Proceedings of the ASME 2017 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference
IDETC/CIE 2017
August 6-9, 2017, Cleveland, Ohio, USA

DETC2017-68337

DEVELOPING AND COMPARING ALTERNATIVE DESIGN OPTIMIZATION FORMULATIONS FOR A VIBRATION ABSORBER EXAMPLE

Siyao Luan
Deborah L. Thurston*
Decision Analysis Laboratory
University of Illinois at Urbana Champaign
Urbana, Illinois, 61801
Email: {luan1, thurston}@illinois.edu

Madhav Arora
James T. Allison
Engineering System Design Lab
University of Illinois at Urbana Champaign
Urbana, Illinois, 61801
Email: {marora6, jtalliso}@illinois.edu

ABSTRACT

In some cases, the level of effort required to formulate and solve an engineering design problem as a mathematical optimization problem is significant, and the potential improved design performance may not be worth the excessive effort. In this article we address the tradeoffs associated with formulation and modeling effort. Here we define three core elements (dimensions) of design formulations: design representation, comparison metrics, and predictive model. Each formulation dimension offers opportunities for the design engineer to balance the expected quality of the solution with the level of effort and time required to reach that solution. This paper demonstrates how using guidelines can be used to help create alternative formulations for the same underlying design problem, and then how the resulting solutions can be evaluated and compared. Using a vibration absorber design example, the guidelines are enumerated, explained, and used to compose six alternative optimization formulations, featuring different objective functions, decision variables, and constraints. The six alternative optimization formulations are subsequently solved, and their scores reflecting their complexity, computational time, and solution quality are quantified and compared. The results illustrate the unavoidable tradeoffs among these three attributes. The best formulation depends on the set of tradeoffs that are best in that situation.

*Address all correspondence to this author.

1 INTRODUCTION

Cagan et al. define the engineering design procedure as the following four iterative steps: definition of search space, problem formulation, solution, and verification and critique [1]. Many researchers focus on developing rigorous methods for the third "solution" step, yet we should keep in mind that the second "formulation" step deserves significant attention. Design problem formulation can be done normatively or descriptively. However, design can never be reduced to a prescriptive procedure exclusive of human input [2]: determination of which relationships to include in analytical models, assessment of the inputs to the models, and determination of an appropriate value measure are the three formulation tasks where human input will always be needed. Design problem formulation not only has its own intrinsic complexities and challenges, but also affects the level of solution difficulty and quality.

A number of design problem formulation studies have been presented in the literature. Cramer et al. identified several problem formulations for multidisciplinary optimization, and their research focuses on how the optimization problem in the solution phase should be posed [3]. Fuge et al. conducted an extensive study in how experienced designers choose design methods, and they propose machine learning algorithms for design method recommendations [4]. This paper addresses a different question: for the same underlying design problem, how can different formulations be developed, and how good are these alternatives compared to each other?

Volkema acknowledges that a design problem statement may lead to a wide range of problem formulations, which then affect the direction of succeeding states [5]. This illustrates how formulation decisions can impact design outcomes, including solution quality, but does not account for alternative formulations for the same problem. As a result, the method due to Volkema is unable to compare different formulations.

Danielescu et al. identify the importance of problem formulation, and point out that studies of the relationship between problem formulation and creative outcomes is lacking [6]. Their problem map framework does not address why the design problem formulation is finalized in a particular form (expressed in mathematical language). In addition, creativity is the only design output property assessed.

Watton and Rinderle identify the benefits of simplifying mechanical design equations to reformulate a design problem [7]. They also discuss briefly the cognitive benefits of these simplifications for designers. However, their work does not address the tradeoffs between the simplification and its potential negative effects.

Similarly, Pomrehn and Papalambros explain explicitly how a design problem can be formulated differently in terms of discreteness in variables [8]. The only metric the authors include in the design outcome evaluation is solution quality. Tradeoffs between solution quality and other metrics are not discussed.

Chapman and Jakiela described the benefits of topology simplification—which is essentially the reduction of features being included in a system model—but do not present a quantitative comparison between various candidate simplifications [9]. These different options are likely to have different effects on several measures of the design solution.

Tailandier and Gaffuri describe how to determine an objective function from quantitative analysis of user preferences [10]. However, it is difficult to compare the results due to the lack of other formulation considerations.

In this paper, we propose an initial formalized strategy to compare and rank alternative design problem formulations for the same underlying design problem. First, Section 2 divides the design problem formulation process into three tasks: comparison metrics, design representation, and the predictive model. Next, Section 3 presents guidelines for completing the formulation process. Section 4 identifies three aspects that are important for evaluating and comparing alternative formulations: the complexity of the design formulation process, solution time required to solve the problem as formulated, and the final quality of the design result. We explain the proposed method by providing an example of implementing this method, detailed in Sections 5–7. Discussion, conclusions, and topics for future work are presented in Section 8.

2 DESIGN PROBLEM FORMULATION

Given a design project, engineers define design objectives and identify specific metrics on which different design alternatives can be explicitly compared against each other. We call this task "comparison metrics formulation." Usually, design engineers formulate comparison metrics by finalizing the objective function of the design problem, unambiguously expressing design objectives using mathematical language.

Design engineers also determine what elements of the system are to be changed during design exploration. Automated exploration using optimization requires an explicit mathematical representation, such as a vector of discrete or continuous variables (design decision variables), that the optimization algorithm can operate on. Often many options exist for design parameterization. Deciding on an appropriate design parameterization is referred to here as the "design representation formulation" task. This is sometimes referred to as "framing". Choosing a design representation in essence defines the design search space, and can have significant impact on the design process and outputs. Some elements of a design representation may be held fixed as constants.

To compare the desirability of distinct designs, we need the ability to predict values for comparison metrics for a given design representation. Creating a mapping between design representation and comparison metrics is the "predictive model formulation" task. A design-appropriate model is sought that can predict accurately the effects of design changes on performance. Typically many options exist for predictive models, and in general exhibit a tradeoff between model fidelity (accuracy with respect to real system behavior) and computational efficiency. A high-fidelity model may support finding designs that perform better in actual implementation, but such models typically involve high-dimension numerical problems that require significant development effort and computational solution cost.

It is important to note that in each of the above three tasks, heuristics are utilized extensively. No formal principle governs which set of arithmetic variables a design objective defines, how many properties are sufficient to represent a design, or how capable a model should be of predicting output accurately. For example, deciding how to model the manufacturing process is an important problem that requires significant amounts expert opinion, and is not straightforward. Efforts are being made to centralize this type of information and making it broadly available (e.g., for manufacturing process information [11]), but judgment must still be employed when defining model formulations. Heuristic methods are often employed. For example, the problem of significant modeling efforts and computational complexity is addressed specifically for manufacturing plant reliability problems [12], using reliability bounds and linear programming to more quickly analyze manufacturing plant reliability than would be possible using simulation.

The three aforementioned tasks form a space, which we call

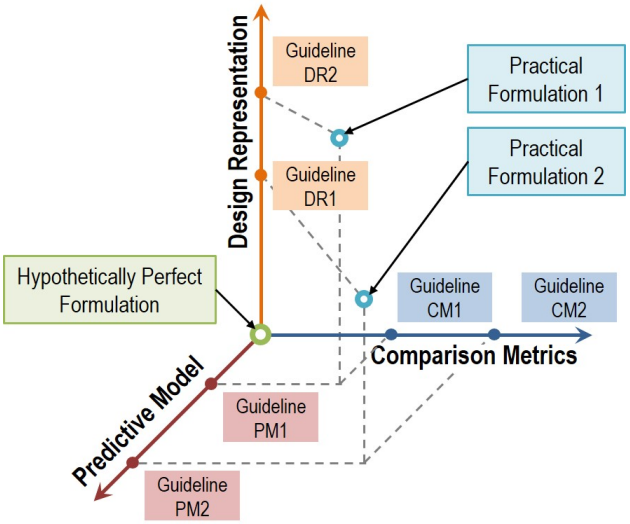

FIGURE 1. DESIGN FORMULATION SPACE WITH TWO ILLUSTRATIVE "PRACTICAL FORMULATIONS" FORMED OUT OF SIX FORMULATION GUIDELINES

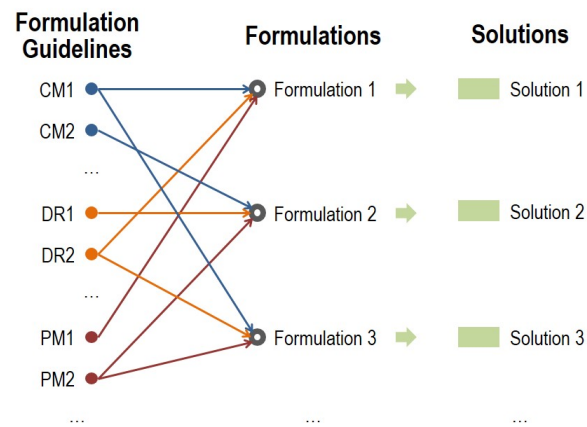

FIGURE 2. RELATIONSHIPS AMONG GUIDELINES, FORMULATIONS AND SOLUTIONS

"design formulation space", as shown in Fig. 1. The three dimensions of this space are comparison metrics, design representation, and predictive model. The origin of the space is an impeccably "perfect" formulation of a design problem that reflects all possible factors (i.e., the substantive rationality formulation). Due to the fact that there does not exist universally acknowledged formal principles (refer to [13] for the definition of "principle") for design formulation, we believe that a "perfect formulation" only exists hypothetically. It is expected that formulations that move closer to the origin in this space improve design outcomes, but often will involve significant solution expense. Desirable advancements in design methodology help move closer to the substantive rationality formulation (and solution), while avoiding excessive computational expense.

3 FORMULATION GUIDELINES

It should be intuitive that the closer a formulation is to the hypothetically perfect formulation, the more sophisticated it is, and the less likely it can be achieved. Accordingly, we introduce the concept of "formulation guidelines", through which designers can create "practical formulations" that yield manageable design problem solution. In other words, "formulation guidelines" help design engineers move away from the hypothetical perfect formulation to enable solution. Sometimes, those guidelines are referred to as "simplifying assumptions" when they clearly reduce the complexity of a formulation.

As shown in Fig. 1, CM1 and CM2 are two guidelines that definitively specify comparison metrics for the design problem; CM1, closer to the hypothetical perfect formulation, ostensibly reflects the true design objective more accurately, but CM2 may be more straightforward and thus easier to handle. Note that different guidelines on the same dimension sometimes only formulate the same problem from different perspectives, and their relative complexity may not be obvious. Also note that any Guideline in Fig. 1 may be a composite of a set of more specific guidelines on the same dimension.

Since the formulation space has three dimensions, a combination of guidelines across all three dimensions can be used to generate a practical formulation. For an example, in Figs. 1 and 2, Guidelines CM1, DR2, and PM1 generate Formulation 1, which then yields Solution 1. One guideline can be part of many formulations. The large number of combinations generates a large number of alternative formulations which, presumably, can accommodate any preference in complexity, computational time and solution quality.

It should be noted that, according to Hazelrigg in [2], problem formulation is not a process exclusive of human input, and the guidelines are a source of human input, because as Fu et al. describe in Ref. [13], guidelines are based on extensive experience and/or empirical evidence. We arrive at the statement that guidelines are created through the repetitive process of formulating similar design problems.

It should also be noted that one guideline usually only provides process direction for a very specific (and common) sub-task during the formulation process, especially for design representation and predictive model tasks. Hence, even for very complex design problems, the list of guidelines is comprised of fairly basic guidelines that are generally applicable to a wide variety formulation tasks. The remainder of this paper discusses the effects of applying different guidelines, and the degree to which the resulting formulations meet expectations.

4 EVALUATION OF FORMULATIONS

Here we explain the motivation for choosing formulation complexity, solution effort, and solution quality as the formulation evaluation criteria. Formulation complexity determines directly the cognitive load level that design engineers must sustain when formulating the problem. The greater the cognitive load, the more difficult and time consuming is the process, and the greater level of expertise required of the designer.

Solution effort reflects the amount of stress the formulation exerts to whoever (or whatever) undertakes the solution process. Computational expense can be used as a metric. We acknowledge that, in addition to formulation decisions, solution effort is affected by algorithmic solution strategy choice. We consider solution strategy to be separate from formulation, but acknowledge the tight interaction between formulation and solution decisions. For example, a simplified formulation may support faster solution strategies (e.g., continuous, smooth functions supporting gradient-based methods), but may not necessarily result in designs that perform well in real implementation. Conversely, if an engineer has a particular solution method in mind, it may motivate particular formulation decisions. For example, using a genetic algorithm for solution often is aided by design representation dimension reduction.

Design quality is reflected by how well the merits of the design solution align with the motivating design objectives. If a design formulation is such that it sensibly conveys the design objectives and reasonably depicts the pertinent properties and relationships, then the design solution should address the design objectives well. In other words, a high-quality design performs its intended overal function(s) well in actual implementation.

It should be noted that the above three objectives may be competing. In this case an intrinsic tradeoff occurs when one wants to improve any one factor by changing relevant formulation guidelines. As a strategy to pursue consistency when making formulation decisions, we later illustrate a method for evaluating these tradeoffs in problem formulation alternatives.

5 EXAMPLE: VIBRATION ABSORBING SYSTEM

5.1 Physical Layout and Application

The proposed formulation comparison method is illustrated here by walking through the design of a simple vibration-absorbing system for a sprung mass subject to a disturbance force. The use of symbols is summarized in Table 1. Figure 3 depicts the physical layout of the dynamic system. A mass, referred to as the primary mass m_1, is linked to the ground by a linear primary spring k_1. $F(t)$ is a disturbance force applied vertically to m_1. The primary mass is kinematically constrained to move only in the vertical direction. The vertical displacement of m_1 is $z_1(t)$. For a given disturbance $F(t)$, we would like to reduce primary mass vibration. The primary spring and mass comprise the primary mass-spring system. Here we assume that $m_1 = 1$ kg and $k_1 = 100$ N/m.

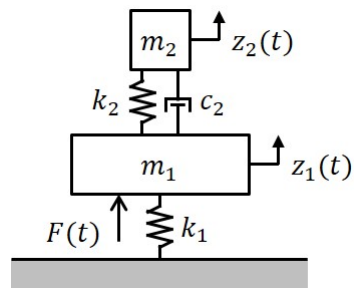

FIGURE 3. SCHEMATIC OF THE DYNAMIC SYSTEM

This mass-spring system can loosely represent a variety of design optimization problems. It can model a machining process platform, such as those employed for a lathe or a mill. In this case the platform is represented by m_1, and the legs supporting the platform correspond to k_1. The force $F(t)$ represents the vibrations induced by machining operations.

5.2 Objectives and Constraints

If we assume this is a machining system, then an important technical performance objective is to improve surface quality of the workpiece. Many factors contribute to surface quality, including material properties, machining process parameters, and system dynamic behavior. A comprehensive formulation would account for all of these and other factors. Here we explore a range of formulations with a range of simplifications. To bound scope, we focus on reducing primary mass vibration, which impacts surface quality. Here we examine design options that include a secondary mass-spring-damper system (m_2, k_2, c_2) that passively vibrates in a way that reduces primary mass vibration. A more sophisticated vibration absorber system could include an active force element between the primary and secondary masses, but is not considered here. These simplifying assumptions push possible formulations away from the origin in Fig. 1, but ease solution difficulty.

The motion of the secondary mass m_2 for the vibration absorber is restricted kinematically to vertical displacements $z_2(t)$. The equations of motion are derived by analytically applying Newton's Second Law, Hooke's Law for elastic springs, and assuming that the damper force is proportional to the relative velocity between the masses. The equations of motion based on these assumptions are:

$$\begin{aligned} m_1\ddot{z}_1 &= k_2(z_2 - z_1) + c_2(\dot{z}_2 - \dot{z}_1) + F - k_1 z_1 \\ m_2\ddot{z}_2 &= -k_2(z_2 - z_1) - c_2(\dot{z}_2 - \dot{z}_1), \end{aligned} \tag{1}$$

where z_1, z_2, and their time derivatives are functions of time. The design variables for this initial formulation are m_2, k_2, and

c_2. In the simplest case k_2 may be constant, but we can also consider a nonlinear spring where k_2 is a function of deflection to produce more sophisticated dynamic behavior. The damping coefficient c_2 may be constant or a function of frequency. $F(t)$ is an input. Here we simulate the system across a time horizon of five seconds: $t_0 = 0$, $t_f = 5$. The initial condition is $z_1(0) = z_2(0) = 5$ and $\dot{z}_1(0) = \dot{z}_2(0) = 0$.

The design of the passive vibration absorber is subject to a set of constraints, including bounds on m_2, k_2, and c_2 related to material properties, manufacturability, geometric dimensions, cost, and other factors.

6 EXAMPLE: ALTERNATIVE OPTIMIZATION FORMULATIONS

In this section we discuss the substantive rationality formulation for the vibration absorber problem, as well as formulation guidelines and resulting practical formulations.

6.1 "Perfect" Formulation

The primary purpose of designing the vibration absorber is to reduce the vibration of m_1 in a way that optimizes workpiece surface quality. The ideal objective function would aim at minimizing the "detrimental impact" of m_1's vibration across all anticipated operating conditions. Suppose that the primary mass m_1 is a platform carrying a lathe. The vibration of m_1 will ultimately lead to unsatisfactory surface quality of the finished part. If the surface quality of the finished part is measurable, then it explicitly reflects the aforementioned "detrimental impact" caused by primary mass vibration and can be used as the "perfect" objective function.

Unfortunately, estimating actual surface quality of machined components is a difficult task. Accurate mathematical models for predicting surface quality are often developed empirically, for example by conducting carefully designed experiments and then analyzing the results using response surface methodology and analysis of variance [14]. This can be a very time-consuming and manufacturing site-specific process (significantly increasing solution effort).

Even with a nearly perfect objective function, challenges still prevail in formulating the design representation and the predictive model. An accurate model must be comprehensive, nonlinear, and high-fidelity model with the following properties:

Comprehensive. This model includes all important system elements, including dynamic coupling between the motor and all the other elements.

Nonlinear. Dependence on frequency, amplitude or any other state will require nonlinear simulation. Inman identifies six important differences between linear and nonlinear systems, and explains how the latter are very complex [15].

High-fidelity. This model is accurate with respect to reality in ways that impact design decisions.

While the above discussion ostensibly characterizes a "perfect" formulation, it is impractical to utilize for design problem solution. Tradeoffs must be made across formulation complexity, solution effort, and solution quality. Design engineers must simplify the formulation to an extent to render solution practical with available resources, while producing acceptable design solution quality.

6.2 Formulation Guidelines

We generated a set of formulation guidelines for the vibration-absorber system design problem. First, these guidelines are described briefly, and then discussed in detail in the subsequent paragraphs.

Guideline 0.1 No dynamic coupling and design coupling. The designs of the secondary system do not alter the properties of the primary system or the input disturbance force F(t).

Guideline 0.2 Given disturbance force expression. F(t) is defined by

$$F(t) = 5\sin(\omega(t)t) \text{where} \omega = \begin{cases} 20t, \text{ if } 0 \leq t \leq 1 \\ 20, \text{ if } t > 1 \end{cases} \tag{2}$$

Guideline 0.3 Restrained framing. The objective of this design is only about maximizing vibration-absorbing performance and satisfying constraints. No other attributes such as cost are considered.

Guideline 1.1 Use maximum $|z_1(t)|$ for $t \geq 4s$ as an objective function to minimize.

Guideline 1.2 Use sum of $|z_1(t)|$ for $0 \leq t \leq 5s$ as objective function to minimize.

Guideline 1.3 Use sum of maximum $|z_1(t)|$ for $t \geq 4s$ for $\omega = 2, 6, 10, 14$, and 18 rad/s as an objective function to minimize.

Guideline 2.1 Include m_2 as a design variable.

Guideline 2.2 Include c_2 as a design variable.

Guideline 2.3 Include k_2 as a design variable.

Guideline 3 Design a viscoelastic (VE) damper instead of a simple linear damper

Simplification 3-1 Assuming relaxation kernel $K(t) = K_1 e^{-K_2 t}$ to simplify viscoelastic damper modeling and design; including K_1 and K_2 as design variables

Guideline 4 Optimize nonlinear spring behavior assuming existing models for telescoping conical springs.

Simplification 4-1 For nonlinear helical compression spring design, assume d, Na, and G are known, and that D_1, D_2, and p are design variables.

Guideline 5 Assume monotonic cubic spline nonlinear spring curve $F(\delta)$ with six control points.

TABLE 1. NOMENCLATURE

Symbol	Definition	Unit	Symbol	Definition	Unit	Symbol	Definition	Unit
$F(t)$	Sinusoidal disturbance force	N	ω	$F(t)$ frequency	rad/s	K_2	$K(t)$ parameter 2	s
$F(\delta)$	Force-deflection curve	N	t	Time	s	d	Conical spring wire diameter	m
$K(t)$	VE damper relaxation kernel	N/m	m_2	Secondary mass mass	kg	D_1	Conical spring min. helix diameter	m
X_D	Design variables		k_2	Secondary spring spring constant	N/m	D_2	Conical spring max. helix diameter	m
$L(X_D)$	Objective function		c_2	Damper damping coefficient	Ns/m	p	Conical spring pitch	m
m_1	Primary mass mass	kg	z_1	Primary mass displacement	mm	Na	Conical spring initial No. of active coils	1
k_1	Primary spring spring constant	N/m	z_2	Secondary mass displacement	mm	G	Shear modulus of elasticity	Pa
A	$F(t)$ amplitude	N	K_1	$K(t)$ parameter 1	N/m	cp_i	$F(\delta)$ spline parameter	N/m

Guideline 6 Apply target matching strategy.

Guideline 7 Using a simple sum of errors loss function in the target matching strategy.

Series 0 guidelines are upper-level, "umbrella" guidelines for this problem, which set an easier starting point for the application of more specific guidelines (they are applied across every formulation considered here). Series 1 guidelines are for objective functions (the "comparison metrics" dimension). Series 2 guidelines and Guidelines 3–5 involve the "design representation" dimension. Guidelines 6 and 7 impact both the "comparison metrics" and "design representation" dimensions.

All the guidelines used in this example are well-established guidelines and commonly found in engineering practice: for example, Guideline 0.1 is often used to decrease nonlinearity, and Guideline 6 is often used to reframe design problems when appropriate. Some of the guidelines are obviously case-specific: for example, Guideline 3 is irrelevant in design projects that do not involve damping elements.

6.2.1 Comparison Metrics Formulation Guidelines As discussed in Section 6.1, using surface quality as the objective function is ideal yet often impractical. This subsection describes three different proxy objective functions employed to reduce cognitive and solution effort while achieving a satisficing solution.

Using Maximum Displacement as Proxy

In Guideline 1.1, we minimize the maximum primary mass displacement magnitude across a four-second time horizon:

$$\min \quad L_1(X_D) = \max(|z_1(t)|) \quad \text{for} \quad t \geq 4s \tag{3}$$

This guideline terminates the formulation effort at the point where the $z_1(t)$ is obtained, ignoring how $z_1(t)$, together with a group of other factors, impact machined surface quality. Different norms of $z_1(t)$ could be used to approximate improvement in surface quality, assuming that $z_1(t) = 0$ is ideal, but recognizing potential error due to metric misalignment, i.e., that reduction of a particular norm of $z_1(t)$ does not always improve surface quality. In other words, a risk in using this guideline is that a design achieving minimum $|z_1(t)|_max$ does not necessarily minimize detrimental impact: a design may yield a relatively low global $z_1(t)$ peak, but still has a large number of high local peaks that result in overall poor surface quality.

Using Sum of Displacement as Proxy

In Guideline 1.2, we minimize the sum of primary mass displacement magnitude over the entire time span of interest, assuming a particular time discretization. This is similar to integration over time. This objective is defined as:

$$\min \quad L_2(X_D) = \sum_{n=1}^{N} |z_1(t_i)|, \text{ where} \quad t = 0, \ldots, 5\text{sec}, \tag{4}$$

and where N is the number of discrete time points used to approximate the time horizon $0 \leq t \leq 5s$. This guideline has a similar risk: minimum $L_2(X_D)$ does not necessarily lead to minimum detrimental impact. A full assessment of metrics such as this would require comparison to actual surface quality across a range of design problem conditions.

Using Transmissibility as Proxy

The above formulations lie in the scope of time-domain simulation. An alternative angle is to look at the impact in the frequency domain and define the objective as minimizing transmissibility.

Transmissibility quantifies the ratio of disturbance and output values as a function of input frequency. While the previous

guidelines only deal with response to the given disturbance frequency ω, these guidelines considers a wider frequency range.

In Guideline 1.3, we minimize the sum of the maximum primary mass displacement magnitude for five different frequencies. The objective function is defined as:

$$\begin{aligned} \min \quad & L_3(X_D) = \sum_{n=1}^{5} max|z_1(t)|_{\omega=omega_i} \\ \text{where} \quad & t \geq 4s,\ \omega_i = 2,6,10,14,18\text{rad/s} \end{aligned} \tag{5}$$

Note that $L_3(X_D)$ does not weight the importance of different ω values. When translating to surface quality, some frequencies may be more important than others (e.g., those near typical operating speeds). An improved version of this formulation may include weights on different frequencies, and calibrated using surface quality data.

6.2.2 Predictive Model Formulation Guidelines

Again, as discussed in Section 6.1, a high-fidelity model that predicts surface quality accurately as a function of design changes may require excessive development effort. Here we discuss model simplifications based on Guidelines 0.1 and 0.2.

Using a Linear Model and Excluding Disturbance Source

If we are to design a damper with a viscoelastic material (VEM), an accurate model involves solution of integro-differential equations (IDEs), which is computationaly expensive. If we assume that the VEM can instead be modeled using a generalized Maxwell model, the resulting equations of motion is a system of linear ordinary differential equations (ODEs), which are much easier to solve. The generalized Maxwell model makes an assumption about how the VEM relaxation kernel may be parameterized, instead of assuming that the relaxation function can take on arbitrary shapes.

Additionally, in the dynamic model, instead of modeling the motor, which is the disturbance source, we can treat the external disturbance force $F(t)$ as given, which may be obtained from experiment or simulation. The dynamic coupling is therefore eliminated between the system components and the disturbance source. This is a potential source of error if the structural dynamic response influences motor behavior.

Using the linear generalized Maxwell model and excluding disturbance source in the dynamic model allows us to perform a much less expensive linear simulation to obtain $z_1(t)$. This assumption also eliminates coupling between the primary/secondary system components and the disturbance source. A result of this is that $F(t)$ is independent of design. We treat these simplifications as default for all formulations presented here.

6.2.3 Design Representation Formulation Guidelines

Here we describe how to determine what aspects of the underlying design problem are to be included in the design formulation, as well as a strategy for reframing the problem.

Designing a Viscoelastic Damper

A standard model of a linear viscous damper has a constant damping coefficient. A vibration absorber with a linear spring and damper is designed properly, it can eliminate vibrations very well for a single specified disturbance frequency. If the input frequency changes, however, vibration amplitude can increase significantly (i.e., the system is not robust to changes in frequency). One possible strategy to improve robustness to frequency changes is to utilize a damper with a frequency-dependent VEM [16]. This is important because the drive motor will pass through a range of frequencies upon startup, even if it normally operates at a fixed speed.

While Guideline 2.2 assumes that we are to design a viscous damper with constant c_2, Guideline 3 prompts us to design a viscoelastic damper. The key to designing a viscoelastic damper is to determine its relaxation kernel $K(t)$, from which we can deduce the response of this damper under different frequencies. $K(t)$ must be monotonically decreasing to make physical sense, so we further assume that $K(t)$ has this property. If we assume that $K(t)$ can be approximated well using an exponential function, we arrive at the single-mode Maxwell model:

$$K(t) = K_1 e^{-K_2 t} \tag{6}$$

The design variables for the viscoelastic damper are K_1 and K_2, as described by Simplification 3-1. This simplification also indicates that our effort terminates at obtaining $K(t)$. More specifically, we have a simplified two-parameter design representation for the VEM, and we are designing the VEM properties directly instead of exploring new material formulations that could achieve the desired relaxation function. This simplification may result in $K(t)$ designs that are not physically realizable if they are not constrained to map back to actual material formulations. The above single-mode Maxwell model can be extended to a multi-mode Maxwell model using a summation of exponential functions, each function involving two design variables. This increases design flexibility (the range of achievable $K(t)$ shapes), but also the design representation dimension (which usually impacts solution effort).

Designing a Nonlinear Spring

Apart from designing a viscoelastic damper, we can also expand the design problem to include a nonlinear spring to further improve the vibration absorbing performance and robustness to frequency changes.

In Guideline 2.3, we design a linear spring, which has a constant k_2 (elastic force varies linearly with deflection). A nonlinear spring has a stiffness parameter that varies with deflection, and is

therefore state-dependent. This nonlinear behavior can produce distinctly different absorber performance, but involves increased computational effort to simulate.

In modeling system response with a nonlinear spring, we consider the following mapping as shown in Fig. 4. Let $\mathbf{X}_(D_Spring)$ be the independent geometric and/or material spring design variables, with which an elastic spring analysis can be performed. The fidelity of the analysis can vary from simple static analysis to more advanced dynamic analyses that acount for structural vibrations. Design engineers can apply heuristics to determining an appropriate level of fidelity. The elastic spring analysis generates a force-deflection curve $F(\delta)$ over the achievable deflection range. This $F(\delta)$ is then used as to model state dependence in the dynamic system simulation. This simulation is used to predict $L(\mathbf{X}_D)$, and can then be used to optimize performance with respect to spring design.

At least three methods of choosing design variables for the nonlinear spring exist. Note that the following guidelines are not only design representation guidelines, but also predictive model guidelines, since each of them requires a different nonlinear spring model.

Topology optimization. This strategy offers the most flexible design representation. It supports essentially arbitrary spring geometries, allowing engineers to explore a very wide range of physically realizable nonlinear springs. It may be a challenge to include manufacturability considerations with established topology optimization methods. The downside of topology optimization is its extreme complexity and difficult in implementation, especially if the objective function differs from the standard compliance objective functions typically used with established topology optimization methods. Difficulty also increases when considering realistic manufacturability constraints. Explanations of topology optimization and case studies are provided in [13] and many other recent articles.

Shape optimization. This is a less flexible design representation in that spring shape can be adjusted widely, but topology cannot change. One possible shape optimization implementation involves a design representation where we assume the boundaries of the spring geometry are defined by splines. The design variables are the spline control points. This method itself can vary in complexity. The spline curves may have different fidelities (e.g., number of control points, spline types). Manufacturability constraints may alsobe necessary.

While the shape optimization method sacrifices some level of flexibility in design representation, it has fewer restrictions

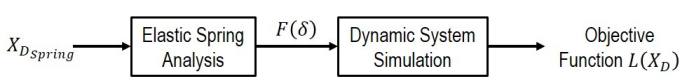

FIGURE 4. MAPPING OF THE SPRING DESIGN OPTIMIZATION PROBLEM

in choosing objective functions, may involve simpler analysis, and is less cognitively demanding. Methods for nonlinear spring design given a prescribed $F(\delta)$ curve are discussed in [17].

Known spring architecture optimization. A further simplification is to assume a particular spring architecture (such as a telescoping conical spring), and then optimize with respect to the parameters for that architecture. This is a significant assumption that reduces design flexibility, but results in a low-dimension design representation, and is particularly helpful for exploring implicitly manufacturable designs that may be commercially available. This approach, however, is less useful at early design stages in that is limits broad exploration to identify enhanced system performance limits and new design insights.

Heuristics often play an important role in this strategy when choosing appropriate architectures. This decision has a significant impact on the formulation and the designs that will be considered during solution. As explained in [18], anchoring heuristics may be in effect when the engineer starts with choosing the "template".

We used this known spring architecture optimization in Guideline 4; specifically, we consider optimizing a telescoping conical spring. A telescoping conical spring is similar to a helical spring, but is tapered. One possible parameterization results in six design variables: d, D_1, D_2, p, Na, and G. In Simplification 4-1, we further reduce the number of design variables by assuming that d, Na, and G are fixed quantities, and that only D_1, D_2, and p are design variables.

Parameterizing $F(\delta)$

The force-deflection function $F(\delta)$ can be parameterized in a number of different ways with varying complexity and fidelity. Free-form parameterization—such as using a high-fidelity spline—offers significant design flexibility. This is similar in concept to designing the relaxation function for VEM directly. Alternative parameterizations, such as piecewise linear functions, may help reduce complexity. Adding simple constraints may be beneficial. For example, for a spring to be stable (only stable state is $\delta = 0$), we can add a monotonicity constraint on $F(\omega)$ to ensure another stable state with $\delta \neq 0$ does not exist. This restricts the design space in a targeted way, reducing the number of designs that are considered and focusing only on stable designs (important for the vibration absorber application). This simplification appears in Guideline 5, which also assumes the use of six control points for cubic spline curves.

As with VEM design, the drawback of this free-form approach is that it may not always be possible to identify a physically realizable spring design that achieves the desired force-deflection curve. This is a highly flexible design representation (close to the origin in Fig. 1 along the design representation axis), but can result in significant error with respect to predicting performance. This error occurs when the desired $F(\delta)$ curve is not realizable, pushing the formulation away from the origin along

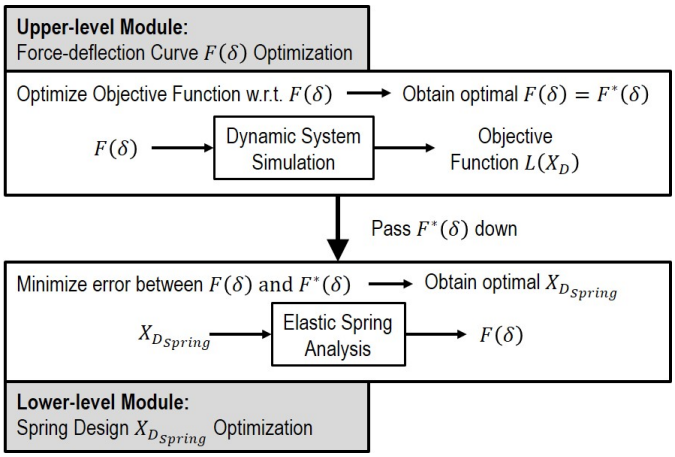

FIGURE 5. ILLUSTRATION OF THE TARGET MATCHING STRATEGY FOR SPRING DESIGN

the predictive modeling axis. This free-form approach, however, may result in important early-stage design insights that can then guide designers in selecting appropriate spring architectures or other design representations that are effective for the application.

Target Matching Strategy

This problem can be re-framed using a target matching strategy, providing an element of modularity. This is illustrated in Fig. 5, and is based in part on the free-form $F(\delta)$ design strategy described above, and the need to work toward physically realizable designs.

We optimize $L(X_D)$ with respect to $F(\delta)$ in the upper-level module. The optimal force-deflection curve $F^*(\omega)$ is an output of the upper-level problem, which is then passed down as a known function to the lower-level module. Here the error between $F^*(\delta)$ and $F(\delta)$ (a physically achievable curve) is minimized. Spring design variables are adjusted to explore $F(\delta)$ possibilities, aiming to match the desired target. In this way, we reframe the vibration absorber design problem as a spring design problem: we only optimize the design to achieve a minimized error. This strategy is our Guideline 6. An alternative method could involve iteration if a strategy can be devised to re-optimize the upper-level problem based on what was learned from the lower-level problem regarding spring design feasibility.

Evaluating Error

To formulate this target matching problem, we need to define a way to evaluate the error between $F(\delta)$ and $F^*(\delta)$, which is the loss function of the lower-level module. We use the simplest sum of errors from Guideline 7:

$$Error = \sum_{n=1}^{6} |F(\delta_i) - F^*(\delta_i)| \tag{7}$$

where δ_i is the x-coordinate (deflection) of the spline curve control points. Guideline 5 specifies that there are six control points.

An improved version of the loss function is one that has a weight for each deflection range. Then, error within specific deflection ranges is treated as less unfavorable if $F(\delta)$ within those ranges affects the system performance very little.

If $F^*(\delta)$ is not close to achievable, steps can be taken to to close the gap between $F(\delta)$ and $F^*(\delta)$. In the lower-level module, we can unlock more flexibility on $\mathbf{X}_{D_{Spring}}$, especially when the lower-level has been formulated in a less flexible way. For example, if we started with the known architecture formulation, we could improve design flexibility by switching to topology or shape optimization. This has the advantage of not requiring simulation when optimizing with respect to a large number of design variables. In addition, another strategy is to add constraints to the upper-level problem based on insights from lower-level solutions to better limit exploration to realizable designs.

6.3 Practical Formulations

From the guidelines described above, we compose six formulation examples, which are summarized in Rows 1–6 of Table 2. Each formulation is programmed and solved using MATLAB®. System dynamics are solved using an adaptive step algorithm, and optimization problems were solved using the `fmincon` function for constrained optimization with the default sequential quadratic programming algorithm. The optimal design variable values and corresponding objective function values are presented in Rows 7–8 of Table 2.

7 EXAMPLE: FORMULATION COMPARISONS

With the design problem solved using the six different optimization formulations, we now compare them against each other based on the three attributes defined earlier.

We acknowledge that for a design problem, its design optimization formulation has an impact on the range of choice of applicable optimization algorithms, as well as the determination of the most efficient algorithm. In other words, problem formulation and problem solution are not two independent decisions, because the formulation, by affecting the choice of the most efficient and effective algorithm, influences computational time and solution quality. With that said, in this paper, we ignore the dependence between problem formulation and problem solution; nonetheless, this dependence and its influence on our method will certainly be studied in future work.

7.1 Attributes

We first explain how the three formulation attributes are quantified. While we propose the following three attributes due to their importance to many design problems, practitioners can include other attributes as required by specific projects, addressing particular needs.

TABLE 2. FORMULATIONS: SUMMARY AND APPLIED GUIDELINES, DESIGN SOLUTION AND EVALUATION RESULTS

Row	Item	*Formulation 1*	*Formulation 2*	*Formulation 3*	*Formulation 4*	*Formulation 5*	*Formulation 6*
1	VE Damper?	No	No	Yes	Yes	No	No
2	Conical Spring?	No	No	No	No	Yes	Yes
3	Target Matching?	No	No	No	No	No	Yes
4	$\mathbf{X}_D$	m_2, c_2, k_2	m_2, c_2, k_2	m_2, c_2, K_1, K_2	m_2, c_2, K_1, K_2	m_2, c_2,D_1, D_2, p	m_2, c_2, cp_i;D_1, D_2, p
5	$L(\mathbf{X}_D)$	$\max(\lvert z_1(t)\rvert)$	$\sum\lvert z_1(t_i)\rvert$	$\sum\lvert z_1(t_i)\rvert$	$\sum\max(\lvert z_1(t)\rvert)_\omega$	$\sum\lvert z_1(t_i)\rvert$	$\sum\lvert z_1(t_i)\rvert$, $\sum\lvert F-F^*\rvert$
6	Guidelines and Simplifications	1.1, 2.1, 2.2, 2.3	1.2, 2.1, 2.2, 2.3	1.2, 2.1, 2.3, 3, 3-1	1.3, 2.1, 2.3, 3, 3-1	1.2, 2.1, 2.2, 4, 4-1, 5	1.2, 2.1, 2.2, 4, 4-1, 5, 6, 7
7	$\mathbf{X}_D^*$	0.9860, 7.4489, 22.0446	1.0000, 4.6922, 20.2336	0.9999, 0.0987, 0.0801, 3.1778	0.3442, 0.1154, 0.2846, 2.1788	0.9982, 3.5181, 0.0854, 0.0843, 0.0497	0.5498, 2.9972; 0.0550, 0.0600, 0.0230
8	$L^*(\mathbf{X}_D^*)$	0.0151	1487	2510	21.7442	282	265; 1540
9	X_{sc}	10%	15%	90%	100%	60%	80%
10	X_{oc}	4	5	13	15	11	14
11	U_c	0.950	0.880	0.140	0.000	0.380	0.145
12	X_t	143 sec.	72 sec.	242 sec.	612 sec.	1050 sec.	280+51 sec.
13	U_t	0.930	1.000	0.830	0.450	0.000	0.740
14	X_q	1718	1487	2510	5906	219	288
15	U_q	0.740	0.780	0.600	0.000	1.000	0.990
16	AU	0.9122	0.9215	0.6959	0.1800	0.8528	0.8916

7.1.1 Task Complexity Task complexity can be evaluated from two perspectives: subjective complexity and objective complexity. Here, we choose the well-established NASA Task Load Index (TLX) to quantify subjective formulation complexity [19]. For simplicity, we sum the six numbers and divide by the maximum possible sum; the lower this ratio is, the less cognitively demanding the formulation is. This ratio is denoted as X_{sc}, the first attribute in our analysis.

For objective complexity, we choose to use the structural complexity quantification method proposed by Sinha and de Weck [20]. This method is a graph-energy based method; it takes into account graph topology. It is well-suited for the formulation evaluation task because of its ability to deal with modularity; our formulations are modular and their modules can be shared among multiple formulations. In our example, the "nodes" in the graphs are the equations, variables and operators such as derivatives. The "links" are how these elements mathematically interact with each other. The quantification result from this method is denoted as X_{oc}, our second attribute.

7.1.2 Computational Time Computational time is taken directly from recorded time required for a particular problem formulation to be solved. It is denoted as X_t. When executing the programs in MATLAB®, the "Run and time" option was selected and the "Profile time" recorded upon completion to evaluate computational expense. It is worth noting that the "Profile time" is affected by code implementation and hardware speed. This helps to estimate relative expense across formulations, but other more rigorous methods could be used for this evaluation.

7.1.3 Solution Quality Solution quality is quantified by an evaluation metric that represents the degree to which the design meets the underlying objectives. As the substantive rationality formulation cannot be achieved in practice, a metric should be chosen that can be evaluated, and is determined to be as close as possible to the ideal objective function. This metric is then used as a benchmark to evaluate formulations that use other objective functions that deviate further from substantive rationality.

For the example problem here, such a metric could be workpiece surface quality (Section 6.1). This could be evaluated, but requires significant investment in model development. To facilitate these studies, we chose a simpler metric as a benchmark. Future work could include high-fidelity surface quality evaluation. The evaluation metric used here is the sum of $\lvert z_1(t_i)\rvert$ over the time span from $t_0 = 0$ to $t_f = 5$ with 5001 evenly spaced points:

$$\text{Evaluation} = \sum_{n=1}^{5001} \lvert z_1(t_i)\rvert \tag{8}$$

This Evaluation value is denoted as X_q.

7.2 AGGREGATE UTILITY FUNCTION

As a strategy to combine the three aforementioned attributes into a single comparison metric, we propose using an aggregate utility function. In this example, we use a linearly-scaled weighted average for the attributes, which essentially mimics the simplest form of a multiattribute utility function under the assumptions of risk neutrality (linear attribute utility function) and equal attribute weights.

X_{sc}, X_{oc}, X_t and X_q are scaled from 0 to 1, 0 being the least desirable and 1 being the most desirable. The resultant values are denoted U_{sc}, U_{oc}, U_t and U_q. U_{sc} and U_{oc} each takes 50% in the formation of U_c. Therefore, the weighted average of Formulation i's score is:

$$\begin{aligned} AU_i =& 0.33(0.25U_s c(x_{sc_i}) + 0.25U_o c(x_{oc_i}) \\ &+ 0.33U_t(x_{t_i}) + 0.33U_q(x_{q_i}) \end{aligned} \tag{9}$$

The attribute values, scaled attribute values, as well as the weighted averages of the formulations are presented in Rows 9–16 of Table 2.

8 DISCUSSION

From the results above, it can be seen that—based on our simple linearly-scaled weighted average—Formulation 2 achieves the highest score among the six formulations. Formulations 1 and 2 are both easy to formulate, fast to solve, and both achieve fairly good solution quality. The different choices of objective function did not impact utility significantly. Formulations 3 and 4 are very difficult to formulate, and while Formulation 3 achieves fair computational time and solution quality, Formulation 4 performs poorly. This result suggests that expanding the design formulation to include VEM design may not be justifiable, and that using an objective function attending to robustness over a range of frequencies does not align with our objective well. A different utility function, benchmark metric, and other differences may lead to a different conclusion. Formulations 5 and 6 have comparable formulation complexity. Formulation 5 achieves the best solution quality, but is very computationally expensive. In contrast, Formulation 6 is much faster to solve and the decrease in solution quality is negligible.

This comparison of formulation results highlights the potential value for practitioners who routinely perform similar yet distinct design tasks. Using our method, they can assess the tradeoffs of formulating design problems with different combinations of guidelines. For example, by switching from Formulation 2 to Formulation 6, the design engineer trades computational efficiency for and lower complexity for better solution quality. Increased complexity costs 0.25 point; longer computational time costs 0.09; better solution quality earns 0.07. There is not a net benefit to switching.

By testing the design engineer's risk attitude and preference on attributes, a full version of a multiattribute utility function can be derived. Suppose that now an engineer faces a design project in which solution quality bears utmost importance, and accordingly attribute preference is updated: k_c=0.2, k_t=0.1 and k_q=0.9. With this change the same level of increased complexity only costs 0.0673, the same increase in computational time costs 0.0112, and the same improvement solution quality earns 0.1531. Now the engineerin is better off switching from Formulation 2 to Formulation 6. As a practitioner accumulates more formulations and their evaluation results for related problems, more data and insight is available to improve aggregate utility value for each design task, given risk attitude and preference on attributes. This approach may be more economical and less time-consuming than current practice.

We acknowledge that these insights required solution of multiple problem formulations. In practice, the resources for such comprehensive studies may not be available. A practitioner may only have time to solve the problem once. How then can an engineer make better-informed formulation decisions? One potential strategy is to systematically study formulation tradeoffs across a wide range of related design problems, and work to extract generalizable guidelines (and possibly formulation principles) based on this data. The work presented in this article is a first step toward this vision.

9 CONCLUSIONS AND FUTURE WORK

In this paper, we identified three important dimensions of problem formulation, and enumerated several formulation guidelines for a vibration absorber system design problem. We discussed the benefits and drawbacks of each guideline, and generated six practical problem formulations. We then examined how each of them performed in terms of complexity, computational time, and solution quality. Finally, we used a simple weighted score to quantify the relative performance of each formulation, showing how formulations can be evaluated in a formalized manner.

To study the potential of this method further, one could develop specific models of the dependence relationships between problem formulation and problem solution quality, as well as including additional attributes in the formulation evaluation process. One could also we can implement our method in more complex design problems, involving more realistic dynamic modeling, more design variables and more constraints. It might also be desirable to use a metric that more accurately aligns with the underlying design objective—for example, a simulated surface quality measure or even an actual surface quality test result—to evaluate the solution quality.

Acknowledgement

This material is based upon work supported by the National Science Foundation under grant no. CMMI-1538234. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. We also extend our thanks to Nate B. Harroun, a former graduate student at University of Illinois at Urbana-Champaign, for contributing to simulation and optimization implementation.

REFERENCES

[1] Cagan, J., Grossmann, I., and Hooker, J., 1997. "A conceptual framework for combining artificial intelligence and optimization in engineering design". pp. 20–34.

[2] Hazelrigg, G. A., 1998. "A framework for Decision-Based engineering design". *Journal of Mechanical Design,* ***120***(4), Dec., pp. 653+.

[3] Cramer, E. J., Dennis, J. E., Frank, P. D., Lewis, R. M., and Shubin, G. R., 1994. "Problem formulation for multidisciplinary optimization". *SIAM Journal on Optimization,* ***4***(4), pp. 754–776.

[4] Fuge, M., Peters, B., and Agogino, A., 2014. "Machine learning algorithms for recommending design methods". *Journal of Mechanical Design,* ***136***(10), pp. 101103+.

[5] Volkema, R. J., 1983. "Problem formulation in planning and design". *Management Science,* ***29***(6), June, pp. 639–652.

[6] Danielescu, A., Dinar, M., MacLellan, C., Shah, J. J., and Langley, P., 2012. "The structure of creative design: What problem maps can tell us about problem formulation and creative designers". In ASME Proceedings — 24th International Conference on Design Theory and Methodology, pp. 437–446.

[7] Watton, J. D., and Rinderle, J. R., 1991. "Improving mechanical design decisions with alternative formulations of constraints". *Journal of Engineering Design,* ***2***(1), Jan., pp. 55–68.

[8] Pomrehn, L. P., and Papalambros, P. Y., 1995. "Discrete optimal design formulations with application to gear train design". *Journal of Mechanical Design,* ***117***(3), pp. 419–424.

[9] Chapman, C. D., and Jakiela, M. J., 1996. "Genetic Algorithm-Based structural topology design with compliance and topology simplification considerations". *Journal of Mechanical Design,* ***118***(1), Mar., pp. 89–98.

[10] Taillandier, P., and Gaffuri, J., 2012. Objective function designing led by user preferences acquisition, Apr.

[11] Bernstein, W. Z., Mani, M., Lyons, K. W., Morris, K. C., and Johansson, B., 2016. "An open Web-Based repository for capturing manufacturing process information". In ASME Proceedings — 21st Design for Manufacturing and the Life Cycle Conference.

[12] Hein, A. M., and Farel, R., 2016. "Estimating reliability bounds on industrial plants".

[13] Fu, K. K., Yang, M. C., and Wood, K. L., 2015. "Design principles: The foundation of design". In ASME Proceedings — 27th International Conference on Design Theory and Methodology, pp. V007T06A034+.

[14] Ku, N., and Andhare, A. B., 2015. "Optimization of surface roughness in turning of Ti-6Al-4V using response surface methodology and TLBO". In ASME Proceedings — 20th Design for Manufacturing and the Life Cycle Conference, pp. V004T05A020+.

[15] Inman, D. J. *Engineering Vibration*. Daniel J. Inman.

[16] Corman, R. E., Rao, L., Bharadwaj, N. A., Allison, J. T., and Ewoldt, R. H., 2016. "Setting material function design targets for linear viscoelastic materials and structures". *R. E. Corman, Lakshmi Rao, N. Ashwin Bharadwaj, James T. Allison, Randy H. Ewoldt,* ***138***(5), Mar., pp. 051402+.

[17] Jutte, C. V., and Kota, S., 2008. "Design of nonlinear springs for prescribed Load-Displacement functions". *Journal of Mechanical Design,* ***130***(8), July, pp. 081403+.

[18] Hallihan, G. M., Cheong, H., and Shu, L. H., 2012. "Confirmation and cognitive bias in design cognition". In ASME Proceedings — 24th International Conference on Design Theory and Methodology, Vol. 7, pp. 913–924.

[19] Hart, S. G., 2006. "Nasa-Task load index (NASA-TLX); 20 years later". In Proceedings of the Human Factors and Ergonomics Society Annual Meeting, no. 50.

[20] Sinha, K., and de Weck, O. L., 2013. "Structural complexity quantification for engineered complex systems and implications on system architecture and design". In ASME Proceedings — 39th Design Automation Conference, Vol. 3A, pp. V03AT03A044+.