Multiple Objective Latin Hypercube Designs for Computer Experiments

Damola Akinlana[1], Lu Lu[2]
[12]Department of Mathematics and Statistics, University of South Florida

Abstract

Optimal Latin hypercube designs (LHD) based on optimizing a single criterion are commonly used space-filling designs for computer experiments. For instance, the maximin-LHD (MmLHD) and minimax-LHD (mMLHD) are popular to ensure a good spread and coverage, respectively, across the full input space while achieving uniform projections in each univariate dimension. The maximum projection-LHD (MaxProLHD) ensures space-filling across all possible subspaces. This paper proposes optimal LHDs to achieve robust and balanced performance across multiple objectives. Using the Pareto optimization approach, we employed the column-wise exchange simulated annealing algorithm and the nondominated sorting genetic algorithm to generate optimal LHDs that simultaneously optimize multiple space-filling characteristics, and the efficiency of these algorithms are compared. The methods are illustrated with examples of varied dimensions of input factors, and the generated optimal LHDs are evaluated based on their performance across simulations with different response surface models and compared with optimal designs from single criterion optimization. The nondominated sorting genetic algorithm proved to be more efficient for generating optimal LHDs with balanced performance across multiple distance metrics.

Keywords: Space-filling designs, maximin distance, minimax distance, maximum projection, Pareto Front approach, Gaussian process models

1. Introduction

Computer experiments have been broadly used in scientific and engineering research to substantially reduce cost of physical experimentation and for situations where physical experiments are prohibitive. However, some applications involve complex computer models and exploration of large input spaces such that computer experiments can still be computationally expensive and time consuming. Fortunately, due to advancement in research brought about by the growth in computer power, techniques have been devised for the design and analysis of computer simulations to be successfully applied to a variety of problems in engineering (Viana, 2015). When simulations are time consuming, computer experiment techniques involve generating a surrogate model to replace the expensive simulation code (Bandler et al, 2004). Gaussian process is a popular family of surrogate models (Sacks et al, 1989) used in the field of computer experiments to replace the role of computer models and makes tasks like sensitivity analysis, uncertainty quantification, validation, and calibration feasible (see Santner et al. (2003) and Fang et al. (2006)).

The first major step for statistical modeling of computer simulations is careful planning and configuration of the inputs. Moreover, in practice, time and computational resources are limited, so an experimental design plays a crucial role in identifying a set of inputs at which the underlying computer simulation will be evaluated. Such a set of points is called the design of the computer experiment. In the vast literature about design for computer experiments, two categories of design criteria have been proposed for evaluating design performance: the geometric or distance-based criteria and the model-based criteria. The distance-based criteria focus on achieving maximum spread or coverage of the geometric locations of the observations in the design space in order to avoid missing any interesting features. On the contrary, the model-based criteria focus on obtaining good understanding of the underlying process based on certain assumptions of the underlying model to evaluate design performance. This paper considers the distance-based criteria, which do not rely on model assumptions and provide more robust performance across a variety of response surfaces.

Space-filling designs refer to the family of distance-based optimal designs. They generally offer a good spread or coverage across the experimental region and great flexibility in capturing different behaviors of responses in different areas of the design region. Depending on the choice of the distance criteria, the term “space-filling” could be used to describe different features for “filling” the design space. “Spacing-filling” could mean to maximally spread out the design points by avoiding any pair of design points being too close to each other, based on the assumption that the information about the response surface provided by two close points will be similar. The notion here is to place points in the experimental region such that they are as far apart as possible. Johnson et al. (1990) proposed the maximin distance criterion, which seeks to maximize the minimum distance between any pair of design points. The minimum distance among the points in the design is given by $min_{ij} d(x_i, x_j)$. Therefore, the maximin distance design can be found by maximizing this minimum distance:

$$\max_{D} \min_{i,j} d(x_i, x_j) \qquad (1)$$

Alternatively, “Space-filling” could also mean to provide the best coverage of the design space. That is, for any point in the experimental region, there should be a design point close by based on the belief that any interesting features of the true underlying relationship are likely to exist in any design region and hence, it is important to avoid missing out any large region of the design space. Let p represent the number of design factors and χ be the experimental region, which in most cases, can be scaled to a unit hypercube, $\chi = [0,1]^p$. Let $D' = \{x_1, \dots, x_n\}$ denote a design of n runs, where each point $x_i \epsilon [0,1]^p$. Then, for any point $x \epsilon [0,1]^p$, the distance to the nearest design point is given by $min_i d(x, x_i)$. Intuitively, the closer a point is to the nearest design point, the more likely it is to be predicted with better precision. Hence, the worst prediction is likely to occur at the point that is farthest from the nearest design point. This worst nearest distance is found by $max_{x \epsilon [0,1]^p} min_i d(x, x_i)$. Again, Johnson et al (1990) proposed the minimax distance criterion, which seeks the best coverage of the design space by minimizing the maximum distance of any arbitrary point in the design space to the nearest design point as in:

$$\min_{D} \max_{x \epsilon \chi} \min_{i} d(x, x_i) \qquad (2)$$

In terms of computing, a maximin distance design is much easier to construct than a minimax distance design because it only requires evaluating the distances among the design points and not the distances from the design points to a grid of points spanning the whole experimental region as in the case of minimax distance design.

Although, both maximin and minimax distance criteria ensure good space-filling property in the full dimensions of all p input variables, their performance when projected onto spaces formed by a subset of variables are not guaranteed (Joseph, 2016). This may not be desirable when there exists effects sparsity (Montgomery, 2017) which is common in computer experiments. In this case, among the many factors that are being explored, only a few may be active factors. It is possible that a space-filling design in the full space of all factors may not be space filling in the projected lower dimension of the active factors. Therefore, it is desirable to have good space-filling property when it is projected into the subspaces of variables. McKay et al (1979) proposed the Latin hypercube design (LHD), which avoids replications in single-dimensional projections of each design factor. To construct a LHD of n runs, the range of each of the factors is divided into n equally spaced intervals. Then, only one coordinate of a design point is sampled from each of the non-overlapping interval. Although the LHD is uniformly spread when projected into the univariate space of each input variable, a randomly selected LHD is not guaranteed a good space-filling property across the full dimension or any subspace of design region. For example, a LHD with n points evenly distributed along the $45°$ diagonal line in the unit square $[0,1]^2$ design space with two input variables defects a good coverage of the design space because all the design points are located on a line leaving the rest of the unit square unexplored.

To ensure a good space filling property for the full space and for projection into the univariate subspace, optimization of the maximin and minimax criterions respectively have been proposed for LHDs. Edwin and Dam (2008) investigated the minimax criterion for LHDs (mMLHD) in two dimensions. They suggested that design points for LHDs should be chosen such that the maximal distance of any point in the input space to the design is minimal. Alternatively, Morris and Mitchell (1995) proposed choosing an LHD that maximizes the minimum distance among the points, which is called a maximin Latin hypercube design (MmLHD). Their proposed criterion for searching for MmLHDs is given by:

$$\min_{D} \left\{ \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{1}{d^k(x_i, x_j)} \right\}^{1/k}. \qquad (3)$$

As $k \to \infty$, this criterion becomes the maximin distance criterion in (1). They suggested using the smallest value of k that gives a maximin distance design because such a design will tend to have fewer pairs of points with minimum distance, which is referred to as the index of the maximin distance design.

For both MmLHD and mMLHD, the Latin hypercube structure ensures good one-dimensional projections, whereas the maximin and the minimax distance criterions ensure good space-filling in the full dimensional space. Particularly the MmLHD focuses on spreading any pair of design points as far as possible and hence tend to push more points towards the edge of the design space, while the mMLHD focuses on not missing any region of the design space, hence often have more runs located around the center of the design region. Even though MmLHDs and mMLHDs are optimized versions of the standard LHDs, the projections of the points of these designs onto subspaces of more than a single design factor are not guaranteed to have good space-filling properties.

The maximum projection criterion (MaxPro) was proposed by Joseph et al. (2015) to ensure good projections in all subspaces of the design factors. That is, having projections as maximin distance designs, across all the subspaces of 2 to $p - 1$ input variables. When a design is projected onto a subspace, the distances between the points are calculated with respect to the factors that define the subspace. Therefore, a weighted Euclidean distance between the points x_i and x_j is define as:

$$d(\boldsymbol{x_i}, \boldsymbol{x_j}; \boldsymbol{w}) = \left\{\sum_{i=1}^{p} w_l |x_{il} - x_{jl}|^2\right\}^{1/2}$$

where $w_l = 1$ for the factors defining the subspace and $w_l = 0$ for the remaining factors. It makes sense to use weights between 0 and 1, which can be viewed as measures of importance for the factors. Let $0 \leq w_l \leq 1$ be the weight assigned to factor l and let $\sum_{l=1}^{p} w_l = 1$, to ensure good projections in all possible subspaces, an appropriate distribution is assigned to $\boldsymbol{w}$ and the reciprocal distance criterion in equation (3) is integrated over that distribution. Thus, the criterion is:

$$\min_D \int \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{1}{d^k(x_i, x_j; \boldsymbol{w})} p(\boldsymbol{w}) d\boldsymbol{w} \tag{4}$$

where $p(\boldsymbol{w})$ is a prior distribution for $\boldsymbol{w}$. In general, this criterion is not easy to evaluate and optimize. Fortunately, if a uniform prior distribution is chosen for $\boldsymbol{w}$ and $k = 2p$, then the criterion simplifies to:

$$\min_D \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{1}{\prod_{l=1}^{p} |x_{il} - x_{jl}|^2} \tag{5}$$

The product in equation (5) ensures that no two coordinates can be the same; otherwise, the objective function can become ∞. Combining the MaxPro criterion with the LHD structure result in a design with improved uniformity when projected in each univariate dimension, regarded as MaxProLHD (Joseph, 2016).

The variations of the LHDs discussed above; MmLHD, mMLHD, and MaxProLHD are substantial improvements to the traditional LHDs and have been used extensively for generating designs for computer experiments. However, the limitation of these designs is that they are optimal based on a single criterion. In this article, we propose an improvement by using the Pareto front optimization approach to flexibly combine multiple space-filling characteristics to generate optimal LHDs with improve and balanced performance. In section 2, we review the Pareto front optimization approach for multiple objective studies. In section 3, we propose two search algorithms; the column-wise exchange simulated annealing algorithm and the nondominated sorting genetic algorithm for finding the Pareto optimal LHDs based on multiple objectives. In section 4, we illustrate the proposed method and compare the performance of the search algorithms with examples of varied dimensions of input factors. The Pareto front optimal designs are compared with optimal LHDs generated from single criterion optimization for a variety of underlying response surface models. Graphical tools (Lu et al 2011 and Lu and Anderson-Cook, 2012) were explored to evaluate and compare designs and to further guide the selection of the most performing designs from the identified Pareto front. Finally, we conclude and summarize our findings in section 5.

2. Pareto Front Optimization for Multiple Objective Studies

For multiple objective design problem, the desirability function (DF) approach (Derringer and Suich, 1980) has conventionally been used for design selection; it transforms the different criteria considered to a common 0-1 desirability scale, where 0 represent the worst criteria value and 1 represent the best criteria value, to create a single summary for quantifying the goodness of a design. The two basic forms of the DF are; additive (where the scaled criteria are combined as a weighted sum) and multiplicative (where the scaled criteria are combined into a product with the weights entering as exponents). However, optimization using DF fails to consider the trade-offs between criteria directly, it instead identifies a best design via a "black box" algorithm that combined the different metrics into a single metric. The implication of this is that, when different decision makers have different priorities, the DF approach pose a difficulty on how these different priorities should be handled. For instance, deciding on new data to collect can depend heavily on the user-specified relative importance of individual criteria, scaling schemes, and type of DF. Thus, making decisions without understanding the potential impact of different subjective choices can pose a risk.

Pareto optimization has been extensively used in many disciplines as a tool for optimizing multiple responses (Kasprzak and Lewis 2001; Gronwald, Hohm, and Hoffmann 2008; Trautmann and Mehnen 2009). As an alternative to the desirability approach, the Pareto front approach, enhanced with graphical tools, has been adapted to design of experiments to aid decision making (Lu et al., 2011). The Pareto optimization approach not only enables the user to gain deeper understanding of the trade-offs among the multiple competing design criteria, but it also offers an approach for objectively and effectively eliminating inferior solutions and identifying a suite of more promising solutions. The method finds the Pareto frontier of non-dominant designs such that no design can improve one criterion without diminishing at least another criterion. A suite of more promising designs is then selected from the Pareto frontier based on all possible combinations of user specified weights, scaling schemes for individual criteria, and the desirability function for combining the multiple criteria. Finally, numerical and graphical tools (Lu et al., 2011) are utilized to select the best solution based on comprehensive understanding of the robustness of solutions to different user priories. A major advantage of this approach is its adoption of a streamlined process that efficiently and objectively reduce the candidate solutions in the first stage and thereafter, a comprehensive evaluation of the impacts of subjective choices is done in the second stage to facilitate an informed and realistic decision.

The general goal of a design optimization problem with multiple objectives is to maximize $C\ (\geq 2)$ criteria simultaneously given constraints on the input factors. Let $\boldsymbol{\xi} = (\boldsymbol{d}_1', \boldsymbol{d}_2', \dots, \boldsymbol{d}_n')' \epsilon \boldsymbol{\Omega}$ denote a design matrix of dimension $n \times p$, where n is the number of design points and p is the number of design factors; the set of all possible $n \times p$ design matrices is denoted by $\boldsymbol{\Omega}$. Let $\boldsymbol{y} = \boldsymbol{F}(\boldsymbol{\xi}) = \left(f_1(\boldsymbol{\xi}), f_2(\boldsymbol{\xi}), \dots, f_C(\boldsymbol{\xi})\right)^T$ denote the vector of criteria values corresponding to the design matrix, $\boldsymbol{\xi}$. Then, the space containing all obtainable criteria vectors is called the criterion space. A solution $\boldsymbol{\xi}_1$ is said to Pareto dominate another solution $\boldsymbol{\xi}_2$ if $f_j(\boldsymbol{\xi}_1) \geq f_j(\boldsymbol{\xi}_2)$ for all $j \in \{1,2,\dots,C\}$ and there exists at least one $j \in \{1,2,\dots,C\}$ such that $f_j(\boldsymbol{\xi}_1) > f_j(\boldsymbol{\xi}_2)$. In this case, the criteria vector $\boldsymbol{F}(\boldsymbol{\xi}_2)$ is said to be dominated by $\boldsymbol{F}(\boldsymbol{\xi}_1)$. The criteria vector corresponding to a particular solution is usually referred to as a point in the criterion space. A solution is Pareto optimal if and only if no other solution dominates it and its corresponding criteria vector is a non-dominated vector. The Pareto optimal set is referred to as the set of Pareto optimal solutions and the corresponding set of criteria vectors as the Pareto front.

When the Pareto optimal solutions and the Pareto front have been obtained, the decision-making process involves careful consideration of the trade-offs between the competing criteria. Moreover, it is necessary to transition from a potentially large set of candidate solutions to a manageable set of solutions. The Utopia point method is common for strategically making selection from a large set of solutions (Lu et al., 2011). The standard

Utopia point method uses a norm and user-specified set of weights to characterize each solution's proximity to the *Utopia point* and then chooses the best solution (based on different weightings for combining the criteria) closest to the Utopia point. A Utopia point, F^0, in the criterion space satisfies:

$$\boldsymbol{F}^0 = \underset{\xi}{Max}\{f_i(\boldsymbol{\xi}) | \boldsymbol{\xi} \in \boldsymbol{\Omega}\} \quad \text{for all} \quad i \epsilon \{1,2,\cdots,C\} \tag{6}$$

if there is a solution $\boldsymbol{\xi}^* \in \boldsymbol{\Omega}$ that satisfies $\boldsymbol{F}(\boldsymbol{\xi}^*) = \boldsymbol{F}^0$, then $\boldsymbol{\xi}^*$ is called a Utopia solution. For multiple objectives optimization problem, the Utopia solution $\boldsymbol{\xi}^*$ usually does not exist, but it can be used as an "ideal" standard for the criteria values to find the best solution(s) from the Pareto optimal set. The general idea of the Utopia point method is to selects preferred points on the Pareto front which minimize the distance to the Utopia point based on a chosen metric. A common choice for measuring distances to the Utopia point is the L_1-norm, formulated as:

$$\min_{\boldsymbol{\xi} \in \boldsymbol{\Omega}^*} \sum_{j=1}^{C} w_j \left| s_j(\boldsymbol{\xi}) - s_j^0 \right|, \tag{7}$$

where $\boldsymbol{\Omega}^*$denotes the Pareto set of solutions, w_j for $j \in \{1,2,\ldots,C\}$ are the weights assigned to the C individual criteria, $s_j(\boldsymbol{\xi})$ is the jth objective function converted to a desirability scale between 0 and 1 for the jth criterion corresponding to solution ξ , and s_j^0denotes the Utopia point value for the jth criterion on the same scale. Typically, we set $s_j^0 = 1$. Each single-weight combination identifies one optimal solution from the Pareto front. However, not all solutions from the Pareto front will be optimal for at least one set of weights based on the chosen distance metric and scaling scheme. Hence, the Utopia point approach selects only a subset of the solutions from the Pareto front and further reduces the potential options to choose from for making a final decision.

Experimenting with different weight choices takes minimal computational effort in the Utopia point approach since every set of weights utilize the same Pareto front found by the search algorithm. In this study, we consider simultaneously optimizing space filling characteristics for LHDs by exploring a fine grid of weight combinations spread over the entire weighting space. Decisions can then be recommended based on choices of several priorities after considering the tradeoffs and robustness of candidate solutions to different subjective choices. Compared to the standard DF method, the Pareto approach provides summaries of available range of solutions, more intuition about the relative performance of different solutions, and quantitative information for making a justifiable coherent decision.

3. Search Algorithms

The random permutation process allows possibility of many LHDs, each satisfying the Latin Hypercube condition of only one point in each level. As discussed earlier, this random procedure poses the possibility of generating a design with poor design qualities. To overcome this problem, the notion of optimizing random LHDs based on design criterion was introduced to generate optimal LHDs. However, due to the huge combinatorial nature of the design problem, finding the optimal LHD can be computationally expensive. Several search algorithms such as; simulated annealing (Morris and Mitchell1995), columnwise-pairwise algorithms (Ye et al, 2000), enhanced stochastic evolutionary algorithms (Jin et al, 2005; Rungrattanaubol and Na-udom, 2016), etc have been proposed in the literature for finding optimal LHDs. Most of the algorithms use an exchange method for searching in the design space. In this work, we employed the column-wise exchange simulated annealing Pareto Front search algorithm and the nondominated sorting genetic algorithm to generate optimal LHDs.

3.2 Column-wise Exchange Simulated Annealing Algorithm

The column-wise exchange simulated annealing (CESA) algorithm has been commonly used in the design of computer experiments due to its natural ability to preserve the Latin hypercube structure (Morris and Mitchell, 1995). In the mechanism of this algorithm, the search begins with a randomly chosen LHD, and proceeds through the examination of a sequence of designs, each generated as a perturbation of the preceding one. By perturbation, it means a new design is created from the current design by interchanging two randomly chosen observations within a randomly chosen column in the current design. The perturbation replaces the current design if it leads to an improvement. Otherwise, it will replace the current design via simulated annealing (Kirkpatrick et al., 1983) approach- a commonly used method to approximate the global optimum of any given function. In the simulated annealing approach, the new design is allowed to replace the existing design (even if it does not lead to an improvement) with gradually decreasing probability to ensure extensive search over the solution space.

For the multiple objective optimization case in this work, given that the number of criteria to simultaneously optimize is k, we define the set $\boldsymbol{W}$, with each element as a weight vector $(\boldsymbol{w_1}, \dots, \boldsymbol{w_k})$, where $\sum_{i=1}^{k} \boldsymbol{w_i} = \mathbf{1}$ and $\boldsymbol{w_i} \in [\mathbf{0}, \mathbf{1}]$. Now, let $C = \sum_{i=1}^{k} \boldsymbol{w_i c_i}$ be the weighted combination of the multiple criteria, referred to as the weighted criterion, where $\boldsymbol{c_i}$ is the scaled value for each criterion. An overview of the proposed column-wise exchange simulated annealing algorithm with Pareto front optimization approach is described by the following steps:

i. Randomly generate a LHD as the starting design. For this current design, $\boldsymbol{\xi}$, evaluate the weighted average of the criterions, denoted as C_c.
ii. Initialize two null sets: the set of Pareto designs denoted as P and the set containing the corresponding Pareto fronts, denoted as PF ; then add $\boldsymbol{\xi}$ to P and add C_c to PF.
iii. A new design, $\boldsymbol{\xi}^*$ is generated by column-wise exchanging the current design and the weighted criterion, C_n is evaluated.
iv. Next, the difference between the weigthed criterion for the current design and that of the newly generated design, that is, $\Delta C = (C_n - C_c)$ is evaluated. Since the goal is to minizmize C, if the value of ΔC is negative, the new design replaces the current design and search continues. Otherwise, the new design will be accpeted with a probability of $P = exp(-\Delta C/t)$, where t is a time-varying parameter tuned to gradually approach zero as the number of iterations increases.
v. The next step is to update the Pareto front and the Pareto optimal set by searching for improvements. The comparison made here is between the new design, $\boldsymbol{\xi}^*$ and the ones in the "current" set of non-dominated designs P. If $\boldsymbol{\xi}^*$dominates at least one of the designs in the "current" set, then add $\boldsymbol{\xi}^*$ to P and remove the designs dominated by $\boldsymbol{\xi}^*$. If $\boldsymbol{\xi}^*$neither dominates nor is dominated by any design in the current set, then just add $\boldsymbol{\xi}^*$to P. If $\boldsymbol{\xi}^*$ is dominated by at least one of the designs in the "current" set, then discard $\boldsymbol{\xi}^*$ and no update is needed for the current Pareto set.
vi. Repeat Steps i-v for a set of weight vectors of interest. Suppose the search has found m non-dominated designs in the set $P(i.e\ P = \{\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \dots, \boldsymbol{\xi}_m\})$ and m associated criterion vectors in the set $PF\ (i.e\ PF = \{F(\boldsymbol{\xi}_1), F(\boldsymbol{\xi}_2), \dots, F(\boldsymbol{\xi}_m)\})$.

vii. Repeat steps i to vi with S different random starts. The set of non-dominated designs are combined across all the random starts to create the final combined Pareto Front.

3.3 Nondominated Sorting Genetic Algorithm

Genetic algorithms (GA) are popular for generating solutions in optimization problems (Mitchell, 1998). They rely on some operators: selection, crossover, and mutation, inspired by evolution theory. The method of the algorithm is an iterative process, and the population in each iteration (otherwise known as generation) are potential solutions evolving toward better solutions for the optimization problem. Traditionally, solutions are coded by a binary string of 0s and 1s, although many other encodings have emerged in literature. To adapt for our application, the original population should be generated as a LHDs. Moreover, the genetic operators of crossover and mutation should be defined such that they operate directly on LHD matrices, thus preserving the LHD properties. Bates and Sienz (2004) proposed using permutation GA, where the encoding ensures that the rule of one point in each level is not contravened. In our analysis, solutions for the GA are coded as a LHD and the initial population is generated randomly, thereby allowing sampling from the range of all possible LHDs in the design space.

In each iteration, several "children" solutions are generated from crossover between randomly selected "parent" solutions and mutation of randomly selected parent solutions, and the fitness of each individual impacts its likelihood of being selected to create new solutions for the next generation. Fitness is defined as the value of an objective function of the optimization problem which quantifies the suitability of each solution. The more suitable solutions have higher chance of surviving to the next generation. Several selection methods have been proposed to rate the fitness of each solution and preferentially select the best solutions (see, De Jong, 1975; Liefrendahl and Stocki (2005) and De Jong (2006)). Liefrendahl and Stocki (2005) and De Jong (2006) proposed ranking the solutions in each iteration from best to worst fitness and then selecting half of the best solutions as survivors at each iteration. To ensure that each generation is at least as fit as the previous generation, an elitist strategy (Eiben and Smith, 2003) is employed in this work, that is, offspring solutions compete against their parent solutions for survival into the next generation.

Crossover is often considered a multiparent operation while mutation is usually a single parent operation. In this work, crossover is used to preserve the desirable features of two existing solutions by merging those solutions into a new solution, while mutation is used to introduce "fresh solution" into the population by randomly generating a slight modification of an existing solution to maintain population diversity (Eiben and Smith, 2003). In our analysis, we employed rank-proportional selection (Hamada et al. 2008; Lu et al, 2013) to choose parent allocations for the crossover operation. Specifically, the probability of selecting the rth ranked candidate of the N_{pop} solutions is $(N_{pop} - r + 1)/(N_{pop}(N_{pop} + 1)/2)$. Once probabilities have been assigned to each solution in the parent set, a crossover operation using permutation encoding (Bates and Sienz, 2004) is used to generate offspring solutions as follows; two parent designs are randomly selected from the "parents set". A random integer k_i $(1 \leq k_i \leq n-1, i = 1,2,\cdots,p)$ is selected as the cut-off point for each variable. To generate a "child" for the i^{th} variable, values in runs 1 to k_i from the "first parent" are selected and values in runs $k_i + 1$ to n from the "second parent" are selected. Finally, a "repair" function is used to replace repeated values in each variable with the missing values in a random order. For instance, for a 10-run size design problem, if the cut-off point for generating a particular variable of the child solution is chosen as 6, the runs of that variable is determine as shown in Table 1. The red line in table 1 signify the cutoff point selected for the variable. Thus, the child variable takes its first six runs from parent 1 and the remaining four runs from parent 2. As observed in Table 1, the value 0.45 is repeated in the child solution, this can then be corrected by using a repair function to replace one of the repeated values by an unselected value from one of the parent solutions.

Table 1: An example of crossover operation

Parent 1	0.23	0.51	0.33	0.45	0.46	0.65	0.91	0.83	0.15	0.52
Parent 2	0.15	0.85	0.35	0.49	0.52	0.37	0.74	0.45	0.69	0.28
Child	0.23	0.51	0.33	0.45	0.46	0.65	0.74	0.45	0.69	0.28

For the mutation operation, one parent design is randomly sampled from the “parents set” and switching is done on two randomly sampled observations for each variable. Since the children’s solutions are generated by either crossover or mutation, the crossover proportion P_c and the mutation proportion P_m satisfy $P_c + P_m = 1$, where the crossover rate and mutation rate are chosen to be 0.5. That is, at each generation, half of the children’s solutions are created via crossover and half via mutation.

For the multiple criteria optimizations, we employed the NSGA-II algorithm (Deb et al., 2002)-a fast-elitist non-dominated sorting GA that uses a rank-based fitness function for generating the Pareto front for multiple objective optimization problems. The basic idea of the algorithm is as follows; for each solution S in the current population, two quantities are calculated; the number of solutions that dominate S (regarded as domination count) and a list of all solutions dominated by S. All solutions that have a domination count of 0 are nondominated and hence are on the first tier Pareto front (PF1). After each solution has been assigned a domination count and the solutions it dominates have been determined, the algorithm loops through the solutions on PF1. For each solution S on PF1, the solutions it dominates are considered; each solution dominated by S has its domination count reduced by one; if it’s new domination count is 0, then that solution is on the second tier PF (PF2). This process is repeated for all solutions on the lower tier PFs.

Once the population of solutions is sorted into tiers of PFs, NSGA-II ranks solutions on a given tier according to their crowding distance (Deb et al., 2002). The crowding distance measures how far a solution’s criteria values are from those of other solutions, with solutions that are farther from others assigned a larger crowding distance and a higher rank. To compute the crowding distance for a single criterion, the observed criterion values are sorted from smallest to largest. The minimum and maximum observed values are assigned an infinite crowding distance. For all other values of the criterion, the crowding distance assigned to a specific value is computed by taking the range of values immediately flanking that value and scaling that according to the range of the criterion. Once the crowding distances have been computed for each criterion separately, the overall crowding distance for a solution is determined as the sum of that solution’s crowding distances for all criteria.

When randomly generated solutions are used to initialize NSGA-II, there is possibility that only a relatively small number of those solutions would be on the PF1. The implication of this for fixed population size is that solutions on lower tier PFs would potentially be used to generate offspring solutions for the next generation, thereby resulting in offspring that likely continue to be inferior. Thus, computation time would be wasted in evaluating these inferior solutions. On the other hand, for later generations as the PF continue to grow, its size could potentially exceed the initial population size, hence using the fixed population size could result in elimination of superior solution on the Pareto front. To avoid this challenge, Chapman et al (2018) modify the NSGA-II by implementing adaptive population sizing (Eskandari et al, 2007). The adaptive population sizing procedure ensures faster convergence of the algorithm with reduced computational effort and allows the number of offspring solutions produced at each generation to be determined dynamically.

To employ adaptive population sizing for NSGA-II, we specify the maximum population size ($maxpop$) and maximum number of offspring ($maxoff$) to be created in each generation. To initialize the algorithm, $popsize$ = $maxpop$ solutions would be randomly generated. The nondominated sorting algorithm would be used to determine which solutions are on the PF1. The number of offspring ($noff_g$) solutions to be created in generation g is then dynamically determined as:

$$noff_g = min\{c_g + [d_g \times |(PF_1)_g|, maxoff]\}$$

where c_g represents the minimum number of offspring solutions to be created in generation g, while d_g represents the rate at which the number of offspring created in generation g grows as the number of solutions on PF1 grows. For implementation in this work, we set c_g = 20 and d_g = 4. Thus, at each generation, a minimum of 20 offspring solutions are created, and for every solution on the PF1, four additional offspring solutions are created. At the end of generation g , there are $PF1_g$ solutions on the PF1. Then the population size for generation $g + 1$ is determined as:

$$popsize_{g+1} = min\{a_g + [b_g \times |(PF_1)_g|, maxpop]\}$$

where, a_g represents the minimum number of solutions to include in the population beyond the PF1 while b_g represents how the size of the population grows in relation to the size of the PF1. For implementation in this work, we define a_g relative to the specified maximum population size (specifically, a_g = 0.2 $* maxpop$) and $b_g = 1$. This ensures that at each generation, the population size is slightly larger than the number of PF1 solutions. The $popsize_{g+1}$ highest ranked solutions are the ones that move on to Generation $g + 1$. There is a drawback to specifying a maximum population size: if $|(PF_1)_g| > maxpop$, then some solutions on the frst tier PF do not move on to the next generation-that is, the PF will be truncated. This could mean that a reasonable, contending solution is not made available to the user in the subjective decision-making stage. To prevent this, if $|(PF_1)_g| > maxpop$, we allow $maxpop$ to grow based on the size of the PF1 (Chapman et al, 2018).

For the GA search procedure, the cycle of fitness evaluation, reproduction, and survival continues until some termination criterion condition is achieved. Specifically, for this work, the termination condition for the algorithm is when the specified number of generations is reached. However, it should be noted that, as a heuristic search algorithm, the GA algorithm does not guarantee finding the absolute optimal solutions. Generally, the longer the search algorithm runs, the better the approximation to the true Pareto front. In practice, a balance is sort between the accuracy of the approximation and the computing time.

4.0 Applications

Using the Pareto optimization approach, the CESA algorithm and the NSGA-II were used to seek optimal LHDs that simultaneously optimize multiple space-filling characteristics. The optimal LHDs generated are thereafter evaluated for estimating different Gaussian process models (Rasmussen and Williams, 2006). Gaussian process models are widely used in analyzing data from computer experiments due to their flexibility in approximating complex surfaces (Adamou, 2014). In terms of the search algorithms considered, the analysis in this work revealed the CESA to be less effective for broad exploration of the design space, and consequently unable to find the PF efficiently. We think that the column-wise exchange mechanism makes the solutions largely dependent on the starting design locations and hence limited the space and efficiency of the search algorithm. On the contrary, the NSGA-II procedure proved to be more efficient in the exploration of the design space and populating the PF.

4.1 Low Dimensional Design Example

Consider the problem of constructing a design of computer experiment with 20 runs and two input variables. For this scenario, there is no need to consider maximum projection criterion because LHD already ensured a uniform distribution in 1-dimension. Thus, we only consider the maximin distance and minimax distance criterions. After several GA searches for the single criterion optimization, the optimal MmLHD and optimal mMLHD found have criterion values 4.0947 and 0.1646 respectively. In calculating the maximin criterion value for the LHD, we employed the simplified alternative maximin criterion proposed by Morris & Mitchell (1995), such that minimizing their alternative maximin criterion is the same as maximizing the standard maximin criterion.

For the multiple objective optimizations, the Pareto front optimization approach and the search algorithms were used to seek optimal LHDs, where both the maximin and minimax criterions are simultaneously considered. The NSGA-II algorithm takes an average of 2 minutes to run 100 generations on a standard desktop computer. Therefore, it takes an average of 3 hours for 10000 generations. Multiple separate GA searches were run in parallel to improve the computational efficiency. The Pareto front, containing 16 optimal LHDs (it includes the MmLHD and mMLHD from the single criterion optimization) was identified after running 70 separate GA searches of 10000 generations each. The criteria values are transformed to the desirability scale between 0 and 1 so that both criteria are in comparable scale as shown in Figure 1.

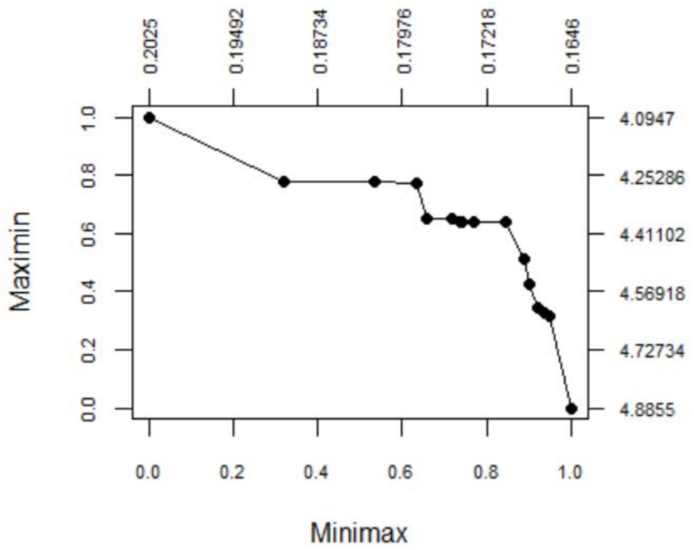

Figure 1: The Pareto optimal solutions based on maximin criterion and minimax criterion. The right and top scale represent the raw values of the criterions.

The designs on the Pareto Front were subjected to further evaluation using graphical summaries, in order to examine design robustness to different weightings of the competing objectives as well as trade-offs between criteria among competing designs. Six optimum designs were selected from the Pareto Front by the Utopia point method (Lu et al., 2011) with L_1 norm across all possible weights. Figure 2 is a mixture plot that shows the weight distributions for the selected designs. The Figure shows that designs 10 and 1 are optimal for large ranges of weights. Specifically, design 10 is optimal for weights ranges between 0.25 and 0.6 for the maximin criterion and between 0.4 to 0.75 for the minimax criterion while design 1 is optimal for weights ranges between 0.725 and 1 for the maximin criterion and between 0 and 0.275 for the minimax criterion. Designs 16, 15 and 4 are optimal for small ranges of weights while design 11 is optimal for only very specific values of weights. A plot such as this can be valuable for the practitioner when there is uncertainty about how different criteria should be valued. From the mixture plot, we can also deduce that; designs 4 and 10 offer some balances when both criterions are simultaneously considered, designs 1 appears optimal when the maximin criterion is valued more while designs 15 and 16 appears optimal when the minimax criterion is valued more.

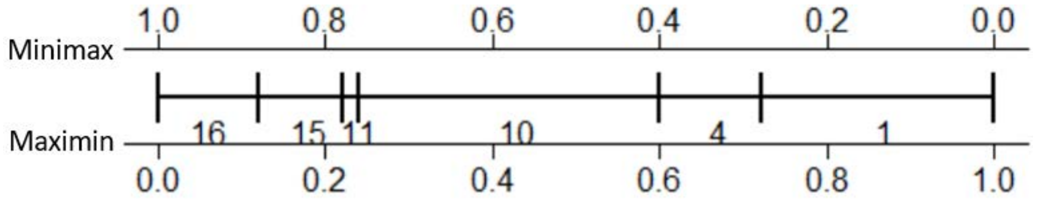

Figure 2: The mixture plot for the six optimum designs.

Figure 3 is the trade-off plot (Lu et al, 2011) showing the criteria values for the five selected designs from the Utopia point approach. In the plot, the criteria values for each of the 6 designs use two scales: the desirability scale between 0 and 1 (inner vertical axis) and the original scale of the criteria values. We sort the designs based on a primary criterion (here chosen to be the maximin criterion). The observations from the trade-off plot are similar to that from the mixture plot discussed above. From Figure 3, notice that design 1, which has optimal performance for the maximin criterion has a poor performance for the minimax criterion. Similarly, design 16 which has optimal performance for the minimax criterion has a poor performance for the maximin criterion. Designs 10 and 4 represent choices with some balanced performance when both criteria are considered simultaneously.

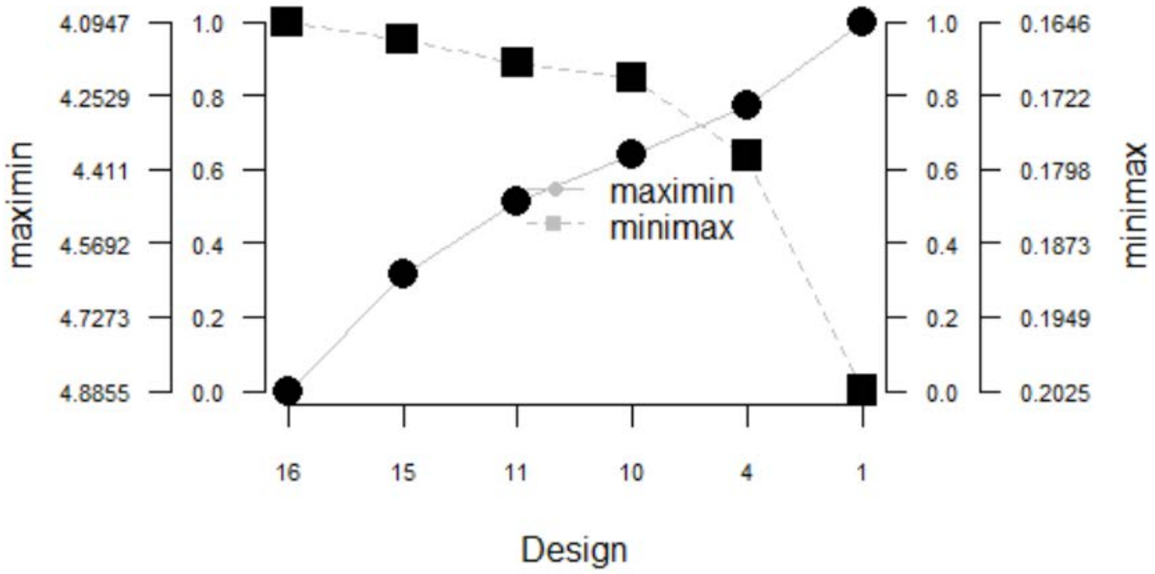

Figure 3: The plot showing trade-offs between the 6 optimum designs.

Next, the performances of the six optimum designs are examined through simulation studies. Two-dimensional data with run size 20 is generated from the response surface model of the form; $z(x) = x_1 exp(-x_1^2 - x_2^2)$, $x_1, x_2 \in [-2,2]^2$ and Gaussian process model with a constant mean is fitted to the data. For the covariance function of the Gaussian process, we considered the product power exponential correlation function using three different choices of the smoothing parameter, v. Using a 100×100 grid points in two-dimensional space, the mean squared prediction error of the designs were estimated.

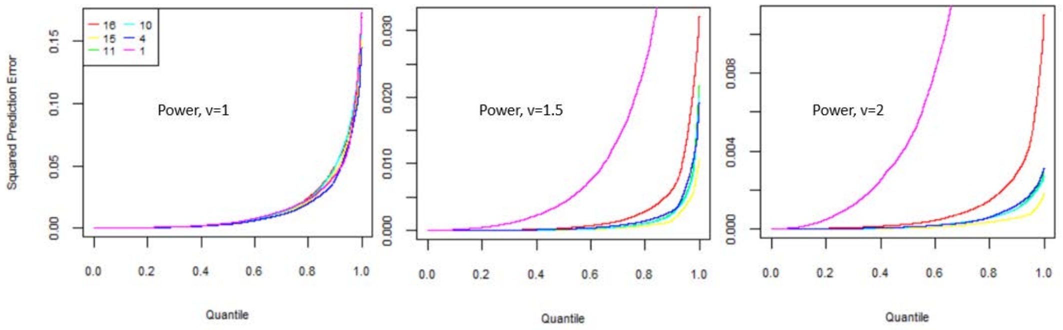

Figure 4: FDS plot for comparing the 6 designs using response surface design; $z(x) = x_1 exp(-x_1^2 - x_2^2)$, $x_1, x_2 \in [-2,2]^2$

Figure 4 shows the fraction of design space (FDS) plot for comparing the six optimum designs. Each curve in the FDS plots represents a single design and it reflects the fraction of the design that has the prediction variance above certain percentage. Intuitively, a design with small, squared prediction error through the design space corresponds to a low flat curve in the FDS plot. Figure 4 shows that the designs with some balance in mM and Mm (10-light blue, 4-blue, 11-green and 15-yellow) generally perform better in terms of the prediction errors when compared with designs that are optimal in single criterion (design

1 (MmLHD) and design 16 (mMLHD)). Specifically, design 15 has the lowest mean squared prediction error. Two alternative forms of the response surface model are explored to further observe the performance of the six designs. The results obtained are found to be consistent.

4.2 High Dimensional Design Example

In this section, we explore the performance of the proposed methodology on a design problem with higher dimensions (more design factors), larger number of runs, and more criteria to be considered. In addition to the maximin and minimax criteria, space-filling over the subspaces of active variables is also a desirable property. Hence, the projection property in design selection needs to be considered, and this property can be quantified by the maximum projection criterion. Consider the problem of constructing a design of computer experiment with 50 runs and 4 input variables. After several GA searches for the single criterion optimization, the optimal MmLHD, MaxProLHD and mMLHD found have criterion values 2.5854, 37.1714 and 0.4062 respectively. Due to largely increased search space, the NSGA-II algorithm took an average of 16 minutes to run 100 generations on a standard desktop computer. The Pareto Front, containing 536 optimal LHDs (includes the MmLHD, MaxProLHD and mMLHD from the single criterion optimization) was identified after running 50 separate GA searches of 10000 generations each.

Similar to the previous example, the impact of different subjective choices and weightings on the solutions on the Pareto Front are examined via graphical summaries. Five optimum designs were selected from the 536 designs on the Pareto Front by the Utopia point method with L_1 norm across all possible weights. Figure 5 shows the 3D mixture plot of the 5 optimum designs. Every point in the triangle corresponds to a weight combination with the sum of the three weights equaling one. Hence, the vertices and the edges correspond to optimizing based on a single criterion and two of the three criteria, respectively. The weight combinations corresponding to the same optimal design are displayed in the same region in the same color. From Figure 5, we see that some designs; 163, 41 and 162 are optimal for large ranges of weights. The 5 optimum designs identified by the Utopia point method do not include the optimal MmLHD MaxProLHD and mMLHD from the single criterion optimization.

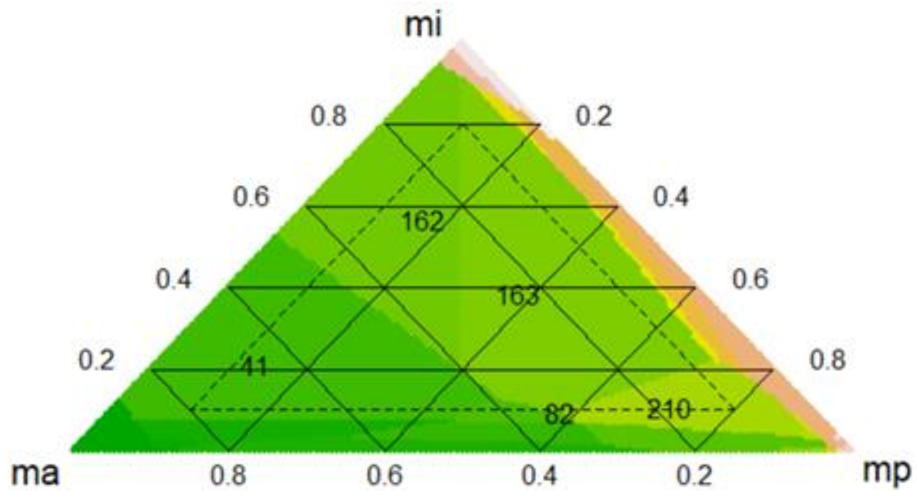

Figure 5: The 3D mixture plot of the five optimum designs.

The trade-offs between the reduced set of designs are shown in Figure 6. The designs are sorted based on the maximin criterion. The observations from the plot are similar to that from the 3D mixture plot. From both plots, design 41 appears to favor the maximin criterion more, design 162 appears to favor the minimax criterion more and design 210 appears to favor the maximum projection criterion more.

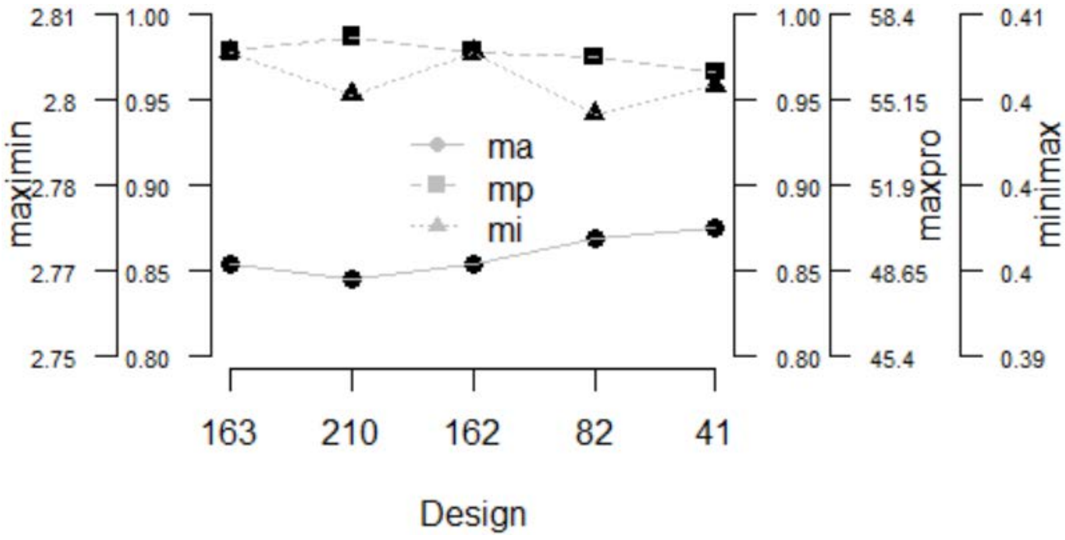

Figure 6: The plot showing trade-offs between the 5 optimum designs

The performances of the five optimum designs together with the optimal designs from the single criterion optimization were then examined via simulation. Four-dimensional data with run size 50 is generated from the response surface model of the form; $z(x) = exp(sin(0.9 \times (x_1 + 0.48)^\wedge 10))$, $x_i \in [0,1]^4$ and Gaussian process is fitted to the data. Using a 10^4 grid points in four-dimensional space, the mean squared prediction errors of the designs were estimated. Figure 7 shows the fraction of design space (FDS) plot for comparing the designs. Specifically, design1(MmLHD), design530 (MaxProLHD) and design 534(mMLHD) are the optimal designs found for single criterion optimization.

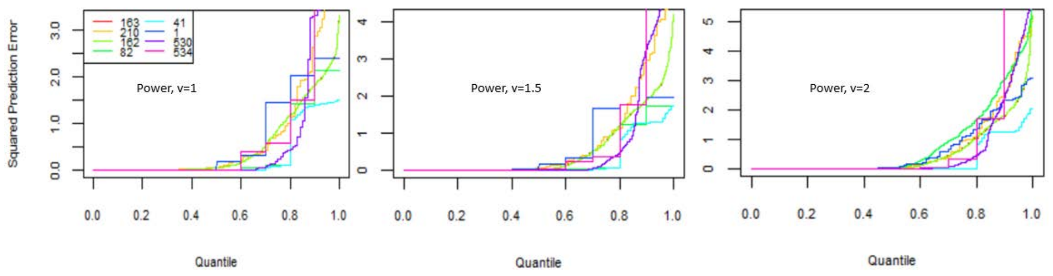

Figure 7: FDS plot for comparing the designs using response surface design; $z(x) = exp(sin(0.9 \times (x_1 + 0.48)^\wedge 10))$, $x_i \in [0,1]^4$

The Figure shows that the MaxProLHD (purple) performs well when the quantile is below around 0.9 but poorly afterwards. The Pareto Front designs: 41(light-blue), 162(light-green) and 163 (red) have more robust performance, especially when preventing the worst-case scenario. Design 41 generally performs well across the design space. Two alternative forms of the response surface model are explored to further observe the performance of the designs and as expected, the results and observations are found to be consistent with the example reported. In general, optimal LHDs from the multiple criteria optimizations are better choices compared to optimal LHDs from single criterion optimization.

5.0 Summary and Conclusion

In the design of computer experiments, most of the existing search algorithms are designed for optimizing single criterion when generating LHDs. However, the fundamental differences in the emphasized criterion often result in suboptimal designs, which performs well on one aspect of the design and relatively poor on other aspects. Simultaneously considering multiple criteria is desired to generate better designs with improved and balanced performance on multiple aspects of the design characteristics. Moreover, the presence of multiple objectives in a problem, in principle, gives rise to a set of Pareto-optimal solutions, instead of a single optimal solution.

Utilizing the pareto optimization approach, the column-exchange simulated annealing algorithm and nondominated sorting genetic algorithm are employed with examples of varied dimensions of input factors, and the generated optimal LHDs are evaluated based on their performance across simulations with different true response surface models and compared with designs from single criterion optimization to show their improved performance. The nondominated sorting genetic algorithm proved more efficient in eliminating noncontending choices from the solution space, thereby allowing identification of the most promising designs. Although, this present work considers simultaneous combinations of the minimax, maximin and maximum projections criterions. The methodology can be extended to combinations of other criterions used for the design of computer experiments.

References

Adamou, M. (2014). Bayesian Optimal Designs for the Gaussian Process Model. PhD Thesis,
University of Southampton.

Bandler, J. W., Cheng, Q. S., Dakroury, S. A., Mohamed, A. S., Bakr, M. H., Madsen, K. and
Sondergaard, J. (2004). Space Mapping: The state of the Art. *IEEE Transactions on Microwave theory and techniques*, *52*(1), 337-361.

Bates, S.J. and Sienz, J. (2004). Formulation of the Optimal Latin Hypercube Design of Experiments Using a Permutation Genetic Algorithm. *45th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Materials Conference*, April 2004, Palm Springs, California.

Chapman, J., Lu, Lu. and Anderson-cook, C.M. (2018). Using Multiple Criteria Optimization and
Two-Stage Genetic Algorithms to Select a Population Management Strategy with Optimized Reliability. *Complexity*. DOI:10.115520187242105

De Jong, K. A. (2006). Evolutionary Computing: A Unified Approach. Cambridge, MA: MIT
Press.

De Jong, K. A. (1975). Analysis of the Behavior of a Class of Genetic Adaptive Systems.

Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T. A. M. T. (2002). A fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, *6*(2), 182-197.

Edwin, R. and Dam, V. (2008). Two-dimensional Minimax Latin Hypercube Design. *Discrete*
Applied Mathematics. 156, 3483-3493.

Eskandari, H., Geiger, C.D. and Lamont, G.B. (2007). Fast PGA: A Dynamic Population Sizing
Approach for Solving Expensive Multiobjective Optimization Problems. *In Evolutionary Multi-Criterion Optimization*. *Springer*, Berlin, Germany, 2007.

Fang, K., Li, R. and Sudjianto, A. (2006). Design and Modelling for Computer Experiments. *Chapman & Hall/CRC*: Boca Raton.

Gronwald, W., Hohm, T. and Hoffmann, D. (2008). Evolutionary Pareto-Optimization of Stably
Folding Peptides, *BMC-Bioinformatics*, 9, 109.

Hamada, M., Wilson, A. G., Reese, C. S., and Martz, H. F. (2008). Bayesian Reliability. New
York: Springer. [473,479,480]

Liefvendahl, M. and Stocki, R. (2006). A Study on Algorithms for Optimization of Latin

Jin, R., Chen, W., Sudjianto, A. (2005). An Efficient Algorithm for Constructing Optimal Design
of Computer Experiments. *Journal of statistical planning and inference***,** 134, 268–287.

Johnson, M. E., Moore, L. M. and Ylvisaker, D. (1990). Minimax and Maximin Distance
Designs. *Journal of Statistical Planning and Inference*, *26*(2), 131-148.

Joseph, V. R. (2016). Space-filling Designs for Computer experiments: A Review. *Quality Engineering*, *28*(1), 28-35.

Joseph, V. R., Gul, E. and Ba, S. (2015). Maximum Projection Designs for Computer
Experiments. *Biometrika*, *102*(2), 371-380.

Kasprzak, E. M. and Lewis, K. E. (2001). Pareto Analysis in Multiobjective Optimization Using
the Collinearity Theorem and Scaling Method. *Structural Multidisciplinary Optimization*, 22, 208–218.

Lu, L. and Anderson-Cook, C. M. (2012). Rethinking the Optimal Response Surface Design for a
First-Order model with two-factor Interactions, When Protecting Against Curvature. *Quality Engineering*, *24*(3), 404-422.

Lu, L., Anderson-Cook, C.M. and Robinson, T.J. (2011). Optimization of Designed Experiments
Based on Multiple Criteria Utilizing a Pareto Frontier. *Technometrics*. 53(4), 353–365.

McKay, M. D., Beckman, R. J. and Conover, W. J. (1979). Comparison of Three Methods for
Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics*, *21*(2), 239-245.

Mitchell, M. (1998). An introduction to genetic algorithms. MIT press.

Montgomery, D. C. (2017). Design and analysis of experiments. *John Wiley and Sons*.

Morris, M. D., & Mitchell, T. J. (1995). Exploratory Designs for Computational
Experiments. *Journal of statistical planning and inference*, *43*(3), 381-402.

Rasmussen, C. and Williams, C. (2006) Gaussian Processes for Machine Learning. Cambridge,
Massachusetts: *MIT press*.

Rungrattanaubol, J. and Na-udom, A. (2016). Heuristic Search Algorithms for Constructing
Optimal Latin Hypercube Designs. *Recent Advances in Information and Communication Technology*. 463, DOI 10.1007/978-3-319-40415-8_18

Sacks, J., Welch, W. J., Mitchell, T. J., & Wynn, H. P. (1989). Design and Analysis of Computer
Experiments. *Statistical science*, 409-423.
Santner, T., Williams, B. and Notz, W. (2003). The Design and Analysis of Computer
Experiments. *Springer*: New York
Trautmann, H. and Mehnen, J. (2009). Preference-Based Pareto Optimization in Certain and Noisy
Environments. *Engineering Optimization*, 41, 23–38.
Viana, F.A.C. (2015). A Tutorial on Latin Hypercube Design of Experiments. *John Wiley & Sons*.
New York
Ye, K.Q., Li, W., Sudjianto, A. (2000). Algorithmic Construction of Optimal Symmetric Latin
Hypercube Designs. Journal of Statistical Planning and Inference. 90, 145–159.