Design Optimization Using Multiple Dominance Relations

Laurence W. Cook,[*] Karen E. Willcox,[†] and Jerome P. Jarrett.[‡]

December 5, 2019

Abstract

A challenge in engineering design is to choose suitable objectives and constraints from many quantities of interest, while ensuring an optimization is both meaningful and computationally tractable. We propose an optimization formulation that can take account of more quantities of interest than existing formulations, without reducing the tractability of the problem. This formulation searches for designs that are optimal with respect to a binary relation *within* the set of designs that are optimal with respect to another binary relation. We then propose a method of finding such designs in a single optimization by defining an overall ranking function to use in optimizers, reducing the cost required to solve this formulation. In a design under uncertainty problem, our method obtains the most robust design that is not stochastically dominated faster than a multi-objective optimization. In a car suspension design problem, our method obtains superior designs according to a k-optimality condition than previously suggested multi-objective approaches to this problem. In an airfoil design problem, our method obtains designs closer to the true lift/drag Pareto front using the same computational budget as a multi-objective optimization.

1 Introduction

If an engineering design problem can be expressed as an optimization problem, then optimization algorithms or heuristics can be exploited to efficiently find desirable designs [1, 2]. An optimization problem is to find designs that are optimal with respect to particular binary relation[1]. Common optimization formulations are single-objective minimization, where this binary relation is induced by using *less than or equal to* in order to compare scalar objectives, and multi-objective optimization, where this binary relation is induced by using Pareto dominance to compare vectors of objectives (and the performance of optimal designs is known as the Pareto front).

In these common optimization formulations, a quantity of interest must either be assigned as an objective or a constraint if it is to be included in the problem. Often a quantity doesn't clearly belong to either of these two categories: it may not be feasible to use additional objectives in a problem that is already computationally expensive to solve, but an appropriate threshold by which to constrain a quantity may not be known. A designer hoping to use optimization for their design problem is faced with a dilemma.

For example, consider transonic airfoil design, in which key design drivers are increasing lift and reducing drag, but other important quantities of interest are trailing edge separation and pitching moment. If all

[*]Postdoctoral Associate, Aerospace Computational Design Lab, Massachusetts Institute of Technology.
[†]Director, Institute for Computational Engineering and Sciences, University of Texas at Austin.
[‡]University Senior Lecturer, Department of Engineering, University of Cambridge.

[1]A binary relation is a comparison of two designs that determines whether one is better than the other, and for an optimal design no better design exists with respect to the relation [3].

four quantities were used as objectives in a multi-objective optimization, trade-offs would exist between all objectives and the optimizer would be trying to explore a large area of design space. Under a strict computational budget, the optimizer will not be able to explore these trade-offs sufficiently to obtain satisfactory designs. An alternative is to constrain the amount of separation and the pitching moment. The appropriate threshold to use for these constraints may be difficult to determine a priori, since they are physical quantities whose value affects other aspects of the overall system. Too strict a constraint can mean there are no feasible designs, and too loose a constraint is likely to be exploited by the optimizer to give airfoils that are unrealistic for the overall system.

We propose that in such a scenario, it would be useful to search for designs on the Pareto front of separation and moment *within* designs on the Pareto front of lift and drag. This allows the separation and moment to be taken into account without increasing the number of objectives or constraints in the problem, which would reduce computational tractability. However, with current optimization methods, obtaining such designs would require a multi-objective optimization of lift and drag to be performed first, and then moment and separation to be taken into account in a second step.

Consider also the problem of designing a car suspension system to maximize ride comfort and road holding. This is a problem that is known to have many conflicting objectives. Various different multi-objective (and many-objective) optimization formulations have been applied to the design of suspension systems [4, 5], but the most appropriate objectives to use is not well established [6]. It would be desirable to consider all the common objectives without having to resort to a large optimization problem, or choose arbitrary values to constrain them by.

A related type of design problem is one in which we wish to consider vector valued quantities instead of scalar quantities of interest. For example, when the performance of a system is uncertain, we may want to consider the cumulative distribution function (CDF) of the quantity of interest. A design should be both robust (often quantified by low variance of the quantity of interest) but also give a high probability of good performance. In many cases minimizing variance can result in designs that are robust, but whose CDFs lie entirely at worse values than other designs; such designs are said to be stochastically dominated [7, 8]. Therefore we would be interested in obtaining the most robust design that is not stochastically dominated, which is equivalent to finding the design with the lowest variance *within* the set of stochastically non-dominated designs.

These examples (and many other design problems) can be formulated as finding designs that are optimal with respect to a binary relation within the set of designs that are optimal with respect to another binary relation. In this paper we develop a method of finding such designs in a single optimization, by proposing an optimization formulation of which they are the solution. As a result, such designs can be obtained more efficiently than existing methods.

In Section 2, we mathematically define the type of design problem we are considering, propose a binary relation whose optimal designs are solutions to this problem, then discuss how this can be used within optimizers to efficiently solve this problem. In Section 3, we apply this formulation to a design under uncertainty problem, to illustrate its relevance to vector-valued performance measures. Then in Sections 4 and 5 we apply this formulation to the two motivating examples discussed in the introduction to illustrate the benefit of being able to solve this type of engineering design problem efficiently. Finally in Section 6 we conclude the paper and discuss avenues for future work.

2 Design Optimization Using Multiple Dominance Relations

This section defines the problem formulation, proposes a binary relation for solving the problem in a single optimization, develops a method of using this relation in several optimization algorithms, and discusses how this accelerates the solution of the design problems discussed in the introduction.

2.1 Problem Definition

We denote a single design as x, which is an element of the set of all admissible designs $\mathbb{X}$ (taking into account any bounds or constraints); we refer to $\mathbb{X}$ as the design space. We denote the performance of a design $q \in \mathbb{Q}$, where $\mathbb{Q}$ is the set of values q can take and is referred to as performance space; q is is given by a function $f(x)\colon \mathbb{X} \to \mathbb{Q}$. We define the performance q such that smaller is better, and so q is to be minimized. The performance, q, can represent any or all information from a simulation analyzing design x and so could contain scalar objectives, one-dimensional functions (e.g. pressure as a function of distance over an aerofoil), higher-dimensional functions (the pressure as a function of 3D spatial coordinates), etc.

We define an optimization problem (similarly to [9, 10]) as the problem of finding the set of optimal designs in $\mathbb{X}$ with respect to a binary relation $\preceq$. A binary relation is a comparison of *two* designs that determines whether one dominates the other, and is usually induced by a comparison of the performance of the designs. An optimal design in $\mathbb{X}$ with respect to a relation, denoted x^*, is one for which no other design dominates it (and so is also referred to as a non-dominated design). Note that for optimal designs with respect to a binary relation to exist, the binary relation must be a preorder[2] over $\mathbb{X}$.

Definition 2.1. The set of optimal designs in $\mathbb{X}$ with respect to $\preceq$, denoted $\min(\mathbb{X}, \preceq)$, is given by:

$$\min(\mathbb{X}, \preceq) \coloneqq \{x \in \mathbb{X} \mid \nexists\, y \in \mathbb{X} \text{ for which } y \preceq x \ \cap \ x \npreceq y\} \tag{1}$$

Figure 1 illustrates a binary relation $\preceq$ defining a preorder over the design space given by $\mathbb{X} = \{a, b, c, d, e\}$. A design from which an arrow originates is dominated by the design at the end of that arrow according to $\preceq$. In this case, $a \preceq b$, $d \preceq a$, $d \preceq c$, $c \preceq b$, and $e \preceq b$ (also note that because $\preceq$ is a preorder and so is transitive, $d \preceq b$). According to Definition 2.1, an optimal design with respect to $\preceq$ is one from which no arrows originates, so in this case $\mathbb{X}_{\text{optimal}} = \{d, e\}$. Note that two designs between which a path along arrows does not exist are *incomparable*[3], so $d||e$, $a||c$, $a||e$, and $c||e$.

To illustrate further, consider two standard optimization formulations: single-objective and multi-objective optimization.

Example 2.2. Single objective-optimization considers a single scalar value to minimize, i.e. $q = f(x)\colon \mathbb{X} \to \mathbb{R}$, and compares the performance of designs using the *less than or equal to* binary relation. This induces the binary relation $\preceq_{\text{so}}$ over $\mathbb{X}$, where $x \preceq_{\text{so}} y \Leftrightarrow f(x) \leq f(y)$. Thus $\min(\mathbb{X}, \preceq_{\text{so}}) \equiv \underset{x\in\mathbb{X}}{\operatorname{argmin}} f(x)$.

Example 2.3. Multi-objective optimization considers m multiple values to minimize, i.e. $q \in \mathbb{R}^m$. The performance q is now a vector of quantities of interest given by m functions, such that $q = f(x) = [f_1(x), f_2(x), \ldots, f_m(x)]$. Multi-objective optimization searches for optimal designs in $\mathbb{X}$ with respect to the (weak) Pareto dominance relation, denoted $\preceq_{\text{Pareto}}$, which is defined in Definition 2.4.

[2]A preorder is a binary relation that is reflexive ($x \preceq x$) and transitive ($x \preceq y \ \cup \ y \preceq z \implies x \preceq z$)

[3]Two designs x and y are incomparable with respect to a binary relation, denoted $x||y$, if neither dominates the other with respect to that relation.

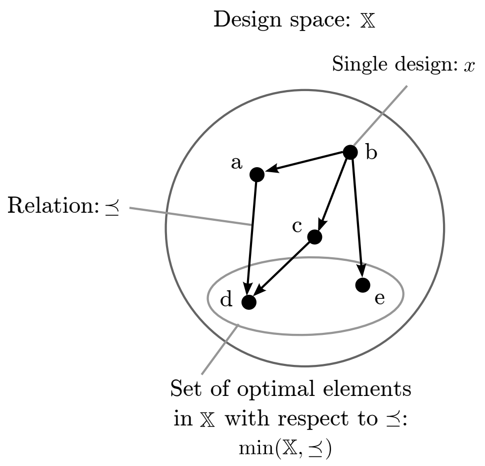

Design space: $\mathbb{X}$
Single design: x
a
b
c
d
e
Relation: $\preceq$
Set of optimal elements in $\mathbb{X}$ with respect to $\preceq$: $\min(\mathbb{X}, \preceq)$

Figure 1: Illustration of a binary relation $\preceq$ giving a preorder over design space.

Definition 2.4. For two designs x and y, x weakly Pareto dominates y, denoted $x \preceq_{\text{Pareto}} y$, if it is at least as good in all objectives, i.e.

$$x \preceq_{\text{Pareto}} y \Leftrightarrow f_i(x) \leq f_i(y) \; \forall i \in 1, \ldots, m \tag{2}$$

The performance vectors of the optimal designs with respect to $\preceq_{\text{Pareto}}$ are known as the Pareto front, denoted PF($\mathbb{X}$):

$$\text{PF}(\mathbb{X}) \coloneqq \; \{[f_1(x), f_2(x), \ldots, f_m(x)] \mid \nexists \; y \in \mathbb{X} \text{ for which } y \preceq_{\text{Pareto}} x \; \cap \; x \npreceq_{\text{Pareto}} y\} \tag{3}$$

Figure 2 illustrates how the Pareto dominance binary relation induces a preorder over design space (the same preorder illustrated on Figure 1). Comparing the performance of each design (given by the values of two objectives on the right) using Pareto dominance induces the arrows between designs on the left.

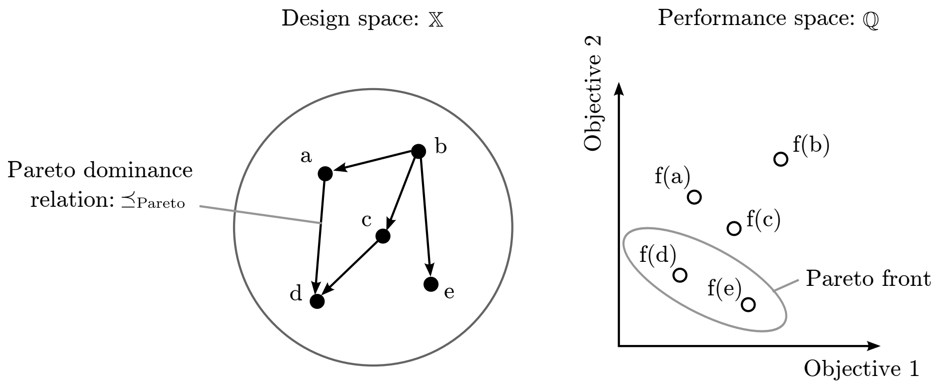

Design space: $\mathbb{X}$
Performance space: $\mathbb{Q}$
Pareto dominance relation: $\preceq_{\text{Pareto}}$
a
b
c
d
e
Objective 2
Objective 1
f(a)
f(b)
f(c)
f(d)
f(e)
Pareto front

Figure 2: Illustration of the Pareto dominance binary relation giving a preorder over design space. On the left are points in design space, and on the right are the corresponding performances.

In this paper, we are interested in finding the optimal set of designs with respect to a binary relation *within* the optimal set of designs according to another binary relation. For this two-relation case, this set of designs can be written as:

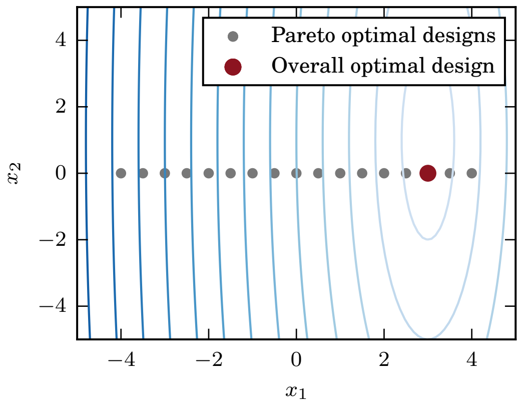
$$\mathbb{X}_{\text{optimal}} = \min(\min(\mathbb{X}, \preceq_1), \preceq_2). \tag{4}$$

Example 2.5. For example, consider the following algebraic function, for which $q = [f_1, f_2, f_3]$, over the 2D design space $\mathbb{X} = [-5, 5]^2$:

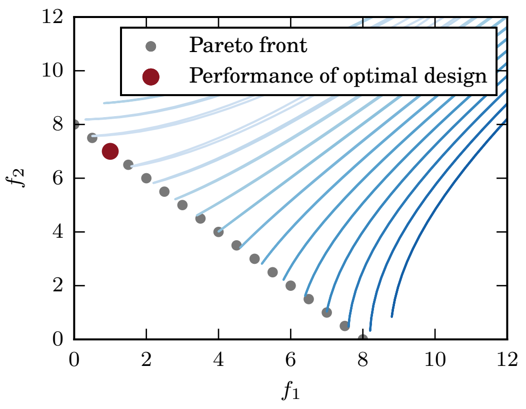
$$f_1 = \sqrt{||x_1 - 4||^2 + 25||x_2 - 0||^2},$$
$$f_2 = \sqrt{||x_1 + 4||^2 + 25||x_2 - 0||^2},$$
$$f_3 = \sqrt{25||x_1 - 3||^2 + ||x_2 - 1||^2},$$

where $||.||$ indicates the L_2 norm.

Let $\preceq_1$ indicate Pareto dominance for f_1 and f_2, and let $\preceq_2$ indicate *less than or equal to* for f_3. We are looking for the following set: $\min(\min(\mathbb{X}, \preceq_1), \preceq_2)$. This set consists only of the design with the smallest f_3 within the set of designs that are Pareto optimal for f_1 and f_2, and this design can be found analytically to be $x = [3, 0]$. Some designs (and their corresponding performance) that are optimal with respect to $\preceq_1$ are plotted in grey on Figure 3, and the design among them with the lowest f_3—the single design in $\min(\min(\mathbb{X}, \preceq_1), \preceq_2)$—is highlighted in red; contours of f_3 are given in blue.



(a) Designs



(b) Performance

Figure 3: Designs and their performance for the algebraic test problem. Optimal designs with respect to Pareto dominance of f_1 and f_2 are given in grey. Contours of f_3 are given in blue.

In general, we consider that we could nest the formulation of Equation 4 further using n binary relations, denoted $\preceq_i, i = 1, \ldots, n$, and look for the set given by Definition 2.6.

Definition 2.6. The set of optimal designs in $\mathbb{X}$ according to n binary relations denoted $\preceq_i, i = 1, \ldots, n$, is given by:

$$\mathbb{X}_{\text{optimal}} \coloneqq \min\left(\left(\ldots \min\left(\min\left(\mathbb{X}, \preceq_1\right), \preceq_2\right)\ldots\right), \preceq_n\right) \tag{5}$$

Definition 2.6 is a problem formulation in that it specifies we are looking for $\mathbb{X}_{\text{optimal}}$, but says nothing about how we might obtain designs in this set. In the following sections we develop an efficient method of obtaining designs given by Equation 5 in a single optimization.

2.2 An overall binary relation using a hierarchy of ranking functions

To use an optimization to find $\mathbb{X}_{\text{optimal}}$ from Equation 5, we need an overall binary relation, denoted $\preceq_{(1,2,\ldots,n)}$, such that $\min\left(\mathbb{X}, \preceq_{(1,2,\ldots,n)}\right) = \min\left(\left(\ldots \min\left(\min\left(\mathbb{X}, \preceq_1\right), \preceq_2\right)\ldots\right), \preceq_n\right)$. We propose such a relation by using a *ranking function* for each of $\preceq_k,\ k=1,\dots,n$ as a measure of how far a design is from true optimal designs, and use lexicographic ordering of these ranking functions:

Definition 2.7. The binary relation $\preceq_{(1,\dots,n)}$, referred to as the *combined binary relation* of $\preceq_k, k=1,\dots,n$, is given by:

$$x \preceq_{(1,\dots,n)} y \;\Leftrightarrow\; R_1(x) < R_1(y) \;\cup\; \Big(R_1(x) = R_1(y) \cap \Big(R_2(x) < R_2(y) \;\cup\; \big(R_2(x) = R_2(y) \cap (\dots)\big)\Big)\Big) \quad (6)$$

where each $R_k, k=1,\dots,n$ is a function that satisfies the following two conditions (and is referred to as a ranking function):

1) Comparison of R_k gives a necessary condition for $\preceq_k$, i.e. $x \preceq_k y \implies R_k(x) \le R_k(y)$.

2) For any $\mathbb{S} \subseteq \mathbb{X},\ R_k(x) = R_k(y)\ \forall x, y \in \min(\mathbb{S}, \preceq_k)$.

This definition can be written more concisely using recursion of the index k in the notation $\preceq_{(k,\dots,n)}$:

$$x \preceq_{(1,\dots,n)} y \;\Leftrightarrow\; R_1(x) < R_1(y) \;\cup\; \big(R_1(x) = R_1(y) \;\cap\; x \preceq_{(2,\dots,n)} y\big) \quad (7)$$

where $x \preceq_{\emptyset} y$ is true $\forall x, y \in \mathbb{X}$.

In the Appendix we give proofs that the combined binary relation in Equation 6 is a preorder, and that $\min(\mathbb{X}, \preceq_{(1,\dots,n)})$ is the set $\mathbb{X}_{\text{optimal}}$ defined in Equation 5—showing that it achieves our goal for this paper.

2.3 Ranking of continuous problems and discrete approximations

In order to use the combined binary relation $\preceq_{(1,2,\dots,n)}$, we need a ranking function for each $\preceq_k$ that satisfies the conditions specified by Definition 2.7. When $\preceq_k$ is the Pareto dominance binary relation, one such ranking function is the Euclidian distance in $\mathbb{Q}$ of the performance of a design from that of its nearest optimal design; another would be the volume enclosed by its performance, a hyperplane perpendicular to each objective, and the Pareto front. These are referred to as distance ranking and volume ranking respectively, and are illustrated in Figure 4 for a single $\preceq_k$ consisting of Pareto dominance of two objectives in a continuous design space.

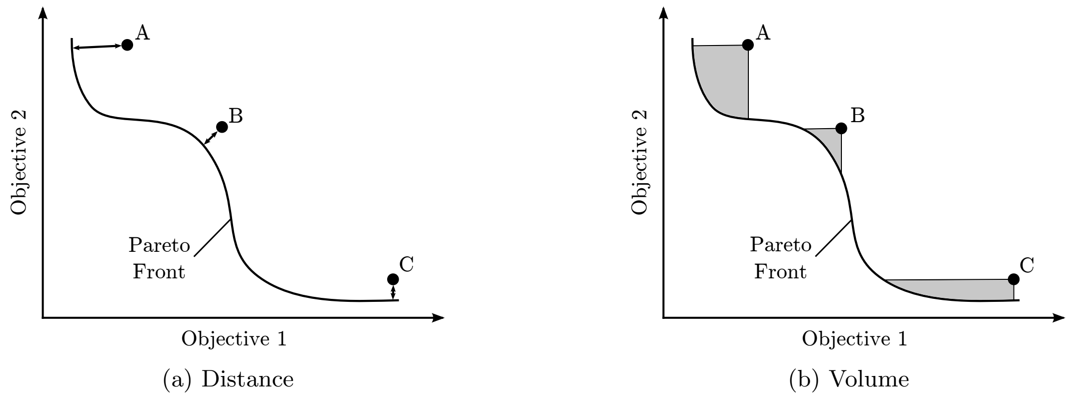


Figure 4: Ranking functions for Pareto dominance in continuous problems. Note, under the distance ranking, $R(C) < R(B) < R(A)$, whereas under the volume ranking $R(B) < R(C) < R(A)$.

The issue preventing these ranking functions from being used directly in $\preceq_{(1,\ldots,n)}$ is that they cannot be evaluated without knowing the optimal designs with respect to $\preceq_k$, which are not known a priori (otherwise we have already solved the problem). Note that while scalarization methods [11]—which use a combination of objectives to define a scalar function whose minimizer is an optimal design of Pareto dominance, e.g. weighted sum—satisfy condition 1, they do not satisfy condition 2, and so are not suitable to use as ranking functions for our goal.

The purpose of defining $\preceq_{(1,\ldots,n)}$ is so that optimizers can be used to efficiently find designs in $\mathbb{X}_{\text{optimal}}$ defined by the problem formulation in Equation 5. Therefore for practical purposes we care not about the actual value of a ranking function for a given design, but rather how the ranking function is used to compare designs visited during an optimization to drive the optimizer towards designs in $\mathbb{X}_{\text{optimal}}$.

We *can* evaluate *approximations* to these rankings by only considering a finite set of designs, denoted $\hat{\mathbb{X}}$, and treating this as a finite representation of the (potentially infinite) design space $\mathbb{X}$. Therefore we use the following binary relation within optimizers:

$$x \hat{\preceq}_{(1,\ldots,n)} y \implies \hat{R}_1(x) < \hat{R}_1(y) \;\cup\; \left(\hat{R}_1(x) = \hat{R}_1(y) \;\cap\; x \hat{\preceq}_{(2,\ldots,n)} y\right) \tag{8}$$

where $\hat{R}_k$ is an approximation to R_k obtained by only considering $\hat{\mathbb{X}}$. As an optimizer explores design space, these approximations drive the optimizer to evaluate designs closer to designs in $\mathbb{X}_{\text{optimal}}$, which in turn will make these ranking approximations more accurate, achieving our goal of finding designs in $\mathbb{X}_{\text{optimal}}$ efficiently.

An approximation to the distance ranking is the non-dominated sorting method of Ref. [12], where the rank of x is the number of times the current non-dominated set needs to be removed from the total set before x would be in the non-dominated set.

Algorithm 1 Determining $\hat{R}_k(x)$ for relation $\preceq_k$ using a distance approximation, given a set of points $\hat{\mathbb{X}}$

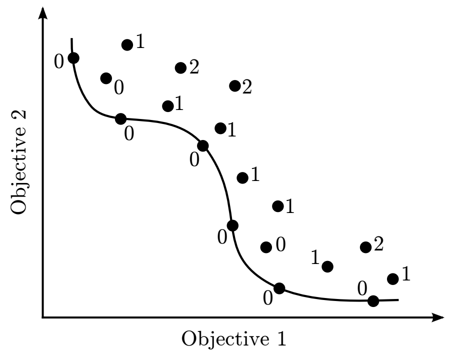
```
rank ← 0
while $\hat{\mathbb{X}}$ is not empty do
    $\mathbb{X}_{\text{current}} \leftarrow \min(\hat{\mathbb{X}}, \preceq_k)$
    for $x_i \in \mathbb{X}_{\text{current}}$ do
        $\hat{R}_k(x_i) \leftarrow$ rank
    rank = rank + 1
    $\hat{\mathbb{X}} \leftarrow \hat{\mathbb{X}} \setminus \mathbb{X}_{\text{current}}$
```

An approximation to the volume ranking can be obtained by counting the number of other designs which dominate any given design [13].

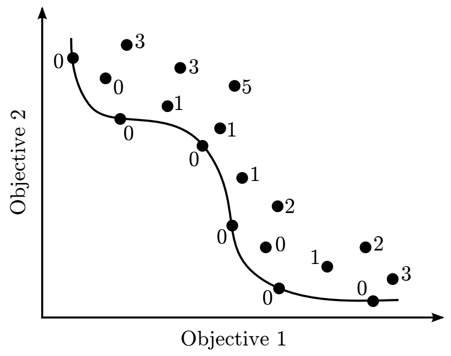
Algorithm 2 Determining $\hat{R}_k(x)$ for relation $\preceq_k$ using a volume approximation, given a set of points $\hat{\mathbb{X}}$

```
for $x_i \in \hat{\mathbb{X}}$ do
    count ← 0
    for $x_j \in \hat{\mathbb{X}}$ do
        if $x_j \hat{\preceq}_k x_i$ and $x_i \hat{\not\preceq}_k x_j$ then
            count = count + 1
    $\hat{R}_k(x_i) \leftarrow$ count
```

These two approximations are illustrated on Figure 5, where each point is labeled by the rank it would be assigned using each of these approximations.



(a) Distance / number of fronts removed



(b) Volume / number of dominating points

Figure 5: Finite approximations to ranking functions

Using a ranking function over a finite approximation to design space is commonplace in population-based evolutionary optimizers (and is why we use the terminology *ranking* functions) [14], where the current population is used as the finite approximation of design space over which a ranking function is evaluated. For example, assigning fitness in multi-objective optimization by counting the number of dominating solutions in a population is known as Pareto ranking [13], and the popular NSGA-II algorithm from Ref. [15] uses non-dominated sorting to assign fitness.

A ranking function in these contexts is used as an operator intrinsic to the workings of a particular algorithm. In contrast, here we use a hierarchy of an arbitrary number of ranking functions in order to solve our proposed optimization formulation of Definition 2.6. Furthermore, we are considering a general context that is independent of any particular algorithm: we first derive the conditions on ranking functions required for continuous problems, and then use approximations on a finite representation of design space. A population within an evolutionary algorithm is just one such representation. Our approach is independent of the particular optimizer used to solve the problem. Indeed, in the entropy-based Bayesian optimization methods of Ref. [16] and Ref. [17], a finite number of representer points are used as an approximation to the overall design space, so our approach is well suited to such optimizers.

Using an approximation to a ranking function, we evaluate our proposed binary relation as follows:

Algorithm 3 Determining whether $x_i \mathrel{\hat{\preceq}}_{(1,\ldots,n)} x_j$

```
k = 1
while $k \leq n$ do
    if $\hat{R}_k(x_i) < \hat{R}_k(x_j)$ then
        return True
    else if $\hat{R}_k(x_i) = \hat{R}_k(x_j)$ then
        $k = k + 1$
    else
        return False
return False
```

Then we can obtain the optimal designs within a given finite set of points $\hat{\mathbb{X}}$ as follows:

Algorithm 4 Get optimal designs in $\hat{\mathbb{X}}$ according to $\hat{\preceq}_{(1,\ldots,n)}$: $\min(\hat{\mathbb{X}}, \hat{\preceq}_{(1,\ldots,n)})$.

```
$\mathbb{S} = \{\}$
**for** $x_i \in \hat{\mathbb{X}}$ **do**
    dominated $\leftarrow$ False
    **for** $x_j \in \hat{\mathbb{X}}$ **do**
        **if** $x_j \hat{\preceq}_{(1,\ldots,n)} x_i$ and $x_j \hat{\not\succeq}_{(1,\ldots,n} x_i$ (using Algorithm 3) **then**
            dominated $\leftarrow$ True
    **if** not dominated **then**
        $\mathbb{S} \leftarrow \mathbb{S} \cup \{x_i\}$
return S
```

2.4 Implementation of the combined binary relation within optimizers

Here we adapt two optimization heuristics in particular to solve our proposed optimization formulation: a multi-objective genetic algorithm (NSGA-II) [15], and a multi-objective particle swarm (MOPS) [18].

When using evolutionary algorithms, we could use only the current population in optimizers as $\hat{\mathbb{X}}$ to get approximations to these ranking functions, but this can lead to deterioration—designs visited later in an optimization being dominated by previously visited designs [19]. Therefore in this paper, we use the set of every design visited during the optimization as $\hat{\mathbb{X}}$.

In general the set $\mathbb{X}_{\text{optimal}}$ from Equation 5 will consist of multiple designs, and so population based optimizers that search for a set of designs in a single run are well suited to solving the optimization formulation considered in this paper. Furthermore, engineering design considers a broad range of applications whose design spaces often lack exploitable structure (linearity, convexity, etc.) [1], and such optimization heuristics are able to effectively explore such design spaces.

The two chosen optimizers (NSGA-II and MOPS) can be adapted to use the combined binary relation of Equation 8 via the following two modifications:

- After the system model has been evaluated for each new candidate population, we evaluate the ranking function of every design visited so far during the optimization for each $\preceq_k, k = 1, \ldots, n$, using Algorithm 1 or 2.
- Whenever a design is normally compared using Pareto dominance, we instead use the combined binary relation of Equation 8.
  - In NSGA-II, Algorithm 4 is used when selecting which designs to breed (the selection step) and which designs to keep in subsequent populations (the replacement step).
  - In MOPS, Algorithm 4 is used when finding the local and global best particles.

The computational cost of ranking each design in $\hat{\mathbb{X}}$ using each of these approximations is $O(nMN^2)$, where N is the number of designs in $\hat{\mathbb{X}}$, n is the number of individual binary relations used, and M is the number of values that define each relation. In many engineering design problems, the computational cost of evaluating $f(x)$ even once will outweigh the cost of doing this ranking, since we can usually only afford $N \simeq O(100)$ to $O(1000)$ evaluations for an optimization.

A limitation of this implementation is therefore memory usage. It relies on building up an approximation to design space ($\hat{\mathbb{X}}$) over the course of an optimization, so requires all previously visited designs and their performance to be recorded. For practical engineering design problems, the limitation of the number of design points that can be explored is usually dominated heavily by the time taken to run the simulation, and so this should not be a concern. However, if an optimization problem is expected to be run for a very high number of iterations, then storing every previously visited design point may not be practical, and so alternative approaches may be preferable. This is an interesting avenue for future work: which design points should be archived, and can we achieve convergence with a finite archive size?

2.5 Illustrative Example

We could find designs in $\mathbb{X}_{\text{optimal}}$ from Equation 5 by doing a standard multi-objective optimization of $\preceq_1$, then choosing designs within this set according to the other binary relations, but doing so can be prohibitively costly when evaluating $f(x)$ is computationally expensive. The purpose of defining $\preceq_{(1,\ldots,n)}$ and its approximation in Equation 8 is to accelerate the convergence of optimizers towards designs in $\mathbb{X}_{\text{optimal}}$, allowing such designs to be found in feasible computational times in challenging engineering design problems.

To demonstrate the acceleration this provides to optimizers, consider the following 4D version of the test problem used in Example 2.5, over a design space $\mathbb{X} = [-5, 5]^4$:

$$\begin{aligned} f_1 &= \sqrt{||x_1 - 4||^2 + 25||x_2 - 0||^2 + 25||x_3 - 0||^2 + 25||x_4 - 0||^2}, \\ f_2 &= \sqrt{||x_1 + 4||^2 + 25||x_2 - 0||^2 + 25||x_3 - 0||^2 + 25||x_4 - 0||^2}, \\ f_3 &= \sqrt{25||x_1 - 3||^2 + 25||x_2 - 0||^2 + 25||x_3 - 0||^2 + ||x_4 - 1||^2}, \end{aligned}$$

where $||.||$ indicates the L_2 norm.

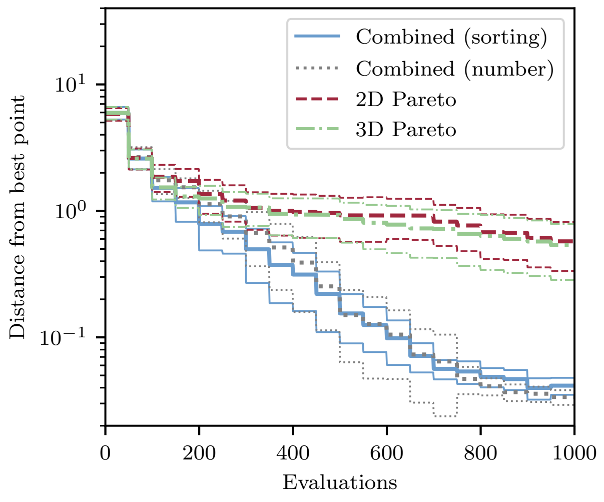
Similarly to Example 2.5, the optimized design (the design in the singleton set $\mathbb{X}_{\text{optimal}}$) is the design in the Pareto front of f_1 and f_2 with the lowest value of f_3, which can be found analytically to be $x = [3, 0, 0, 1]$. We consider using the following three binary relations in an optimization to find this design: Pareto dominance of f_1 and f_2; Pareto dominance of f_1, f_2, and f_3; the combined binary relation $\preceq_{(1,2)}$ with $\preceq_1$ indicating Pareto dominance of f_1 and f_2, and $\preceq_2$ indicating *less than of equal to* using f_3.

We use a population size of 50 with a computational budget of 1000 designs evaluated, and run each optimizer 20 times using the same set of randomly generated initial design points for each of the three cases. Figure 6 shows convergence in terms of average Euclidian distance from $x = [3, 0, 0, 1]$. To evaluate the combined binary relation, we use both the ranking function that counts the number of times the current non-dominated set would be removed before a design is optimal as an approximation to distance ranking (labeled sorting), as well as the ranking function that counts the number of designs that dominate a design as an approximation to volume ranking (labeled number).

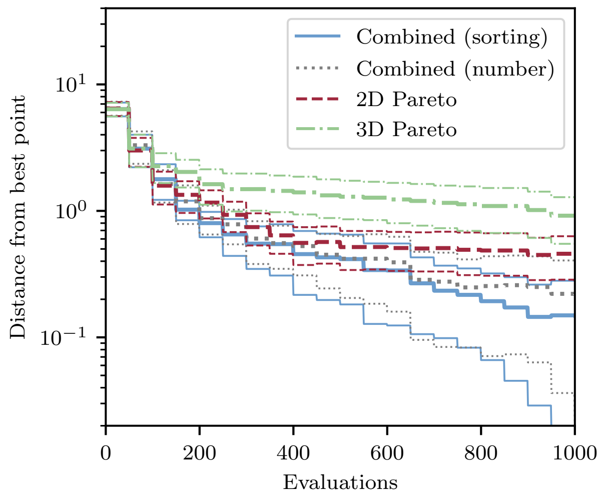
This acceleration is brought about because fewer designs are incomparable according to the combined binary relation $\preceq_{(1,\ldots,n)}$ than just $\preceq_1$, increasing the *selection pressure* exerted on an optimization algorithm to explore in a particular direction of design space [20]. This is because, given a collection of candidate designs, fewer of them are optimal, so an optimizer will consider fewer search directions in which to search for better designs.

The benefit of the proposed formulation is similar for both ranking functions, therefore for the remainder of this paper we just use the number approximation, as it is simpler to implement.

Note that the effect of this increase in selection pressure by reducing the number of incomparable designs is well recognized in the field of many-objective optimization, which has proposed various alternatives to Pareto dominance that extend better to more objectives [21, 22, 23, 24, 25]. Often this is achieved by trying to obtain information from a designer to quantify their underlying preferences using techniques from the



(a) Particle Swarm



(b) Genetic Algorithm

Figure 6: Convergence of optimizations on a simple test problem using Pareto dominance and the combined binary relation (±0.5 standard deviations shown as thin line).

field of multi-criteria decision making [26, 27, 28], such as goals [29], reference points [30], tradeoffs [31] etc. Physical programming [32, 33] similarly elucidates information from a designer to convert a multi-objective problem into a single-objective one.

In contrast, this paper is not trying to quantify designers' preferences (which can be difficult to do, as exemplified by the vast range of techniques available to try to achieve this); instead we propose that some design problems (such as the motivating examples given in the introduction) are usefully expressed in such a way that cannot be solved efficiently using existing optimization formulations, so we define the combined binary relation $\preceq_{(1,2,\ldots,n)}$ to efficiently solve such design problems using optimizers.

Still, a limitation is that the designer still has to decide on which binary relations to use within this formulation, and a hierarchy to place them in. The approach offers more power through more flexibility, but it may be a challenge to set up appropriate optimization problems.

2.6 Constraint Handling

In the problem definition, we defined $\mathbb{X}$ to implicitly include all constraints on the design, but it is straightforward to include constraints in the formulation by setting the first binary relation $\preceq_1$ to be any preorder whose optimal designs are the feasible set of designs. This includes many of the constraint handling approaches that have been proposed for evolutionary algorithms [34].

Denoting such a binary relation as $\preceq_{\text{constraint}}$, and the set of feasible designs as $\mathbb{X}_{\text{feasible}}$, our optimal set of designs defined in Equation 5 becomes:

$$\begin{aligned}\mathbb{X}_{\text{optimal}} &= \min\left(\left(\ldots\min\left(\min\left(\mathbb{X},\preceq_{\text{constraint}}\right),\preceq_2\right),\ldots\right),\preceq_n\right)\\ &= \min\left(\left(\ldots\min\left(\mathbb{X}_{\text{feasible}},\preceq_2\right),\ldots\right),\preceq_n\right)\end{aligned}$$

Note that ranking methods have previously been proposed to handle constraints in evolutionary algorithms. For example constraints in Ref. [35] are handled by ranking populations according to how many other designs are more feasible in more constraints than each design in the population; the constraint handling method in Ref. [36] ranks populations according to constraint violation and according to Pareto dominance, and sums these ranks to assign fitness; and the method in Ref. [37] ranks populations according to each constraint function separately, then sums these ranks to determine the best infeasible solutions to include in the next population. These methods are specific to constraints with fixed thresholds and particular evolutionary optimizers, whereas in this paper we consider the more general problem of finding the set $\mathbb{X}_{\text{optimal}}$ defined in Equation 5.

Having formulated an optimization problem to find these designs, and discussed how optimizers can solve this formulation using finite approximations to design space, in Section 3 we apply this formulation to an algebraic design under uncertainty problem to illustrate its relevance to vector-valued performance measures. Then in Sections 4 and 5 we apply this formulation to the two motivating examples discussed in the introduction to illustrate the benefit of being able to solve this type of engineering design problem efficiently.

3 Design Under Uncertainty

In design under uncertainty problems, using a few scalar objectives such as statistical moments can miss out important behavior of the full probability distribution of the quantity of interest. This has motivated the development of several optimization formulations that consider the probability distribution in its entirety [38, 39, 40, 41]. Similarly in Ref. [8], where we used multiple dominance criteria whose optimal sets were guaranteed to have a non-empty intersection, giving a trivial overall preorder over design space to use in optimization.

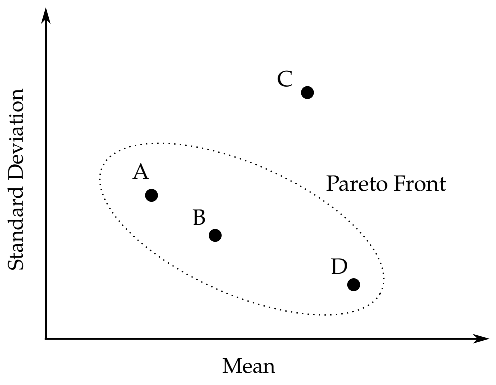
For example, if the cumulative distribution function (CDF) of a design lies entirely to the left of the CDF of another design, then the first design has a higher probability than the second design of achieving a lower value than any given value of the quantity of interest (the CDF of an uncertain quantity of interest X, evaluated at x, gives the probability of a X taking any value less than x). This is known as stochastic dominance, and is not taken into account if only statistical moments are considered, as illustrated in Figure 7, where moments and the CDF are shown for four hypothetical designs. Designs A, B and D are all on the Pareto front of moments but design D is guaranteed to give a worse quantity of interest than design A or B. Furthermore, the CDF of design B does not cross that of design A, so is stochastically dominated.

In many design cases, we would not be interested in stochastically dominated designs, but robustness (indicated by low values of the standard deviation of the quantity of interest) is still a desirable quality, so we may want to find the most robust design that is not stochastically dominated.

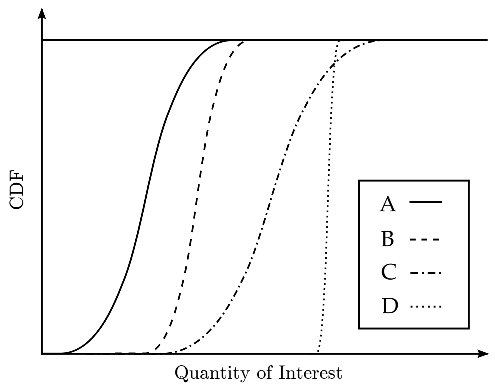
We cannot formulate this problem using the approaches of Ref. [8], but can using the approach developed in this paper. To demonstrate, we consider the following test problem:

$$g(x,u) = 9 + 2\left(5u_1u_2x_1 + x_2u_2^2 + x_2u_2^3 + 2.5||x - [0.1, 0.4]||^2\right), \tag{9}$$

where $x \in [-0.5, 0.5]^2$ is the vector of design variables that can be controlled, and u is the realization of an uncontrollable random input, denoted U, which is uniformly distributed over $[-1, 1]^2$. We estimate the behavior of g under uncertainty using $M = 1000$ independent and identically distributed (i.i.d) sampled values from the distribution of U, denoted $u_i, i = 1, \ldots, M$. The function $g(x, u)$ is evaluated for a given x at each sampled value of U to give sampled values of the output, denoted g_i and given by $g_i = g(x, u_i), i = 1, \ldots, M$.



(a) Pareto Dominance



(b) Stochastic Dominance

Figure 7: Illustration of Pareto dominance of moments and stochastic dominance of corresponding CDFs for hypothetical designs (adapted from [8]).

From these sampled values we use the following objectives:

$$f_{\text{mean}} = \frac{1}{M}\sum_{i=1}^{M} g_i \tag{10}$$

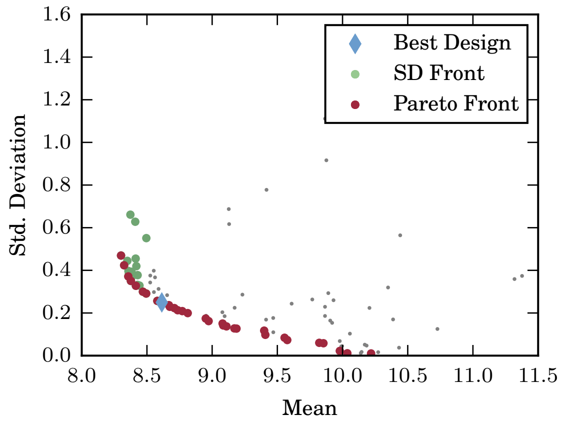
$$f_{\text{std}} = \sqrt{\frac{1}{M-1}\sum_{i=1}^{M}(g_i - f_{\text{mean}})^2} \tag{11}$$

$$f^j_{\text{cdf}} = g_i^{(j)} \tag{12}$$

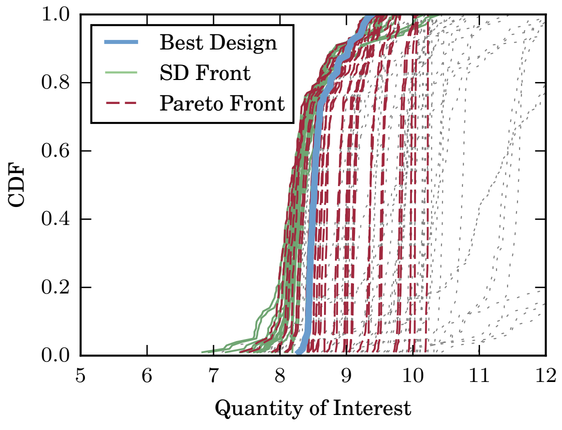
where $g_i^{(j)}$ denotes the j^{th} smallest value out of the sampled values $g_i, i = 1, \ldots, M$. We can then search for optimal designs of $\preceq_{(1,2)}$ with $\preceq_1$ indicating Pareto dominance of $f^j_{\text{cdf}}, j = 1, \ldots, M$, and $\preceq_2$ indicating lower (or equal) f_{std}. If we were to consider this as a many objective problem, we are dealing with 1002 separate objectives, and so trying to quantify a designer's preferences in order to weight these objectives would be infeasible.

We illustrate the benefit of our approach by running an optimization (particle swarm) with a strict computational budget (150 design points considered) finding optimized designs of three binary relations: Pareto dominance of moments (f_{mean} and f_{std} as objectives), stochastic dominance ($f^j_{\text{cdf}}, j = 1, \ldots, M$ as objectives), and the proposed combined binary relation $\preceq_{(1,2)}$. These represent three methods of finding the design specified in the problem formulation: that with the lowest f_{std} that is stochastically non-dominated. Figure 8 gives the optimized designs when just using moments, Figure 9 gives the optimized designs when using stochastic dominance, and Figure 10 gives the optimized designs when using the combined binary relation. In each case we highlight in green the non-dominated designs with respect to stochastic dominance (labeled SD Front), the Pareto non-dominated designs in red (labeled Pareto Front), and the design on the stochastically non-dominated front with the lowest f_{std} out of those visited during the optimization in blue (labeled Best Design).

Using this computational budget, the stochastic dominance-based optimization focused on an area of design space giving moments in the upper left quadrant of Figure 9a, and so has found many designs with poor mean and variance but with a small chance of good performance (with long left-tailed CDFs). This has caused it to be unable to obtain designs close to the true best design under this computational budget.

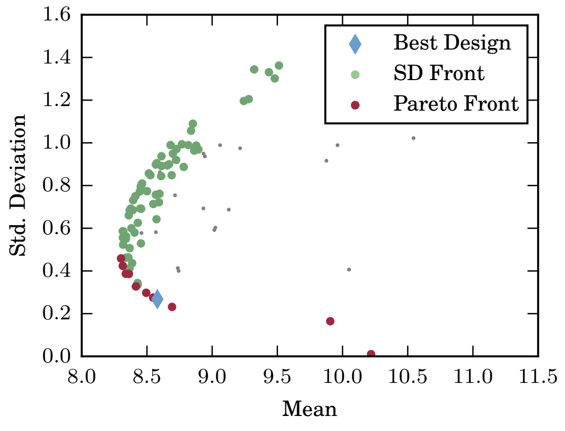


(a) Moments

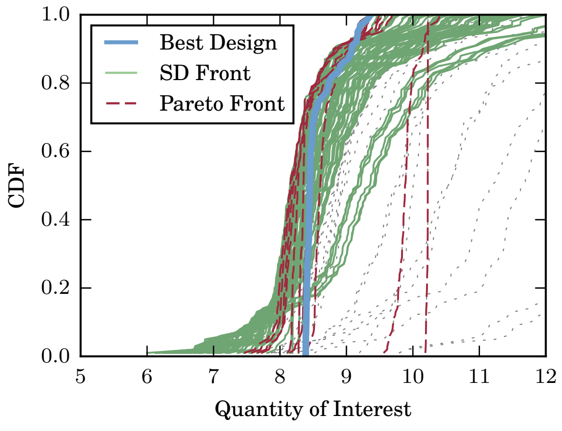


(b) CDFs

Figure 8: Moments and CDFs of designs visited when using Pareto dominance of moments in optimizations



(a) Moments



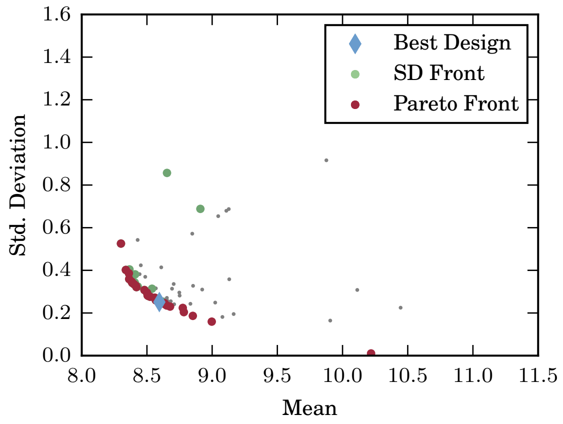
(b) CDFs

Figure 9: Moments and CDFs of designs visited when using stochastic dominance in optimizations

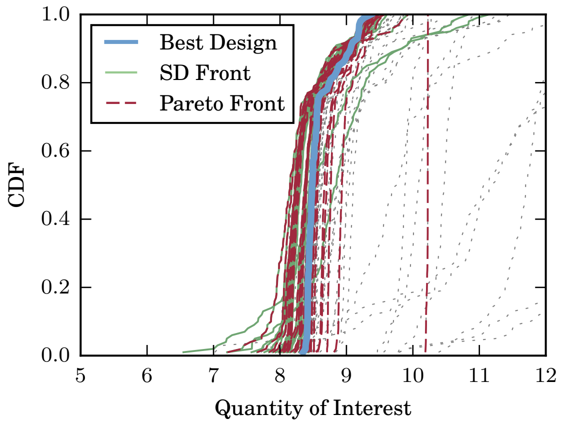
Similarly, the moment-based optimization focused on the bottom right of Figure 8a, where designs with low variance are located (but which we can see from the CDFs are stochastically dominated), also causing it to miss the designs specified in the problem formulation. In contrast, by using the combined binary relation, the optimization was able to focus in around designs of most interest using the given computational budget.

To quantify this behavior, we run 20 of these optimizations, and plot the average distance of the best point found so far by the optimizer to the true best design vs. number of function evaluations on Figure 11.

For both optimizers, using the combined binary relation accelerated the convergence of the optimizers towards the best design specified in the problem formulation. Considering the average distance for the moment-based formulation after the budget has been exhausted, using the combined binary relation achieves this average distance in under half the number of function evaluations, and in fact with the particle swarm optimizer we see computational savings of $\simeq 70\%$.

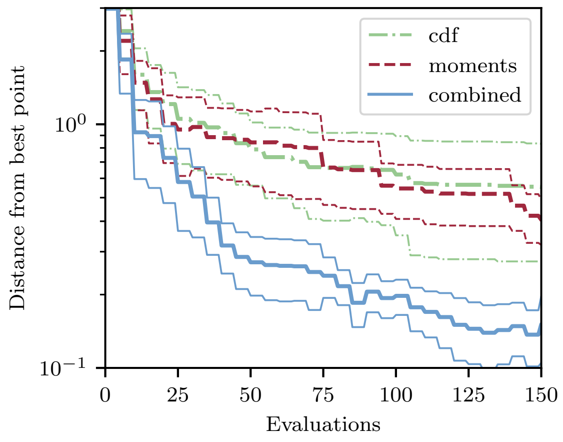


(a) Moments

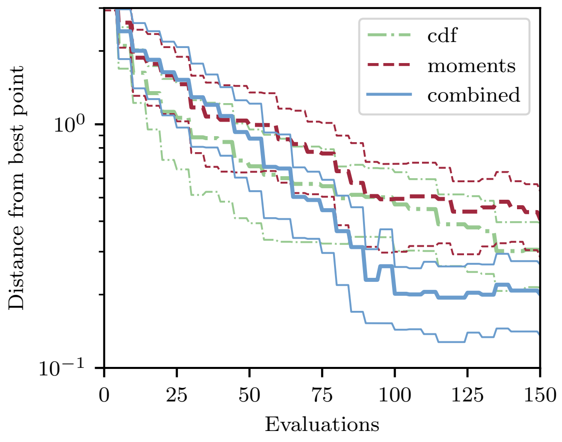


(b) CDFs

Figure 10: Moments and CDFs of designs visited when using the combined binary relation in optimizations



(a) Particle Swarm



(b) Genetic Algorithm

Figure 11: Using the combined criteria to accelerate optimizations towards the most robust design that is not stochastically dominated (± 0.5 standard deviations shown as thin line).

4 Half Car Model

To investigate the usefulness of this formulation for engineering design, we first consider the problem of designing a car suspension system to maximize ride comfort and road holding. This is a problem that is well known to have many conflicting objectives, and which are the most appropriate objectives to use in optimizations is not well established [6]. Therefore multi-objective optimization approaches have been proposed for the design of suspension systems [4]. The difficulty in choosing objectives effectively has also led to techniques from many-objective optimization being proposed [5].

Here we consider the half-car suspension system design problem from [5], and compare our approach to this design problem to the three-objective optimization approach proposed in that paper. This problem considers the design of two vibrating systems: a half-car suspension system modelled by mass-spring-damper subsystems for the road, car body and tyres, and a person located above the centre of gravity of the car modelled by a collection of mass-spring-damper subsystems. These two systems are treated as uncoupled, such that the response of the car to the road is used as a separate input to drive the response of the person.

The equations of motion for these two systems can be derived from the diagram given in Figure 12, as detailed in Ref. [5]. We use the same values for the stiffness, mass, and damping parameters as in Ref. [5]. We examine the response of the car to driving over a bump in the road represented by a half sinusoid of height 0.015m, as illustrated in Figure 13.

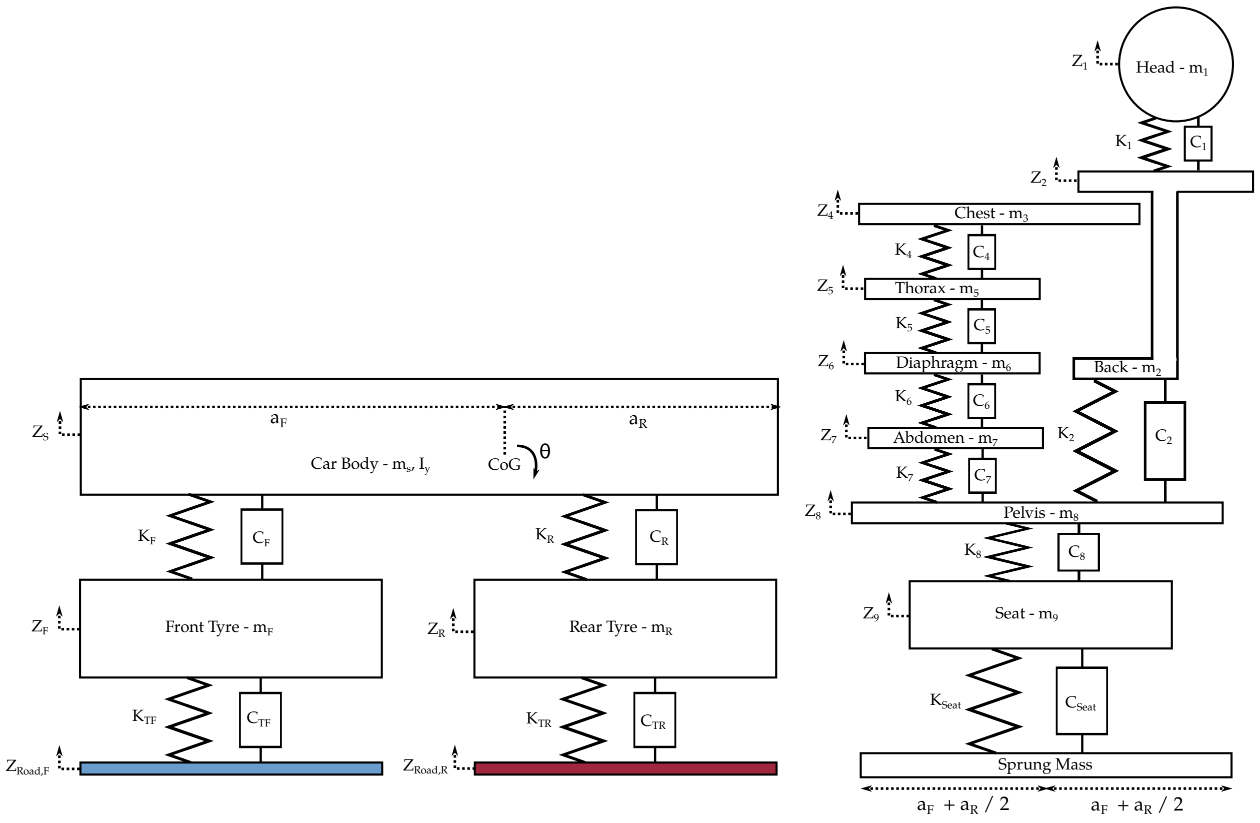


Figure 12: Diagram of half-car model. Car (left) and human (right). Adapted from [5].

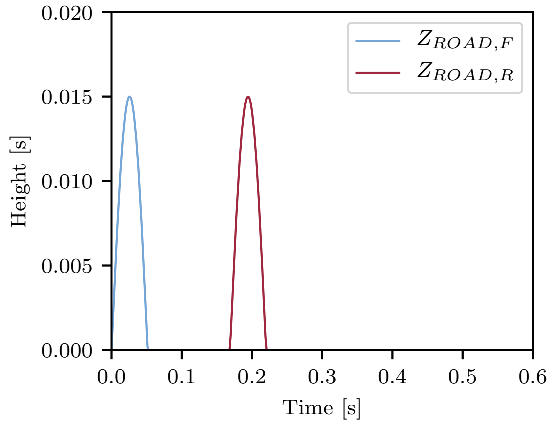


Figure 13: Input to the half car model

There are 6 common objectives to use when designing suspension systems of this type [4, 6, 5], given below:

$$\begin{aligned} f_1 &= \mathrm{var}(\ddot{z}_S) \\ f_2 &= 0.5\left(\mathrm{var}(ST_F) + \mathrm{var}(ST_R)\right) \\ f_3 &= 0.5\left(\mathrm{var}(TD_F) + \mathrm{var}(TD_R)\right) \\ f_4 &= VDV_{head} \\ f_5 &= CF_{head} \\ f_6 &= \mathrm{var}(\ddot{\theta}), \end{aligned}$$

where z_S is the displacement of the car body, ST_F is the suspension travel for the front wheel, TD_F is the tire deflection for the front wheel, VDV_{head} is the vibration does factor at the head, equal to $\left(\int \ddot{z}_1^4 dt\right)^{(1/4)}$ (where z_1 is the displacement of the head in the body model), and CF_{head} is the crest factor at the head, equal to $\max(\ddot{z}_1)/\mathrm{rms}(\ddot{z}_1)$ (where rms denotes the root mean squared). Objectives f_1, f_4 and f_5 are measures of the ride comfort, and objectives f_2, f_3, and f_6 are measures of the road holding of the vehicle.

In Ref. [5], they propose only optimizing objectives f_1, f_2, and f_3 to keep the problem of a reasonable size, and then using the k-optimality approach of Ref. [42] over all 6 objectives as a post-optimization analysis to determine the most desirable designs to present as the results of the optimization to a designer.

As an alternative, we define three binary relations: Pareto dominance using objectives f_1 and f_2 (denoted $\preceq_1$), Pareto dominance using objectives f_3 and f_4 (denoted $\preceq_2$), and Pareto dominance using objectives f_5 and f_6 (denoted $\preceq_3$). We use optimization to search for designs in the following set: $\min(\mathbb{X}, \preceq_{(1,2,3)})$.

These binary relations are chosen because the main trade-off for this problem is between ride comfort and road holding, and so the first binary relation uses the two primary objectives that measure each of these two features, then the second and third binary relations are chosen as one more each from the objectives that measure these features.

The design variables for this optimization are given in Table 1, along with their bounds and initial values.

Name	Notation	Initial Value	Lower Bound	Upper Bound
Rear wheel linear stiffness	K_R	$6e4$	$3.2e4$	$1.5e5$
Rear wheel non-linear stiffness	$K_{R,NL}$	$1e6$	$5e5$	$3e8$
Rear wheel damping factor	C_R	$6e3$	$2e3$	$1e4$
Rear wheel mass	m_R	100	50	100
Front wheel linear stiffness	K_F	$6e4$	$3.2e4$	$1.5e5$
Front wheel non-linear stiffness	$K_{F,NL}$	$1e6$	$5e5$	$3e8$
Front wheel damping factor	C_F	$6e3$	$2e3$	$1e4$
Front wheel mass	m_F	50	50	100

Table 1: Optimization Design Variables and Bounds

We run the NSGA-II algorithm with a population size of 50 for 30 generations, for three optimization formulations, and plot the objectives of the optimized designs on Figure 14. The first case uses Pareto dominance of the three objectives f_1, f_2 and f_3, the second case uses Pareto dominance of the four objectives f_1, f_2, f_3 and f_4, and the third cases uses our proposed approach as described above (labelled "Search"). The same initial population is used for all three cases.

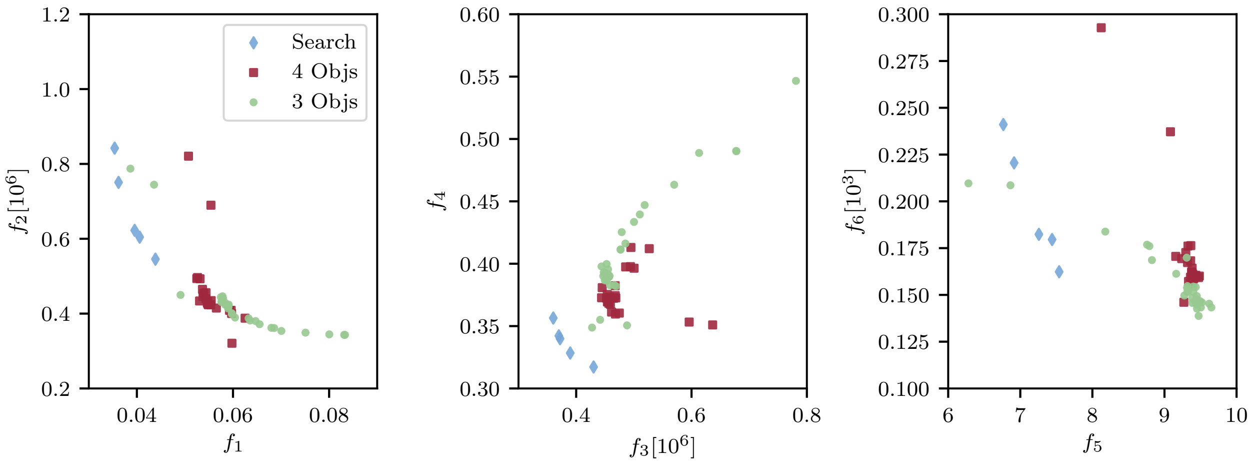


Figure 14: Pareto fronts for the scalar objectives

By looking at the objectives f_1 and f_2 (left figure) on Figure 14, we can see that the three objective optimization and our approach obtained designs on two separate sections on this Pareto front. However, when looking at the objectives f_3 and f_4 (centre figure), it is more evident that our approach was focussing on a smaller area of design space, and has consequently found designs that Pareto dominate those found by the other two cases under objectives f_3 and f_4. The objectives f_5 and f_6 weren't considered by the two standard multi-objective optimizations and were considered at the bottom of the hierarchy in our approach, but the right figure still illustrates how the different cases ended up with designs in different regions of design space.

To investigate the time and frequency response of designs resulting from these optimizations, one design from the middle of the Pareto front for each of the three optimizations are selected. The time series response and frequency spectrum are given for $\ddot{z}_S$ on Figure 15, and for z_F on Figure 16.

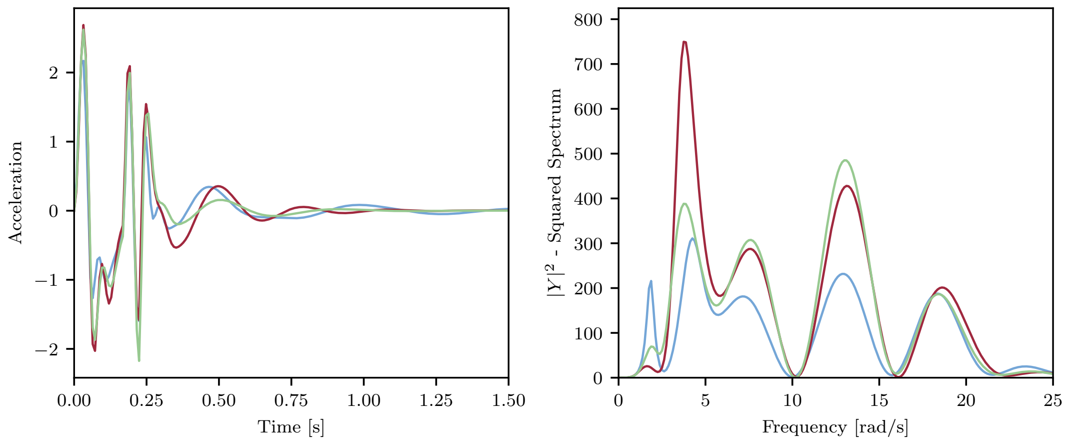


Figure 15: $\ddot{z}_S$ time response and frequency spectrum for optimized designs.

Furthermore, we evaluate the k-optimality of the optimized solutions (an approach proposed in Ref. [42], and used in Ref. [5] for this half-car problem): a design is k-optimal if it is not Pareto dominated by any other

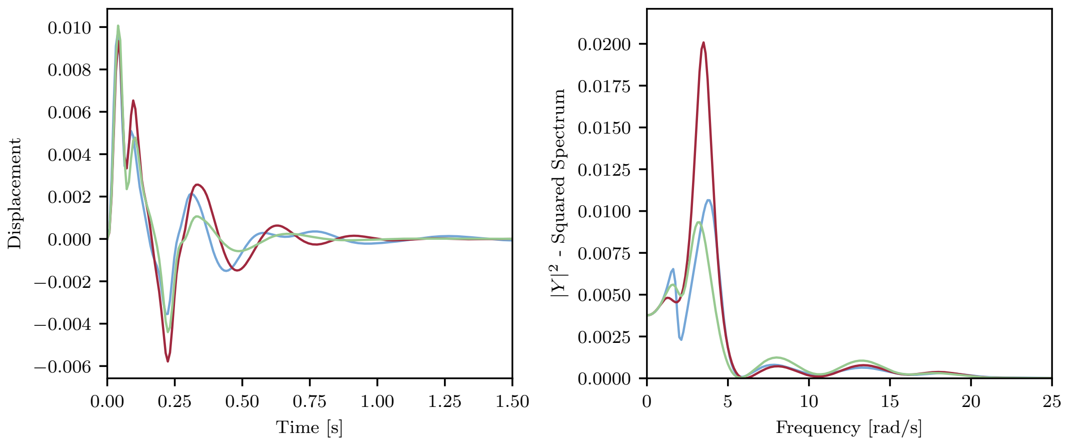


Figure 16: z_F time response and frequency spectrum.

solution for all possible k-element subsets of objectives. The maximum k-optimality value of any solution in blue is 3, whereas the maximum of any solution of the other two fronts is 2. This suggests that a superior design has been obtained using our combined binary relation than the other two optimization formulations.

5 Transonic Airfoil Design

In transonic airfoil design, key design drivers are increasing lift and reducing drag, but other important considerations are trailing edge separation and pitching moment. Low trailing edge separation ensures that the aerofoil is not close to buffet (a dangerous aero-elastic instability in the flow). Low pitching moment ensures that drag induced by the need to trim the aircraft (counteracting this pitching moment using the tail) is also low.

As discussed in the introduction, the appropriate threshold to use if separation and pitching moment were constrained is difficult to know a priori, and using all four objectives in a 4D multi objective optimization may be too computationally expensive. Therefore it would be useful to be able to take account of separation and pitching moment in an optimization in such a way that doesn't require them to be constrained, and that can obtain satisfactory designs under a strict computational budget.

We propose this can be achieved by searching for designs on the Pareto front of separation and moment *within* the designs on the lift/drag Pareto front. The formulation developed in this paper can do so efficiently by using optimizers to search for optimal designs with respect to $\preceq_{(1,2)}$, with $\preceq_1$ indicating Pareto dominance of lift and drag, and $\preceq_2$ indicating Pareto dominance of trailing edge separation and pitching moment. In this section we demonstrate how this formulation can obtain airfoils with better characteristics than a standard multi-objective optimization formulation using the same computational budget.

5.1 Setup

We consider designing an airfoil at a fixed angle of attack of 3°, a free stream Mach number of $M = 0.8$, and a Reynolds number of 6.5×10^6. We use the open source CFD solver SU2 (along with an Spalart-Allmaras turbulence model) to evaluate the performance of a given airfoil [43], which has been validated on various aerodynamics problems, making it suitable for the purposes of this study. SU2 directly outputs lift, drag, and moment coefficients, and as a measure of separation we evaluate the total area under the graph of distance along surface vs. surface skin friction whenever skin friction is negative.

We parameterize a design space by perturbing a baseline mesh (that is provided with SU2) using a series of Hicks-Henne bump functions on the upper and lower surface, using SU2's built in mesh deformation capability. The mesh for the baseline geometry is shown on Figure 17.

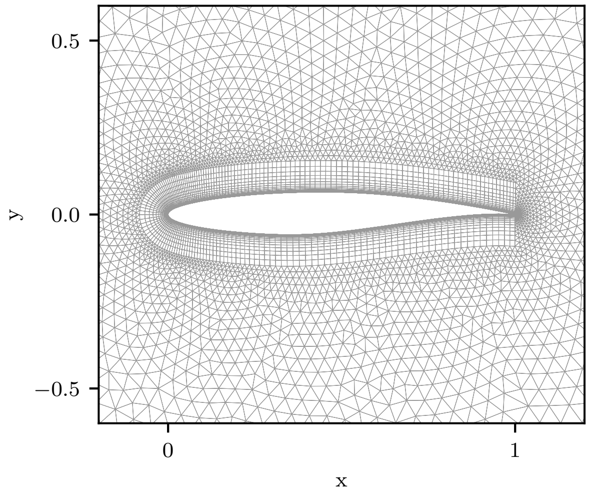


Figure 17: Baseline Mesh

We also constrain the shape of the airfoil such that a box of dimensions 0.5×0.07 chord lengths (which just fits inside the baseline geometry) must fit inside any new airfoil, to represent structural requirements.

We use ten design variables that represent the magnitude of the Hicks-Henne bump functions. The locations on the airfoil and bounds of the ten design variables are given in Table 2.

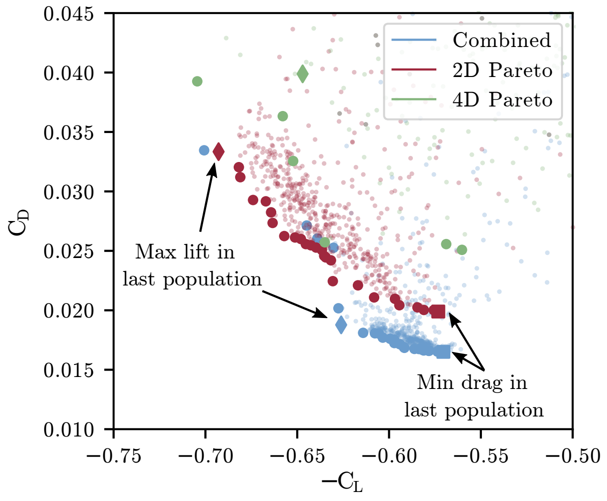
Location [x/chord]	Surface	Lower Bound [$\times 10^{-3}$]	Upper Bound [$\times 10^{-3}$]
0.05	Upper	-10	10
0.2	Upper	-15	15
0.45	Upper	-15	15
0.7	Upper	-10	15
0.85	Upper	-5	10
0.05	Lower	-10	10
0.2	Lower	-15	15
0.45	Lower	-15	15
0.7	Lower	-10	15
0.85	Lower	-5	10

Table 2: Location on the airfoil of the ten Hicks-Henne bump functions, and bounds on their magnitude.

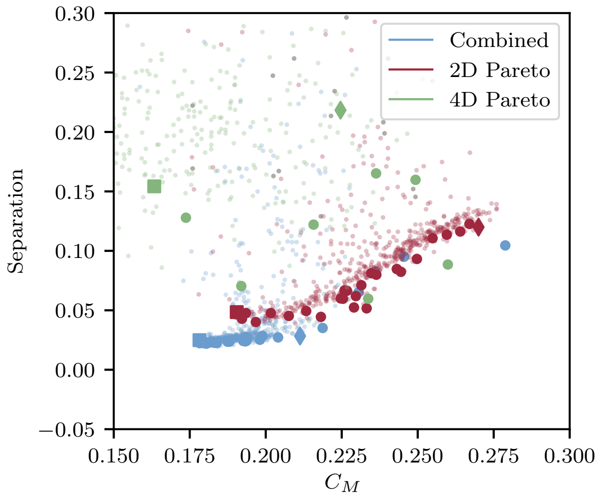
We use the particle swarm optimizer with a computational budget of 600 CFD evaluations, and run this optimizer to solve three formulations (using the same initial population in each case): multi-objective optimization of just lift and drag (labeled 2D Pareto), multi-objective optimization of lift, drag, moment, and amount of separation (labeled 4D Pareto), and our combined binary relation using Pareto dominance of lift and drag as $\preceq_1$, and Pareto dominance of moment and separation as $\preceq_2$ (labeled combined).

5.2 Results

On Figure 18, we plot the designs visited during these three optimization cases as small dots, and we plot the resulting Pareto front of lift and drag as large circles (except in the optimization using of all four objectives, we plot as large circles the Pareto front of lift, drag, separation, and moment). We also plot the minimum drag design from the last few populations of each optimization as a square, and the maximum lift design as a diamond.



(a) Lift and Drag



(b) Moment and Separation

Figure 18: Objectives of designs resulting from the three transonic airfoil design optimization cases.

Using four objectives in Pareto dominance means that under the given computational budget, the optimizer was not able to converge significantly towards designs on the actual Pareto front of lift and drag, causing it to end up with designs that are dominated in all four objectives.

The main comparison is between the 2D Pareto front of lift and drag and that of the combined criterion, and there are two principal observations to be made. The combined criterion focused the search in lift/drag space to only a particular region of the trade-off (lower drag and lower lift), enabling it to converge to better designs in this part of design space, meaning that nearly every design in the Pareto front of lift/drag from this optimization dominates a design on the Pareto front when using just lift and drag on Figure 18a.

Secondly, because we have taken these objectives into account in the combined criterion, on Figure 18b the designs resulting from this optimization have a better moment and separation than those from the standard optimization, also dominating many of those designs (in red) under these two objectives.

This clearly illustrates the advantage of using such a combined criterion when we have a strict computational budget: we focus the search towards the regions of most interest by taking account of more of the information available to a designer, allowing us to reach the designs of most interest more quickly.

To further investigate the airfoils resulting from this optimization, we consider in more detail the minimum drag and maximum lift designs from the standard multi-objective optimizations using Pareto dominance, and those from using the combined criterion. On Figure 19 we plot the airfoil shapes, on Figure 20a we plot the (negative) upper and lower surface pressure distribution, and on Figure 20b we plot the upper surface friction distribution.

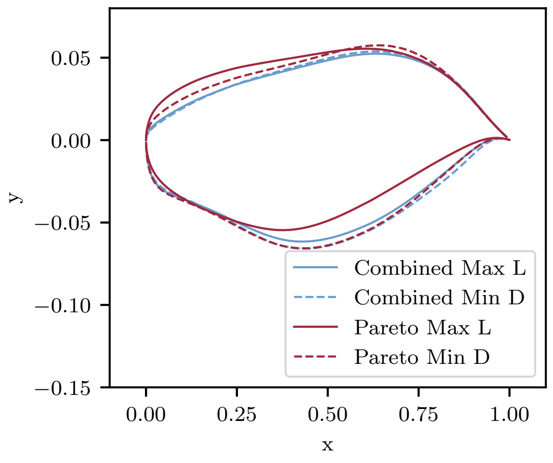
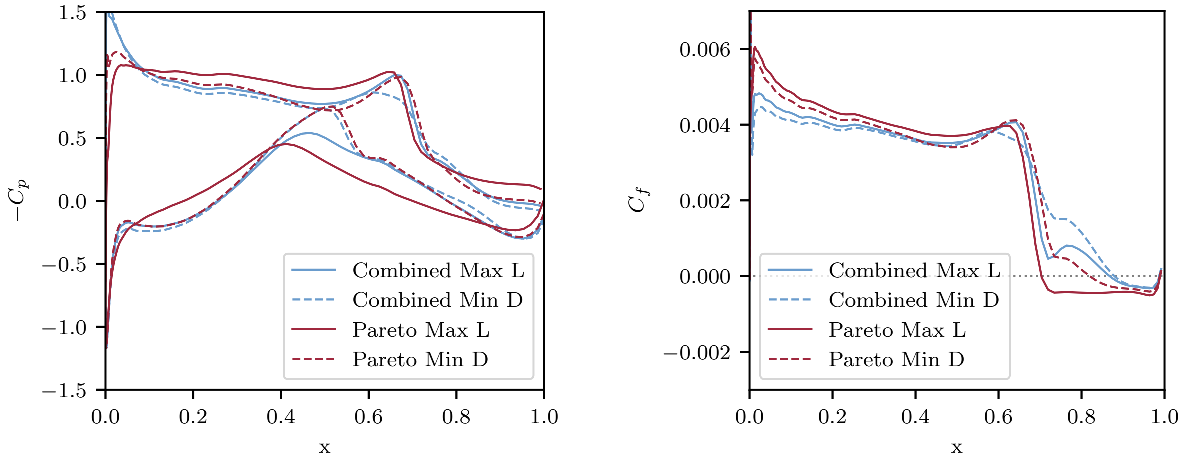


Figure 19: Airfoil Shapes



(a) Pressure distributions

(b) Upper surface friction distributions

Figure 20: Pressure and friction distributions for selected airfoils

Figure 19 illustrates that the four selected airfoils have noticeably different shapes, indicating that the optimizers were exploring different regions of design space in each case; Figure 20b illustrates the lower trailing edge separation achieved by the designs found using the combined binary relation compared to those found using Pareto dominance. Furthermore, Figure 20a shows that the minimum drag design found using the combined binary relation has a smaller shock than the other three, which along with the lower skin friction towards the leading edge seen on Figure 20b, explains the improvement in drag and trailing edge separation behavior compared to the other designs. The more desirable pressure and friction distributions exhibited by the designs found using the combined binary relation are a result of the optimization being able to more thoroughly explore this region of design space compared to the Pareto dominance optimization. This in turn is a result of the combined binary relation having fewer incomparable designs than Pareto dominance, driving the optimizer towards this region of design space faster, illustrating the advantage of being able to express design problems in the form of Equation 5 and the utility of the formulation developed in this paper for solving such problems efficiently.

6 Conclusion

We have proposed an optimization formulation that searches for optimal designs with respect to a binary relation *within* the set of optimal designs with respect to another binary relation. In engineering design problems with many quantities of interest, this formulation allows quantities that would otherwise be discarded to be included in an optimization, without reducing the tractability of the problem or constraining the quantities at arbitrary values.

We developed a method of solving this formulation in a *single* optimization, reducing the computational cost required to find designs that are solutions to this formulation. The method defines a *combined* binary relation using a hierarchy of ranking functions; the optimal designs with respect to this combined binary relation are solutions to the proposed formulation. We then illustrate how this combined binary relation can be usefully approximated for use within optimizers. This opens up more powerful ways of framing design problems that previously could not be solved efficiently using optimization.

We applied the developed method to two such design problems. First, we applied it to an algebraic design under uncertainty problem: finding the most robust design that is not stochastically dominated. It achieved computational savings of over 70% compared to a standard multi-objective optimization.

Second, we applied it to a transonic airfoil design problem in which it is desirable to account for the amount of trailing edge separation and pitching moment but we do not know appropriate values by which to constrain them. Here, given the same computational budget as a standard multi-objective optimizer, the new formulation obtained designs that have both a better lift and drag as well as better moment and separation than the nearest designs found using the multi-objective optimization. This resulted from optimizers using the combined binary relation being able to focus the search onto more desirable regions of the lift/drag Pareto front in order to make better use of the computational budget.

A primary avenue for future work is to investigate the effectiveness of other types of optimizer at solving our proposed formulation, particularly those that already make use of a finite representation of design space (such as entropy search [16, 17]). Furthermore, future work will investigate more efficient single-objective optimizers in the case where we are only searching for a single design (like in the design under uncertainty application). In this case, the combined binary relation gives a total order over design space, and so we should be able to make use of efficient single-objective optimization algorithms to find this design more quickly than population-based heuristics.

Additionally, it would be useful to investigate more advanced ways of selecting which designs to include in our approximation of design space when evaluating the approximate ranking functions, since in problems where we can afford more system model evaluations, storing every visited design to use as an approximation of design space might become prohibitively expensive.

Appendix

Here we show that that 1) the combined binary relation ($\min(\mathbb{X}, \preceq_{(1,\dots,n)})$) from Equation 6 is a preorder, such that it doesn't violate transitivity and so $\min(\mathbb{X}, \preceq_{(1,\dots,n)})$ exists for any given design space $\mathbb{X}$, and 2) $\min(\mathbb{X}, \preceq_{(1,\dots,n)})$ is the set $\mathbb{X}_{\text{optimal}}$ defined in Equation 5.

Note that comparison of R_k is not a sufficient condition for $\preceq_k$, i.e. $R_k(x) \leq R_k(y)$ does not in general imply $x \preceq_k y$. Condition 2 states within any subset of design space, the ranking of any optimal designs with respect to a given relation must be equal.

The combined binary relation is based on lexicographic ordering, so is a preorder, but we also give a proof here:

Lemma 6.1. The combined binary relation $\preceq_{(1,\dots,n)}$ is a preorder over $\mathbb{X}$, meaning it is reflexive[4] and transitive[5].

Proof. For each $k \in 1,\dots,n$: $R_k(x) = R_k(x)\ \forall x \in \mathbb{X}$, so from Definition 2.7, $\preceq_{(k,\dots,n)}$ is reflexive if $\preceq_{(k+1,\dots,n)}$ is reflexive. Furthermore, if $x \preceq_{(k,\dots,n)} y \cap y \preceq_{(k,\dots,n)} z$ for $x, y, z \in \mathbb{X}$, then one of the following four cases is true:

- $R_k(x) < R_k(y) \cap R_k(y) < R_k(z) \implies R_k(x) < R_k(z) \implies x \preceq_{(k,\dots,n)} z$
- $R_k(x) < R_k(y) \cap R_k(y) = R_k(z) \implies R_k(x) < R_k(z) \implies x \preceq_{(k,\dots,n)} z$
- $R_k(x) = R_k(y) \cap R_k(y) < R_k(z) \implies R_k(x) < R_k(z) \implies x \preceq_{(k,\dots,n)} z$
- $R_k(x) = R_k(y) \cap R_k(y) = R_k(z) \implies x \preceq_{(k+1,\dots,n)} y \cap y \preceq_{(k+1,\dots,n)} z$

so $\preceq_{(k,\dots,n)}$ is transitive if $\preceq_{(k+1,\dots,n)}$ is transitive.

Therefore $\preceq_{(1,\dots,n)}$ is reflexive and transitive if $\preceq_{(2,\dots,n)}$ is reflexive and transitive, ..., which is true if $\preceq_{(n)}$ is reflexive and transitive, which is true because $x \preceq_{\emptyset} y$ is true $\forall x, y \in \mathbb{X}$. Therefore $\preceq_{(1,\dots,n)}$ is reflexive and transitive, and is thus a preorder. $\square$

Theorem 6.2. *The set of optimal designs with respect to the n binary relations $\preceq_k$, $k = 1,\dots,n$, min$((\dots \text{min}(\text{min}(\mathbb{X}, \preceq_1), \preceq_2)\dots), \preceq_n)$, is equivalent to the set of minimal elements of the binary relation $\preceq_{(1,\dots,n)}$, min$(\mathbb{X}, \preceq_{(1,\dots,n)})$.*

Proof. Let x^* be an arbitrary design in $\min(\mathbb{X}, \preceq_{(1,\dots,n)})$, so $\nexists y \in \mathbb{X}$ such that $y \preceq_{(1,\dots,n)} x^* \cap x^* \npreceq_{(1,\dots,n)} y$.
Assume $x^* \notin \min(\mathbb{X}, \preceq_1)$, then $\exists y \in \mathbb{X}$ such that $y \preceq_1 x^* \cap x^* \npreceq_1 y$, which from condition 1 implies $R_1(y) < R_1(x^*)$, implying $y \preceq_{(1,\dots,n)} x^*$, which contradicts x^* being in $\min(\mathbb{X}, \preceq_{(1,\dots,n)})$.
Therefore $x^* \in \min(\mathbb{X}, \preceq_1)$, and condition 2 implies that $\forall x \in \min(\mathbb{X}, \preceq_1), R_1(x^*) = R_1(x)$.
Assume $x^* \notin \min(\min(\mathbb{X}, \preceq_1), \preceq_2)$, then $\exists y \in \min(\mathbb{X}, \preceq_1)$ such that $y \preceq_2 x^* \cap x^* \npreceq_2 y$, which implies $R_1(x^*) = R_1(y) \cap R_2(y) < R_2(x^*)$, implying $y \preceq_{(1,\dots,n)} x^*$, which contradicts x^* being in $\min(\mathbb{X}, \preceq_{(1,\dots,n)})$.
Therefore $x^* \in \min(\min(\mathbb{X}, \preceq_1), \preceq_2)$, and condition 2 implies that $\forall x \in \min(\min(\mathbb{X}, \preceq_1), \preceq_2), R_2(x^*) = R_2(x)$.
This same argument holds up to $\preceq_n$, so $x^* \in \min((\dots \min(\min(\mathbb{X}, \preceq_1), \preceq_2), \dots), \preceq_n)$ $\square$

Note that another binary relation that combines binary relations could be constructed using lexicographic ordering of the binary relations $\preceq_k, k = 1,\dots,n$ themselves instead of ranking functions. Therefore one could try (as was briefly mentioned in Ref. [9], and similarly to the priority favor binary relation defined in Ref. [44]) to define a hierarchy of binary relations as follows:

$$x \preceq_{(1,\dots,n)} y \Leftrightarrow x \preceq_1 y \ \cup\ \big(x \parallel y \ \cap\ x \preceq_{(2,\dots,n)} y\big), \tag{13}$$

[4]A binary relation $\preceq$ is reflexive if $\forall\ x, x \preceq x$

[5]A binary relation $\preceq$ is transitive if $x_A \preceq x_B \ \cap\ x_B \preceq x_C \implies x_A \preceq x_C$

where $x \parallel y$ indicates x and y are incomparable. However this binary relation violates transitivity and so it is not a preorder. Therefore optimal elements with respect to it may not exist and it can give rise to cyclic behavior when used by an optimizer [9], so is not well suited for use in optimization.

References

[1] Keane, A. J. and Nair, P. B., *Computational Approaches for Aerospace Design: The Pursuit of Excellence*, Wiley, New York, 2005.

[2] Antoniou, A. and Lu, W.-S., *Practical Optimization*, Springer Science and Business Media, 2007.

[3] Enderton, H. B., *Elements of Set Theory*, Academic Press, 1977.

[4] Georgiou, G., Verros, G., and Natsiavas, S., "Multi-objective optimization of quarter-car models with a passive or semi-active suspension system," *Vehicle System Dynamics*, Vol. 45, No. 1, 2007, pp. 77–92.

[5] Papaioannou, G. and Koulocheris, D., "An approach for minimizing the number of objective functions in the optimization of vehicle suspension systems," *Journal of Sound and Vibration*, Vol. 435, 2018, pp. 149–169.

[6] Koulocheris, D. V., Papaioannou, G. D., and Christodoulou, D. A., "Assessment of the optimization procedure for the nonlinear suspension system of a heavy vehicle," *International Journal for Vehicle Mechanics, Engines and Transportation Systems*, Vol. 42, No. 2, 2016, pp. 19–35.

[7] Shapiro, A., Dentcheva, D., and Ruszczynski, A., *Lectures on Stochastic Programming: Modeling and Theory*, SIAM, 2009.

[8] Cook, L. W. and Jarrett, J. P., "Optimization Using Multiple Dominance Criteria for Aerospace Design Under Uncertainty," *AIAA Journal*, Vol. 56, No. 12, 2018.

[9] Zitzler, E., Thiele, L., and Bader, J., "On Set-Based Multiobjective Optimization," *IEEE Transactions on Evolutionary Computation*, Vol. 14, No. 1, 2010.

[10] Berghammer, R., Friedrich, T., and Neumann, F., "Convergence of set-based multi-objective optimization, indicators and deteriorative cycles," *Theoretical Computer Science*, 2012.

[11] Marler, R. and Arora, J., "Survey of Multi-Objective Optimization Methods for Engineering," *Structural and Multidisciplinary Optimization*, Vol. 26, No. 6, 2004, pp. 369–395.

[12] Goldberg, D. E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.

[13] Fonseca, C. M. and Fleming, P. J., "Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization," *Proceedings of the 5th International Conference on Genetic Algorithms*, 1993.

[14] Garza-Fabre, M., Pulido, G. T., and Coello, C. A. C., "Ranking Methods for Many-Objective Optimization," *Mexican International Conference on Artificial Intelligence*, 2009.

[15] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T., "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, Vol. 6, No. 2, 2002, pp. 182–197.

[16] Villemonteix, J., Vazquez, E., and Walter, E., "An informational approach to the global optimization of expensive-to-evaluate functions," *Journal of Global Optimization*, Vol. 44, No. 4, 2009, pp. 509–534.

[17] Hennig, P. and Schuler, C. J., "Entropy Search for Information-Efficient Global Optimization," *Journal of Machine Learning Research*, Vol. 13, 2012, pp. 1809–1837.

[18] Coello, C. A., Pulido, G. T., and Lechuga, M. S., "Handling Multiple Objectives With Particle Swarm Optimization," *IEEE Transactions on Evolutionary Computation*, Vol. 8, No. 3, 2004.

[19] Laumanns, M., Thiele, L., Deb, K., and Zitzler, E., "Combining convergence and diversity in evolutionary multiobjective optimization," *Evolutionary Computation*, Vol. 10, No. 3, 2002, pp. 263–282.

[20] Branke, J., "Consideration of Partial User Preferences in Evolutionary Multiobjective Optimization," *Multiobjective Optimization. Springer, Berlin.*, 2008, pp. 157–178.

[21] Batista, L. S., Campelo, F., Guimaraes, F. G., and Ramirez, J. A., "A Comparison of Dominance Criteria in Many-Objective Optimization Problems," *IEEE Congress of Evolutionary Computation, CEC*, 2011, pp. 2359–2366.

[22] Corne, D. W. and Knowles, J. D., "Techniques for Highly Multiobjective Optimisation: Some Non-dominated Points are Better than Others," *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation - GECCO*, 2007, pp. 773–780.

[23] Hester, P. T., Velasquez, M., and Hester, P. T., "An Analysis of Multi-Criteria Decision Making Methods," *International Journal of Operations Research*, Vol. 10, No. 2, 2013.

[24] Chand, S. and Wagner, M., "Evolutionary Many-Objective Optimization: A Quick-Start Guide," *Surveys in Operations Research and Management Science*, Vol. 20, No. 2, 2015, pp. 35–42.

[25] Li, B., Li, J., Tang, K., and Yao, X., "Many-Objective Evolutionary Algorithms: A Survey," *ACM Computing Surveys*, Vol. 48, No. 1, 2015.

[26] Rachmawati, L. and Srinivasan, D., "Preference Incorporation in Multi-objective Evolutionary Algorithms: A Survey," *IEEE International Conference on Evolutionary Computation*, 2006.

[27] Li, K., Chen, R., Min, G., and Yao, X., "Integration of Preferences in Decomposition Multiobjective Optimization," *IEEE Transactions on Cybernetics*, Vol. 48, No. 12, 2018, pp. 3359–3370.

[28] Coello, C. A. C. C., "Handling Preferences in Evolutionary Multiobjective Optimization: A Survey," *Proceedings of the 2000 Congress on Evolutionary Computation.*, 2000.

[29] Fonseca, C. M., Fleming, P. J., Fonseca, CM, Fleming, and PJ, "Multiobjective optimization and multiple constraint handling with evolutionary algorithms - Part I: A unified formulation," *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans.*, 1998.

[30] Deb, K. and Sundar, J., "Reference Point Based Multi-Objective Optimization Using Evolutionary Algorithms," *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, 2006, pp. 635–642.

[31] Ruiz, A. B., Saborido, R., and Luque, M., "A Preference-Based Evolutionary Algorithm for Multiobjective Optimization: The Weighting Achievement Scalarizing Function Genetic Algorithm," *Journal of Global Optimization*, Vol. 62, 2015, pp. 101–129.

[32] Messac, A., "Physical programming - Effective optimization for computational design," *AIAA Journal*, Vol. 34, No. 1, 1996, pp. 149–158.

[33] Messac, A., "From Dubious Construction of Objective Functions to the Application of Physical Programming," *AIAA Journal*, Vol. 38, No. 1, 2000, pp. 155–163.

[34] Mezura-Montes, E. and Coello Coello, C. A., "Constraint-Handling in Nature-Inspired Numerical Optimization: Past, Present and Future," *Swarm and Evolutionary Computation*, Vol. 1, No. 4, 2011, pp. 173–194.

[35] Coello Coello, C. A., "Constraint-Handling Using An Evolutionary Multiobjective Optimization Technique," *Civil Engineering and Environmental Systems*, Vol. 17, No. 4, 2000, pp. 319–346.

[36] Ho, P. Y. and Shimizu, K., "Evolutionary Constrained Optimization Using an Addition of Ranking Method and a Percentage-Based Tolerance Value Adjustment Scheme," *Information Sciences*, Vol. 177, No. 14, 2007.

[37] Ray, T., Singh, H. K., Isaacs, A., and Smith, W., "Infeasibility driven evolutionary algorithm for constrained optimization," *Studies in Computational Intelligence*, Vol. 198, 2009, pp. 145–165.

[38] Seshadri, P., Constantine, P., Icacarino, G., and Parks, G., "A Density-Matching Approach for Optimization Under Uncertainty," *Computer Methods in Applied Mechanics and Engineering*, Vol. 305, 2016, pp. 562–578.

[39] Quagliarella, D., Petrone, G., and Iaccarino, G., "Optimization Under Uncertainty Using the Generalized Inverse Distribution Function," *Modeling, Simulation and Optimization for Science and Technology. Springer, Dordrecht.*, 2014, pp. 171–190.

[40] Cook, L. W. and Jarrett, J. P., "Horsetail Matching: A Flexible Approach to Optimization Under Uncertainty," *Engineering Optimization*, 2017, pp. 549–567.

[41] Cook, L. W., Jarrett, J. P., and Willcox, K. E., "Extending Horsetail Matching for Optimization Under Probabilistic, Interval, and Mixed Uncertainties," *AIAA Journal*, 2017, pp. 849–861.

[42] Das, I., "A preference ordering among various Pareto optimal alternatives," *Structural Optimization*, Vol. 18, No. 1, 1999, pp. 30–35.

[43] Palacios, F., Colonno, M. R., Aranake, A. C., Campos, A., Copeland, S. R., Economon, T. D., Lonkar, A. K., Lukaczyk, T. W., Taylor, T. W. R., and Alonso, J. J., "Stanford University Unstructured (SU2): An Open-source Integrated Computational Environment for Multi-Physics Simulation and Design," *51st AIAA Aerospace Sciences Meeting*, Grapevine, Texas.

[44] Schmiedle, F., Drechsler, N., GroBe, D., and Drechsler, R., “Priorities in Multi-Objective Optimization for Genetic Programming,” *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation*, 2001.