Struct Multidisc Optim (2013) 47:645–657
DOI 10.1007/s00158-012-0855-8

RESEARCH PAPER

Divergent exploration in design with a dynamic multiobjective optimization formulation

S. K. Curtis · C. A. Mattson · B. J. Hancock · P. K. Lewis

Received: 13 April 2012 / Revised: 14 September 2012 / Accepted: 21 October 2012 / Published online: 25 November 2012

Abstract Formulation space exploration is a new strategy for multiobjective optimization that facilitates both divergent exploration and convergent optimization during the early stages of design. The formulation space is the union of all variable and design objective spaces identified by the designer as being valid and pragmatic problem formulations. By extending a computational search into the formulation space, the solution to an optimization problem is no longer predefined by any single problem formulation, as it is with traditional optimization methods. Instead, a designer is free to change, modify, and update design objectives, variables, and constraints and explore design alternatives without requiring a concrete understanding of the design problem a priori. To facilitate this process, we introduce a new vector/matrix-based definition for multiobjective optimization problems, which is dynamic in nature and easily modified. Additionally, we provide a set of exploration metrics to help guide designers while exploring the formulation space. Finally, we provide an example to illustrate the use of this new, dynamic approach to multiobjective optimization.

Keywords Conceptual design · Multiobjective optimization · Design space exploration

Nomenclature

g Vector of inequality constraints
h Vector of equality constraints
p Vector of fixed design parameters
w Diagonal matrix of objective identifiers
x Vector of design variables or design objects
y Vector of independent design objects
z Vector of dependent design objects
$\boldsymbol{\mu}$ Vector of design objectives

Subscripts and superscripts

$[]_l$ Lower bound
$[]_u$ Upper bound
$[]^{(0)}$ Benchmark
$[]^U$ Utopia
$[]^N$ Nadir
$[]^*$ Optimal
$\underline{[]}$ Formulation space

1 Introduction

Advancements in computational power have transformed the way engineers perform product design, especially during the later, detailed stages of the design process. Computer-aided design software, finite element analysis, computational fluid dynamics, and numerical optimization are just a few of the computational tools at the designer's disposal. However, many of these tools, especially numerical optimization, are rarely utilized during early, conceptual design stages. There are many reasons for this, but most stem from the nature of conceptual design, which is typically qualitative and fluid in nature. Numerical optimization, on the other hand, usually requires quantitative, well-defined problems to solve. Thus, there is a disconnect between numerical optimization and conceptual design, which if resolved would allow designers

S. K. Curtis · C. A. Mattson (✉) · B. J. Hancock · P. K. Lewis
Dept. of Mechanical Engineering, Brigham Young University, Provo, UT 84602, USA
e-mail: mattson@byu.edu

to benefit from the power of computational assistance and make more informed decisions earlier in the design process.

Conceptual design has been defined in various ways by several researchers (Pahl et al. 2007; Raymer 2006; Ulrich and Eppinger 2004). A common thread between all these definitions is the exploration and discovery of design possibilities/requirements, coupled with the analysis and selection of design concepts for further development. Prevalent activities include benchmarking, conducting market research, abstracting the problem, sketching new ideas, brainstorming, building rudimentary prototypes, and testing design concepts with simple experimentation. Analytical models, when available, are typically low fidelity and computationally inexpensive, which in many cases is seen as an advantage, as they allow a designer to quickly explore a large design space (Kuehmann and Olson 2009; Wang and Shan 2007). Clearly, if computational assistance is to be utilized during conceptual design, then an analytical model in some form is needed. Therefore, we will assume the definition of a design concept from Mattson and Messac (2003), where a concept is defined as an idea that has evolved to the point that there is a parametric model that represents one or more aspects of its performance.

Decisions made during conceptual design generally have the largest impact on the success or failure of a product (Homan and Thornton 1998; Ishii 1995; Mattson and Messac 2002; Wang 2001). Accordingly, several researchers have begun to address the difficulties of implementing numerical optimization techniques during this stage of product development. For example, to capture and represent qualitative objectives, researchers have turned to interactive genetic algorithms (Brintrup et al. 2007, 2008; Gong and Yuan 2011; Takagi 2001), fuzzy logic systems (Huber et al. 2008; Oduguwa et al. 2007), and preference based modeling (Barnum and Mattson 2010). Concept selection via multiobjective optimization is possible by generating and analyzing an s-Pareto frontier, which is the collection of non-dominated designs from a *set* of concepts (Mattson and Messac 2003; Mattson et al. 2009). Several applications where multidisciplinary design optimization has been applied during conceptual design include aircraft configurations (Morino et al. 2006), communication satellites (Hassan and Crossley 2002), and multistage space launch vehicles (Qazi and Linshu 2005). Thus, there is significant promise that numerical optimization can be utilized to an even greater extent during conceptual design.

One challenge with conceptual design optimization that has yet to be fully addressed is its dynamic nature—design parameters, variables, constraints, objectives, and limits are likely to change and evolve over time. With traditional optimization, one must know and clearly define the design parameters, variables, constraints, and objectives before optimization can begin (Arora 2004); however, when the optimization problem definition is improperly formulated (i.e., objectives and constraints are erroneously assumed), the designer will likely be unsatisfied with the results (Balling 1999; Stump et al. 2009). Therefore, in a conceptual design environment, designers may be averse to attempting any numerical optimization because the results may be invalidated as design specifications are updated. In order to be more effective in conceptual design, the optimization problem formulation should be dynamic in nature—easy to formulate, reformulate, and expand into regions beyond the space defined by the initial parameterization (Agate et al. 2010). This need for new Multidisciplinary Design Optimization (MDO) strategies that allow for dynamic problem formulations has been identified as a key research objective in the MDO community (Simpson and Martins 2011). The need for a dynamic problem formulation stems from the desire to (i) facilitate divergent exploration, and (ii) include the designer as an integral part of the optimization loop.

In this paper, we present a new strategy for multiobjective optimization that enables the designer to quickly manipulate an optimization problem and explore the design space in a divergent manner. With this new strategy, the solution to an optimization problem is no longer predefined by any single problem formulation, as it is with traditional optimization methods. Instead, the designer is free to dynamically form the solution as he or she explores the design space as an integral part of an iterative design loop.

The remainder of this paper is organized as follows: we begin in Section 2 by discussing the current multiobjective optimization problem formulation. In Section 3 we introduce a novel concept in numerical optimization which is fundamental to the remainder of this paper—formulation space exploration. In Section 4 we introduce a new multiobjective optimization problem formulation that facilitates divergent exploration. In Section 5 we present exploration metrics to help guide and quantify the exploration process, and in Section 6 we illustrate formulation space exploration with a conceptual design problem. Finally, in Section 7 we offer concluding remarks.

2 Technical preliminary

The generic deterministic multiobjective optimization problem formulation is typically given as Problem 1 (P1):

$$\min_{\mathbf{x}} \{\mu_1(\mathbf{x}, \mathbf{p}), \mu_2(\mathbf{x}, \mathbf{p}), ..., \mu_n(\mathbf{x}, \mathbf{p})\} \quad (n \geq 2) \tag{1}$$

subject to inequality constraints $g_r(\mathbf{x}, \mathbf{p}) \leq 0$ $\{r = 1, 2, ..., n_g\}$, equality constraints $h_s(\mathbf{x}, \mathbf{p}) = 0$ $\{s = 1, 2, ..., n_h\}$, and side constraints $x_{l,i} \leq x_i \leq x_{u,i}$ $\{i = 1, ..., n_x\}$, where

μ_i denotes the i-th generic design objective function; $\mathbf{x}$ is a vector of design variables; $\mathbf{p}$ is a vector of fixed design parameters; and n_g, n_h, and n_x, are the total number of inequality constraints, equality constraints, and design variables, respectively. As a note, $\boldsymbol{\mu}$, $\mathbf{g}$, and $\mathbf{h}$ may be linear or non-linear functions of $\mathbf{x}$ and $\mathbf{p}$.

As formulated above, *P1* yields a set of optimal design alternatives—those belonging to the Pareto frontier. Each solution comprising the frontier is said to be *Pareto optimal*, which means there are no other designs for which *all* objectives are better satisfied (Belegundu and Chandrupatla 1999; Messac and Mattson 2002; Miettinen 1999; Steuer 1986). We generally seek Pareto solutions because they indicate that the objectives have been improved as much as possible without sacrificing the performance of another competing objective (Miettinen 1999).

Other important definitions associated with *P1* include the *utopia point* and the *nadir point*. The utopia point, $\boldsymbol{\mu}^U$, is the point where every objective is simultaneously at its best, or

$$\boldsymbol{\mu}^U = \left[\mu_1^U, \mu_2^U, ..., \mu_n^U\right]^T \tag{2}$$

where μ_i^U is defined as

$$\mu_i^U = \min_{\mathbf{x}} \ \mu_i(\mathbf{x}, \mathbf{p}) \tag{3}$$

subject to the inequality, equality, and side constraints on *P1*. Likewise, the nadir point, $\boldsymbol{\mu}^N$, is the point where every objective is simultaneously at its worst, or

$$\boldsymbol{\mu}^N = \left[\mu_1^N, \mu_2^N, ..., \mu_n^N\right]^T \tag{4}$$

where μ_i^N is defined as

$$\mu_i^N = \max_{\mathbf{x}} \ \mu_i(\mathbf{x}, \mathbf{p}) \tag{5}$$

subject to the same constraints. Typically, neither the utopia or nadir points are on the Pareto frontier nor are they realizable; however, they are helpful for characterizing the bounds of an optimization problem search space (Messac and Mattson 2004).

Problem 1 is well suited for optimization routines used later in the design process when design objectives and constraints are well known and the goal is to converge to the optimal solution. However, in early-stage design, design objectives and constraints may be unknown and the goal is often to diverge and explore many design alternatives, building designer confidence that a better design was not overlooked (Ulrich and Eppinger 2004).

3 Formulation space exploration

A primary objective of this paper is to provide a new optimization method to evolve the design space quickly in a divergent exploratory phase as new design requirements and preferences are identified. We will introduce the fundamental concept behind this method with the aid of a simple, yet popular engineering problem: the two-bar truss. The truss is depicted and labeled at the top of Fig. 1. The graphs below the truss, labeled (a) through (d), represent a traditional view of the *design space* for the two-bar truss. The graphs in (e) and (f) introduce a new concept in optimization, which is centered on the idea that the design variable space and objective space of a particular formulation represent only a portion of a larger space known as the *formulation space*, the exploration of which is beneficial to the designer.

The two-bar truss at the top of Fig. 1 is composed of circular tubing. The solid lines indicate the undeflected state of the truss, while the dashed lines represent the deflected state. The independent design parameters and variables defined in the figure include the height (H), base length (B), tube diameter (d), tube wall thickness (t), material density (ρ), modulus of elasticity (E), and vertical load (P). The mass (M), stress (σ), buckling stress constraint (σ_{buckling}), and deflection (δ), are calculated using a strength/mechanics of materials model (Fox 1971). Using these equations, we can easily formulate an optimization problem for the truss. For instance, we could minimize δ and M by changing t and H, subject to inequality constraints on σ and σ_{buckling}.

Figure 1a represents the design variable space for t and H. Here, each design variable is shown on one of two orthogonal axes for this two-variable problem. Clearly we are not limited in concept to two dimensions, though we choose this now for illustration purposes. The shaded region is the *feasible* design variable space. Any point in the space that resides in this region satisfies all constraints placed on the design variables. The points or designs in the design variable space map to the design objective space through the objective functions. The design objective space for δ and M is shown in Fig. 1b. Again, the set of feasible designs in the design objective space is represented by the shaded region. All solutions in this region satisfy all the constraints of the problem formulation. Recall that for two or more objectives, the Pareto frontier (see Section 2) exists if two or more objectives are in conflict—which is the case here, since a decrease in material (and therefore, mass) can be expected to result in an increase in the deflection (δ) of the truss.

It is essential to note that both the design variable space and the design objective space are completely defined by the optimization problem formulation. In fact, if we reformulate our optimization problem to include B as a design variable (i.e., minimize δ and M by changing t, H, and B

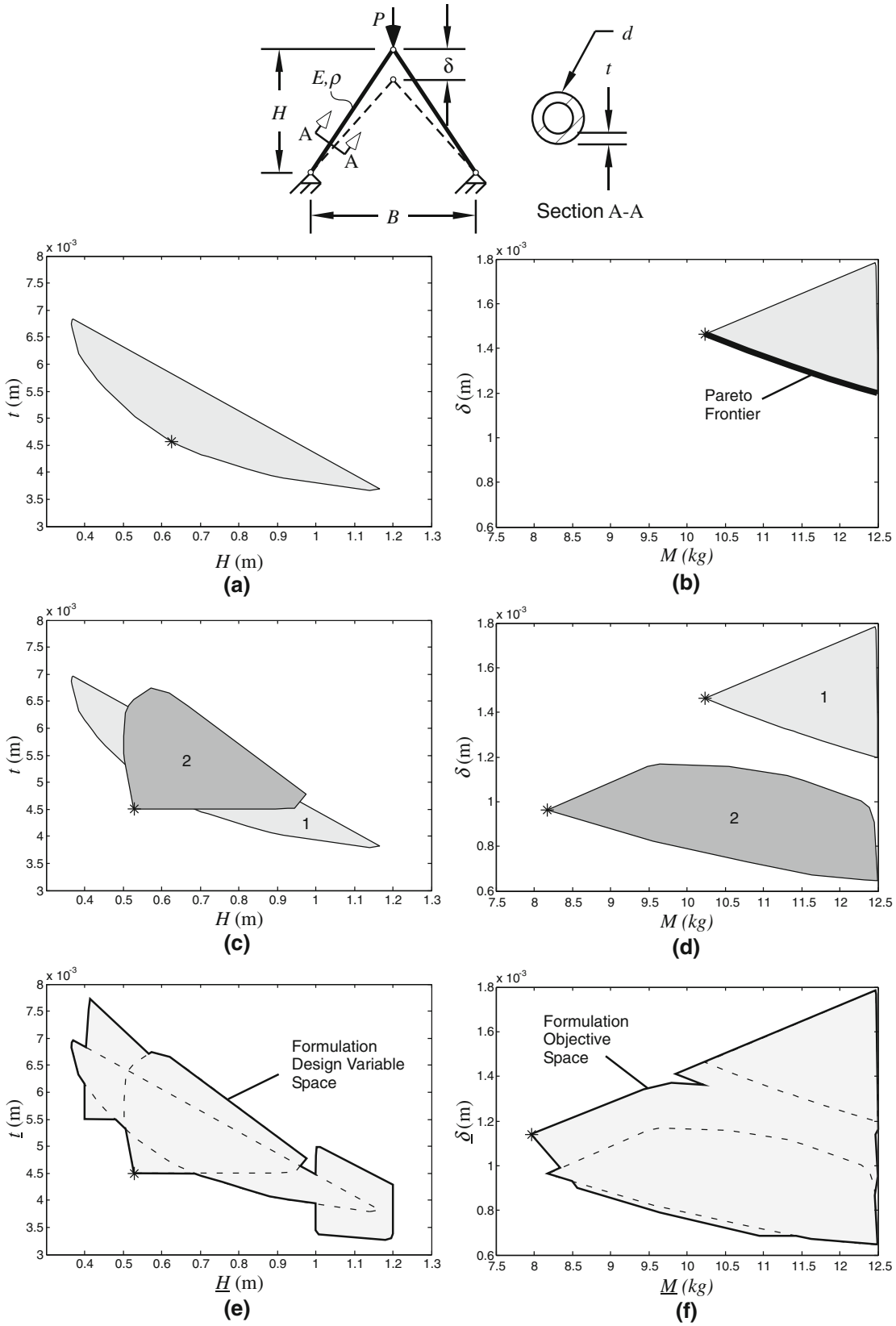

Fig. 1 *Top*: two-bar truss. **a** Feasible design variable space for Formulation 1. **b** Feasible design objective space for Formulation 1. **c** Feasible design variable space for Formulations 1 and 2. **d** Feasible design objective space for Formulations 1 and 2. **e** Formulation design variable space. **f** Formulation design objective space

subject to the same inequality constraints on σ and σ_{buckling}), then we get different design variable and objective spaces, which are the darker shaded regions in Fig. 1c and d. The new design variable space in (c) is actually a projection of a 3-dimensional space onto the t-H plane, due to the adjusted values of the design variable B. While this space appears to overlap with the one defined by the previous formulation (shown as the lighter shaded region), it maps to a completely separate design objective space in (d). The design of least mass, seen as an asterisk in the graphs, is different depending on the optimization formulation. Thus, a common argument against numerical optimization methods is that the optimal solution to the problem is predefined by the problem formulation—in other words, the optimal solution is defined before the search begins. For many practical problems this predefinition is not a drawback since numerical optimization is employed to simply carry out the mundane search for the solution that the designer knows he or she wants. For other design problems, not of this nature, the designer is genuinely interested in exploring the design options *without having to form a concrete understanding of the problem or definition of the formulation*. In such cases, which are abundant in early design, an alternative concept for numerical optimization is needed.

The new strategy presented here expands the exploration of design possibilities to another space—the *formulation space*, meaning the optimization problem formulation space. Exploration into this space is divergent, as it expands from the traditional consideration of design variable and design objective spaces for only *one* problem formulation. Figure 1e and f illustrate the formulation space for design variables t and H, and the formulation space for design objectives δ and M, respectively. For notation purposes, a bar is placed under the symbol to indicate that it is in the formulation space. To elaborate, consider Fig. 1e, which is the formulation space for design variables. Here, we see a large, shaded region labeled as the formulation design variable space. The regions enclosed by the dashed lines represent the design variable spaces formed by the previous optimization problem formulations from above. As shown, the formulation space encompasses the previous formulations, and expands into design spaces that we have not explicitly introduced here. The formulation space is the union of all design variable and objective spaces identified by the designer as being valid and pragmatic problem formulations.

The fundamental technical concept of this paper is that by expanding the exploration into the formulation space, the optimal solution is no longer predefined by the optimization formulation; instead, the solution is formed through divergent exploration of the formulation space. Importantly, this places the designer at the center of the optimization loop, where his or her judgment can be utilized to rationally interpret the results of the computational search.

Divergence in early design is crucial to avoid missing a potentially superior solution, only to later discover its existence and have to perform costly design iterations (Ulrich and Eppinger 2004). With *P1*, once the problem is formulated, convergence begins and divergence can no longer occur unless modifications are made to the programming of the problem formulation. Changing the formulation in *P1* after it has been executed is non-trivial, as the designer must transition from a creative, explorative mind-set to an analytical mental disposition to reprogram the optimization problem. This is not conducive to design exploration; the designer is less likely to ask "what if" questions if he or she must exert a significant amount of effort to reformulate the optimization problem. In CAD modeling, studies have shown that premature idea fixation is likely to occur if the perceived cost of changing the model is too high (Robertson and Radcliffe 2009). The same is true for optimization—if the perceived cost of reformulating the optimization problem is high, then a designer will not likely explore the formulation space. Thus, a dynamic optimization problem formulation is required, one that allows the designer to optimize and reoptimize with modified variables, constraints, or objectives at a low cost.

4 Dynamic multiobjective optimization problem

Formulation space exploration requires us to look at optimization problem statements in a new light—one in which design variables can seamlessly turn into design parameters, or inequality constraints into design objectives, etc. As optimization problem formulations change, so do the individual components; what was a design parameter in one formulation could be implemented as a design objective in the next formulation. Thus, to avoid confusion and to illustrate that the designer does not have to commit to variables, parameters, constraints, or objectives, we will refer to these components of an optimization formulation as *design objects*. With this understanding, we present a generic dynamic multiobjective optimization problem as Problem 2 (P2):

$$\min_{\mathbf{y}} \left\{\mu_1(\mathbf{x}), \mu_2(\mathbf{x}), ..., \mu_{n_x}(\mathbf{x})\right\} \quad (n_x \geq 2) \tag{6}$$

subject to the side constraints

$$y_{l,i} \leq y_i \leq y_{u,i} \quad \{i = 1, ..., n_y\} \tag{7}$$

$$z_{l,i} \leq z_i \leq z_{u,i} \quad \{i = 1, ..., n_z\} \tag{8}$$

where

$$\boldsymbol{\mu} = \mathbf{w} * \mathbf{x} \tag{9}$$

$$\mathbf{w} = \begin{bmatrix} w_{1,1} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & w_{n_x,n_x} \end{bmatrix} \tag{10}$$

$$\mathbf{x} = \left[y_1, y_2, ..., y_{n_y}, z(y)_1, z(y)_2, ..., z(y)_{n_z}\right]^T \tag{11}$$

where **x** is a vector composed of independent design objects (model inputs), **y**, and dependent design objects (model outputs), **z**; **w** is a diagonal matrix where each element along the diagonal is a member of the set $\{-1, 0, 1\}$; n_x, n_y, n_z are the number of design objects, independent design objects, and dependent design objects, respectively.

Problem 2 is very similar to Problem 1, with a few exceptions. Beginning with (6) in *P2*, the dynamic multiobjective optimization problem is minimized over all independent design objects in **y** instead of only the design variables in **x**. In fact, assume the nature of **x** has changed; it now includes all independent and dependent design objects, whereas in *P1*, **x** only contained independent design variables. The role of each design object in **x** is determined by the lower and upper bounds on **y** and **z** in (7) and (8), as well as the values in the diagonal of **w** in (10). If in (7), $y_{l,i} = y_{u,i}$, then y_i (or x_i) is a design parameter; otherwise, y_i is a design variable. Likewise, if in (8), $z_{l,i} = z_{u,i}$, then z_i (or x_{i+n_y}) is an equality constraint; otherwise, z_i is an inequality constraint. Thus, the inequality, equality, and side constraints for *P1* are satisfied with (7) and (8). If in (10) $w_{i,i} = 0$, then the corresponding x_i is not a design objective. If $w_{i,i} = 1$, then x_i is an objective that is minimized; if $w_{i,i} = -1$, then x_i is an objective that is maximized. The conditions that determine a design object's behavior are summarized in Table 1.

Importantly, *P1* and *P2* will yield the same Pareto frontiers; however, by describing the multiobjective optimization problem with *P2*, formulations are more easily manipulated, allowing the designer to quickly and efficiently explore all feasible and pragmatic design spaces. If, for example, the designer wants to switch a design variable to a minimized objective, then he or she simply changes the corresponding value in the **w** matrix—no additional programming is necessary. Likewise, if he or she wants to change a parameter to a design variable, only the values in $\mathbf{y}_l$ and $\mathbf{y}_u$ need to be changed. To illustrate, consider Table 2, which presents the number of required line changes to generic pseudocode (see Appendix) to modify an optimization problem formulated with *P1* and *P2*. The number of line changes are tallied for adding and deleting design objects from the formulation, as well as modifying or mutating existing design objects. Changes to values in the code were not counted, as such actions are considered trivial; for example, if the upper bound on a design variable changes from a value of 3.50 to 4.00, this is not included in the tally. As shown, on average, *P2* requires 14 less lines of code to change than *P1*.

We note that there are some drawbacks to the dynamic optimization formulation. The required number of changes to lines of pseudocode for adding a design object is greater for *P2* than for *P1*. However, this discrepancy will be often be compensated for by the decreased number of changes to lines of pseudocode for modifying that new object after it has been created. Because a designer will often have little prior experience with this new design object in his or her formulation, we suspect that modifications will be required often, mitigating to some extent the negative aspect of this drawback. Also, depending on the optimization algorithm used, computational efficiency may decrease. Since *P2* minimizes over all independent design objects (**y**) including

Table 1 Design object behavior in *P2*

Design object x_i	Condition
Minimized objective	$w_{i,i} = 1$
Maximized objective	$w_{i,i} = -1$
Non-objective	$w_{i,i} = 0$
Design parameter	$y_{l,i} = y_{u,i}$
Design variable	$y_{l,i} \neq y_{u,i}$
Equality constraint	$z_{l,i} = z_{u,i}$
Inequality constraint	$z_{l,i} \neq z_{u,i}$

Table 2 Required changes to lines of pseudocode to add, delete, or mutate design objects in *P1* and *P2*

Action	Design object	*P1*	*P2*
Add design object (New)	Objective (Independent)	5	5
	Objective (Dependent)	3	6
	Constraint	4	5
	Variable	4	4
	Parameter	2	4
	Sub total	18	24
Delete design object	Objective	1	0
	Constraint	1	0
	Variable	4	4
	Parameter	2	4
	Sub total	8	8
Modify or mutate design object (Existing)	Constraint to objective	1	0
	Variable to objective	1	0
	Parameter to objective	7	0
	Parameter to variable	6	0
	Inequality constraint to equality	3	0
	Add bound to constraint	2	0
	Sub total	20	0
	Total	46	32

Numerical value changes are not counted

those that act as fixed design parameters, many gradient-based optimization algorithms will attempt to perturb fixed independent design objects and waste function calls. While any computational inefficiency is obviously undesirable, this is generally not a significant problem during conceptual design, because the models are typically computationally inexpensive (see Section 1). Moreover, other non gradient-based algorithms such as simulated annealing or genetic algorithms will see no significant efficiency losses. A closer look at the benefits and drawbacks of *P2* is provided in Section 6.

5 Exploration metrics

To aid the designer during the exploration process, we provide several exploration metrics. These metrics help the designer to determine (i) how well the exploration process has expanded the formulation space, (ii) how much improvement to objectives has been added through formulation space exploration, and (iii) when the exploration process is no longer diverging. These metrics loosely correspond to metrics of ideation effectiveness, as proposed by Shah and Vargas-Hernandez (2003)—namely novelty, variety, quality, and quantity. The difference here is that we are exploring design alternatives rather than generating new design concepts. Our metrics describe improvements made to the formulation space in terms of novelty, preferred variety, and quality.

For each of the metrics introduced in this section, we assume that there is a baseline formulation objective space, denoted with the superscript $[]^{(0)}$. This benchmark design space is the first to comprise the formulation space. If there is no improvement over the benchmark, then the value of the metric is zero. Higher values of each metric indicate improvement. We also assume that the last or current formulation in the formulation space contains the critical design objects, as determined by the designer. In other words, if at the beginning of the design exploration process the designer is only interested in two objectives, but by the end of the exploration process he or she is interested in three, then the three objectives are used in the calculations for the metrics of all previous formulation spaces. In this manner, the design spaces that make up the formulation space can be compared to one another. Finally, we note that feasibility is assumed in formulation space exploration; by definition, a space that is not pragmatic or valid is not included in the formulation space.

5.1 Novelty

Novelty, as defined by Shah and Vargas-Hernandez (2003), is a measure of how well the exploration process expands the search into regions that are *not perceived to be within the design space*. Thus, during formulation space exploration any region of the formulation space outside the original design objective space is considered novel. The metric for formulation space novelty, $\underline{M}_n$, is given by

$$\underline{M}_n = \frac{\|\underline{\Omega}\| - \|\Omega^{(0)}\|}{\|\Omega^{(0)}\|} \tag{12}$$

where $\underline{\Omega}$ is the diagonal of the hypercube containing the entire formulation objective space, $\Omega^{(0)}$ is the diagonal of the hypercube containing the original design objective space, and in general $\Omega^{(i)}$ is the diagonal of the hypercube containing the i-th space (Messac and Mattson 2004), or

$$\Omega^{(i)} = \boldsymbol{\mu}^{N,(i)} - \boldsymbol{\mu}^{U,(i)} \tag{13}$$

and $\boldsymbol{\mu}^{N,(i)}$ and $\boldsymbol{\mu}^{U,(i)}$ are the nadir point and utopia point, respectively, for the i-th space. This is depicted graphically for a two-dimensional formulation space in Fig. 2a. The darker shaded region represents the original design space within the formulation space, shown as the lighter shaded region. The hypercubes containing both the original design objective space and the formulation space are also shown—$\underline{M}_n$ measures the difference in the vector lengths that connects the utopia and nadir points of both spaces. We

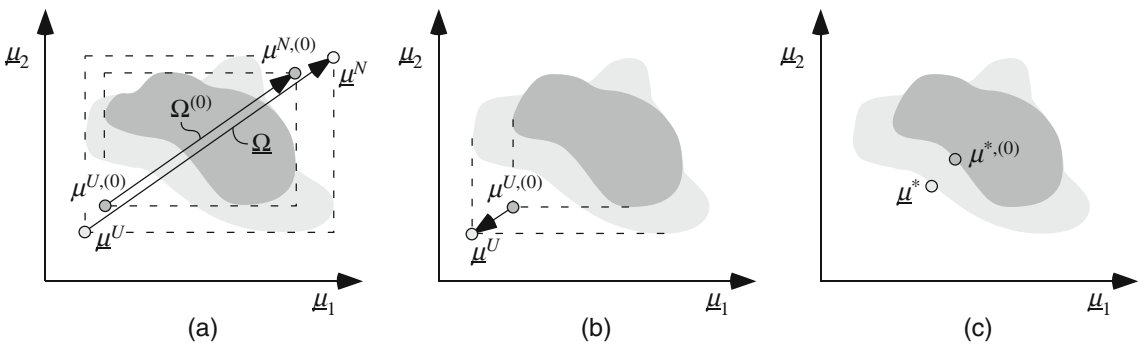

Fig. 2 **a** Novelty. **b** Preferred variety. **c** Quality

note that if design preferences are truly known, $\underline{M}_n$ is not highly valued, as it can reward exploration into regions that are not desirable. If the designer has an interest in divergently exploring a product's design space, however, this metric provides him or her with a way to quantify to what extent new design alternatives are being discovered due to the exploration process. Even if a new optimal solution is not found, the designer may gain confidence in his or her original solution after having seen the potential benefits and drawbacks of other, previously unconsidered, designs.

5.2 Preferred variety

Preferred variety is a measure of how well the formulation space expands into desirable regions. The metric for preferred variety, $\underline{M}_v$, is given by

$$\underline{M}_v = \frac{\|\underline{\boldsymbol{\mu}}^U - \boldsymbol{\mu}^{U,(0)}\|}{\|\boldsymbol{\mu}^{U,(0)}\|} \tag{14}$$

Preferred variety is depicted graphically for a two-dimensional space in Fig. 2b. In the figure, as the vector between $\underline{\boldsymbol{\mu}}^U$ and $\boldsymbol{\mu}^{U,(0)}$ grows in length, $\underline{M}_v$ also increases. If objectives are constrained to be positive, $\underline{M}_v$ will be bounded between 0 and 1; otherwise, values greater than 1 for $\underline{M}_v$ will be possible, indicating the percent improvement of the formulation space over the initial space. In situations where a designer has not yet settled on specific objective weights, this metric may be of particular interest to him or her as a means of showing overall improvement of the Pareto frontier.

5.3 Quality—best design alternative

The metric for the quality of the exploration process, $\underline{M}_q$ is measured in terms of the "best" design alternative (see Fig. 2c), or

$$\underline{M}_q = \frac{J(\boldsymbol{\mu}^{*,(0)}) - J(\underline{\boldsymbol{\mu}}^*)}{|J(\boldsymbol{\mu}^{*,(0)})|} \tag{15}$$

where $\boldsymbol{\mu}^{*,(0)}$ is the best design alternative at the beginning of the exploration process, $\underline{\boldsymbol{\mu}}^*$ is the best design alternative at the end of formulation exploration, and J is an aggregate objective function (AOF). The AOF is used here to determine "quality" for two reasons. First, if formulated properly, the AOF helps to capture the designer's preference in the search. Second, the majority of engineering applications in optimization involve the formation of an AOF, making it a practical measure for quality (Messac and Puemi-Sukam 2000). Many methods exist for formulating the AOF such as weighted sum methods (Steuer 1986), compromise programming methods (Chen et al. 1999), and physical programming (Messac and Mattson 2002). The most suitable method for each specific problem is determined by the designer. If an AOF is never formally defined, then this metric can be calculated using an even weight for all objectives, essentially assigning values to solutions based upon their Euclidean distances to the origin.

5.4 Exploration value to effort ratio

Formulation space exploration must be advantageous in order to be useful. As the name suggests, the exploration value to effort ratio is defined as

$$\epsilon \equiv \frac{\text{value}}{\text{effort}} \tag{16}$$

where value can be defined as any individual metric or combination of the metrics defined above. For instance, value could be assessed by the best design alternative with (15). The effort can be measured in terms of coding complexity (Halstead 1977; McCabe 1976), number of function calls in the optimization algorithm, or computation time. Regardless of method, there is an associated cost with formulation space exploration; therefore, it is pivotal to monitor the metrics described in this section to ensure that value is being added through the exploration process. When the exploration process ceases to produce novel design alternatives, improve objective values, or discover new "best" designs according to the user-defined AOF, the formulation space is no longer diverging and subsequent exploration will likely decrease ϵ.

5.5 Discussion

The metrics are intended to be a design tool, where the designer can review the metrics after each optimization formulation to determine if "value" is being added by the exploration process. This is critical because exploration is likely to occur in more than two or even three dimensions, making it difficult to fully comprehend the formulation space. For instance, the formulation spaces for the two bar truss example shown in Fig. 1e and f are only two-dimensional representations of multidimensional spaces (i.e., the true variable and objective formulation spaces respectively span the variables B, t, H, and d, and the objectives M, δ, σ, and σ_{buckling}). The metrics, however, easily extend into any number of dimensions. At the end of the two bar truss exploration process, $M_n = 5.315$, $M_v = 3.346$, and $M_q = 5.624$. Each metric indicates that the exploration process has added significant value over the original formulation.

The metric values for the exploration sequence can be tracked and presented graphically to the designer after each new formulation (see Fig. 4 in Section 6). When the met-

ric values stop improving, the designer will know that the formulation space may no longer be diverging in a useful manner. Since the designer is reviewing the results of the optimization process after each iteration, the designer literally becomes part of the optimization loop, using his or her judgment to help guide the exploration process.

A noted weakness in the provided metrics is their dependency on the original formulation. Each metric is scaled to show an improvement with respect to that formulation, which means that identical final formulation spaces may exhibit differing values for these metrics, due to differences in the original formulations of the exploration sequences. This dependency suggests that these metrics are more effectively used as a means of recognizing the amount of improvement of a particular design space for a given sequence of formulations, rather than as a comparison between different formulation sequences.

Table 3 Summary of inputs and outputs to the antisubmarine warfare aircraft concept model

Inputs	Outputs
Weight of payload	Wetted aspect ratio
Weight of crew	Maximum lift to drag ratio
Cruise speed	Lift to drag ratio during cruise
Wing aspect ratio	Lift to drag ratio during loiter
Wetted area ratio	Unknown fuel-weight fractions of mission
Cruise range	Take-off weight
First loiter time	
Second loiter time	
Known fuel-weight fractions of mission	
Empty weight fraction model coefficients	

6 Case study: conceptual sizing of an aircraft

The purpose of this case study is to illustrate how to use formulation space exploration presented in Section 3 to search a product's design space in both a divergent and convergent manner during conceptual design. As part of the study, we quantify the benefits and limitations of the dynamic optimization problem formulation from Section 4, and show how to interpret and utilize the metrics from Section 5. The focus here is not to defend the practicality of the proposed product or its analysis model, but rather to show how the methods presented in this paper could be used in the development of a new product.

The case study is based upon the conceptual sizing problem of an antisubmarine warfare aircraft presented by Raymer (2006), where a rudimentary analytical model for sizing any aircraft from a conceptual sketch is developed from statistical and historical data. The conceptual sketch for this case study is shown in Fig. 3a, and the mission profile in (b). The model inputs and outputs are summarized in Table 3. A total of 17 independent design objects and 11 dependent design objects are included in the model. This model is well suited for formulation space exploration because it is a well-known example of conceptual design, where the expressed intent for its use is in *evaluation and refinement, with the customer, of the design requirements* (Raymer 2006).

Eight optimization problem formulations are created here using *P2* as the template; each formulation is summarized below:

- **Formulation 0**—Maximize cruise range and minimize take-off weight subject to lower and upper constraints on all model outputs and by allowing the wing aspect ratio and cruise range to vary between side constraints. All other model inputs are fixed independent design objects (i.e., fixed design parameters).
- **Formulation 1**—Same as Formulation 0, except we allow the weight of the payload to vary.
- **Formulation 2**—Same as Formulation 1, except we allow the wetted area ratio vary.
- **Formulation 3**—Same as Formulation 2, except we add the weight of the payload as a maximized objective and the total fuel-weight fraction as a minimized objective.

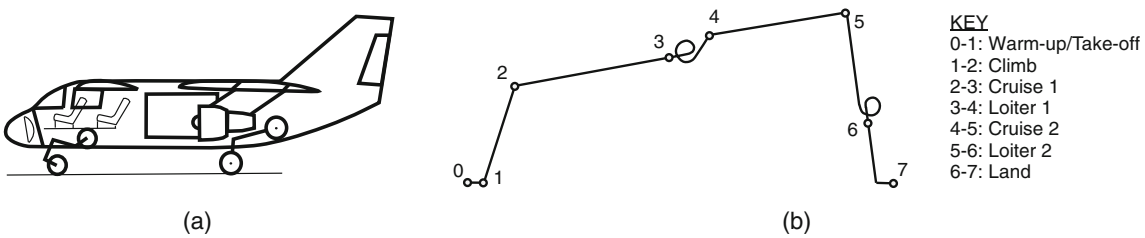

Fig. 3 **a** Antisubmarine warfare aircraft conceptual sketch. **b** Flight mission details. Images are adapted from Raymer (2006)

- **Formulation 4**—Same as Formulation 3, except we remove the total fuel-weight fraction as a minimized objective.
- **Formulation 5**—Same as Formulation 4, except we change input values to simulate a composite aircraft instead of one fabricated out of aluminum.
- **Formulation 6**—Same as Formulation 4, except we allow the cruise speed to vary and change input values to simulate a low-bypass turbofan engine rather than a high-bypass turbofan.
- **Formulation 7**—Same as Formulation 4, except we allow the first mission loiter time to vary.

The results from the formulation exploration process are shown in Fig. 4. In (a), a two-dimensional slice of the formulation space is shown for cruise range and take-off weight (shown as the lighter shaded region, with the baseline formulation shown as the darker shaded region and all other formulations shown as dashed lines); in reality, the formulation space is a three-dimensional since we have three objectives in this problem. For this reason, we have also plotted exploration metrics in (b), which easily extend into n-dimensional space. The novelty metric at the end of the exploration process is 2.13, meaning that the diagonal of the hypercube containing the entire formulation space is 2.13 times larger than the baseline. The preferred variety metric ends at a value of 0.44, meaning that the distance between the formulation utopia point and the baseline utopia point 0.44 times the distance of the baseline utopia point. The quality metric is at 0.65, indicating that the AOF value, which in this case is determined with a weighted sum, of the formulation space is 0.65 times better than the baseline AOF value.

An important aspect of the exploration metrics is to help indicate when the formulation space is no longer diverging. From Fig. 4b, we see that the average slope of the exploration metrics near the end of our exploration is low, which is a good indicator that the process may be complete. Note that at Formulation 5, the quality metric (M_q) reached its maximum value. If Formulation 5 had been formulated first in the exploration process, there would have been no improvement in M_q in subsequent formulations. In other words, the value of M_q would never increase and the value of M_q would remain at zero. However, there may have been improvement in the other metrics. If the designer was only interested in M_q, then they may not have explored the remaining formulations.

The benefits from formulation space exploration are not only measured by the metrics above, but by what information is gleaned from the process. It is clear that various trade-off studies are possible through formulation space exploration. Obtaining the Pareto frontier for the formulation space provides a rich set of design alternatives from which the designer can "shop" for the most suitable solution (Balling 1999). For example, consider points α and β in Fig. 4a. Point α is a design alternative from the first optimization formulation, and point β is a design alternative from the three-dimensional formulation space Pareto frontier, originating from Formulation 5 where a composite aircraft is simulated. These designs are juxtaposed in

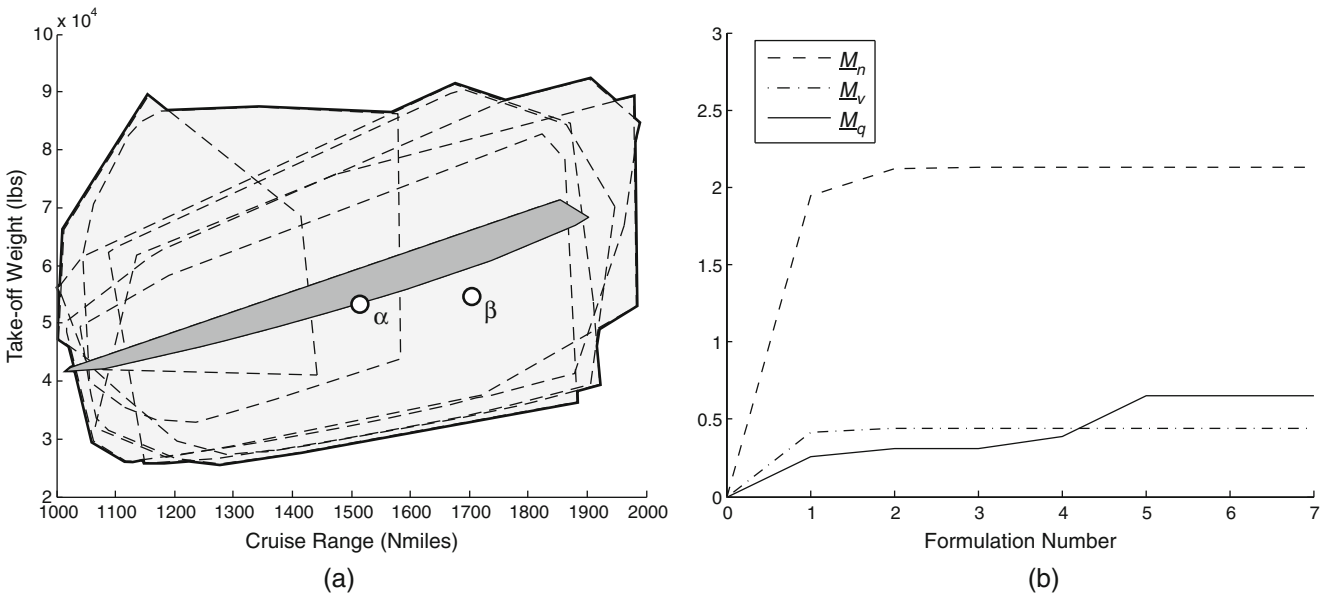

Fig. 4 **a** Formulation objective space for cruise range and take-off weight. *Darker shaded region* is baseline formulation. **b** Novelty, preferred variety, and quality metrics versus formulation number

Table 4 Details about two design alternatives in the formulation space

	Weight of payload (lbs)	Wing aspect ratio	Wetted area ratio	Range (Nm)	Weight fraction coefficient	Wetted aspect ratio	Max lift to drag ratio	Total fuel weight fraction	Take-off weight (lbs)
Design α	10,000	8.40	5.50	1520	1.00	1.53	16.96	0.36	53,639
Design β	10,154	8.17	5.19	1715	0.95	1.57	17.24	0.39	54,184

Table 4. While the weight of the payload and take-off weights are similar, the range for Design β is significantly greater.

We note that formulation space exploration can be done with *P1* or *P2*. However, with *P2* it is simpler to formulate and reformulate the optimization problem. According to Table 2, the formulation space exploration process presented in this section would require 27 lines of code to be changed if *P1* is used, whereas with *P2* no lines of code need to be changed (only the values of $\mathbf{y}_l$, $\mathbf{y}_u$, $\mathbf{z}_l$, $\mathbf{z}_u$, and $\mathbf{w}$ need to be changed). However, as stated in Section 4, there are some limitations. To illustrate, we formulate the baseline problem (Formulation 0) using *P1* and *P2*, and execute it with various optimization algorithms including two gradient based methods with a weighted-sums AOF: sequential quadratic programming (SQP) and Interior-point; and an evolutionary algorithm: genetic algorithm. The computation time and function call count for each algorithm are summarized in Table 5. As shown, *P1* outperforms *P2* when using the gradient-based algorithms; however, no significant difference is seen in the evolutionary algorithm. The baseline problem includes 2 design variables as part of 17 independent design objects. When the ratio of design variables to independent objects is low, as it is here, *P2* does not perform as well as *P1* with gradient-based algorithms. As the ratio approaches one, the computational efficiency differences between *P1* and *P2* diminishes. For this example, the computation time difference between *P1* and *P2* is only a few seconds.

Another way to compare the performance between *P1* and *P2* is with the exploration value to effort ratio (see Section 5). In this example, we choose value (see (16)) to be the sum of all exploration metrics above—$\underline{M}_n$, $\underline{M}_v$, and $\underline{M}_q$—multiplied by 100 for proper scaling. The value added is the same whether we use *P1* or *P2*. Effort is approximated as the time it takes to code each formulation plus the computation time. This method for approximating effort does not take into account the time spent interpreting results, planning future formulations, etc. Nonetheless, it is sufficient for our purpose here. We will assume that it takes 10 s to change each line of code (27 lines for *P1* and 0 for *P2*). Using the genetic algorithm computation times from Table 5, the exploration value to effort ratio for *P1* is 1.08, and for *P2* is 11.47. If we instead use an SQP algorithm in conjunction with the normal constraint method (Messac and Mattson 2004) to generate 30 Pareto optimal solutions, the value to effort ratio for *P1* is 0.94, and for *P2* is 1.18. In each case, *P2* outperforms *P1*.

Table 5 Comparison of computation performance for *P1* and *P2* on an Intel Core 2 Quad 2.67 GHz processor

	SQP		Interior-point		Genetic algorithm	
	Time (s)	Function count	Time (s)	Function count	Time (s)	Function count
P1	2.37	48	2.66	77	28.60	1,240
P2	9.08	288	11.18	369	28.07	1,240

7 Conclusion

We have presented an optimization strategy that facilitates both convergence and divergence in design. Using this strategy, a computational search is not confined to the search space defined initially by an optimization problem formulation. Instead, a designer may search the formulation space, which we have defined as the union of all design variable and objective spaces identified by the designer as being valid and pragmatic problem formulations. This can open the door to early stage design divergence with computational assistance. As part of this strategy, we have introduced a generic, vector/matrix-based, dynamic multiobjective optimization problem that allows the designer to easily modify and adapt the optimization problem as needed. Additionally, we have provided a set of exploration metrics to help guide the designer during the formulation space exploration process. In so doing, we provide designers with a method to obtain valuable design information earlier in the design process, and ultimately make better decisions in the early stages of design.

Acknowledgement This research was partially supported by National Science Foundation Grant 0954580.

Appendix: Pseudo Codes

Standard Multiobjective Optimization Code (P1)

```
Define Design Variable Limits...
Define Design Parameter Values...
Construct x0
Construct xL
Construct xU
Construct P
Call [x*, mu*] = optimize(x0, xL, xU, P)

function [objectives] = objectiveFunction(x, P)
Call [outputs] = model(x, P)
Calculate objectives

function [g, h] = constraintFunction(x, P)
Call [outputs] = model(x, P)
Define Equality Constraint Values...
Calculate h...
Define Inequality Constraint Values...
Calculate g...

function [outputs] = model(x, P)
Extract x...
Extract P...
Calculate outputs...
```

Dynamic Multiobjective Optimization Code (P2)

```
Define Independent Design Object Limits...
Define Dependent Design Object Limits...
Construct y0
Construct yL
Construct yU
Construct zL
Construct zU
Define w...
Call [x*] = optimize(y0, yL, yU, zL, zU, w)

function [objectives] = objectiveFunction(y, w)
Call [z] = model(y)
Calculate x = [y;z]
Calculate objectives = w*x

function [constraints] = constraintFunction(y, zL, zU)
Call [z] = model(y)
Calculate constraints = [zL-z; z-zU]

function [z] = model(y)
Extract y...
Calculate z...
```

References

Agate J, de Weck O, Sobieszczanski-Sobieski J, Arendson P, Morris A, Spieck M (2010) MDO: assessment and direction for advancement—an opinion of one international group. Struct Multidisc Optim 40(1):17–33

Arora JS (2004) Introduction to optimum design. Elsevier Academic Press

Balling R (1999) Design by shopping: a new paridigm? In: Third world congress of structural and multidisciplinary optimization, vol 1, pp 295–297

Barnum GJ, Mattson CA (2010) A computationally-assisted methodology for preference-guided conceptual design. J Mech Des 132(12):121003

Belegundu A, Chandrupatla T (1999) Optimization concepts and applications in engineering. Prentice Hall, Englewood Cliffs

Brintrup AM, Ramsden J, Tiwari A (2007) An interactive genetic algorithm-based framework for handling qualitative criteria in design optimization. Comput Ind 58:279–291

Brintrup AM, Ramsden J, Takagi H, Tiwari A (2008) Ergonomic chair design by fusing qualitative and quantitative criteria using interactive genetic algorithms. IEEE Trans Evol Comput 12(3):343–354

Chen W, Wiecek M, Zhang J (1999) Quality utility—a compromise programming approach to robust design. ASME J Mech Des 121(2):179–187

Fox RL (1971) Optimization methods in engineering design. Addison-Wesley

Gong D, Yuan J (2011) Large population size IGA with individuals' fitness not assigned by user. Appl Soft Comput 11:936–945

Halstead MH (1977) Elements of software science. Elsevier North-Holland, Amsterdam

Hassan RA, Crossley RA (2002) Multi-objective optimization of conceptual design of communication satellites with a two-branch tournament genetic algorithm. In: 43rd AIAA/ASME/ASCE/AHS/ASC structures, structural dynamics, and materials conference

Homan BS, Thornton AC (1998) Precision machine design assistant: a constraint-based tool for the design and evaluation of precision machine tool concepts. Artif Intell Eng Des Anal Manuf (AIEDAM) 12(5):419–429

Huber M, Petersson O, Baier H (2008) Knowledge-based modeling of manufacturing aspects in structural optimization problems. Adv Mater Res 43:111–122

Ishii K (1995) Life-cycle engineering design. J Mech Des 117: 42–47

Kuehmann CJ, Olson GB (2009) Computational materials design and engineering. Mater Sci Technol 25(4):472–478

Mattson CA, Messac A (2002) A non-deterministic approach to concept selection using s-Pareto frontiers. In: ASME IDETC/CIE2002, Montreal, Quebec, Canada, DETC2002/DAC-34125

Mattson CA, Messac A (2003) Concept selection using s-Pareto frontiers. AIAA J 41(6):1190–1204

Mattson CA, Muller A, Messac A (2009) Case studies in concept exploration and selection with s-Pareto frontiers. Int J Prod Dev 9(1/2/3):32–59. Special issue on space exploration and design optimization

McCabe TJ (1976) A complexity measure. IEEE Trans Softw Eng 2(4):308–320

Messac A, Mattson CA (2002) Generating well-distributed sets of Pareto points for engineering design using physical programming. Optim Eng 3(4):431–450

Messac A, Mattson CA (2004) Normal constraint method with guarantee of even representation of complete Pareto frontier. AIAA J 42(10):2101–2111

Messac A, Puemi-Sukam C (2000) Aggregate objective functions and Pareto frontiers: required relationships and practical implications. Optim Eng 1:171–188

Miettinen KM (1999) Nonlinear multiobjective optimization. International series in operations research & management science. Kluwer Academic Publishers

Morino L, Bernardini G, Mastroddi F (2006) Multi-disciplinary optimization for the conceptual design of innovative aircraft configurations. Comput Model Eng Sci 13(1):1–18

Oduguwa V, Roy R, Farrugia D (2007) Development of a soft computing based framework for engineering design optimisation with quantitative and qualitative search spaces. Appl Soft Comput 7(1):166–188

Pahl G, Beitz W, Feldhusen J, Grote KH (2007) Engineering design: a systematic approach. Springer

Qazi M, Linshu H (2005) Rapid trajectory optimization using computational intelligence for guidance conceptual design of multistage space launch vehicles. In: AIAA guidance, navigation and control conference

Raymer DP (2006) Aircraft design: a conceptual approach, 4th edn. American Institute of Aeronautics and Astronautics, Reston

Robertson BF, Radcliffe DF (2009) Impact of cad tools on creative problem solving in engineering design. Comput-Aided Des 41(3):136–146

Shah JJ, Vargas-Hernandez N (2003) Metrics for measuring ideation effectiveness. Des Stud 24:111–134

Simpson TW, Martins JRRA (2011) Multidisciplinary design optimization for complex engineered systems: report from a national science foundation workshop. J Mech Des 133:101002

Steuer RE (1986) Multiple criteria optimization, theory computations and applications. Wiley, New York

Stump G, Lego S, Yukish M, Simpson T, Donndelinger J (2009) Visual steering commands for trade space exploration: user-guided sampling with example. J Comput Inf Sci Eng 9(4):044501:1–10

Takagi H (2001) Interactive evolutionary computation: fusion of the capabilities of EC optimization and human evaluation. Proc IEEE 9(9):1275–1296

Ulrich KT, Eppinger SD (2004) Product design and development, 3rd edn. McGraw-Hill/Irwin

Wang J (2001) Ranking engineering design concepts using a fuzzy outranking preference model. Fuzzy Sets Syst 119:161–170

Wang GG, Shan S (2007) Review of metamodeling techniques in support of engineering design optimization. J Mech Des 129:370–380