstract

The Method of Imprecision (M_I) is a formal method, utilizing the mathematics of fuzzy sets, for incorporating the natural level of imprecision that occurs throughout the engineering design process. This paper presents the details of the Level Interval Algorithm (LIA) used internally by the M_I, and its extensions to permit application to engineering design problems in industry where monotonicity cannot be guaranteed, only discrete values may be available for some variables (and hence continuity must be relaxed), and engineering analyses are expensive and must be minimized.

Computation problems that reach beyond the scope of the LIA, such as singularities, *etc.*, are also examined, showing that the LIA behaves no worse than conventional calculations in the presence of these difficulties.

1 Introduction

Imprecision is an integral part of the engineering design process, not imprecision in thought or logic, but rather the intrinsic vagueness of a preliminary, incomplete description. Obtaining precise information upon which to base decisions is usually impossible, yet it is critical to make early engineering design decisions on a sound basis.

At the stage where concepts are being generated, the description of a design is nearly completely vague or imprecise (fuzzy). This imprecision is reduced during the design process until ultimately the final description is precise (crisp), except for tolerances, which represent the allowable limits on stochastic manufacturing and material variations.

The need for a methodology to represent and manipulate imprecision is greatest in the early, preliminary phases of engineering design, where the designer is most unsure of the final dimensions and shape, materials and properties, and performance of the completed design.

* Associate Professor, Mail Code: 104-44, Caltech, Pasadena, CA 91125

[†] Assistant Professor

Additionally, the most important decisions, those with the greatest effect on overall cost, are made in these early stages [12, 38, 41, 43].

Our work focuses on the development of methods for representing and manipulating imprecise descriptions of designs to permit the designer to compare alternatives during the preliminary design phase. Because design imprecision concerns the choice of design variable values used to describe an artifact or process, we use the designer's preference to quantify the imprecision with which design variables are known. Preference, as used here, denotes either subjective or objective information that may be quantified and included in the evaluation of design alternatives. We call our approach the *Method of Imprecision* (M_I) [2, 20, 23, 24, 25, 26, 27, 36, 45, 46, 47, 48]. Other researchers have also recently contributed to this area: Diaz [6, 7], Hamburg [11, 13], Knosala and Pedrycz [18], Müller and Thäringen [22], Posthoff [29], Rao [30, 31, 32, 33, 34], Sakawa and Kato [35, 37], Thurston and Carnahan [39, 40], and Zimmermann and Sebastian [50, 51, 52]. The following sections present a brief review of how imprecision is used to facilitate decision-making in engineering design using the M_I, followed by a description of the calculation methods (and a few difficulties).

2 The Method of Imprecision

The M_I begins with one or more design alternatives, at the concept level. The designer's preferences are then applied to each of the variables that (imprecisely) describe the design. Commonly, performance specifications will also be imprecisely described and can be elicited from customers through market surveys, *etc.* The imprecise design variables are then induced onto the performance space, by use of the algorithm described below. The design preferences induced onto the performance variables are compared to the specifications. At that point, decisions regarding the feasibility of each alternative concept can be made, and promising ranges of design variables can be identified.

Definitions and Notation. *Design variables* are denoted d_i , and the valid design variable values within the *design variable space* (DVS) form a subset \mathcal{X} . The set of valid values for d_i is denoted \mathcal{X}_i . The preference that a designer has for values of d_i , the i th design variable, is represented by a preference function on \mathcal{X}_i , called the *design preference*: $\mu_d(d_i)$.

Performance variables are denoted p_j . For each performance variable p_j there must be a mapping f_j such that $p_j = f_j(\vec{d})$. The mappings f_j can be any calculation or procedure to measure the performance of a design, including closed-form equations (*e.g.*, for stress, weight, speed, cost, *etc.*), iterative solutions, heuristic methods, "black box" calculations, testing of prototypes, or consumer evaluations. The subset of valid performance variable values \mathcal{Y} is mapped from \mathcal{X} and the set of valid values for p_j is denoted \mathcal{Y}_j . The *performance variable space* (PVS) is the dependent set of performances evaluated for each design in the DVS. In order to compare design alternatives, design preferences are mapped onto the PVS via the extension principle [49], discussed below.

Specifications and requirements also embody design imprecision, even though most are written as if they were crisp, *e.g.*, "This device must have a range of at least 250 km." Such a requirement implies that given two designs arbitrarily close together, one with a range of 250 km and one just below, the first would be acceptable but not the second, as shown by the dashed line in Figure 1. Specifications and requirements in the real world are commonly fuzzy. Often the designer must ask questions to distinguish the underlying fuzzy constraint so that the final design will satisfy the customer's actual requirements even though it may violate the crisp constraint initially given. The fuzziness of constraints and the fuzziness of preliminary design

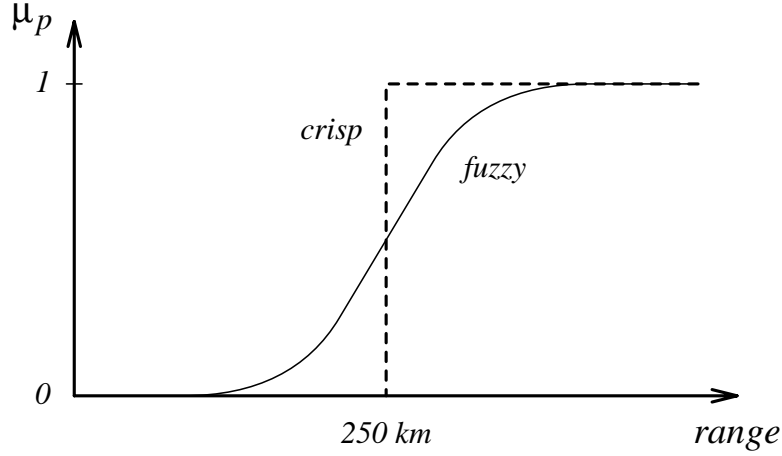


Figure 1: Example imprecise functional requirement.

variables are both forms of design imprecision and can be represented in exactly the same way. The customer's preference (requirements) for values of p_j , the j th performance variable, is represented by a preference function called the *functional requirement*: $\mu_p(p_j)$. The solid line in Figure 1 shows a fuzzy functional requirement.

Quantifying Imprecision. Utility and risk-aversion are quantified in utility theory via the lottery method [17]. Unfortunately no such formal method exists for eliciting preference [4, 5, 9, 14, 15]. However, limits of acceptability for variable values, whether communicated formally or established informally by experience, are familiar to engineers in industry [42]. Such acceptable limits correspond to intervals over which preference is greater than zero. This suggests that rather than determine the preference μ_d at each value of d , as shown in Figure 1, it may be more natural to determine the intervals in d , called α -cuts, over which μ_d equals or exceeds certain preference values α . The use of intervals encourages the passing of set-based design information between engineering groups early in the design process [42], and permits the early release of possible sets of design data from one engineering group to the next in advance of precise design information. This approach has many advantages over the traditional “point-by-point” design iteration. The M₀J can extend set-based concurrent design by providing preference information over the possible range of design data.

Imprecision Calculations. After specifying design preferences μ_{d_i} on \mathcal{X}_i and functional requirements μ_{p_j} on \mathcal{Y}_j (and an aggregation function [36, 23]), the next step is to determine the induced values of μ_{d_i} on \mathcal{Y} (design preferences mapped onto the performances), given by Zadeh's *extension principle* [49]:

$$\mu_d(\vec{p}) = \sup_{\vec{d}: \vec{p} = \vec{f}(\vec{d})} [\mu_d(\vec{d})] \quad (1)$$

A simple one-dimensional example of Zadeh's extension principle is shown in Figure 2. The performance p achieved for each value of the design variable d is given by the function f , which is a curve in this simple example. The corresponding $\mu_d(d)$ can be mapped onto p , producing $\mu_d(p)$, the design preference mapped onto the performance space (as illustrated by the dashed lines in Figure 2). For higher dimension design problems, each p will be a function of many d 's, and each function f will be a hyper-surface.

An algorithm to compute Zadeh's extension principle (and thus to calculate $\mu_d(\vec{p})$) is the *Level Interval Algorithm* (LIA), first proposed by W. M. Dong and F. S. Wong [8] as the “Fuzzy Weighted Average” algorithm and also called the “Vertex Method”. Note that in the simple

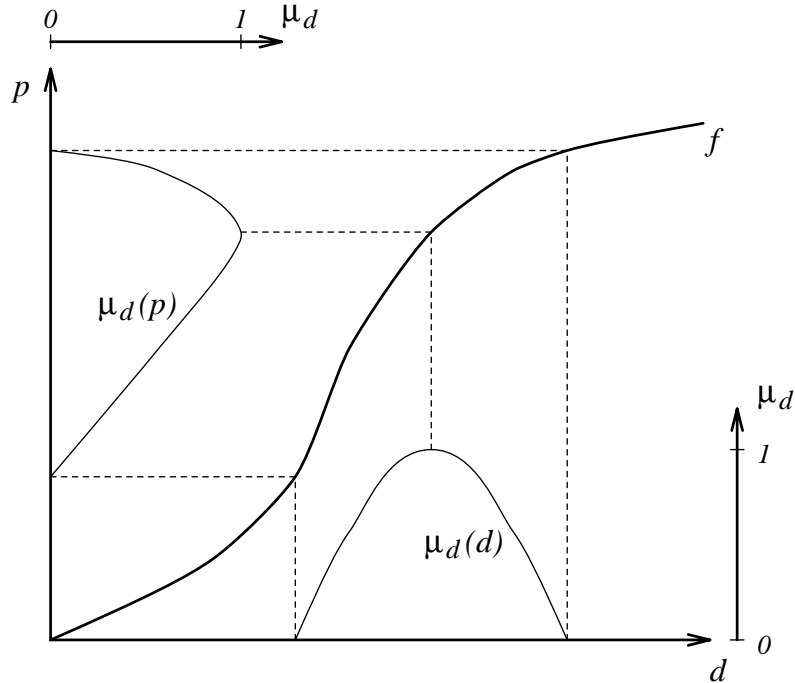


Figure 2: Zadeh's extension principle.

example above f is non-linear. Non-monotonic and discrete functions can also be used, as introduced in [26, 48], and reviewed below.

Once the imprecision on each design variable ($\mu_d(\vec{d})$) is induced onto the PVS, the induced preferences are combined with the functional requirements ($\mu_p(\vec{p})$) to obtain an overall preference ($\mu_o(\vec{p})$). The point (or points) with the highest preference correspond to the performance of the overall most preferred design(s). The design problem is to find the corresponding set of design variables (\vec{d}^*) that produce the maximum overall preference ($\mu_o^*(\vec{d}^*)$). In the typical engineering design case, where the inverse mapping ($\vec{f}^{-1} : \mathcal{Y} \rightarrow \mathcal{X}$) doesn't exist, $\mu_o(\vec{d})$ can still be obtained point by point [19].

3 The Computational Model for Design Imprecision

As indicated above, one of the central elements of the procedure to represent and manipulate imprecision in engineering design developed by the authors, and briefly described above, is the algorithm to induce preferences from independent variables to dependent ones. The following sections describe the algorithm as originally presented in the literature, and some of the extensions recently developed and implemented.

3.1 Computation of the Extension Principle

Kaufmann and Gupta [16, 44] describe an analytical method of calculating a fuzzy output (an application of Zadeh's extension principle) from imprecise inputs. This method is based on α -cuts [16] and interval arithmetic [21]. Although the method is straightforward in its approach, the manipulation of symbols and the solution of expressions that include high order polynomials, both in the numerator and denominator (for extended division), make this method infeasible for computer-assisted design applications. This is compounded by the fact that the exact so-

lution to the analytical application of the extension principle can be shown to be equivalent to an unwieldy non-linear programming problem [3]. A discrete numerical approach is therefore necessary to meet the computational requirements for handling many design variables. The section below discusses useful numerical techniques.

3.1.1 The Level Interval Algorithm (LIA)

Many discrete and analytical methods exist in the literature for carrying out extended operations with fuzzy sets (or fuzzy numbers). The Fuzzy Weighted Average (FWA) algorithm, as presented by W. M. Dong and F. S. Wong in [8], outlines a simple and efficient algorithm that is useful for carrying out engineering design calculations. This algorithm is extended below for generalized real functions of fuzzy variables, and the extended form is referred to as the *level interval algorithm* (LIA). Comparing the algorithm to the analytical method outlined in [16], LIA uses the interval analysis techniques as described; yet, LIA simplifies the process extensively by discretizing the membership functions of the input fuzzy numbers into a prescribed number of α -cuts. Performing interval analysis for each α -cut and combining the resultant intervals, the output is a discretized fuzzy set, the performance variable output of input preference functions for the case of a design calculation. Dong and Wong also include a combinatorial interval analysis technique in order to avoid the problems of the multiple occurrence of variables for division and multiplication in an algebraic equation expression. A condensed version of the algorithm from [8] has been provided below (where the terminology has been changed to reflect the application to design calculations). Later sections describe the implementation and extension of this algorithm.

There are conditions which must be satisfied for application of the algorithm: the preference functions must satisfy the normality and convexity conditions and must be continuous over the design variables (\vec{d}), no singularities of the functions can occur over \vec{d} (*i.e.*, no division by zero can occur, and no zero arguments can occur in $f_j(\vec{d})$ for each \vec{d}_i for the unary operations, such as the natural logarithm and the square root), and only monotonic regions of multi-valued functions, *e.g.*, sine and cosine, are computed for a given \vec{d}_i .

The algorithm is as follows: for N real imprecise design variables, $\vec{d}_1, \dots, \vec{d}_N$, let d_i ($i \in [1, N]$) be an element of \vec{d}_i . Given a performance variable represented by the mapping:

$$p = f(d_1, \dots, d_N) \quad d_i \in \vec{d}_i,$$

let \tilde{P} be the fuzzy output of the mapping. The following steps lead to the solution of \tilde{P} .

1. For each \vec{d}_i , discretize the preference function into a number of α values, $\alpha_1, \dots, \alpha_M$, where M is the number of steps in the discretization.
2. Determine the intervals for each variable $\vec{d}_i, i = 1, \dots, N$ at each α -cut, $\alpha_j, j = 1, \dots, M$.
3. Using one end point from each of the N intervals for each α_j , combine the end points into an N -ary array such that 2^N distinct permutations exist for the array.
4. For each of the 2^N permutations, determine $p_k = f(d_1, \dots, d_N), k = 1, \dots, 2^N$. The resultant interval for the α -cut, α_j , is then given by

$$\tilde{P}^{\alpha_j} = [\min(p_k), \max(p_k)].$$

DPs (units)	$\alpha = 0$	$\alpha = 1$	$\alpha = 0$
n	1.0	3.0	5.0
l/r	60.0	100.0	160.0
E (GPa)	75.0	150.0	225.0
K (simply-supported)	1.0	1.0	1.0

Table 1: Design Variable Data for Column Equation

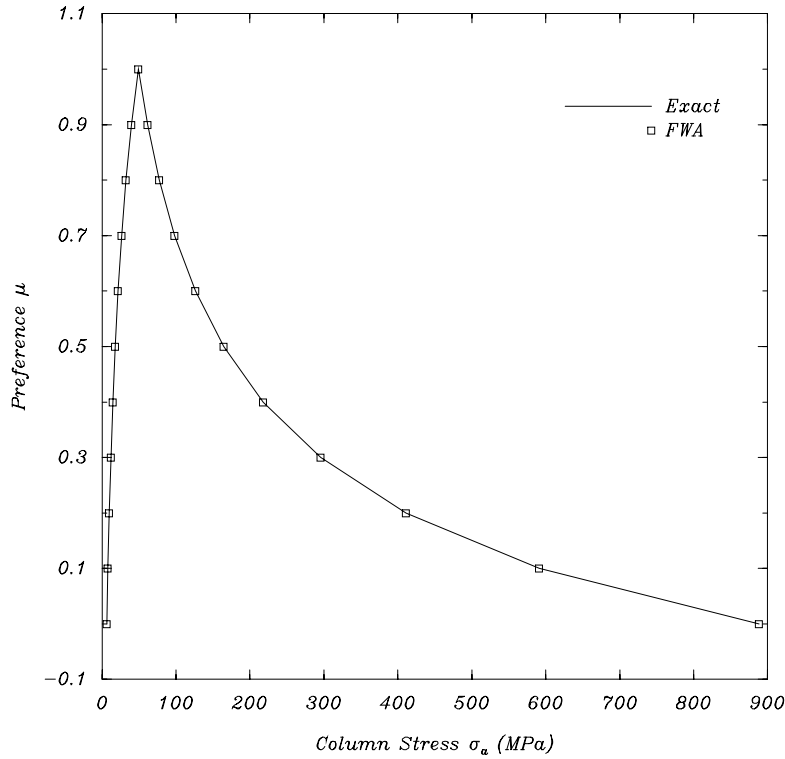


Figure 3: Example column calculation: Exact and LIA.

Figure 3 shows the results of the analytical application of the extension principle to the column-stress equation.

$$\sigma_a = \frac{\pi^2 E}{n \left(\frac{Kl}{r}\right)^2}. \quad (2)$$

The input variables for the maximum allowable stress σ_a are triangular preference functions as listed in Table 1.

3.1.2 Extending LIA for Internal Extrema

The LIA is valid only for real-valued functions f , and corresponding interval extensions $F(\mathcal{X}_0)$ that do not include internal bounded *extrema* for the intervals in question, $\mathcal{X} \in \mathcal{X}_0$. This is because only the endpoints (at a given α -cut) of the input variables d_i $i = 1, \dots, N$ are used in the computation. An extension to the LIA will now be introduced to determine the correct bounds $P^{\alpha_j}(\vec{D}^{\alpha_j})$ for a given α -cut α_j with the following procedure:

1. For each α -cut α_j , determine if an internal extremum exists for the α -cut intervals of d_i , $i = 1, 2, \dots, n$, $p = f(d_1, d_2, \dots, d_n)$. This may be accomplished by either analytically or numerically solving:

$$\frac{\partial p}{\partial d_i} = 0$$

for each d_i as a set of simultaneous equations. Section 3.2 below introduces an additional extension to efficiently locate internal extrema by use of optimization methods [20].

2. Denote the extrema by ξ_l and the values of the d_i that make up a given ξ_l by $\varepsilon_{l,i}$. If none of the points $\varepsilon_{l,i}$ lie within \mathcal{X}_0 , the extrema can be ignored and the standard LIA applied as before.
3. If every $\varepsilon_{l,i}$ lies within the α -cut \mathcal{X}_α , calculate $\xi_l = f(\varepsilon_{l,i})$.
4. If not every $\varepsilon_{l,i}$ lies within the α -cut, let q span those that do not, $\varepsilon_{l,q}$, and m span those that do, $\varepsilon_{l,m}$. Calculate all of the extrema values outside \mathcal{X}_α (but within \mathcal{X}_0) as $\xi_l = f(\varepsilon_m, d_q)$. This will be a two factorial set across the end points of the α -cut for all \tilde{d}_q .
5. Continuing with the standard LIA, using one end point from each of the N intervals for each α_j , combine the end points into an N -ary array such that 2^N distinct permutations exist for the array. For each of the 2^N permutations, k determine $p_k = f(d_1, \dots, d_N)$.
6. The resultant interval for the α -cut α_j is given by

$$P^{\alpha_j} = [\min(p_k, \xi_l), \max(p_k, \xi_l)]$$

over all k, l .

To illustrate the application of this algorithm, consider the function $z : \mathbb{R} \rightarrow \mathbb{R} \mid z(x) = 2.927x^3 - 1.927x$, which is a cubic equation passing through $(-1, -1)$, $(0, 0)$, and $(1, 1)$ with two local extremum, as shown in Figure 4. (The function z is to be interpreted as a performance variable p .)

Let \tilde{X} be a triangular imprecise number with preference function $\mu(x|\tilde{X})$, where $\mu(x|\tilde{X})$ equals 0 outside $\{x|x \in (-1, 1)\}$, equals 1 at $x = 0$, and linearly increases to $\mu(0) = 1$ from the endpoints $\mu(-1) = 0$ and $\mu(1) = 0$. (\tilde{X} is to be interpreted as a design variable d .) Substituting \tilde{X} and the equation for z into the LIA, the incorrect preference function for \tilde{Z} , shown in Figure 5, is obtained. The problem is the two local extrema in the function z . Hence the algorithm of Section 3.1.2 must be applied.

Substituting \tilde{X} and the equation for z into the extended LIA for functions with extrema, the correct preference function for \tilde{Z} , shown in Figure 6, is obtained. Jump discontinuities in preference occur at $z = \pm 0.602$. These result from the local extrema in the function z , shown in Figure 4 ($x = \pm 0.42$). At z values just above $z = 0.602$, only points in the neighborhood of $x = 0.926$ are in the pre-image of these z ; the map is one-to-one. But at z values just below $z = 0.602$, the pre-image contains points in the neighborhood of both $x = -0.42$ and $x = 0.926$. Values of z below the local maximum in Figure 4 ($x = -0.42$, $z = 0.602$) have three points x in their pre-image. Points in the neighborhood of $x = -0.42$ have higher preference than points in the neighborhood of $x = 0.926$, and hence at $z = 0.602$ there is a jump discontinuity in preference.

This type of discontinuity in output preference is easily understood; it arises from the multiplicity of points in the map's pre-image (in this case, at most three points). The multiplicity

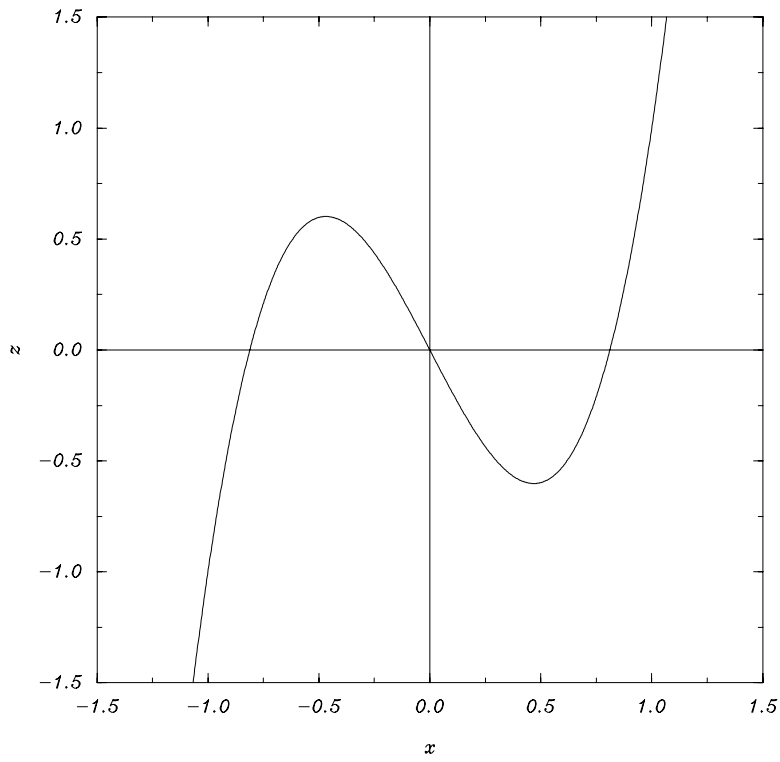


Figure 4: The cubic equation $z = 2.927x^3 - 1.927x$.

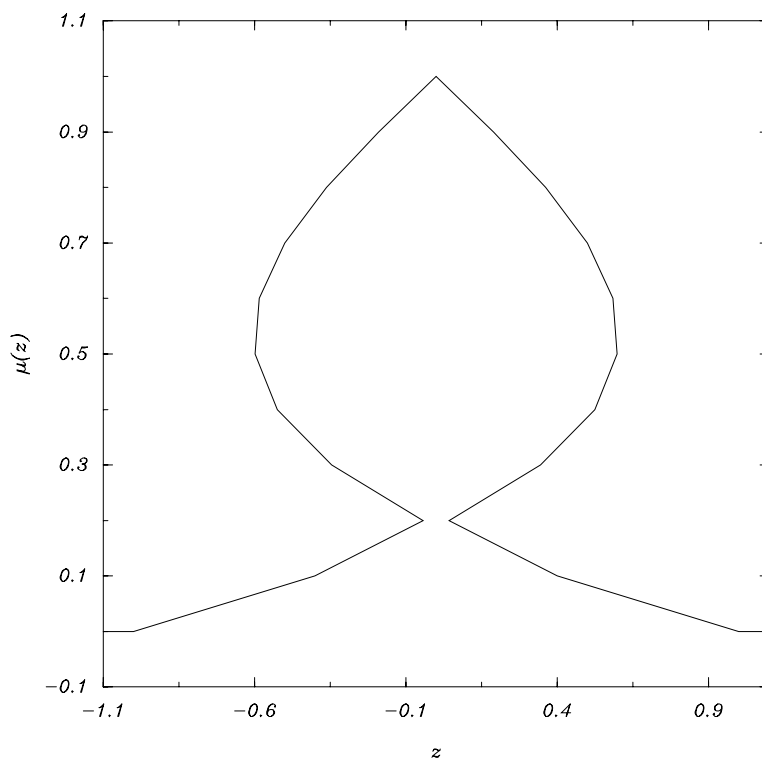
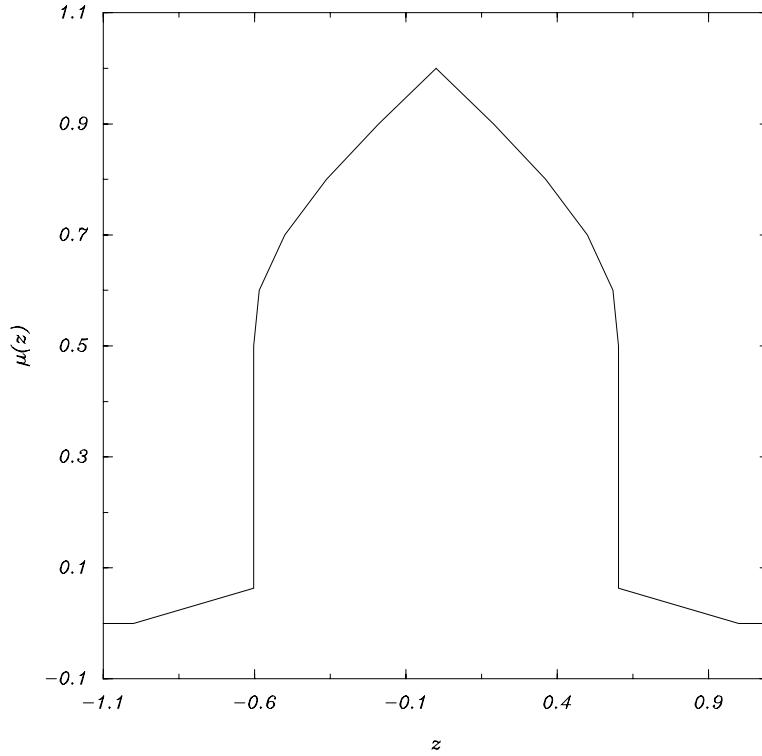


Figure 5: Incorrect LIA solution for $\mu(z|\tilde{Z})$.

Figure 6: Output preference function $\mu(z|\tilde{Z})$.

here is finite, and can be dealt with using the extended LIA. Real concerns arise, however, when the output preference functions exhibit unboundedness, either in support or in the number of discontinuities. Representation and calculation of the imprecise preference functions then becomes difficult, as will be demonstrated in Section 4.

3.2 Optimization

One of the major limitations of the LIA stems from the assumption that the extreme values of f_j will occur at the corner points of the D_{α_k} n -cube [20]. The algorithm may thus be improved by relaxing this assumption. The problem restated is to find:

$$\begin{aligned} p_{j_{\min}}^{\alpha_k} &= \min\{p_j = f_j(\vec{d}) \mid \vec{d} \in D_{\alpha_k}\} \\ p_{j_{\max}}^{\alpha_k} &= \max\{p_j = f_j(\vec{d}) \mid \vec{d} \in D_{\alpha_k}\} \end{aligned} \quad (3)$$

Finding extrema within a subspace is a constrained optimization problem.

In choosing an optimization technique, a trade-off must be made between computational cost and robustness. Traditional calculus-based optimization methods converge in relatively few function evaluations but seek only local minima. Randomized search methods such as genetic algorithms offer greater robustness [10] but require more function evaluations. Where function evaluations are relatively expensive, as is common in engineering design, traditional optimization methods are a satisficing solution.

The traditional optimization algorithm utilized here is Powell's method, which begins as a one-at-a-time search. After each iteration a heuristic determines whether to replace the direc-

tion of maximum decrease with the net direction moved during the last iteration. This allows minimization down valleys while avoiding linear dependence in the set of search directions [1].

An important feature for a practical computational tool is a means to trade-off the number of function evaluations against accuracy. Such an adjustment enables the designer to use the same program to obtain quick estimates as well as precise evaluations. This is implemented as a user-specified fractional precision that defines termination criteria for the optimization algorithm.

Suppose that it is necessary to incur the minimum number of function evaluations. A fractional precision of 1 would be specified, creating automatically satisfied termination criteria, and the optimization would proceed through exactly one iteration of a one at a time search using the maximum step size. The algorithm begins at one corner of the search space D_{α_k} , and checks corners in each of the n directions given by d_1, \dots, d_n , moving to the minimum each time. It expends $n + 1$ function evaluations to find each end point, and therefore $2n + 2$ per α -cut, as compared to 2^n per α -cut for the LIA. This is a substantial improvement, but the α -cut interval obtained is only correct if f_j is monotonic: none of the interior points of the D_{α_k} n -cube are evaluated. Minimizing function evaluations carries the cost of implicitly assuming monotonicity.

If f_j is known to be monotonic, $2n + 2$ is not the minimum number of function evaluations. The first pass of the optimization algorithm identifies the direction for each d_i in which f_j increases. Subsequent extrema can then be directly evaluated, without the need for searching. Hence where f_j is monotonic, $n + 2$ function evaluations are required for the first α -cut and 2 for each subsequent α -cut. Experimental design methods [28] may also be used to identify design variables with near-linear effects, and remove those directions from the optimization search [20].

4 Anomalies in Imprecise Calculations

4.1 Introduction

The extended-operations algorithm (LIA) presented above provides a basis for computing with sets of *imprecise* variables in preliminary engineering design. Even though the algorithm and extensions can be applied to standard computing functions, certain limitations apply. For the LIA without the optimization extension: the preference functions must satisfy the normality and convexity conditions and must be continuous over \tilde{d} ; no singularities of the functions can occur over \tilde{d} and only monotonic regions of multi-valued functions, *e.g.*, sine and cosine, are computed for a given \tilde{d}_i . For the optimization extended algorithm, the continuity and monotonicity requirements are relaxed, though singularities must still be avoided. This section considers cases in which one or more of these conditions are violated.

4.2 Unbounded Preference Functions

Consider the possibility of functions which operate on well-formed imprecise numbers and create output preference functions whose support becomes unbounded. If such functions exist, they would fail when applying the methods of Section 3.

Consider an imprecise number \tilde{X} of “about $1/4$, and possibly negative”, *i.e.*, let \tilde{X} be a triangular imprecise number with preference function $\mu(x|\tilde{X})$, where $\mu(\frac{1}{4}) = 1$, and $\mu(x|\tilde{X}) = 0$ outside $\{x|x \in (\frac{1}{4} - s, \frac{1}{4} + s)\}$, and s will be gradually increased to observe the variation in

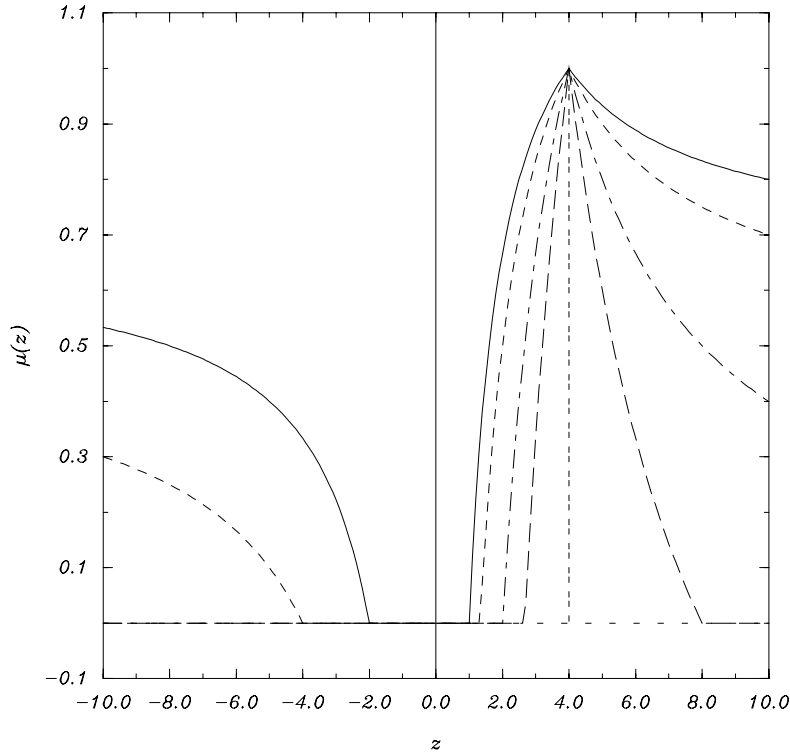


Figure 7: Output preference functions $\mu(z)$.

calculation results. Such an imprecise number will be expected to become ill-defined with inversion (when s becomes large enough to encompass zero), since the inverse of points in the neighborhood of zero become unbounded.

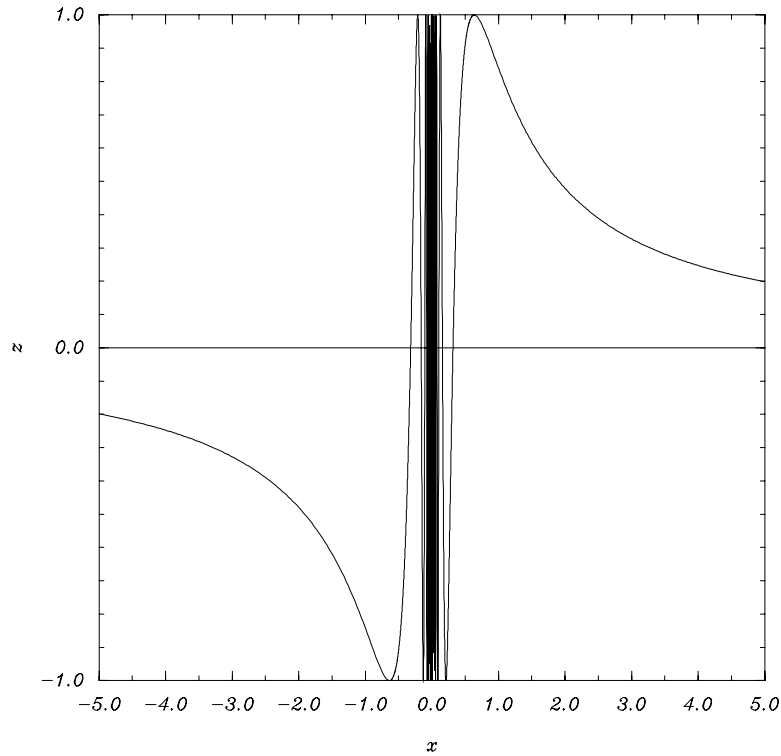
The imprecise variable \tilde{X} centered at $1/4$ was systematically increased in support to include negative numbers to observe the effects of inclusion of zero within the support. The effect on the output of the inverse function $z : \mathbb{R} \rightarrow \mathbb{R} \mid z(x) = 1/x$ was observed. Substituting \tilde{X} and the equation for z into the extension principle, the equation for the preference of any z becomes:

$$\mu(z) = \mu(x), \text{ where } x : x = 1/z \tag{4}$$

since $z = f(x) = 1/x$ is one-to-one.

The results are shown in Figure 7. As s increases, the preference function for z begins to extend to include non-zero preference for unbounded values; all calculations are performed with the nominal \tilde{X} remaining at $1/4$. This is a case of a function which operates on a well-formed imprecise number and produces a preference function with unbounded support. This occurs because \tilde{X} includes 0, a singularity of the function z . Hence the output preference includes $z(0) = 1/0 = \infty$. When the input preference function $\mu(x|\tilde{X})$ includes both positive and negative points in the neighborhood of zero, the output preference function $\mu(z)$ has support over both positive and negative values in the “neighborhood” of infinity. Therefore the output preference function will appear as shown in Figure 7.

The LIA will fail, of course, since zero is a singularity of z , and hence cannot occur in \tilde{X} . This result seems to indicate that using preference curves to represent designer uncertainty will sometimes fail. But consider what the result indicates when the preference function is

Figure 8: The function $z = \sin(1/x)$.

interpreted as the degree of designer preference. The unbounded results on z are due to the preferences stated on x . The problem has been incorrectly formulated, and must be reformulated to hedge the preferences on x away from the singular point zero, just as the designer would have to do if imprecision were not used. This method warns the designer of the problem, whereas using crisp or single valued calculations (in the example, just using $x = 1/4$) would not.

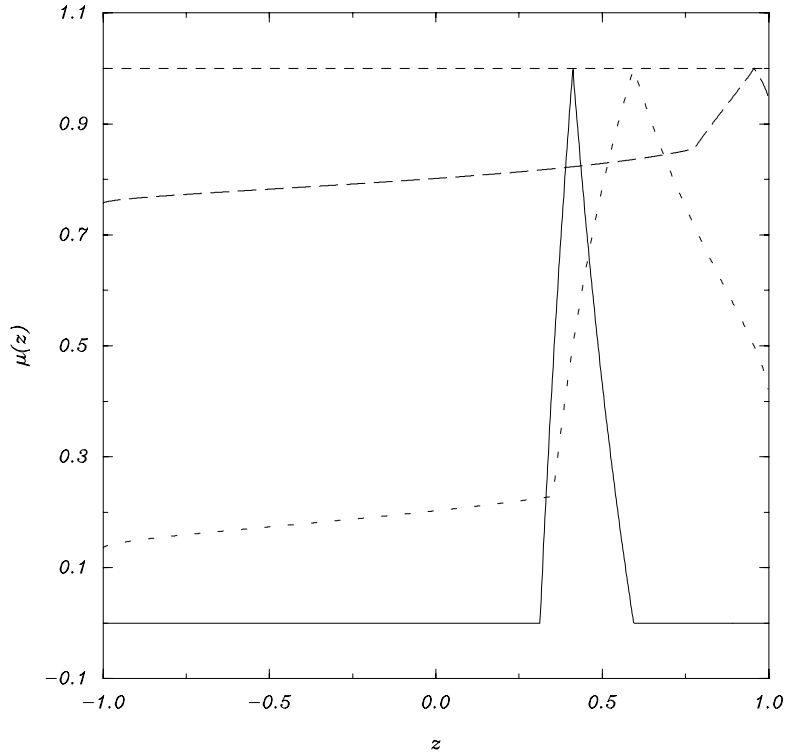
4.3 Singular Points and Preference Functions

Consider any function which operates on well-formed imprecise numbers and creates an output preference function requiring a limit process for evaluation. Such could be the case of a function with a singularity within the support of the design variable's preference. To demonstrate this, consider $z : \mathbb{R} \rightarrow [-1, 1] \subset \mathbb{R}$ such that: —

$$z(x) = \begin{cases} \sin(1/x) & x \neq 0 \\ 0 & x = 0 \end{cases}$$

This function has a non-removable singularity at $x = 0$ (multiplying by $(x-0)^n$ will not create a locally regular function for any n). As $x \rightarrow 0$, $z(x)$ oscillates an unbounded number of times between -1 and 1 (refer to Figure 8). The question then arises as to the output preference behavior when the input imprecise number \tilde{X} includes in its support the singular point $x = 0$.

Let \tilde{X} be an imprecise number with preference function $\mu(x|\tilde{X})$, where $\mu(x|\tilde{X}) = 0$ outside $\{x|x \in (-\pi + s, \pi)\}$, $\mu(\frac{\pi}{2}) = 1$, and s is gradually decreased from $\frac{3}{2}\pi$ to 0 to observe the

Figure 9: Output preference functions $\mu(z)$.

variation in calculation results. Figure 9 shows the corresponding output preference functions resulting from application of the extension principle to the imprecise numbers \tilde{X} .

When the support for the input \tilde{X} does not include the singular point 0 (*i.e.*, $s > \pi$), the output preference function remains perfectly well defined. When the input preference function does include the singular point for any support range, the output preference function would be expected to become not well defined, since the function z itself becomes oscillatory (unstable). But note that the output preference function remains well defined. This is due to the supremum in the extension principle definition, and the selected input preference function $\mu(x|\tilde{X})$. There is always a point x in each z value's pre-image such that the preference of that x is greater than the singular point preference. Since this supremum preference is greater than the singular point preference, the preference for all z are well defined. Hence the preference $\mu(z)$ is well defined when $x = 0$ does not have the peak preference among all x . When $x = 0$ has the peak preference, the *sup* definition must be explicitly used, since the point with maximum preference in the pre-image of any z does not exist, only a least upper bound exists. All values of z have in their pre-image points x arbitrarily close to zero. In this case, the peak preference of $x = 0$ would be the least upper bound of preference for all z , *i.e.*, $\mu(z) = 1 \forall z$.

Calculation of the preference for $z = \sin(1/x)$ poses difficulty. There are numerous and possibly an infinite number of internal extrema within the support of the design variable (depending if $x = 0$ is in the support). Hence the methods of Section 3.1.2 are impractical as posed. They must be extended using a limiting process. That is, the support of $\mu(x)$ must be split into monotonic intervals of z between the zeros of the slope of z as discussed and solved in Section 3.1.2. The difference here is that there are an unbounded number of such intervals. However, these intervals will become smaller in a limit as will their contribution to $\mu(z)$. The

limiting process can be terminated with a convergence criterion.

Given that functions exist which exhibit such computational difficulty, a mechanism must be found to identify when the methods of Section 3 will fail. This problem's key feature, which causes the real-time methods to fail, is the presence of the singularity of the function z in the design variable \tilde{X} 's support. The same can be said of the previous example involving unbounded support ($z = 1/x$). In both cases, the output preference functions are defined and exist, but they exhibit difficulty in representation or evaluation. These effects are a direct result of the singularity of the function being within the support of the design variable.

Note in the case of $z = \sin(1/x)$, the singularity requires particular care, in that $\mu(z)$ is defined, even though there is a singularity of z within \tilde{X} . Any conclusions drawn by a designer when observing such a resulting preference curve for z may be misleading. Failure to meet the criteria of the real-time techniques of Section 3 indicates that the functions being considered (z) are incompatible with the specified design variable preference functions ($\mu(x|\tilde{X})$), and hence is a warning to the designer that the values desired for \tilde{X} should be re-considered.

5 Interpretability of Imprecision Results

Consider the possible existence of continuous functions $f : \mathcal{D} \rightarrow \mathcal{P}$ that operate on well-formed design variable preference functions and with them create output performance variable preference functions which oscillate in preference from 0 to 1 as the output z approaches a limit value \bar{z} . The existence of any such function would question the viability of the entire proposed method, since such a function would be an interpretable mapping from a design variable set, where the preferences are known, to an output set where the preferences are un-interpretable. Via construction, it will be shown (though not formally proven) that this is only possible for functions which are infinitely multi-valued (have an infinite number of branches), but need not be singular. The existence of any such function which maps specified preferences into un-interpretable preferences is not due to the fuzzy extension principle, but rather to the multi-valued character of the function itself.

Such a function z must convert a well-formed, convex input preference function $\mu(x|\tilde{X})$ into a preference function with a non-removable singularity. Such a membership function might be of the form:

$$\mu(z) = 1/2 \sin(1/z) + 1/2,$$

or more simply behave like:

$$\mu(z) \sim \sin(1/z),$$

i.e., behave un-interpretable in the neighborhood of $z = 0$. Denoting $z = f(x)$ the "equation" above can be inverted to derive:

$$f(x) = 1/\arcsin(\mu(z|\tilde{Z})).$$

Simplifying through the extension principle (Equation 1), this can be equated to:

$$f(x) = 1/\arcsin(\mu(x|\tilde{X})).$$

Now define $\mu(x)$ as a convex function over all real values with range $[0, 1]$, such as e^{-x^2} . This construct:

$$z : \mathbb{R} \rightarrow \mathbb{R} \mid z = 1/\arcsin(e^{-x^2})$$

is based on $\mu(z)$ and $\mu(x)$. Using different functions which exhibit the same behavior as the chosen $\mu(z)$ and $\mu(x)$ will construct a similar z , though it may not be as easily expressible.

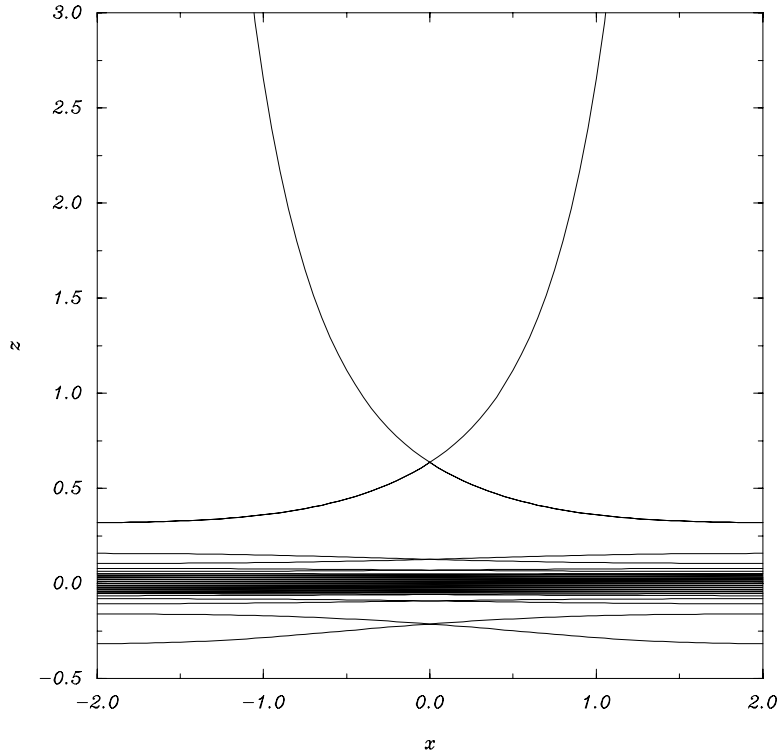


Figure 10: $z = \frac{1}{\arcsin(e^{-x^2})}$.

The created z is defined over all x and contains no singularities. It is, however, multi-valued (refer to Figure 10). Considering all of the branches, the pre-image of each value z has either zero or two points x .

To observe the effects of z on an imprecise number \tilde{X} , consider \tilde{X} where $\mu(x|\tilde{X}) = 0$ outside $\{x|x \in (-1, 1)\}$, $\mu(0) = 1$, and $\mu(x|\tilde{X})$ smoothly increases to the peak at 0 between -1 and 1 . Substituting \tilde{X} and the equation for z into the extension principle, the resulting $\mu(z)$ is shown in Figure 11.

Note that for z values in the neighborhood of zero, the preference function becomes arbitrary, from 0 through 1 in preference. That is, moving a small amount (δ) in the neighborhood of $z = 0$ will result in large changes in $\mu(z)$, from zero to one. Also note the intervals on z where the preference function is not mapped from any x (for example, all z values between $-\infty$ and $\sin(1/\pi)$). These z values require complex x ; there are no real x in the pre-image of these z . The number of such intervals in a neighborhood of z becomes unbounded as z approaches zero (and the length of the each interval approaches zero length). These intervals arise since f is not a surjection; *i.e.*, $Z = f(\mathbb{R}) \not\subseteq \mathbb{R}$, but $Z \subset \mathbb{R}$. According to the interpretation of preference used here, $\mu(z) \simeq 0$ for the values of z having no real points x in the pre-image. Therefore $\mu(z) = 0$ for such points to define $\mu(z) \forall z$, as reflected in Equation 1.

These results would suggest that a preference function could become un-interpretable, since this preference function became un-interpretable as $z \rightarrow 0$. But this was entirely due to the unbounded number of branches considered in z : z is not a function. For any particular branch, however, the mapping is well defined. This result (infinitely many choices as $z \rightarrow 0$) exists using crisp numbers as well as imprecise numbers.

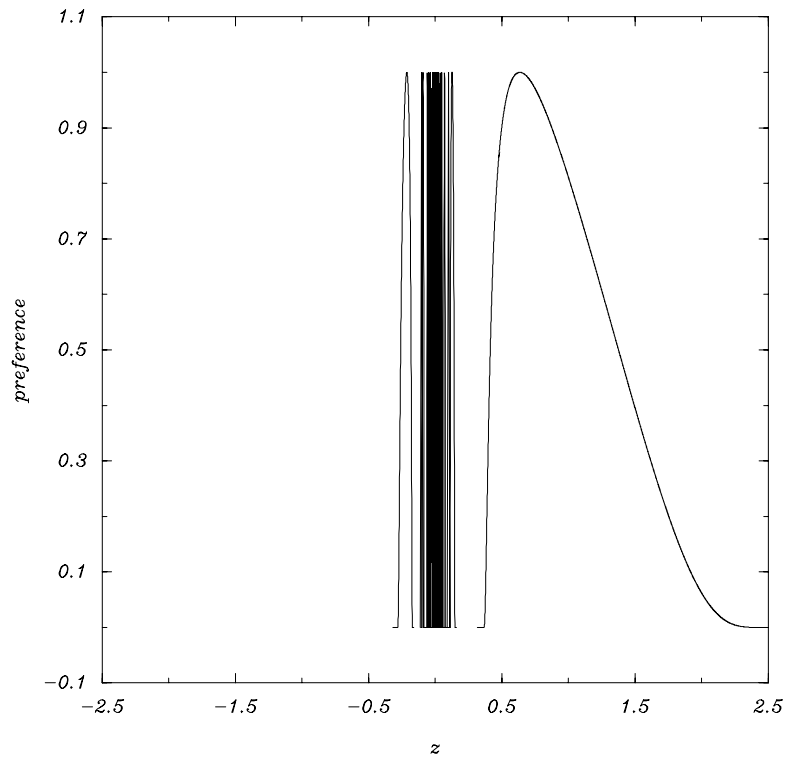


Figure 11: Output preference function $\mu(z)$.

Note z was constructed based on the desired output function $\mu(z)$ being un-interpretable at a value. The only way to exhibit this behavior was with a multi-valued function, where the multi-valued character arose as a direct consequence of the desired un-interpretability. Thus the conclusion: only multi-valued functions can lead to un-interpretability, entirely due to the indecision on which branch is being used in the function z itself, not in the fuzzy mathematics. Once a branch is selected, the problem reduces to a simple case which the methods of Section 3 can accommodate. Replacing the specific *sine* used for $\mu(z)$ by any general periodic function and replacing the specific $\mu(x)$ by a general convex imprecise number in the construction would demonstrate the conclusion for the general case (though this is not a formal proof). Overall, if the performance expression is interpretable, then the use of the fuzzy mathematics in design calculations will also produce interpretable results.

6 Conclusion

Imprecision and uncertainty occur throughout the engineering design process. Many methods for incorporating uncertainty (*e.g.*, utility theory, probability methods, Taguchi's method, *etc.*) are in common use, but methods to represent imprecision in engineering design are few [2]. The Method of Imprecision (M_I) is a formal method for incorporating the natural level of imprecision that occurs throughout the engineering design process. This paper has presented the details of the Level Interval Algorithm (LIA) used internally by the M_I, and its extensions to permit application to engineering design problems in industry where monotonicity cannot be guaranteed, only discrete values may be available for some variables (and hence continuity

must be relaxed), and engineering analyses are expensive and must be minimized.

Computation problems that reach beyond the scope of the LIA, such as singularities, *etc.*, have also been examined. It has been shown that the LIA behaves no worse than conventional calculations in the presence of these difficulties.

The approach embodied in the M₀I encourages the designer and customer to specify preferences on design and performance variables (specifications), and thus promotes design communication to evolve from individual “point” designs to (fuzzy) sets of designs. Since a range of possible design variable values can be communicated to down-stream design processes earlier than a completed individual design, the M₀I can facilitate (fuzzy) set-based concurrent design.

Acknowledgments

This material is based upon work supported, in part, by: the National Science Foundation under NSF Grant Numbers DDM-9201424 and DMI-9523232. The work was also made possible in part by a Faculty Early Career Development Award from the National Science Foundation. The authors acknowledge the support of the Leaders for Manufacturing Program, a collaboration between MIT and US industry. Any opinions, findings, conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the sponsors.

References

- [1] ADBY, P., AND DEMPSTER, M. *Introduction to Optimization Methods*. Chapman and Hall, London, 1974.
- [2] ANTONSSON, E. K., AND OTTO, K. N. Imprecision in Engineering Design. *ASME Journal of Mechanical Design 117(B) (Special Combined Issue of the Transactions of the ASME commemorating the 50th anniversary of the Design Engineering Division of the ASME.)* (June 1995), 25–32. Invited paper.
- [3] BAAS, S., AND KWAKERNAAK, H. Rating and ranking of multiple-aspect alternatives using fuzzy sets. *Automatica 13* (1977), 47–48.
- [4] BANDEMER, H., Ed. *Modelling Uncertain Data*, 1st ed. Akademie Verlag, Berlin, 1993.
- [5] BARRETT, C. R., PATTANAIK, P. K., AND SALLES, M. On choosing rationally when preferences are fuzzy. *Fuzzy Sets and Systems 34*, 2 (1990), 197–212.
- [6] DIAZ, A. R. Fuzzy set based models in design optimization. In *Advances in Design Automation - 1988* (New York, Sept. 1988), S. S. Rao, Ed., vol. DE-14, ASME, pp. 477–485.
- [7] DIAZ, A. R. A strategy for optimal design of hierarchical systems using fuzzy sets. In *The 1989 NSF Engineering Design Research Conference* (College of Engineering, University of Massachusetts, Amherst, June 1989), J. R. Dixon, Ed., NSF, pp. 537–547.
- [8] DONG, W. M., AND WONG, F. S. Fuzzy weighted averages and implementation of the extension principle. *Fuzzy Sets and Systems 21*, 2 (Feb. 1987), 183–199.
- [9] DUTTA, B., PANDA, S. C., AND PATTANAIK, P. K. Exact choice and fuzzy preferences. *Mathematical Social Sciences 11* (1986), 53–68.
- [10] GOLDBERG, D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, Massachusetts, 1989.
- [11] HAMBURG, I., AND HAMBURG, P. Fuzzy logic, A user friendly technology for the management of uncertain knowledge in engineering design process. In *Modelling Uncertain Data* (Berlin, Mar. 1993), H. Bandemer, Ed., Akademie Verlag, pp. 153–157. Mathematical Research series, Volume 68, Lectures from the GAMM-Workshop: Modelling Uncertain Data, held in Freiberg from March 21-24, 1992.

- [12] HOLMES, M. F. Machine dynamics, The need for greater productivity. In *Research Needs in Mechanical Systems* (New York, 1984), K. N. Reid, Ed., ASME, pp. 140–159.
- [13] HSU, Y.-L., LIN, Y.-F., AND SUN, T.-L. Engineering design optimization as a fuzzy control process. In *Proceedings of the Fourth IEEE International Conference on Fuzzy Systems (FUZZ-IEEE/IFES'95)* (Mar. 1995), vol. 4, IEEE, pp. 2001–2008.
- [14] KACPRZYK, J., AND FEDRIZZI, M., Eds. *Combining Fuzzy Imprecision with Probabilistic Uncertainty in Decision Making*, vol. 310 of *Lecture Notes in Economics and Mathematical Systems*. Springer-Verlag, New York, 1988.
- [15] KACPRZYK, J., AND ROUBENS, M., Eds. *Non-Conventional Preference Relations in Decision Making*, vol. 301 of *Lecture Notes in Economics and Mathematical Systems*. Springer-Verlag, New York, 1988.
- [16] KAUFMANN, A., AND GUPTA, M. M. *Introduction to Fuzzy Arithmetic: Theory and Applications*. Electrical/Computer Science and Engineering Series. Van Nostrand Reinhold Company, New York, 1985.
- [17] KEENEY, R., AND RAIFFA, H. *Decisions with Multiple Objectives: Preferences and Value Trade-offs*. Cambridge University Press, Cambridge, U.K., 1993.
- [18] KNOSALA, R., AND PEDRYCZ, W. Evaluation of design alternatives in mechanical engineering. *Fuzzy Sets and Systems* 47, 3 (1992), 269–280.
- [19] LAW, W. S., AND ANTONSSON, E. K. Implementing the Method of Imprecision: An Engineering Design Example. In *Proceedings of the Third IEEE International Conference on Fuzzy Systems (FUZZ-IEEE '94)* (June 1994), vol. 1, IEEE, pp. 358–363. Invited paper.
- [20] LAW, W. S., AND ANTONSSON, E. K. Optimization Methods for Calculating Design Imprecision. In *Advances in Design Automation - 1995* (Sept. 1995), vol. 1, ASME, pp. 471–476.
- [21] MOORE, R. E. *Methods and Applications of Interval Analysis*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1979.
- [22] MÜLLER, K., AND THÄRIGEN, M. Applications of fuzzy hierarchies and fuzzy madm methods to innovative system design. In *Proceedings of the Third IEEE International Conference on Fuzzy Systems* (June 1994), vol. 1, IEEE, pp. 364–367.
- [23] OTTO, K. N., AND ANTONSSON, E. K. Trade-Off Strategies in Engineering Design. *Research in Engineering Design* 3, 2 (1991), 87–104.
- [24] OTTO, K. N., AND ANTONSSON, E. K. Design Parameter Selection in the Presence of Noise. *Research in Engineering Design* 6, 4 (1994), 234–246.
- [25] OTTO, K. N., AND ANTONSSON, E. K. Modeling Imprecision in Product Design. In *Proceedings of the Third IEEE International Conference on Fuzzy Systems (FUZZ-IEEE '94)* (June 1994), vol. 1, IEEE, pp. 346–351. Invited paper.
- [26] OTTO, K. N., LEWIS, A. D., AND ANTONSSON, E. K. Approximating α -cuts with the Vertex Method. *Fuzzy Sets and Systems* 55, 1 (Apr. 1993), 43–50.
- [27] OTTO, K. N., LEWIS, A. D., AND ANTONSSON, E. K. Determining optimal points of membership with dependent variables. *Fuzzy Sets and Systems* 60, 1 (Nov. 1993), 19–24.
- [28] PHADKE, M. *Quality Engineering Using Robust Design*. Prentice Hall, Englewood Cliffs, NJ, 1989.
- [29] POSTHOFF, C., AND SCHLOSSER, M. Learning from examples for the construction of fuzzy evaluations. In *Proceedings of the Third IEEE International Conference on Fuzzy Systems* (June 1994), vol. 1, IEEE, pp. 368–371.
- [30] RAO, S. S. Description and optimum design of fuzzy mechanical systems. *ASME Journal of Mechanisms, Transmissions, and Automation in Design* 109 (Mar. 1987), 126–132.

- [31] RAO, S. S., AND DHINGRA, A. K. Integrated optimal design of planar mechanisms using fuzzy theories. In *Advances in Design Automation - 1989* (New York, Sept. 1989), vol. DE-15-2, ASME, pp. 161–168.
- [32] RAO, S. S., AND DHINGRA, A. K. Applications of fuzzy theories to multi-objective system optimization. NASA Contractor Report 177573, NASA Ames Research Center, 1991.
- [33] RAO, S. S., DHINGRA, A. K., AND KUMAR, V. Nonlinear membership functions in the fuzzy optimization of mechanical and structural systems. Tech. Rep. 90-1175-CP, AIAA, New York, 1990.
- [34] RAO, S. S., SUNDARARAJU, K., PRAKASH, B., AND BALAKRISHNA, C. Multiobjective fuzzy optimization techniques for engineering design. *Computers and Structures* 42, 1 (1992), 37–44.
- [35] SAKAWA, M., AND KATO, K. An interactive fuzzy satisficing methods for large-scale multi-objective linear programs with fuzzy numbers. In *Proceedings of the Fourth IEEE International Conference on Fuzzy Systems (FUZZ-IEEE/IFES'95)* (Mar. 1995), vol. 3, IEEE, pp. 1155–1162.
- [36] SCOTT, M. J., AND ANTONSSON, E. K. Aggregation Functions for Engineering Design Trade-offs. In *9th International Conference on Design Theory and Methodology* (Sept. 1995), vol. 2, ASME, pp. 389–396.
- [37] SHIH, C. J., AND WANGSAWIDJAJA, R. A. S. Multiobjective fuzzy optimization with random variables in a mix of fuzzy and probabilistic environment. In *Proceedings of the Fourth IEEE International Conference on Fuzzy Systems (FUZZ-IEEE/IFES'95)* (Mar. 1995), vol. 3, IEEE, pp. 1163–1170.
- [38] SULLIVAN, L. P. Quality function deployment. *Quality Progress* (June 1986), 39–50.
- [39] THURSTON, D. L., AND CARNAHAN, J. Fuzzy ratings and utility analysis in preliminary design: evaluation of multiple attributes. *ASME Journal of Mechanical Design* (Sept. 1990). Submitted for review.
- [40] THURSTON, D. L., AND CARNAHAN, J. Fuzzy ratings and utility analysis in preliminary design evaluation of multiple attributes. *ASME Journal of Mechanical Design* 114, 4 (Dec. 1992), 648–658.
- [41] ULRICH, K. T., AND PEARSON, S. A. Does product design really determine 80% of manufacturing cost? Working Paper MSA 3601-93, MIT, Sloan School of Management, Cambridge, MA, Aug. 1993.
- [42] WARD, A. C., LIKER, J. K., SOBEK, D. K., AND CRISTIANO, J. J. Set-based concurrent engineering and Toyota. In *Design Theory and Methodology – DTM '94* (Sept. 1994), vol. DE68, ASME, pp. 79–90.
- [43] WHITNEY, D. E. Manufacturing by design. *Harvard Business Review* 66, 4 (July 1988), 83–91.
- [44] WOOD, K. L. *A Method for Representing and Manipulating Uncertainties in Preliminary Engineering Design*. PhD thesis, California Institute of Technology, Pasadena, CA, 1989.
- [45] WOOD, K. L., AND ANTONSSON, E. K. Computations with Imprecise Parameters in Engineering Design: Background and Theory. *ASME Journal of Mechanisms, Transmissions, and Automation in Design* 111, 4 (Dec. 1989), 616–625.
- [46] WOOD, K. L., AND ANTONSSON, E. K. Modeling Imprecision and Uncertainty in Preliminary Engineering Design. *Mechanism and Machine Theory* 25, 3 (Feb. 1990), 305–324. Invited paper.
- [47] WOOD, K. L., ANTONSSON, E. K., AND BECK, J. L. Representing Imprecision in Engineering Design – Comparing Fuzzy and Probability Calculus. *Research in Engineering Design* 1, 3/4 (1990), 187–203.
- [48] WOOD, K. L., OTTO, K. N., AND ANTONSSON, E. K. Engineering Design Calculations with Fuzzy Parameters. *Fuzzy Sets and Systems* 52, 1 (Nov. 1992), 1–20.

- [49] ZADEH, L. A. Fuzzy sets. *Information and Control* 8 (1965), 338–353.
- [50] ZIMMERMANN, H.-J., AND SEBASTIAN, H.-J. Optimization and fuzziness in problems of design and configuration. In *Proceedings of the Second IEEE International Conference on Fuzzy Systems* (San Francisco, CA, June 1993), IEEE, Pergamon Press, pp. 1237–1240.
- [51] ZIMMERMANN, H.-J., AND SEBASTIAN, H.-J. Fuzzy design - Integration of fuzzy theory with knowledge-based system-design. In *Proceedings of the Third IEEE International Conference on Fuzzy Systems* (June 1994), vol. 1, IEEE, pp. 352–357.
- [52] ZIMMERMANN, H.-J., AND SEBASTIAN, H.-J. Intelligent system design support by fuzzy-multi-criteria decision making and/or evolutionary algorithms. In *Proceedings of the Fourth IEEE International Conference on Fuzzy Systems (FUZZ-IEEE/IFES'95)* (Mar. 1995), vol. 1, IEEE, pp. 367–374.