Integrating a human designer's preferences in Multidisciplinary Design Optimization

Paul Reverdy*, Akhil Reddy†, Luigi Martinelli‡, Naomi E. Leonard§
Department of Mechanical and Aerospace Engineering, Princeton University, Princeton, NJ 08544

Multidisciplinary Design Optimization (MDO) is a powerful engineering tool that allows designers to incorporate information from all relevant design disciplines simultaneously. In aerospace applications, for example, MDO has been used to produce designs that incorporate both the structural and aerodynamic disciplines. It is not generally possible to optimize the objectives of all disciplines simultaneously, so producing an optimal design requires a human designer to balance the tradeoffs between the various objectives. We propose and implement a novel system that helps the designer explore the various possible tradeoffs and systematically find their most preferred design. We show that the system converges to the most preferred design in a simulated task and discuss how it could be used in an industrial MDO problem.

I. Introduction

Engineering design is an iterative optimization process incorporating multiple objectives, often representing different design disciplines. Multidisciplinary Design Optimization (MDO) is a numerical tool often used in engineering design, which allows designers to find optimal designs by incorporating information from all relevant design disciplines. However, it is not generally possible to optimize the objectives of all disciplines simultaneously, meaning tradeoffs must be made among the conflicting objectives in order to find an optimal design. Ultimately, it is a human designer who expresses a preference about which tradeoffs are acceptable and which design to build. This makes the human designer an essential part of any MDO system.

Frameworks and tools for MDO have been developed in the literature over the past several decades. The review [1] and the more recent survey [2] provide a summary of the existing literature. While much work has been done to develop algorithms for finding the Pareto frontier of efficient tradeoffs, comparatively little work has considered the human designer. Our focus is on studying MDO with an emphasis on the human designer.

In parallel with the MDO literature, Multiple Criteria Decision Analysis (MCDA) has been developed in the decision theory literature [3]. As its name implies, MCDA studies decisions that incorporate multiple objectives and explicitly considers the preference information that is required to discriminate among efficient tradeoffs. Through its connection with economics and psychology, the MCDA literature provides frameworks for eliciting this preference information from human decision makers, as well as methods for encoding the preferences mathematically in utility functions. By integrating tools from the MDO and MCDA literatures, we provide a framework in the present paper for rationalizing the process of engineering design by explicitly including the human designer in the system.

Such a framework is valuable within a given design exercise, but even more so over the course of a series of design exercises. An important aspect of design practice is that an experienced designer will tend to find a good solution much more quickly than a novice. By performing many design exercises, the experienced designer develops intuition about the nature of good solutions, which allows him/her to more quickly search the design space. This intuition is likely to be largely subconscious and as such difficult to quantify. Our framework provides a method for eliciting this intuition using preferences and a way to quantify the intuition using utility theory. Once quantified, an experienced designer's intuition can be used both for self improvement and to aid novice designers whose intuition is less developed.

We are not the first researchers to notice the parallels between engineering design and decision theory. As early as 1991, Sykes and White [4] proposed a framework for incorporating designer preferences into an intelligent computer-aided design process using utility functions. Their approach is conceptually similar to ours but makes more restrictive assumptions in the utility model. Despite the potential value of the ideas in [4], few papers extended their framework. In 2001, the US National Research Council's Board on Manufacturing and Engineering Design produced a report [5]

*preverdy@princeton.edu, Ph.D. Candidate, D309 Engineering Quadrangle, Student Member, AIAA
†asreddy4@gmail.com, Currently Research Assistant at the Cleveland Clinic, Cleveland, OH 44106
‡gigi@phantom2.princeton.edu, Associate Professor, D302C Engineering Quadrangle, Associate Fellow, AIAA
§naomi@princeton.edu, Professor, D234 Engineering Quadrangle

that highlighted the role of decision making in engineering design and recommended that "constructive dialogue should be encouraged to allow the best aspects of each method and theory for decision making in engineering design to be incorporated into common practice." Our paper contributes to this dialogue by connecting the engineering design and decision theory literatures in the context of MDO, which is a well-established tool with a well-defined end user and research community. To make the connection concrete, we construct a system to solve MDO problems using the utility-based framework. The system uses designer preferences as feedback to iteratively converge on a solution.

The remainder of the paper is structured as follows. Section II lays out our specification of the MDO problem. Section III introduces the mathematical formulation of preferences, details the conditions under which they can be represented in terms of a utility function, and explains how to use the utility function in an MDO framework. Section IV presents an MDO system constructed using the utility-based framework. Section V presents the results of a simulation study where the utility-based MDO system was used to solve a test design problem and discusses how to apply the system to an industrially-relevant problem. Finally, Section VI offers conclusions and prospectives for future work.

II. The MDO problem

The MDO problem can be specified as follows. Let there exist n scalar-valued design variables $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in X \subseteq \mathbb{R}^n$. Let $\mathbf{q} : X \to Q \subseteq \mathbb{R}^m$ represent m scalar-valued objective functions of the design determined by the n design variables. Then $\mathbf{q}(\mathbf{x}) = (q_1(\mathbf{x}), q_2(\mathbf{x}), \ldots, q_m(\mathbf{x}))$ represents a performance assessment of the design $\mathbf{x}$. For example, $\mathbf{q}(\mathbf{x})$ may be computed using a numerical analysis of the design represented by $\mathbf{x}$. The designer's goal is to pick $\mathbf{x}$ so as to maximize the objectives $\mathbf{q}$:

$$\max_{\mathbf{x} \in X} \quad q_i(\mathbf{x}), \quad i = 1 \ldots, m. \tag{1}$$

The MDO problem (1) is not a well-defined optimization problem, since $\mathbf{q}$ is a vector. There is no single well-defined ordering of two points $\mathbf{q}, \mathbf{q}' \in Q$, and as such, no well-defined optimal point. However, one can define a partial order on Q as follows.

Definition 1. *For two points $\mathbf{q}, \mathbf{q}' \in Q$ and $\mathbf{q} \neq \mathbf{q}'$, point $\mathbf{q}$ is said to* dominate *$\mathbf{q}'$ if $q'_i \leq q_i$ for all $i = 1, \ldots, m$. That is, $\mathbf{q}$ is at least as good as $\mathbf{q}'$ in every individual objective. A point $\mathbf{q}$ is said to be* non-dominated[a] *if there is no other point that dominates it.*

The set of non-dominated points is known as the Pareto front, and consists of solutions where no single objective can be improved without worsening at least one other objective.

Computation of the Pareto front is a well-studied problem (see, e.g., [6]), so it is reasonable to assume that the Pareto front, or at least a local approximation, is known. All points on the Pareto front are efficient in the sense that they are optimal for some value of the tradeoff among objectives, but the solution of an engineering design exercise generally requires the selection of a single solution. Additional constraints are needed to isolate an optimal solution among the points on the Pareto front. In particular, the designer's preferences among tradeoffs, if they are known, can be used to define an optimal solution.

Two different common approaches to defining the optimal solution are the game-theoretic approach and the scalarization approach. In the game-theoretic approach [7], one considers the optimization problem as a game where each objective is represented by a player, each of which must bargain with the others to optimize its individual objective. The single optimum arises as the equilibrium of this bargaining process. In the scalarization approach [8, 9], one defines a scalar objective function $J : Q \to \mathbb{R}$ and then searches for the value of $\mathbf{x}$ that maximizes $J(\mathbf{q}(\mathbf{x}))$. Our framework takes the scalarization approach.

The standard scalarization technique defines $J(\mathbf{q}(\mathbf{x}))$ as the weighted sum

$$J(\mathbf{q}(\mathbf{x})) = \sum_{i=1}^{m} w_i q_i(\mathbf{x}), \tag{2}$$

where the weight $w_i \geq 0$ is a constant that represents the importance of objective i to the designer. However, the solution to this scalarized problem depends heavily on the choice of weights, the correct values of which are not generally clear. The use of constant weights in Equation (2) implies that the relative importance of each objective q_i is independent of its value. While such an assumption may be justified in a local region around a point $\mathbf{q} \in Q$, it is not valid in general. For example, consider a design that has a target value for some objective q_i. If q_i is far from its target value, it will be more heavily weighted than if it is close to the target.

Ultimately, it is the designer who uses his/her judgement to pick the optimal design. Under mild assumptions detailed in Section III, a scalar objective function can be used to represent the designer's judgement. We seek to learn

[a] Some authors call such a point Pareto-optimal.

the designer's objective function, since it encodes the designer's intuition. An experienced designer is likely to have good intuition that is useful in solving the problem. The functional form we use for $J(\mathbf{q})$ is a sum over objectives as in (2), but it relaxes the assumption of linearity and allows the relative importance of each objective to depend on its value.

III. Utility and preferences

For many years economists have studied decision-making processes with particular emphasis on theories of rational decision making, meaning decisions that reflect a consistent ranking of choice alternatives [10]. Since Samuelson [11], research has focused on formulating rational decision making in terms of maximizing a scalar utility function that can be inferred from preference information revealed by an individual's choice behavior. Our framework applies the revealed preference paradigm to the MDO problem by estimating the designer's utility function on objective space and using this as the scalarization function $J(\mathbf{q}(\mathbf{x}))$.

A. Preferences

For the MDO problem, preferences represent the designer's judgement. Let $\succ$ be a *preference relation* and $\sim$ be a *indifference relation* between elements in Q. We assume that given any pair of designs $\mathbf{x}, \mathbf{x}'$ with corresponding objective values $\mathbf{q}, \mathbf{q}'$, the designer's judgement will take the form of a *preference* consisting of a choice of exactly one of the following three options:

- $\mathbf{q} \succ \mathbf{q}'$: the designer prefers $\mathbf{q}$ to $\mathbf{q}'$,
- $\mathbf{q} \prec \mathbf{q}'$: the designer prefers $\mathbf{q}'$ to $\mathbf{q}$, or
- $\mathbf{q} \sim \mathbf{q}'$: the designer is indifferent between $\mathbf{q}$ and $\mathbf{q}'$.

The MCDA literature has studied the case of fuzzy preferences (i.e., where the designer has uncertainty as to which preference to express) and variable strength preferences (i.e., where some preferences may be stronger than others) [12], but these extensions are beyond the scope of this paper.

The question of how to encode preferences in a scalar function is the subject of utility theory [3]. In utility theory, two important properties of a preference relation are transitivity and completeness.

Definition 2. *A preference relation $\succ$ on a set Q is said to be* transitive *if, for every $\mathbf{q}, \mathbf{r}, \mathbf{s} \in Q$, $\mathbf{q} \succ \mathbf{r}$ and $\mathbf{r} \succ \mathbf{s}$ implies $\mathbf{q} \succ \mathbf{s}$.*

This is a basic consistency condition that is required for the preferences to rank the possible outcomes.

Definition 3. *A preference relation on a set Q is said to be* complete *if for each pair $\mathbf{q}, \mathbf{r} \in Q$, either $\mathbf{q} \succ \mathbf{r}, \mathbf{r} \succ \mathbf{q}$, or $\mathbf{q} \sim \mathbf{r}$.*

In other words, the preference set includes an opinion about each pair of outcomes. We assume that the designer's preferences are transitive and complete.

A fundamental result in utility theory due to Debreu [13] is the following.

Theorem 1. *If a preference relation on a set $Q \subseteq \mathbb{R}^m$ is complete and transitive, then there exists a real-valued utility function representation.*

By Theorem 1 there is a *utility function* $v : Q \to \mathbb{R}$ with the property that $v(\mathbf{q}) > v(\mathbf{r}) \Leftrightarrow \mathbf{q} \succ \mathbf{r}$ and $v(\mathbf{q}) = v(\mathbf{r}) \Leftrightarrow \mathbf{q} \sim \mathbf{r}$. A point $\mathbf{q}^*$ that maximizes the value of $v(\mathbf{q})$ is known as a *most preferred* point. In the context of the design problem, where $v(\mathbf{q})$ represents the designer's preferences, $\mathbf{q}^*$ is a most preferred solution for the designer, i.e., it corresponds to a design he/she would choose to build. For purposes of presentation, we refer to $\mathbf{q}^*$ as the most preferred solution. The goal of our approach is to find $\mathbf{q}^*$ by learning the designer's utility function.

B. Learning utility functions

We learn the designer's utility function by posing a series of queries in which the designer is required to express a preference between pairs of points $\mathbf{q}, \mathbf{q}'$ in objective space Q. We use the UTA (UTilités Additives) method, developed by Jacquet-Lagrèze and Siskos [12, Chapter 8] to estimate the utility function that represents the designer's preferences. UTA assumes that there are known bounds $\underline{q_i}$ and $\bar{q}_i$ on each objective value q_i such that $q_i \in [\underline{q_i}, \bar{q}_i]$, and that the

decision maker's preferences are monotonic in each q_i, i.e., that larger values of each criterion are weakly preferred to smaller ones. UTA then assumes that the utility function takes the additive form

$$v(\mathbf{q}) = \sum_{i=1}^{m} v_i(q_i) \tag{3}$$

subject to normalization constraints

$$\begin{cases} \sum_{i=1}^{m} v_i(\bar{q}_i) = 1 \\ v_i(\underline{q}_i) = 0, \ \forall i = 1, 2, \ldots, m, \end{cases} \tag{4}$$

where v_i are non-decreasing real-valued functions on Q, termed *marginal value functions*. Their monotonicity follows from the assumed monotonicity of the decision maker's preferences.

Sykes and White [4] employ a similar model, which in our notation takes the form

$$v(\mathbf{q}) = \sum_{i=1}^{m} w_i u_i(q_i),$$

where $u_i(q_i)$, $i \in \{1, \ldots, m\}$ are *value score functions* obeying $0 \leq u_i(q_i) \leq 1$, assumed known, and $w_i \geq 0$ is the weight on objective i, normalized such that $\sum_{i=1}^{m} w_i = 1$. The quantity $w_i u_i(q_i)$ is analogous to the marginal value function v_i. The major difference as compared to UTA is in the assumption that the value score functions $u_i(q_i)$ are known. Many different functional forms are possible; Sykes and White [4] suggest some examples and refer the reader to the decision sciences literature for further information.

UTA makes no assumption about the functional form of v_i and instead fits the marginal value functions using linear interpolation. By fitting the functions directly, UTA makes the method simpler to use by avoiding the additional steps Sykes and White's approach requires to pick a functional form. The UTA interpolation procedure is as follows. Let A be the set of points the designer has considered in Q. These are the points over which the designer has been queried and expressed a preference. For each alternative $\mathbf{a} \in A \subseteq Q$ define an estimate v' of the utility function v as

$$v'(\mathbf{a}) = \sum_{i=1}^{m} v_i(a_i) + \sigma(\mathbf{a}), \ \forall \mathbf{a} \in A, \tag{5}$$

where $\sigma(\mathbf{a})$ is an error relative to the true value function $v(\mathbf{a})$ and a_i is the i^{th} component of $\mathbf{a}$. Further, let each interval $[\underline{q}_i, \bar{q}_i]$ be divided into $\alpha_i - 1$ equal intervals with end points that we denote q_i^j for $j = 1, 2, \ldots, \alpha_i$. The value $v_i(a_i)$ is approximated by linear interpolation between the points $v_i(q_i^j)$, which UTA picks to minimize the error σ.

The set $A = \{\mathbf{a}_1, \ldots, \mathbf{a}_l\}$ is sorted according to the preference relation, so that $\mathbf{a}_1$ is the most preferred alternative and $\mathbf{a}_l$ is the least preferred one. Therefore, for each consecutive pair of alternatives $(\mathbf{a}_k, \mathbf{a}_{k+1})$, either $\mathbf{a}_k \succ \mathbf{a}_{k+1}$ ($\mathbf{a}_k$ is preferred) or $\mathbf{a}_k \sim \mathbf{a}_{k+1}$ (the designer is indifferent). Define the difference in their corresponding utilities with added error as

$$\Delta(\mathbf{a}_k, \mathbf{a}_{k+1}) = v'(\mathbf{a}_k) - v'(\mathbf{a}_{k+1}). \tag{6}$$

By the definition of the utility representation, one of the following holds

$$\begin{cases} \Delta(\mathbf{a}_k, \mathbf{a}_{k+1}) \geq \delta \ \Leftrightarrow \ \mathbf{a}_k \succ \mathbf{a}_{k+1} \\ \Delta(\mathbf{a}_k, \mathbf{a}_{k+1}) = 0 \ \Leftrightarrow \ \mathbf{a}_k \sim \mathbf{a}_{k+1}, \end{cases} \tag{7}$$

where δ is a small positive number representing a threshold for significant discrimination between two equivalence classes of the preference relation $\succ$.

The assumption that the marginal value functions $v_i(q_i)$ are monotonically increasing in q_i implies the following set of constraints on the values $v_i(q_i^j)$ at the interpolation points q_i^j

$$v(q_i^{j+1}) - v_i(q_i^j) \geq s_i \quad \forall j = 1, 2, \ldots, \alpha_i - 1, \ i = 1, 2, \ldots, n, \tag{8}$$

with $s_i \geq 0$ being indifference thresholds defined on each criterion q_i. Jacquet-Lagrèze and Siskos [14] set $s_i = 0$ and show that setting a positive threshold is not necessary, but can be useful in certain special circumstances.

UTA estimates the marginal value functions by solving the following linear program which minimizes the total error of the linear interpolation by choosing the function values at the interpolation points:

$$
\begin{aligned}
\min_{v_i(q_i^j)} \quad & \sum_{\mathbf{a}\in A} \sigma(\mathbf{a}) \\
\text{subject to} \quad & \left.\begin{array}{l} \Delta(\mathbf{a}_k, \mathbf{a}_{k+1}) \geq \delta \text{ if } \mathbf{a}_k \succ \mathbf{a}_{k+1} \\ \Delta(\mathbf{a}_k, \mathbf{a}_{k+1}) = 0 \text{ if } \mathbf{a}_k \sim \mathbf{a}_{k+1} \end{array}\right\} \quad \forall k \\
& v_i(q_i^{j+1}) - v_i(q_i^j) \geq 0 \qquad \forall i \text{ and } j \\
& \sum_{i=1}^{n} v_i(\bar{q}_i) = 1 \\
& v_i(\underline{q}_i) = 0,\ v_i(q_i^j) \geq 0,\ \sigma(\mathbf{a}) \geq 0 \quad \forall \mathbf{a} \in A,\ \forall i \text{ and } j.
\end{aligned}
\tag{9}
$$

Tunable parameters in this linear program include the utility discrimination threshold δ and the number of interpolation points α_i. To set up the linear program, UTA takes as inputs the set of upper and lower bounds $\{\underline{q}_i, \bar{q}_i\}$ on the objectives q_i and the set A of alternatives sorted according to the designer's preference relationship. The set A can be represented parsimoniously as a binary search tree.

C. Design Choice Hierarchy

As new alternatives are added to the set that the designer considers, we use a binary search tree to maintain the sorted set A of alternatives. We call the set A represented as a binary search tree the *design choice hierarchy*. Introducing a new alternative into the set A corresponds to inserting a new element into the tree. The insertion procedure starts at the root of the tree and compares the new element to the value there, moving on to the left or right subtree if the new element is smaller or larger, respectively, than the root, and recursively applying this procedure until it reaches an external node of the tree.

Each comparison operation corresponds to a query that must be posed to the human designer, who will reply with a preference. Responding to a query is tiring for the designer, so the number of queries required should be minimized. Binary search trees are known to be efficient data structures for inserting new elements: the average number of comparisons required to insert a new element in a tree of l elements is $O(\log l)$ (as compared to the worst-case performance $O(l)$). Using the design choice hierarchy structure is efficient with respect to the designer's cognitive load, and the insertion procedure ensures that the preferences encoded in the design choice hierarchy will be transitive, as required for the utility function representation.

IV. Utility-based MDO system

Figure 1 shows our proposed implementation of a utility-based MDO system based on the above elements. The system works interactively with the designer, incorporating his/her feedback in the form of preferences, to find the most preferred solution point $\mathbf{q}^*$ in objective space. The system sequentially presents the designer with pairs of objective vectors $\mathbf{q}, \mathbf{q}'$ and asks him/her to express a preference for one of the elements in the pair. At the beginning of an MDO problem, the system performs an initialization step where a set of points that cover objective space are used for queries. In the next step the system iteratively improves the estimate of $v(\mathbf{q})$ and $\mathbf{q}^*$ using the designer's preferences as feedback at each iteration. The idea is that the system uses a continuously improving estimate of $v(\mathbf{q})$ to guide the preference query process.

A. Initialization step

The system requires a set of preferences in order to estimate the utility function, which is the basis of the optimization process. Therefore, we begin solving the MDO problem by picking a set of points in objective space Q and constructing the design choice hierarchy over these points. The initialization step has the designer consider a variety of points in objective space Q, thereby exploring the full space. Such an exploration step is analogous to current industrial practice, where the initial step in a design optimization is often a large simulation study which computes a rough approximation of the design-objective map $\mathbf{q} : X \to Q$ at a set of points that cover the space X of design variables.

We consider both deterministic and stochastic methods to choose the set of initialization points. In either case, we choose a coordinate system for the objective space Q. To create a deterministic discretization, we pick a uniform grid of points in this coordinate system. To create a stochastic discretization, we pick the set of points according to a uniform distribution on Q. The extension to non-uniform discretization grids or distributions is straightforward and

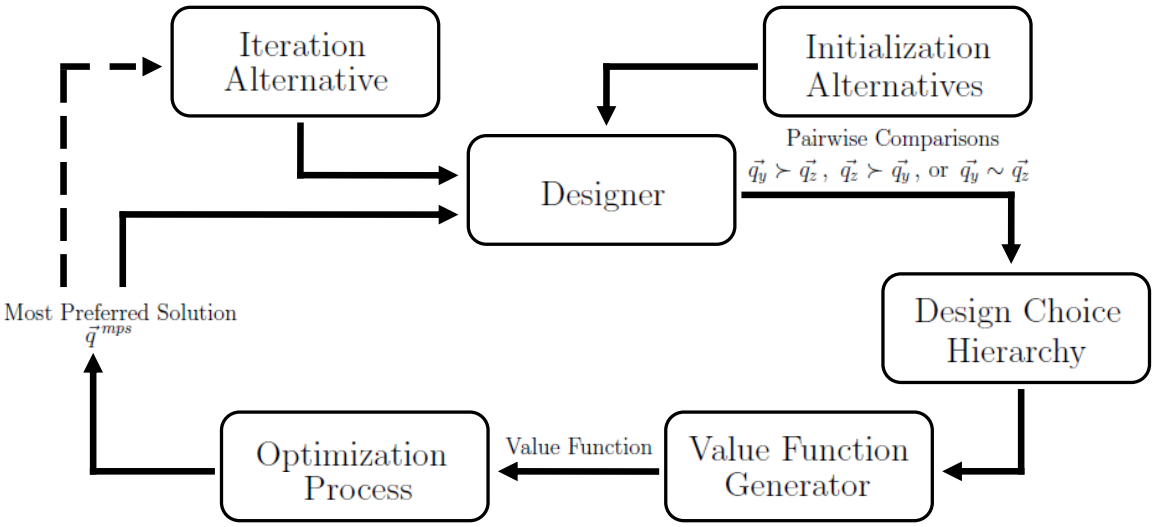


Figure 1. Schematic of utility-based MDO system design.

may be useful when one has prior knowledge about the location of interesting areas in objective space, but we do not pursue this extension here.

B. Utility function estimation

We construct the designer's choice hierarchy using a binary search tree structure as detailed above. Based on the design choice hierarchy with t alternatives, we produce the estimate $v^t(\mathbf{q})$ of the utility function $v(\mathbf{q})$ by solving the linear program (9) using Matlab's standard solver `linprog`. We use $\delta = 0.001$ and $\alpha_i = 26$, so we have $\alpha_i - 1 = 25$ subintervals for each objective i.

C. Generation of iteration alternatives

Upon producing $v^t(\mathbf{q})$, we use the value function to generate new alternative points to be considered by the designer. The initialization step carries out a general exploration of the objective space, so the iteration step can be used to focus on regions that the initialization step showed to be good. The optimum $\mathbf{q}_t^* = \arg\max_{\mathbf{q}} v^t(\mathbf{q})$ is the system's current estimate of the designer's most preferred solution $\mathbf{q}^*$. Thus, points nearby $\mathbf{q}_t^*$ are likely to be good and as such are targets for new iterations. We consider two methods for generating new alternatives:

1. Use the current optimum $\mathbf{q}_t^*$ as the next $(t+1)$ alternative.
2. Use both the current optimum $\mathbf{q}_t^*$ and a nearby point randomly chosen from the space within a radius r of $\mathbf{q}_t^*$ as the next two ($t+1$ and $t+2$) alternatives.

The optimum $\mathbf{q}_t^*$ is simple to compute because of the additive linear interpolation structure of $v(\mathbf{q})$. Method 1 can be described as a pure exploitation strategy, as it exploits the currently-known information about the utility function to recommend a new alternative. Method two can be described as a mixed exploration-exploitation strategy, since the randomly chosen alternative attempts to explore the space near the current optimum $\mathbf{q}_t^*$.

D. Termination condition

The system should eventually complete its execution and output a final estimate of the most preferred solution once a termination condition is reached. There are two natural candidates for the termination condition:

1. Terminate when the magnitude of the change in the estimated optimum $\mathbf{q}_t^*$ is below some threshold. This results in an estimate that has some guarantee of being close to the true optimum, but convergence may take an arbitrarily large number of iterations, i.e., designer time.

2. Terminate after a fixed maximum number of iterations or preference queries. Each iteration or query is costly in terms of computational and human cognitive resources, so this termination condition gives an upper bound on the total cost of the design exercise.

In many practical cases, the solution of the design exercise is cost or time constrained, so we will focus on the second termination condition.

V. Results

We use our system in simulation to solve an academic test problem and characterize the convergence performance in terms of two metrics. We show that the system converges to the optimal design, and we study ways in which our proposed iteration scheme affects the rate of convergence. To illustrate the relevance of our system to industrial problems, we describe how to apply the system to an industrially-relevant turbo-machinery test problem from [15].

A. Performance metrics

We measure the performance of our system using two metrics: one that measures local convergence to the true optimum design and one that measures global convergence throughout the objective space Q. As a local measure we consider the Euclidean distance $\|\mathbf{q}_t^* - \mathbf{q}^*\|$ between the current and the true optimum. This measure is equal to 0 when the system recommendation is equal to the true optimum, and bounded above by $\sqrt{2}$ in the academic test problem due to the size of the space Q. As a global measure, we consider Kendall's tau, τ, which compares two orderings of a given set A. We study the set of considered alternatives A and apply Kendall's tau to the orderings implied by the currently estimated utility function $v^t(\mathbf{q})$ and the true utility function $v(\mathbf{q})$. Kendall's tau is defined such that it takes value $+1$ if the two orderings are identical, -1 if one ordering is the reverse of the other, and 0 if the two orderings are uncorrelated. Therefore, $\tau = +1$ represents global convergence of $v^t(\mathbf{q})$ to $v(\mathbf{q})$.

B. An academic test problem

We tested our system in simulation with a simplified version of the DTLZ2 test problem described in [16], specified by

$$\max \begin{cases} q_1 = r\cos(\theta)\cos(\gamma) \\ q_2 = r\cos(\theta)\sin(\gamma) \\ q_3 = r\sin(\theta), \end{cases} \qquad \text{subject to} \begin{cases} 0 \le r \le 1 \\ 0 \le \theta \le \dfrac{\pi}{2} \\ 0 \le \gamma \le \dfrac{\pi}{2} \end{cases}. \tag{10}$$

The problem (10) is constructed such that the Pareto front is the surface of the unit sphere in the first octant: $\{\mathbf{q} = (q_1, q_2, q_3) | q_1, q_2, q_3 \ge 0,\ q_1^2 + q_2^2 + q_3^2 = 1\}$. We simulate a human designer by specifying the following utility function:

$$v(\mathbf{q}) = \frac{2cq_1}{0.15 + q_1} + \frac{3cq_2}{0.15 + q_2} + \frac{4cq_3}{0.15 + q_3}, \tag{11}$$

where $c \approx 0.1278$ is a normalization constant. The true most preferred solution for this utility function is

$$\mathbf{q}^* = \arg\max_{\mathbf{q}} v(\mathbf{q}) \approx (0.4948, 0.5797, 0.6473). \tag{12}$$

The simulated designer expresses preferences by computing the utility value for the alternatives in question and preferring the alternative with the higher utility.

We performed simulations using the simulated designer and measured the convergence of the system to the true most preferred solution (12), as defined by the Euclidean distance between the system's inferred value of $\mathbf{q}^*$ and its true value. As can be seen in Figure 2, the system converges as the number of initialization alternatives increases, but including feedback by also adding the current inferred value of $\mathbf{q}^*$ as an alternative increases the rate of convergence. This can be seen in Figure 3, which shows mean distance D between the current estimate of the optimum $\mathbf{q}_t^*$ and the true optimum $\mathbf{q}^*$. Figure 3a shows results for simulations using Scheme 1 and Figure 3b shows results for Scheme 2. The horizontal axes for both panels are comparable since they count the total number of alternatives considered in the process. By comparing the value of D for any given number of total alternatives considered, we see that Scheme 2 consistently converges more quickly than Scheme 1.

For both figures, the initialization used a set of points chosen from a uniform random distribution over the objective space Q. The initialization and iteration stages interact to affect the convergence properties of the system, as can be seen by considering the values of D along the two vertical dotted lines in each panel. These represent, for example, a

total budget of alternatives that can be considered in a given design exercise. A relevant issue for implementation is how to allocate alternatives between initialization and iteration since initialization ensures a good coverage of the space, but iteration is important to ensure that the the system converges on the optimum. For Scheme 1, an intermediate fraction of initialization is optimal, as can be seen from the dotted line at $N+m=50$: the curve with the smallest distance (i.e., error) is the one with $N=30$ initialization alternatives, rather than the ones with $N=10$ or 50. For Scheme 2, looking at $N+2m=50$ shows that more iteration is optimal, i.e., the curve with $N=10$ initialization alternatives yields the smallest error. This is likely because the second point added in a randomly-chosen location effectively serves as an additional mechanism to ensure good coverage of the space, replacing the need for an extensive initialization step.

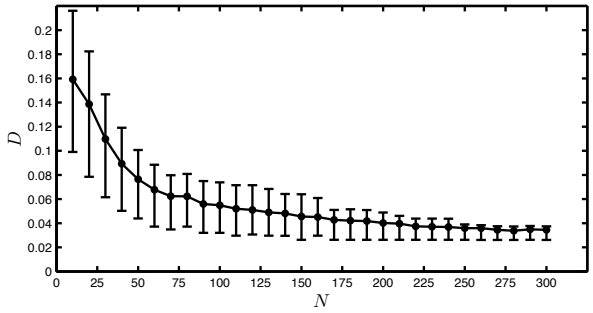
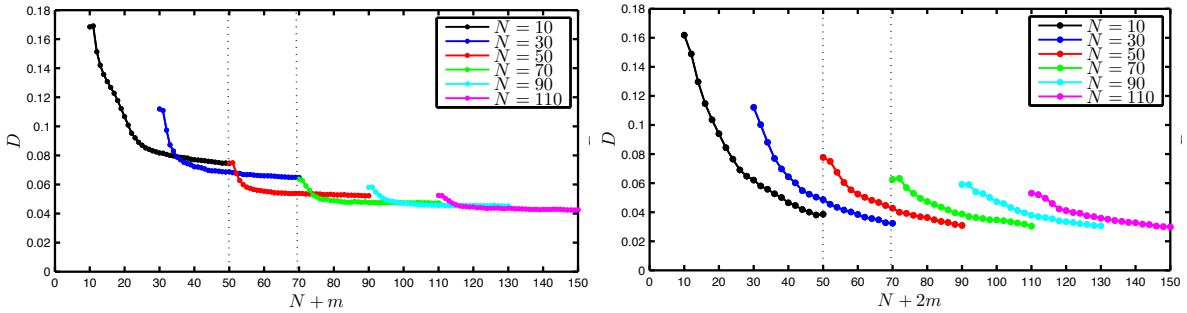


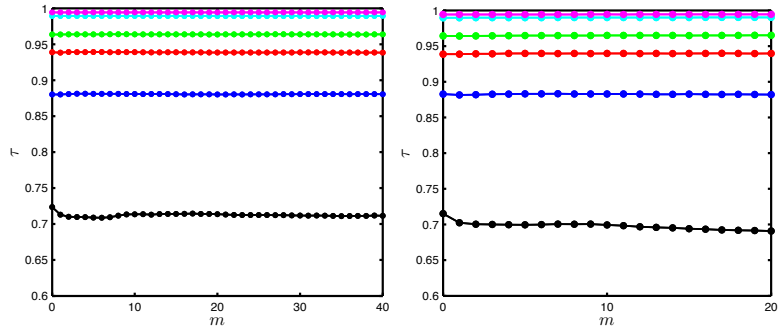
Figure 2. Euclidean distance D between the system's current estimate $\mathbf{q}_t^*$ of the most preferred solution and the true most preferred solution $\mathbf{q}^*$ as a function of the number of initialization alternatives N. The estimate converges to the true value as N increases. The data were generated from 500 simulations of the system and the error bars indicate the range of the middle 50% of the data.



a) Scheme 1: one point added per iteration. **b) Scheme 2: two points added per iteration.**

Figure 3. Euclidean distance D from the true most preferred solution as a function of the number of total alternatives considered, including initialization alternatives N and iteration alternatives m. The curves represent the mean value of D based on data from 500 simulations of the system. In panel a), the system was simulated using iteration Scheme 1, where the current estimate of the optimum $\mathbf{q}_t^*$ is used for the next alternative. In panel b), the system was simulated using iteration Scheme 2, which added both the current optimum $\mathbf{q}_t^*$ and a nearby point randomly chosen from the space within a radius $r=0.1$ of $\mathbf{q}_t^*$. Scheme 2 consistently converges more quickly than Scheme 1, as can be seen by comparing the value of D at any given number of total alternatives considered. For a fixed budget of alternatives that can be considered, the optimal allocation of designer time between initialization and iteration depends on iteration scheme, as can be seen by comparing where the different curves intersect the vertical dotted lines.

The convergence behavior due to the iteration process is largely local around the true optimum rather than global throughout the whole space Q, which can be seen by considering the global convergence measure τ, as plotted in Figure 4. Recall that τ is a measure of global convergence in the sense that it compares the rankings of the considered alternatives according to the inferred utility function $v'(\mathbf{q})$ and the true one $v(\mathbf{q})$. The measure is normalized to lie in $[-1,1]$, with $\tau=0$ corresponding to uncorrelated rankings and $\tau=1$ corresponding to identical rankings, i.e., convergence for the considered alternatives. Figure 4 shows τ as a function of the number m of iterations completed for the two schemes and the same numbers N of initialization alternatives as in Figure 3. In all cases, τ is essentially constant as a function of m, meaning that the the rankings of points away from the optimum do not converge as m increases. If all one cares about is finding the most preferred design $\mathbf{q}^*$, precisely measuring the value of $v(\mathbf{q})$ at points far from $\mathbf{q}^*$ is irrelevant, so this lack of global convergence is inconsequential. However, if the goal is to accurately measure the utility function throughout Q, the points should be chosen to cover the space.



a) Scheme 1. b) Scheme 2.

Figure 4. Kendall's tau, τ, between the rankings of all considered alternatives implied by the inferred utility function $v^I(\mathbf{q})$ and the true utility function $v(\mathbf{q})$ as a function of the number of iteration alternatives m. The curves represent the mean value of τ based on data from 500 simulations of the system for the different numbers of initialization alternatives N plotted in Figure 3. The larger values of N correspond to more complete coverage of the space and therefore better global convergence, which is reflected in the larger value of τ. However, iteration has little effect on global convergence, as seen by the flatness of the curves as a function of m. Iteration primarily causes local convergence to $\mathbf{q}^*$ in a neighborhood of the optimum.

C. An industrially-relevant test problem

In recent work, Shankaran and Vandeputte [15] studied a multidisciplinary turbo-machinery problem: the tradeoff between aerodynamics and mechanical integrity for a turbine blade. In this section, we pose their problem in our framework and show how our method can complement their game-theoretic approach to finding the optimal tradeoff.

The problem in [15] aims to maximize the mid-span aerodynamic efficiency of a turbine blade while ensuring that a mechanical frequency of interest is above a particular value. The problem begins with a baseline design that has acceptable aerodynamic efficiency, but has one torsional frequency that is close to resonance. There is a tradeoff between aerodynamic and structural performance, so finding an "optimal" solution is nontrivial. The authors of [15] computed an approximate Pareto front for their problem and considered several game-theoretic approaches for selecting an optimal point on the Pareto front. Our method provides an alternative approach for selecting the optimal point.

Figure 5 shows the Pareto front generated in solving the turbine blade problem as well as several potential optimal points representing different solution concepts. In the game theoretical approach, each objective q_i is represented by a player i, who participates in the game by picking a set of design variables $\mathbf{x}_i$. A solution consists of the equilibrium that emerges from the interaction of the various players. In the lower left corner of the figure is the Nash non-cooperative game solution **N**, which represents the disagreement point for the players and represents, in some sense, a worst-case solution. This solution is Pareto suboptimal, as it is inside the front. By cooperating, the players can bargain to pick a point that is efficient, i.e., Pareto optimal. Game theory shows that the efficient points of interest are in the cone with the solution **N** as the origin, so the relevant part of objective space is bounded by the cone and the Pareto front.

Solution concepts for cooperative games are generally formulated in terms of maximizing some scalar function g. For example, the Nash bargaining solution **NB** [17] maximizes the function $g = \Delta_1\Delta_2$, where Δ_i is the increase of objective q_i relative to the non-cooperative solution **N**. Another cooperative solution concept due to Kalai and Smordinsky [18] maximizes $g = \Delta_1/\Delta_2$, i.e., the ratio of the improvements. In [15], the authors introduced and studied the System Optimal Cooperative Solution (SOCS), which maximizes a scalar objective function measuring the overall performance of the system. In particular for their turbo-machinery problem, they sought to minimize specific fuel consumption (*sfc*), which they approximate as a linear function of the objective values q_i:

$$g(\mathbf{q}) = sfc = aq_1 + bq_2, \tag{13}$$

where a and b are constants that have previously been estimated by the designers. The resulting equilibrium solution emerges from optimizing $g(\mathbf{q})$ subject to the constraint that $\mathbf{q}$ lie on the Pareto front, and consists of the tangent point between the Pareto front and level curves of g.

The above problem can be posed in our framework (1) as follows: the $n = 5$ design variables $(x_1, x_2, x_3, x_4, x_5)$ are, namely, blade stagger angle, cross-sectional area, trailing edge feature, position of maximum thickness, and radius of the leading edge. Shankaran and Vandeputte note that the dominant design variables are stagger (x_1) and position of maximum thickness (x_4). The two objectives (q_1, q_2) are aerodynamic efficiency η and percentage change in the

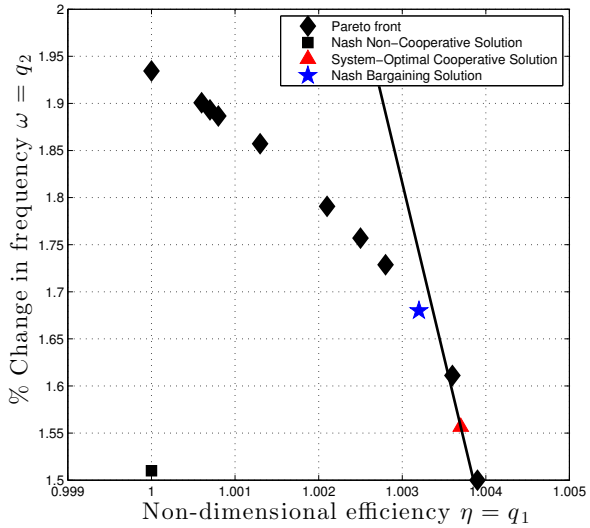


Figure 5. The Pareto front for the turbine blade problem together with three game-theoretically relevant points: the Nash non-cooperative solution N, which sets the disagreement point for the system, and the Nash Bargaining (NB) and System-Optimal Cooperative (SOCS) solutions. The Nash Bargaining and SOCS solutions emerge as tangent points between an objective function and the Pareto front. The straight line defines the objective function for the SOCS, which authors of [15] argue is the appropriate engineering solution to the game. Adapted from [15] with permission.

torsional frequency ω from its baseline value, so concretely we define

$$q_1 = \eta,\ q_2 = \omega.$$

Note that the frequency objective could also be encoded as a constraint on the allowable points in objective space, but we follow [15] and pose it as a second optimization objective. The intersection of the Pareto front and the cone defined by $\mathbf{N}$ provides the bounds $(q_i, \bar{q}_i), i = 1, 2$ required for our method.

The utility function $v(\mathbf{q})$ provided by our method constitutes an alternative to the objective function (13). The benefit of using $v(\mathbf{q})$ is that it 1) allows for a more general function than the linear form assumed by (13), and 2) it represents the human designer's intuition. The constants a and b in (13) (known as derivative information) are assumed known in [15], but must be estimated by performing additional studies. While there is a place for such studies, the resulting derivative information is likely to be somewhat arbitrary. Our framework would provide a rational method for integrating designer preferences and experience into the production of such derivative information, and therefore can complement existing approaches.

Figure 6 shows how we would apply our framework to the MDO workflow. The first two steps constitute formulating the MDO problem and computing the non-cooperative Nash equilibrium and Pareto front, as in the game-theoretic approach in [15]. In the third and final step, we would employ our framework to learn the designer's utility function $v(\mathbf{q})$ and find the most preferred solution $\mathbf{q}^*$ by posing a series of preference queries to the designer.

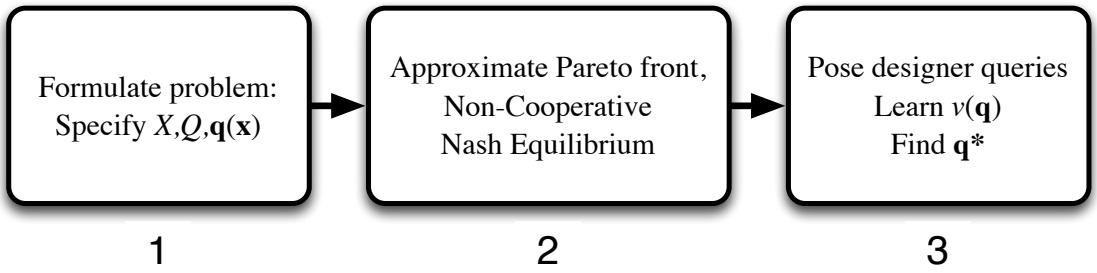


Figure 6. The MDO workflow modified to incorporate our utility-based framework. Steps 1 and 2 follow standard practice as detailed in [15] to formulate the MDO problem and find the Pareto front of efficient points. Step 3 is where we incorporate our utility-based framework to learn the designer's utility function $v(\mathbf{q})$ and find the designer's most preferred design $\mathbf{q}^*$, which is the "optimal" point.

In Section V.B we showed that our system works on a test problem, and we are seeking to apply it to an industrial problem. We would carry out a study by performing the first two steps in the workflow following current practice and then conducing step three in two stages: once with experienced designers to verify that the system recovers a baseline "optimal" design, and a second time with less experienced designers for comparison with the baseline. This would provide a way to compare the judgements made by different designers, both between individuals with a given level of experience and between groups with differing levels of experience. A further interesting question would be how the different designers' utility functions $v(\mathbf{q})$ compare with the SOCS objective (13). If the approaches using the utility function $v(\mathbf{q})$ and the SOCS objective $g(\mathbf{q})$ result in similar optima, this provides evidence that the SOCS objective is an accurate representation of the designers' decisions in the sense that both result in the same solution. A perhaps more interesting outcome would be for the two approaches to disagree on the optimal point, which would suggest that the derivative information alone is insufficient to represent the designers' decision process. Whether or not designers' decisions reflect derivative information is an empirical question and either outcome would be informative.

VI. Conclusions

A utility-based MDO framework provides a natural way to incorporate the human designer's preferences about performance tradeoffs in the context of multiple disciplines. We have designed a system based on such a framework that uses feedback from the designer to iteratively converge on the optimal design. We have shown through simulation that it converges to the correct solution in the context of a test problem. Further work remains to be done to test the framework in the context of an applied problem such as the turbo-machinery problem in Section V.C and to enhance the system's iteration scheme.

The implementation here works in objective space Q, assuming that the Pareto front is known. For several reasons, it may be preferable to work directly in the design variable space X. Computing the Pareto front may be costly, and ultimately the optimal design is parametrized in terms of the design variables, so it may be desirable to know $\mathbf{x}^* = \arg\max_{\mathbf{x}} v(\mathbf{q}(\mathbf{x}))$. Since the mapping from $\mathbf{x}$ to $\mathbf{q}$ may be computationally expensive and its inverse is not generally known in closed form, if $\mathbf{x}^*$ is required, it should be found directly. It may be desirable to include both the design variables and the objectives in the utility function, in which case the utility function should take the form $v = v(\mathbf{x}, \mathbf{q}(\mathbf{x}))$.

Further work should also be done to provide convergence guarantees and to find iteration schemes that converge at an optimal rate. One way to do so would be to cast the MDO problem as a best arm identification problem in the multi-armed bandit framework, as studied by [19]. In previous work [20], we have developed a framework for human-machine collaboration in the context of multi-armed bandit problems which could be extended to the MDO problem. Analysis using the multi-armed bandit framework would facilitate proving convergence guarantees, as well as providing bounds on the best possible convergence rate and developing iteration schemes that achieve these bounds. Such schemes would give an optimal strategy for searching the design space as a function of the available resources, e.g., simulation time, and as such would be of great practical importance.

Our utility-based MDO framework has the potential to rationalize current MDO practice. We have shown promising initial results and are beginning to apply our framework to industrially-relevant problems. We have a clear path towards a number of analytical results that would provide performance guarantees for the system, and we anticipate fruitful collaboration with industry going forward.

Acknowledgement

The authors would like to thank Sriram Shankaran of General Electric Global Research for helpful feedback and for providing data on the industrial test problem.

References

[1] Sobieszczanski-Sobieski, J. and Haftka, R., "Multidisciplinary aerospace design optimization: survey of recent developments," *Structural Optimization*, Vol. 14, 1997, pp. 1–23.

[2] Martins, J. R. R. A. and Lambe, A. B., "Multidisciplinary Design Optimization: A Survey of Architectures," *AIAA Journal*, Vol. 51, No. 9, 2013, pp. 2049–2075.

[3] Keeney, R. L. and Raiffa, H., *Decision analysis with multiple conflicting objectives*, Wiley & Sons, New York, 1976.

[4] Sykes, E. A. and White III, C. C., "Multiobjective intelligent computer-aided design," *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 21, No. 6, 1991, pp. 1498–1511.

[5] Board on Manufacturing and Engineering Design, *Theoretical Foundations for Decision Making in Engineering Design*, The National Academies Press, Washington, D.C., 2001.

[6] Das, I. and Dennis, J., "Normal-Boundary Intersection: A New Method for Generating the Pareto Surface in Nonlinear Multicriteria Optimization Problems," *SIAM Journal on Optimization*, Vol. 8, No. 3, 1998, pp. 631–657.

[7] Tang, Z., Désidéri, J.-A., and Périaux, J., "Multicriterion aerodynamic shape design optimization and inverse problems using control theory and Nash games," *Journal of Optimization Theory and Applications*, Vol. 135, No. 3, 2007, pp. 599–622.

[8] Pascoletti, A. and Serafini, P., "Scalarizing vector optimization problems," *Journal of Optimization Theory and Applications*, Vol. 42, No. 4, 1984, pp. 499–524.

[9] Eichfelder, G., "An adaptive scalarization method in multiobjective optimization," *SIAM Journal on Optimization*, Vol. 19, 2009, pp. 1694–1718.

[10] Grüne-Yanoff, T., "Paradoxes of Rational Choice Theory," *Handbook of Risk Theory*, edited by S. Roeser, R. Hillerbrand, P. Sandin, and M. Peterson, Springer Netherlands, 2012, pp. 499–516.

[11] Samuelson, P. A., "A Note on the Pure Theory of Consumer's Behaviour," *Economica*, Vol. 5, No. 17, 1938, pp. 61–71.

[12] Figueira, J., Greco, S., and Ehrgott, M., *Multiple criteria decision analysis: state of the art surveys*, Vol. 78, Springer, 2005.

[13] Debreu, G., "Representation of a Preference Ordering by a Numerical Function," *Decision Processes*, edited by R. M. Thrall, C. H. Coombs, and R. L. Davis, chap. 11, Wiley, 1954, pp. 159–165.

[14] Jacquet-Lagrèze, E. and Siskos, J., "Assessing a set of additive utility functions for multicriteria decision-making, the UTA method," *European Journal of Operational Research*, Vol. 10, No. 2, 1982, pp. 151–164.

[15] Shankaran, S. and Vandeputte, T., "Game-theoretic models for cooperative equilibrium solutions of interacting engineering sub-systems," *Proceedings of the ASME Turbo Expo*, 2014, pp. GT2014–25293.

[16] Deb, K., Thiele, L., Laumanns, M., and Zitzler, E., "Scalable multi-objective optimization test problems," *Proceedings of the Congress on Evolutionary Computation (CEC-2002)*, 2002, pp. 825–830.

[17] Nash, J., "The Bargaining Problem," *Econometrica*, Vol. 18, No. 2, 1950, pp. 155–162.

[18] Kalai, E. and Smorodinsky, M., "Other solutions to Nash's bargaining problem," *Econometrica*, Vol. 43, No. 3, 1975, pp. 513–518.

[19] Audibert, J.-Y., Bubeck, S., and Munos, R., "Best arm identification in multi-armed bandits," *COLT: 23rd Conference on Learning Theory*, 2010, pp. 41–53.

[20] Reverdy, P., Srivastava, V., and Leonard, N. E., "Modeling Human Decision-making in Generalized Gaussian Multi-armed Bandits," *Proceedings of the IEEE*, Vol. 102, No. 4, 2014, pp. 544–571.