The Canadian Journal of Statistics
Vol. 00, No. 00, 2019, Pages 000–000
La revue canadienne de statistique

CVX based algorithms for constructing various optimal regression designs

Weng Kee WONG[1] and Julie ZHOU[2*]

[1]*Department of Biostatistics, University of California, Los Angeles, CA 90095-1772, U.S.A.*
[2]*Department of Mathematics and Statistics, University of Victoria, Victoria, British Columbia, Canada V8W 2Y2*

Key words and phrases: Convex optimization; dose−response model; equivalence theorem; multi-objective design; multi-stage design; semi-definite programming.

MSC 2010: Primary 62K05; secondary 62K20.

Abstract: CVX based numerical algorithms are widely and freely available for solving convex optimization problems but their applications to solve optimal design problems are limited. Using the CVX programs in MATLAB, we demonstrate their utility and flexibility over traditional algorithms in statistics for finding different types of optimal approximate designs under a convex criterion for nonlinear models. They are generally fast and easy to implement for any model and any convex optimality criterion. We derive theoretical properties of the algorithms and use them to generate new *A*-, *c*-, *D*- and *E*-optimal designs for various nonlinear models, including multi-stage and multi-objective optimal designs. We report properties of the optimal designs and provide sample CVX program codes for some of our examples that users can amend to find tailored optimal designs for their problems. *The Canadian Journal of Statistics* 00: 000–000; 2019 © 2019 Statistical Society of Canada

Résumé: Des algorithmes numériques basés sur CVX gratuits et largement répandus sont disponibles pour résoudre des problèmes d'optimisation convexe, mais leur application pour trouver un plan d'expérience optimal est limitée. Les auteurs démontrent l'utilité et la flexibilité des programmes CVX par rapport aux algorithmes traditionnels de statistique pour trouver différents types de plans approximativement optimaux sous un critère convexe pour les modèles non linéaires. Ces programmes sont généralement rapides et faciles à implémenter, quel que soit le modèle et le critère d'optimalité. Les auteurs déterminent les propriétés théoriques des algorithmes et les exploitent pour générer des nouveaux plans *A*-, *c*-, *D*- et *E*-optimaux pour différents modèles non linéaires, y compris des plans optimaux multi-stades et multi-objectifs. Ils décrivent les propriétés des plans optimaux et fournissent des exemples de code CVX pour certains de leurs exemples qu'un usager pourrait modifier afin de l'adapter à son problème d'optimisation d'un plan d'expérience. *La revue canadienne de statistique* 00: 000–000; 2019 © 2019 Société statistique du Canada

1. INTRODUCTION

An algorithmic approach is a practical way to find optimal designs. There are many types of algorithms for computing optimal designs and they come with different assumptions. Historically, many of the algorithms are deterministic, meaning that repeated runs of the algorithm with the same input produce the same design when it stops using the same stopping criterion. Some algorithms have a proof of convergence to the optimum and others are ad-hoc or based

Additional Supporting Information may be found in the online version of this article at the publisher's website.
**Author to whom correspondence may be addressed.*
E-mail: jzhou@uvic.ca

on heuristics. The former includes traditional ones, like Fedorov-type and exchange-type of algorithms, developed mainly for linear models and more than 40 years ago. They are well described in several design monographs, such as Fedorov (1972) and Silvey (1980).

The traditional algorithms generally work well but can be problematic when there are many variables to optimize and the model is nonlinear. They may stall or break down because of the huge computational burden. For instance, Broudiscou, Leardi & Phan-Tan-Luu (1996) claimed that traditional algorithms in statistics for finding optimal designs cannot be used to find non-standard designs, such as asymmetrical *D*-optimal designs. They found the algorithms performed poorly, were difficult to handle and ultimately abandoned them for genetic algorithms. Similarly, Royle (2002) reported that the traditional exchange algorithms are not practical for finding large spatial designs because they can be computationally prohibitive when the criterion is computationally expensive to evaluate or the design space is too large. More recent algorithms include multiplicative algorithms, cocktail algorithm (Yu, 2011) and an algorithm proposed by Yang, Biedermann & Tang (2013), where they showed their algorithms outperformed several traditional algorithms for finding optimal designs in several ways. Mandal, Wong & Yu (2015) provides a brief review of current numerical approaches for finding optimal designs, including metaheuristic algorithms.

Apart from the above mentioned deterministic algorithms, there are also mathematical programming methods that seem highly appropriate for finding optimal designs. For reasons that are not very clear, these tools are considerably less used in statistics for finding optimal designs, until recently. Papp (2012) reviewed semi-definite programming as a tool for finding optimal designs and noted its stability and theoretical guarantee of the efficiency. Using mainly polynomial models, he showed that running times required for finding various optimal designs are negligible. Additionally, Duarte & Wong (2014) and Duarte, Wong & Atkinson (2015) used semi-infinite programming tools to find optimal designs, and Cuervo, Goos & Sorensen (2016) employed a programming method to find *D*-optimal designs for linear models with multiple constraints on the design space. Ye, Zhou & Zhou (2017) and Wong, Yin & Zhou (2017) further demonstrated the utility of semi-definite programming to find various types of optimal designs for linear models, and Wong, Yin & Zhou (2018) found various types of optimal designs for multi-response multi-factor experiments for nonlinear models (NMs) using theory along with semi-definite programming tools.

CVX based numerical algorithms are widely used in engineering and available in MATLAB for solving convex optimization problems. However, their applications to solve optimal design problems are limited. Gao & Zhou (2017) was probably the first to apply a CVX program to find optimal moments of *D*-optimal designs for polynomial and trigonometric regression models. Our main goal in this article is to demonstrate key advantages CVX programs have over traditional methods in statistics for finding optimal designs, including two-stage designs or multi-objective optimal designs for any model. Additionally, we derive theoretical results helpful for studying properties of the optimal designs and the algorithms. In particular, we show a generalization of the scale invariance property of *D*-optimal designs and how the choice of the discretized points in the design space affect the quality of the generated design. We provide new optimal designs and also show CVX based algorithms can find multi-stage or multi-objective optimal designs.

Section 2 describes statistical background and Section 3 studies CVX based algorithms for constructing optimal designs, including conditions for checking whether a design is optimal among all designs on the selected design space. In Section 4 we derive several properties of the algorithms and the generated optimal designs, and in Section 5 we present various applications. Section 6 proposes another CVX based algorithm for finding optimal designs for large design space and models with several covariates. Section 7 concludes with proofs in the Appendix. MATLAB codes for four examples are given in the Supplementary Material.

2. BACKGROUND

Suppose we have a statistical model for studying the relationship between a response variable Y and a set of independent variables $\mathbf{x}$ from a known compact set $S \subset R^k$, where k is the dimension of $\mathbf{x}$. Design issues concern selecting points from S in some optimal way to observe the responses. We assume there are resources to take n independent observations and the mean response at $\mathbf{x}$ is $f(\mathbf{x}, \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is a q-dimensional vector of unknown parameters in the model.

There are two types of designs: approximate and exact. We focus on approximate designs and denote such a design with m points by $\xi = \{(\mathbf{x}_i, w_i),\ i = 1, \dots, m\}$, where $\mathbf{x}_1, \dots, \mathbf{x}_m \in S$, $w_1, \dots, w_m$ are the corresponding weights that sum to unity. The implemented design takes roughly nw_i observations at $\mathbf{x}_i$, $i = 1, \dots, m$, subject to $nw_1 + \cdots + nw_m = n$ after rounding each nw_i to a positive integer. Approximate designs are easier to study and find because there is a unified theory for finding them and confirming their optimality using equivalence theorems.

Following convention, the worth of a design is measured by its Fisher information matrix. We denote this matrix by $\mathbf{I}_\xi(\boldsymbol{\theta}) = \sum_{i=1}^m w_i \mathbf{I}_{\mathbf{x}_i}(\boldsymbol{\theta})$, where $\mathbf{I}_{\mathbf{x}_i}(\boldsymbol{\theta})$ is the information from an observation at $\mathbf{x}_i$. The asymptotic covariance matrix of the maximum likelihood estimator $\hat{\boldsymbol{\theta}}$ of $\boldsymbol{\theta}$ is proportional to the inverse matrix of $\mathbf{I}_\xi(\boldsymbol{\theta})$. Many optimality criteria are formulated as a convex function $\Phi(\cdot)$ and the design sought is $\xi^* = arg\,min_\xi \Phi(\mathbf{I}_\xi^{-1}(\boldsymbol{\theta}))$, and the minimization is over all designs on S. Examples of Φ are the determinant and the trace functions. For NMs, $\mathbf{I}_\xi(\boldsymbol{\theta})$ depends on the true value $\boldsymbol{\theta}$ and ξ^* is called a locally optimal design, or simply an optimal design.

Finding analytical solutions to optimal design problems for NMs and generalized linear models (GLMs) is challenging unless the model is simple. The problem is compounded when the model has several factors or the design space has complicated constraints or errors have a complex correlation structure. We believe this in part explains why the bulk of design work concerns models with one or two factors; see for example, Dassel & Rawlings (1988), Dette & Biedermann (2003), Biedermann, Dette & Zhu (2006), and Fang, Wiens & Wu (2006). This suggests that relying solely on an analytical approach to find optimal designs can be limiting, when we have a high dimensional NM. More effective and powerful algorithms are needed to find optimal designs for more complicated design problems. The next section presents our algorithms for finding various types of optimal designs and how they can be programmed to generate tailor made designs for the user.

3. CVX BASED ALGORITHMS

CVX is a modelling system, developed for solving disciplined convex optimization problems (Grant & Boyd, 2013), and it is available in MATLAB. CVX can solve several types of optimization problems, including linear and quadratic programming, second-order cone programming, semi-definite programming, and other complex convex optimization problems. There are various algorithms to solve convex optimization problems (Boyd & Vandenberghe, 2004), including steepest descent method and Newton's method for unconstrained minimization problems, Newton's method for minimization problems with equality constraints, and interior point methods for inequality constrained minimization problems. Here we briefly discuss interior point methods. Consider a minimization problem with two inequality constraints,

$$\min_{\mathbf{v}}\ g_0(\mathbf{v}), \quad \text{subject to: } g_1(\mathbf{v}) \le 0,\ g_2(\mathbf{v}) \le 0,\ \mathbf{A}\mathbf{v} = \mathbf{b}, \tag{1}$$

where $\mathbf{v} = (v_1, \dots, v_{k_0})^\top$, $\mathbf{A}$ is a $l_0 \times k_0$ constant matrix with rank l_0, $\mathbf{b}$ is a constant vector, and g_0, g_1 and g_2 are convex and twice continuously differentiable functions from R^{k_0} to R.

There are two key steps in interior point methods. The first step transforms the problem in (1) into an equality constrained problem by a logarithmic barrier function as follows,

$$\min_{\mathbf{v}} \; g_0(\mathbf{v}) - \frac{1}{\gamma}\sum_{i=1}^{2}\log(-g_i(\mathbf{v})) \tag{2}$$
$$\text{subject to: } \mathbf{A}\mathbf{v} = \mathbf{b},$$

where $-\sum_{i=1}^{2}\log(-g_i(\mathbf{v}))$ is called a barrier function, and γ is a positive parameter. As $\gamma \to \infty$, the solution of Equation (2) converges to that of Equation (1).

The second step solves a sequence of problems in Equation (2) defined by an increasing sequence of γ, say $0 < \gamma^{(0)} < \gamma^{(1)} < \gamma^{(2)} < \ldots$. For a fixed γ, Newton's method or other methods can be applied to find a solution for Equation (2). A barrier approach or path-following approach is then applied to solve the sequence of problems, using the solution for Equation (2) with $\gamma = \gamma^{(j)}$ as the starting point for (2) with $\gamma = \gamma^{(j+1)}$, $j = 0, 1, 2, \ldots$, Boyd & Vandenberghe (2004, Chapter 11) provide a detailed discussion about the choice of $\gamma^{(j)}$, stopping criteria and convergence analysis, which are carefully implemented in CVX. Other techniques such as solving dual problems and using sparsity features are also discussed to speed up the methods by reducing the number of variables. The dual problems for optimal design problems also provide interesting geometric interpretations for optimal designs and there are a few examples in Boyd & Vandenberghe (2004, Section 7.5). In summary the CVX system contains various optimization methods that are fully supported by theory.

The convex optimization problems need to be written in certain forms before they can be solved by CVX. The next subsection describes a few commonly used optimal design criteria and how they may be formulated in ways that CVX can optimize directly after the design space is discretized by the user into a set S_N of N grid points from which optimal design points are selected by CVX. The number N is also user-selected and can be very large. Frequently, the N grid points are equally spaced throughout the set S and there is a new proposal to use grid set that adapts after each iteration (Duarte, Wong & Dette, 2018). In Section 3.2 we present CVX algorithms and conditions for checking whether the CVX-generated designs are optimal.

3.1. Optimality Criteria

Given a statistical model, the error distribution, and a discretized design space, $S_N = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, we denote an approximate design on S_N by $\xi_N = \{(\mathbf{x}_i, w_i),\ i = 1, \ldots, N\}$ where the weight vector at these points, $\mathbf{w} = (w_1, \ldots, w_N)^\top$, satisfies

$$w_i \geq 0, \quad i = 1, \ldots, N, \quad \sum_{i=1}^{N} w_i = 1. \tag{3}$$

Given a design criterion, the optimization problem is to determine points in S_N that receive a positive weight and become support points of the optimal design. Berger & Wong (2009) provide details for several optimality criteria. For D-optimality, which is the most widely used criterion, we minimize the objective function

$$\mathcal{L}_D(\xi_N) = -\log\left(\det(\mathbf{I}_{\xi_N}(\boldsymbol{\theta}))\right), \tag{4}$$

or

$$\mathcal{L}'_D(\xi_N) = -\left(\det(\mathbf{I}_{\xi_N}(\boldsymbol{\theta}))\right)^{1/q}, \tag{5}$$

over all vectors $\mathbf{w}$ satisfying Equation (3).

Other criteria include A-, A_s-, c-, I- and L-optimality. Like D-optimality, each of these criteria can be formulated as a convex function of $\mathbf{w}$. Collectively, these criteria are embedded in the following class of objective functions and our goal then is to find a weight distribution on S_N that optimizes the criterion,

$$\mathcal{L}_{AC}(\xi_N) = \text{ trace } \left(\mathbf{C}^\top \, \mathbf{I}_{\xi_N}^{-1}(\boldsymbol{\theta}) \, \mathbf{C} \right), \tag{6}$$

where the choice of the $q \times l$ matrix $\mathbf{C}$ with $l \leq q$ determines the optimality criterion. For singular $\mathbf{I}_{\xi_N}(\boldsymbol{\theta})$, $\mathcal{L}_D(\xi_N)$ and $\mathcal{L}_{AC}(\xi_N)$ are defined to be $+\infty$.

It is important that $\mathbf{I}_{\xi_N}(\boldsymbol{\theta})$ is linear in $\mathbf{w}$, so $\mathcal{L}_D(\xi_N)$, $\mathcal{L}'_D(\xi_N)$ and $\mathcal{L}_{AC}(\xi_N)$ are convex functions of $\mathbf{w}$ (Boyd & Vandenberghe, 2004; Bose & Mukerjee, 2015). We then apply CVX to find optimal designs after writing the convex optimization problem in a general form:

$$\min_{\mathbf{w}} \mathcal{L}(\xi_N), \quad \text{subject to: } w_i \geq 0, \; i = 1, \ldots, N, \; \sum_{i=1}^{N} w_i = 1, \tag{7}$$

where $\mathcal{L}(\xi_N)$ can be $\mathcal{L}_D(\xi_N)$, $\mathcal{L}'_D(\xi_N)$, $\mathcal{L}_{AC}(\xi_N)$, or other convex functions of $\mathbf{w}$ discussed later in Section 4.4.

3.2. Algorithm I

We propose an algorithm to construct optimal designs via CVX as follows.

Algorithm 1 (Computing optimal designs via CVX in MATLAB).

Step (i): Input parameter value for $\boldsymbol{\theta}$ and design points $\mathbf{x}_1, \ldots, \mathbf{x}_N$.
Step (ii): Compute the $q \times q$ information matrices $\mathbf{I}_{\mathbf{x}_i}(\boldsymbol{\theta})$ at each $\mathbf{x}_i$ for $i = 1, \ldots, N$.
Step (iii): Use CVX to solve Equation (7) for $\mathbf{w}$ and denote the solution by $\hat{\mathbf{w}}$.
Step (iv): Check optimality of $\hat{\mathbf{w}}$.

This algorithm is simple and can be applied to any model and various optimality criteria. Steps (i) and (ii) depend on the model and design space, while the design criterion is specified in Step (iii) through the objective function $\mathcal{L}(\xi_N)$ in problem (7). In Step (iv), we verify if the numerical result $\hat{\mathbf{w}}$ is an optimal design.

Similar to Bose & Mukerjee (2015) and Wong, Yin & Zhou (2017), the optimality conditions are stated in Lemma 1. Let ξ_N^* be the optimal design with weight vector $\hat{\mathbf{w}}$, let $h_{Ti}(\hat{\mathbf{w}}) = \text{trace}\left(\mathbf{C}^\top \left(\mathbf{I}_{\xi_N^*}^{-1}(\boldsymbol{\theta}) \, \mathbf{I}_{\mathbf{x}_i}(\boldsymbol{\theta}) \mathbf{I}_{\xi_N^*}^{-1}(\boldsymbol{\theta}) - \mathbf{I}_{\xi_N^*}^{-1}(\boldsymbol{\theta}) \right) \mathbf{C} \right)$, and let $h_{Di}(\hat{\mathbf{w}}) = \text{trace}\left(\mathbf{I}_{\xi_N^*}^{-1}(\boldsymbol{\theta}) \, \mathbf{I}_{\mathbf{x}_i}(\boldsymbol{\theta}) \right) - q$, $i = 1, \ldots, N$. The latter two functions represent the negative directional derivative of the criterion evaluated at ξ_N^* in the direction of the degenerate design at $\mathbf{x}_i$.

Lemma 1. *The optimality conditions are:*

(i) ξ_N^* is a D-optimal design on S_N if and only if it satisfies $h_{Di}(\hat{\mathbf{w}}) \leq 0$, for all $i = 1, \ldots, N$;
(ii) ξ_N^* is an optimal design that minimizes $\mathcal{L}_{AC}(\xi_N)$ on S_N if and only it satisfies $h_{Ti}(\hat{\mathbf{w}}) \leq 0$, for all $i = 1, \ldots, N$.

The proof of Lemma 1 is omitted; it is similar to the proof of Lemma 2 in Wong, Yin & Zhou (2017). In practice, conditions in Lemma 1 are replaced by

$$h_{Di}(\hat{\mathbf{w}}) \leq \delta \;\text{ or }\; h_{Ti}(\hat{\mathbf{w}}) \leq \delta, \;\text{ for all } i = 1, \ldots, N, \tag{8}$$

where δ is a small positive number, say 10^{-4}; see Bose & Mukerjee (2015).

To compare two designs ξ^1 and ξ^2, we use two efficiency measures: $\mathrm{Eff}_D(\xi^1, \xi^2) = \mathcal{L}'_D(\xi^1)/\mathcal{L}'_D(\xi^2)$ for D-optimality, and $\mathrm{Eff}_{AC}(\xi^1, \xi^2) = \mathcal{L}_{AC}(\xi^2)/\mathcal{L}_{AC}(\xi^1)$ for AC-optimality. If $\mathrm{Eff}_D(\xi^1, \xi^2) < 1$, ξ^2 is more D-efficient than ξ^1. Likewise if $\mathrm{Eff}_{AC}(\xi^1, \xi^2) < 1$, ξ^2 is more AC-efficient than ξ^1.

3.3. Programming in MATLAB

Example 1 illustrates how we implement Algorithm 1 in MATLAB to find optimal designs for a model frequently used in biological sciences. All computation times reported in the article are from a PC equipped with an Intel(R) Core(TM)2 Quad CPU Q9550@2.83 GHz.

Example 1. Consider the K-compartmental model given by

$$E(Y) = f(x, \boldsymbol{\theta}) = \sum_{j=1}^{K} \theta_j \, exp(-\theta_{K+j} x), \quad x \geq 0, \; 0 < \theta_{K+1} < \theta_{K+2} < \cdots < \theta_{2K}.$$

Our goal is to compute D-optimal designs for the model on a user-selected discrete design space $S_N \subset [a, b]$. Here we use N equally spaced points, given by $x_i = a + (b - a)(i-1)/(N-1)$, $i = 1, \ldots, N$, which is commonly used. For this model, we have $\mathbf{I}_x(\boldsymbol{\theta}) = (\partial f(x, \boldsymbol{\theta})/\partial \boldsymbol{\theta})\,(\partial f(x, \boldsymbol{\theta})/\partial \boldsymbol{\theta})^\top$. Earlier results for this model include those from Dette, Melas & Wong (2006), who noted that the D-optimal designs do not depend on the true values of $\theta_1, \ldots, \theta_K$ and found the number of the support points in the D-optimal designs when $K \leq 3$. The derivation of the optimal design becomes very challenging when $K \geq 4$, as the information matrix approaches singularity.

Our MATLAB program that implements Algorithm 1 for this problem is given in the Supplementary Material, and it can find D-optimal designs for $K \geq 4$. The output of the program gives the support points of the D-optimal design and their positive weights, the value of the function $-\mathcal{L}'_D(\xi_N)$, the computation time and the maximum value of $h_{Di}(\hat{\mathbf{w}})$ in Equation (8). For $K \leq 3$, our results are consistent with those in Dette, Melas & Wong (2006). Our results for $K = 4$ are new, and Figure 1 shows the D-optimal designs when $(\theta_5, \ldots, \theta_8) = (0.1, 0.6, 2.3, 5.5)$ and $S_N \subset [0, 10]$ for different values of N. As N gets large, the support points are clustered around 8 points including the two boundary points 0 and 10. Let $d(x, \boldsymbol{\theta}) = \mathrm{trace}\left(\mathbf{I}^{-1}_{\xi^*_N}(\boldsymbol{\theta})\, \mathbf{I}_x(\boldsymbol{\theta})\right) - q$ and Figure 2 checks the conditions in Equation (8) by the plot of $d(x, \boldsymbol{\theta})$ versus x for $N = 801$. We observe that $d(x, \boldsymbol{\theta}) \leq 0$ for all $x \in [0, 10]$, so the design found by our MATLAB program is D-optimal. Table 1 presents the computation time, objective function, D-efficiency and the optimal design for $N = 801$. These results indicate that Algorithm 1 is fast and effective for finding optimal designs, and the optimal designs are highly efficient for a moderate value of N.

4. PROPERTIES OF ALGORITHM 1 AND OPTIMAL DESIGNS

We provide more details and discuss several important issues related to the computation using Algorithm 1, including a convergence result, scale transformation of design space, transformation of the information matrix, and multi-stage and multi-objective designs.

4.1. Convergence Result

Let S be a given compact set. The set S_N ($\subset S$) is formed by Cartesian product of equally spaced points in each design variable. Let ξ^c be the optimal design on S and let ξ^*_N be the optimal design on S_N, found by solving Equation (7). Yang, Biedermann & Tang (2013, Theorem 5) derived an efficiency lower bound for any continuous design on a discretized design space S_N. We have the following result for ξ^*_N as $N \to \infty$.

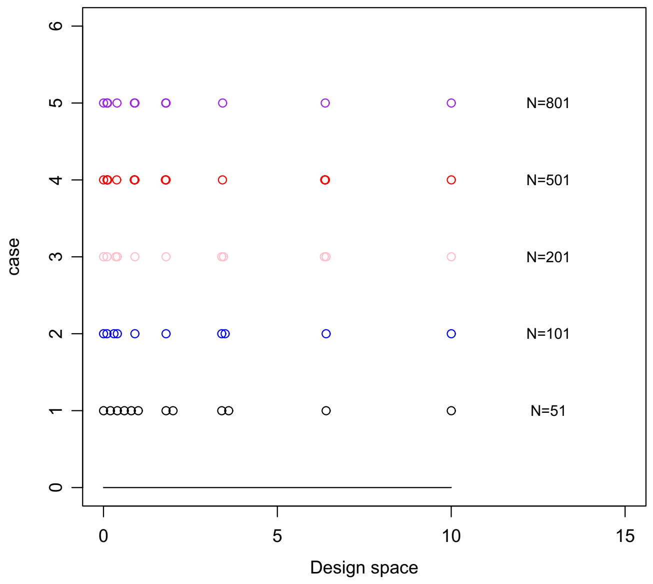

FIGURE 1: Support points of the D-optimal designs for various values of grid size N.

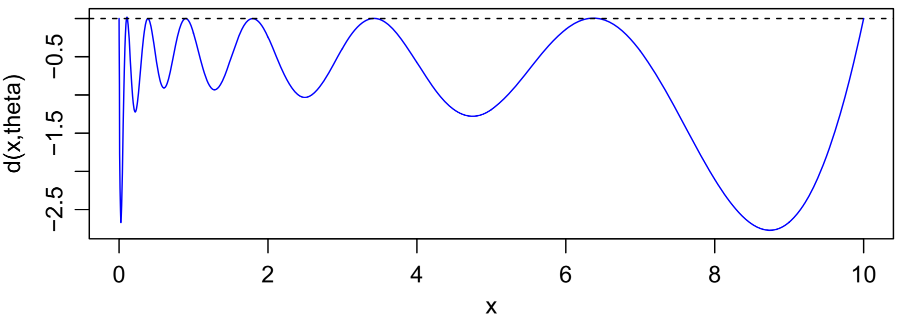

FIGURE 2: Plot of $d(x, \boldsymbol{\theta})$ versus x for the CVX-generated design with $N = 801$.

Theorem 1. *If all the elements of $\boldsymbol{I}_{\boldsymbol{x}}(\boldsymbol{\theta})$ are bounded and continuous functions of $\boldsymbol{x} \in S$, then $\mathcal{L}(\xi_N^*) \to \mathcal{L}(\xi^c)$ as $N \to \infty$.*

The proof of Theorem 1 is in the Appendix. The sequence of designs ξ_N^* may also converge to ξ^c in distribution. In Example 1, $\mathcal{L}(\xi_N^*)$ converges as N gets large, and the support points shown in Figure 1 seem to converge to the 8 points in ξ_{801}^*.

4.2. Scale Transformation of Design Space

D-Optimal designs for NMs and GLMs are usually not scale invariant. We derive a general relationship between D-optimal designs and the design spaces for various models. To fix ideas, let $\xi^*(S_N, \boldsymbol{\theta})$ be the D-optimal design on the design space S_N with true model parameter value $\boldsymbol{\theta}$. Let $\mathbf{V} = \mathrm{diag}(s_1, \ldots, s_k)$ be a $k \times k$ diagonal matrix with $s_j > 0$ $(j = 1, \ldots, k)$ and let $S_N^V = \{\mathbf{V}\mathbf{x}_1, \ldots, \mathbf{V}\mathbf{x}_N\}$ be the transformed design space. This transformation scales each design variable x_j by a factor s_j, for $j = 1, \ldots, k$.

TABLE 1: Results for Example 1 with $K = 4$ and $S = [0, 10]$.

Algorithm 1	$N = 51$	$N = 101$	$N = 201$	$N = 501$	$N = 801$
Time (s)	0.8892	1.1388	1.7784	1.9188	3.2292
$-\mathcal{L}'_D(\xi^*_N)$	0.0034	0.0037	0.0037	0.0037	0.0037
$\mathrm{Eff}_D(\xi^*_N, \xi^*_{801})$	0.9295	0.9973	0.9973	1.0000	1.0000

ξ^*_{801}											
Support points	0.0000	0.1000	0.1125	0.3875	0.8875	0.9000	1.7875	1.8000	3.4250	6.3750	10.0000
Weights	0.1250	0.0032	0.1218	0.1250	0.0369	0.0881	0.0854	0.0396	0.1250	0.1250	0.1250

Theorem 2. *Suppose $\xi^*(S_N, \boldsymbol{\theta}) = \{(\boldsymbol{x}^*_i, w^*_i),\ i = 1, \dots, m\}$ is a D-optimal design for a model with parameter value $\boldsymbol{\theta}$ on S_N. If there exists a constant $q \times q$ non-singular matrix $\boldsymbol{T}$ (not depending on $\boldsymbol{x}$ and $\boldsymbol{\theta}$) and parameter value $\boldsymbol{\theta}^V$ such that*

$$\boldsymbol{I}_{V\boldsymbol{x}}(\boldsymbol{\theta}^V) = \boldsymbol{T}\,\boldsymbol{I}_{\boldsymbol{x}}(\boldsymbol{\theta})\,\boldsymbol{T}^\top, \ \textit{for all } \boldsymbol{x} \in S_N, \tag{9}$$

then $\xi^*(S^V_N, \boldsymbol{\theta}^V) = \{(\boldsymbol{V}\boldsymbol{x}^*_i, w^*_i),\ i = 1, \dots, m\}$ is a D-optimal design for the model with parameter value $\boldsymbol{\theta}^V$ on design space S^V_N.

The proof of Theorem 2 is in the Appendix. The result is a generalization of the scale invariance property in linear models and can be applied to any model. For linear models, $\mathbf{I}_{\mathbf{x}}(\boldsymbol{\theta})$ does not depend on parameter $\boldsymbol{\theta}$, so the condition in Equation (9) becomes $\mathbf{I}_{\mathbf{Vx}}(\boldsymbol{\theta}) = \mathbf{T}\,\mathbf{I}_{\mathbf{x}}(\boldsymbol{\theta})\,\mathbf{T}^\top$ and the result in Theorem2 becomes a scale invariance property of D-optimal designs. For NMs and GLMs, the D-optimal designs on S_N and S^V_N are related through different parameter values: $\boldsymbol{\theta}$ and $\boldsymbol{\theta}^V$. Our experience is that Equation (9) holds for many models and matrix $\mathbf{T}$ is usually diagonal and depends on the scale matrix $\mathbf{V}$. Example 2 illustrates some of these ideas. In Section 4.3 and Example 3 we use Theorem 2 to handle the situation when we have an ill-conditioned information matrix.

Example 2. Consider the logistic model with two variables and an interaction term, and the probability of a response is

$$P(\boldsymbol{x}, \boldsymbol{\theta}) = \frac{exp(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2)}{1 + exp(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2)}, \ 0 \le x_1 \le b_1, \ 0 \le x_2 \le b_2,$$

where $\boldsymbol{\theta} = (\theta_0, \dots, \theta_3)^\top$ is the parameter vector. We use N_1 and N_2 equally spaced grid points from $[0, b_1]$ and $[0, b_2]$, respectively, to form S_N with $N = N_1 N_2$. A direct calculation shows the information matrix for this model at a single observation is

$$\boldsymbol{I}_{\boldsymbol{x}}(\boldsymbol{\theta}) = \frac{\exp(z^\top \boldsymbol{\theta})}{(1 + \exp(z^\top \boldsymbol{\theta}))^2}\, z\, z^\top, \ \ \textit{with}\, z = (1, x_1, x_2, x_1 x_2)^\top.$$

Let $\boldsymbol{V} = diag(s_1, s_2)$, $\boldsymbol{\theta}^V = (\theta_0, \theta_1/s_1, \theta_2/s_2, \theta_3/s_1 s_2)^\top$ and let $\boldsymbol{T} = diag(1, s_1, s_2, s_1 s_2)$. It is easy to verify that Equation (9) holds, and the same is true when $k > 2$ and there are several interaction terms in the model.

4.3. Transformation on the Information Matrix

Sometimes the information matrix is ill-conditioned and computational problems can arise. Consequently, the optimal design problem in Equation (7) with objective function involving $\mathbf{I}_{\xi_N}^{-1}(\boldsymbol{\theta})$ may not be easy to solve. For such situations, we suggest to decompose the information matrix and write it as $\mathbf{I}_{\xi_N}(\boldsymbol{\theta}) = \mathbf{D}\,\mathbf{B}\,\mathbf{D}^\top$ before feeding it into Algorithm 1, where $\mathbf{B}$ and $\mathbf{D}$ are $q \times q$ non-singular matrices. This strategy is especially helpful if it is easier to find $\mathbf{D}^{-1}$ and $\mathbf{B}^{-1}$ numerically than $\mathbf{I}_{\xi_N}^{-1}(\boldsymbol{\theta})$. For many models, scaling the design space can result in a different form of $\mathbf{I}_{\xi_N}(\boldsymbol{\theta})$, and by Equation (9) we have $\mathbf{I}_{\mathbf{x}}(\boldsymbol{\theta}) = \mathbf{T}^{-1}\mathbf{I}_{\mathbf{V}\mathbf{x}}(\boldsymbol{\theta}^V)\,\mathbf{T}^{-\top}$, which leads to $\mathbf{I}_{\xi_N}(\boldsymbol{\theta}) = \mathbf{T}^{-1}\left(\sum_{i=1}^N w_i \mathbf{I}_{\mathbf{V}\mathbf{x}_i}(\boldsymbol{\theta}^V)\right)\mathbf{T}^{-\top}$. Our strategy is to work with the inverse of $\left(\sum_{i=1}^N w_i \mathbf{I}_{\mathbf{V}\mathbf{x}_i}(\boldsymbol{\theta}^V)\right)$ in Algorithm 1. There are other transformations that can be used, such as a simple one that scales $\mathbf{I}_{\xi_N}(\boldsymbol{\theta})$ by a constant factor for all the elements in the matrix. The transformations discussed here can be applied to all optimality criteria, even though Equation (9) is developed to show a property of D-optimal designs. We now show that an appropriate transformation on the information matrix can help find c-optimal designs effectively in Example 3.

Example 3. Guess, Crump & Peto (1977) proposed a Probit like dose response model to ascertain the probability that an animal will have a type of cancer tumour as the dose of a compound varies. This postulated probability as a function of the dose x assumes all the coefficients in the model are non-negative and is given by $P(x, \boldsymbol{\theta}) = 1 - exp(-(\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3))$, $x \in [0, b]$. Hoel & Jennrich (1979) used theory and found c-optimal designs for estimating the excess probability $P(t, \boldsymbol{\theta}) - P(0, \boldsymbol{\theta})$ of an animal developing cancer if it is continuously exposed to a carcinogenic compound at dose t, and $P(0, \boldsymbol{\theta})$ is the probability due to background sources. We applied Algorithm 1 and were able to confirm the results in Hoel & Jennrich (1979) for the case when $t = 0.5$ by computing the c-optimal for estimating $g_1(\boldsymbol{\theta}) = P(0.5, \boldsymbol{\theta}) - P(0, \boldsymbol{\theta})$ when $b = 500$. Frequently, differences between scalar quantities are measured by their difference or their ratio and usually the former is simpler to estimate and draw inference. Our CVX based program is flexible in that if we wish to estimate the ratio of the two quantities, Algorithm 1 can directly find the optimal design for estimating $g_2(\boldsymbol{\theta}) = P(0.5, \boldsymbol{\theta})/P(0, \boldsymbol{\theta})$ with the same set of nominal values. Table 2 shows selected optimal designs for these two objectives for various values of N. The optimal designs for estimating $g_2(\boldsymbol{\theta})$ are new. All the optimal designs have 4 support points with unequal weights. Algorithm 1 is fast, taking about 11 s when $N = 5{,}001$. The efficiency is very high for small value of N; $Eff_{AC}(\xi_6^*, \xi_{5,001}^*)$ is 0.9190 for estimating $g_1(\boldsymbol{\theta})$ and 0.9416 for estimating $g_2(\boldsymbol{\theta})$. For this problem, we transformed the objective function by rescaling the variable x appropriately using $x' = x/b$. The detailed transformation and the MATLAB codes are given in the Supplementary Material.

4.4. Two-stage Designs and Multi-objective Designs

Algorithm 1 can also be used to find other types of optimal designs, including multi-stage and multi-objective designs. In multi-stage design, let ξ_0 be the first stage design, let ξ_N be the design at the next stage, and let n_0 and n_1 be the number of observations to be taken at the first and second stages, respectively. The combined design is denoted by $\xi_0 + \xi_N$, which is defined in Yang, Biedermann & Tang (2013). The information matrix of the combined design is $\mathbf{I}_{\xi_0+\xi_N}(\boldsymbol{\theta}) = \frac{n_0}{n_0+n_1}\mathbf{I}_{\xi_0}(\boldsymbol{\theta}) + \frac{n_1}{n_0+n_1}\mathbf{I}_{\xi_N}(\boldsymbol{\theta})$. The commonly used criteria can be formed similarly as in (4), (5), or (6) by replacing the information matrix $\mathbf{I}_{\xi_N}(\boldsymbol{\theta})$ by $\mathbf{I}_{\xi_0+\xi_N}(\boldsymbol{\theta})$ to find an one-stage design.

An alternative is that, given the first stage design, we want to improve it in the second stage design by minimizing the objective function over all second-stage designs ξ_N to find the

TABLE 2: c-Optimal designs for Example 3 with $\boldsymbol{\theta} = (0.01, 0.000267377, 0, 0)^\top$.

N	Time (s)	Optimal design					$\mathcal{L}_{AC}(\xi_N^*)$
		Estimating $g_1(\boldsymbol{\theta})$					
6	0.9828	Support points	0.0000	100.0000	300.0000	500.0000	1.1142×10^{-5}
		Weights	0.2315	0.5364	0.1887	0.0434	
51	0.5928	Support points	0.0000	80.0000	340.0000	500.0000	1.0252×10^{-5}
		Weights	0.2739	0.5359	0.1414	0.0488	
501	1.6068	Support points	0.0000	83.0000	342.0000	500.0000	1.0240×10^{-5}
		Weights	0.2668	0.5324	0.1488	0.0520	
5,001	11.5288	Support points	0.0000	82.6000	342.4000	500.0000	1.0240×10^{-5}
		Weights	0.2677	0.5325	0.1479	0.0519	
		Estimating $g_2(\boldsymbol{\theta})$					
6	1.2168	Support points	0.0000	100.0000	300.0000	500.0000	0.2192
		Weights	0.4493	0.3844	0.1352	0.0311	
51	1.0764	Support points	0.0000	80.0000	340.0000	500.0000	0.2065
		Weights	0.4859	0.3794	0.1001	0.0346	
501	1.7316	Support points	0.0000	83.0000	342.0000	500.0000	0.2064
		Weights	0.4810	0.3769	0.1053	0.0368	
5,001	11.8561	Support points	0.0000	82.6000	342.4000	500.0000	0.2064
		Weights	0.4815	0.3770	0.1048	0.0367	

optimal designs. For fixed sample sizes n_0 and n_1 in the multi-stage design problem, we note that $\mathbf{I}_{\xi_0+\xi_N}(\boldsymbol{\theta})$ is still linear in $\mathbf{w}$ and Algorithm 1 can be directly applied for finding multi-stage optimal designs. In addition, this two-stage design approach can be easily extended to construct sequential designs.

Our optimal designs depend on the true value of $\boldsymbol{\theta}$, which is frequently unknown in practice. Nominal values are needed and even though they can come from previous studies, they may not agree. Because a wrong set of nominal values can produce very inefficient designs, a multiple-objective optimal design is helpful to incorporate the uncertainty in each set of the nominal values. To fix ideas, suppose there are two possible nominal values, $\boldsymbol{\theta}^*$ and $\boldsymbol{\theta}^{**}$ for $\boldsymbol{\theta}$ and we wish to design a study so that the implemented design is efficient regardless which of the two sets of nominal values is valid. One approach is to modify objective functions in Equations (4) and (6) as follows:

$$\mathcal{L}_D^\alpha(\xi_N) = -(1-\alpha)\ \log\left(\det(\mathbf{I}_{\xi_N}(\boldsymbol{\theta}^*))\right) - \alpha\ \log\left(\det(\mathbf{I}_{\xi_N}(\boldsymbol{\theta}^{**}))\right),$$

$$\mathcal{L}_{AC}^\alpha(\xi_N) = (1-\alpha)\ \mathrm{trace}\left(\mathbf{C}^\top\ \mathbf{I}_{\xi_N}^{-1}(\boldsymbol{\theta}^*)\ \mathbf{C}\right) + \alpha\ \mathrm{trace}\left(\mathbf{C}^\top\ \mathbf{I}_{\xi_N}^{-1}(\boldsymbol{\theta}^{**})\ \mathbf{C}\right),$$

where α is a user-selected number in [0, 1] and reflects the strength of belief which set is closer to the truth.

When $\alpha = 0$ we only use the information obtained from assuming $\boldsymbol{\theta}^*$, and when $\alpha = 1$ we only use the information from $\boldsymbol{\theta}^{**}$. If there are more than two possible values of $\boldsymbol{\theta}$ to be considered, weighted average objective functions can be defined similarly. Since $\mathcal{L}_D^\alpha(\xi_N)$ and $\mathcal{L}_{AC}^\alpha(\xi_N)$ are still convex functions of $\mathbf{w}$, we modify Algorithm 1 slightly for finding optimal designs: in Step (i) we input two parameter values $\boldsymbol{\theta}^*$ and $\boldsymbol{\theta}^{**}$ and design points $\mathbf{x}_1, \ldots, \mathbf{x}_N$, in Step (ii) we compute $\mathbf{I}_{\mathbf{x}_i}(\boldsymbol{\theta}^*)$ and $\mathbf{I}_{\mathbf{x}_i}(\boldsymbol{\theta}^{**})$ for each i, and Steps (iii) and (iv) are the same. The conditions in Lemma 1 are now changed to verify if a design is a multi-objective optimal design as follows.

Theorem 3. *For a user-selected choice of N, $S_N \subset S$ and α in* $[0, 1]$,

(i) the design ξ_N^* is optimal for minimizing $\mathcal{L}_D^\alpha(\xi_N)$ on S_N, if and only if it satisfies, for all $i = 1, \ldots, N$,

$$(1-\alpha)\, trace\left(\boldsymbol{I}_{\xi_N^*}^{-1}(\boldsymbol{\theta}^*)\, \boldsymbol{I}_{\boldsymbol{x}_i}(\boldsymbol{\theta}^*)\right) + \alpha\, trace\left(\boldsymbol{I}_{\xi_N^*}^{-1}(\boldsymbol{\theta}^{**})\, \boldsymbol{I}_{\boldsymbol{x}_i}(\boldsymbol{\theta}^{**})\right) - q \le 0;$$

(ii) the design ξ_N^* is optimal for minimizing $\mathcal{L}_{AC}^\alpha(\xi_N)$ on S_N, if and only if it satisfies, for all $i = 1, \ldots, N$,

$$(1-\alpha)\, trace\left(\boldsymbol{C}^\top \left[\boldsymbol{I}_{\xi_N^*}^{-1}(\boldsymbol{\theta}^*)\, \boldsymbol{I}_{\boldsymbol{x}_i}(\boldsymbol{\theta}^*) \boldsymbol{I}_{\xi_N^*}^{-1}(\boldsymbol{\theta}^*) - \boldsymbol{I}_{\xi_N^*}^{-1}(\boldsymbol{\theta}^*)\right] \boldsymbol{C}\right) +$$

$$\alpha\, trace\left(\boldsymbol{C}^\top \left[\boldsymbol{I}_{\xi_N^*}^{-1}(\boldsymbol{\theta}^{**})\, \boldsymbol{I}_{\boldsymbol{x}_i}(\boldsymbol{\theta}^{**}) \boldsymbol{I}_{\xi_N^*}^{-1}(\boldsymbol{\theta}^{**}) - \boldsymbol{I}_{\xi_N^*}^{-1}(\boldsymbol{\theta}^{**})\right] \boldsymbol{C}\right) \le 0.$$

The proof of Theorem 3 is deferred to the Appendix. For practical applications, we relax the conditions in Theorem 3 the same way as we did in Equation (8).

5. APPLICATIONS

We now apply Algorithm 1 to find different types of optimal designs for various models including situations when the design space is irregularly shaped or models with several design variables.

Much of the design space assumed in the literature is the prototype cuboid or similar form. Sometimes irregularly shaped design spaces arise in practice. Design problems for an irregularly shaped design space can pose additional theoretical challenges. Using Example 4 as an example, we show that Algorithm 1 can find optimal designs for a logistic model with two factors on various design spaces.

Example 4. We revisit Example 2 with the model now defined on design spaces in Figure 3. Design spaces in (a) and (b) are regularly shaped and those in (c) and (d) are irregularly shaped. We discretize interval $[0, 1]$ *or* $[0, 2]$ with 101 equally spaced points for each factor space, so S_N contains $N = 101^2, 101^2$, 9,326, and 5,151 points, respectively, for the design spaces in (a), (b), (c) and (d). Figure 3 displays the support points and the weights of D-optimal designs. It shows that the number of support points in the D-optimal designs for this model depends on the nominal value of $\boldsymbol{\theta}$ and the shape of the design space. For these design spaces the number of support points is 4, 5, or 6.

Example 5. This is an application with several design variables (large k and q) in Yang, Zhang & Huang (2011), who studied a logistic model with multiple design variables and no interactions. A direct calculation shows that the information matrix is

$$\boldsymbol{I}_{\boldsymbol{x}}(\boldsymbol{\theta}) = \frac{exp(\theta_0 + \theta_1 x_1 + \cdots + \theta_k x_k)}{\left(1 + exp(\theta_0 + \theta_1 x_1 + \cdots + \theta_k x_k)\right)^2} \begin{pmatrix} 1 \\ \boldsymbol{x} \end{pmatrix} (1\, \boldsymbol{x}^\top), \quad with\, \boldsymbol{x}^\top = (x_1, \ldots, x_k).$$

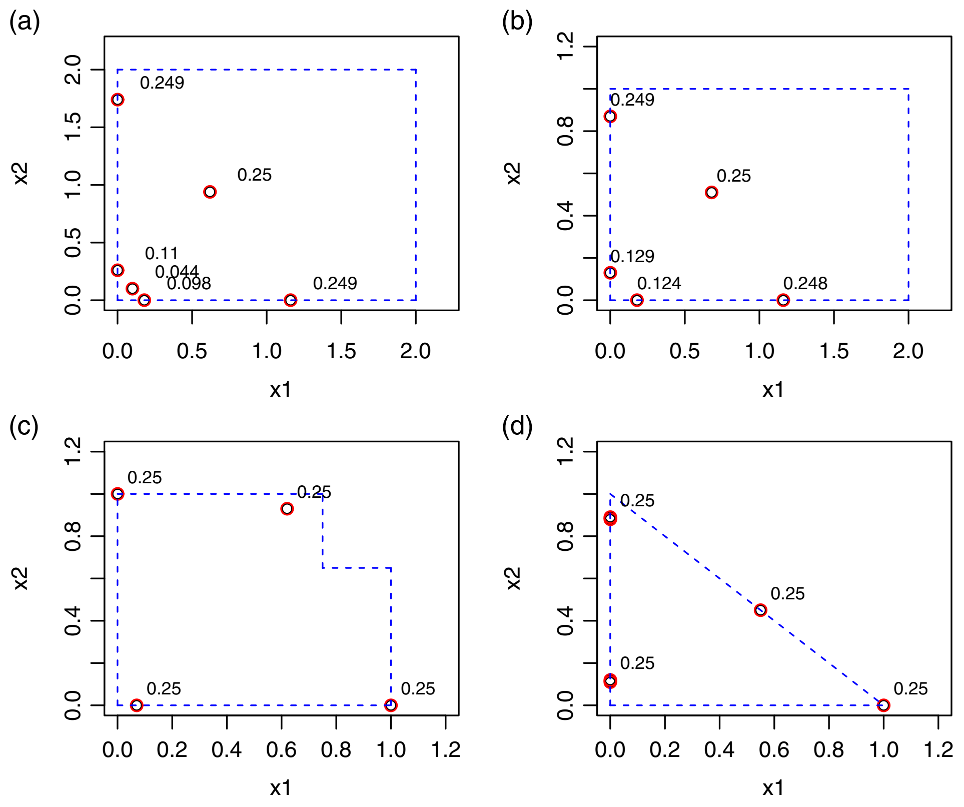

FIGURE 3: *D*-Optimal designs for Example 4 on various design spaces bounded by the dotted lines. The support points and weights are indicated by the circles and numbers, respectively. In (a) and (c), $\boldsymbol{\theta} = (-2, 3, 2, 1)^\top$; in (b) and (d) $\boldsymbol{\theta} = (-2, 3, 4, 1)^\top$.

TABLE 3: *A*- and *D*-optimal designs for Example 5 found by Algorithm 1.

	$k = q - 1$	$N = 2^k$	No. of points in ξ_N^*	Objective function	Time (s)
D-Optimality				$\mathcal{L}_D'(\xi_N^*)$	
	7	128	8	−0.0200	0.8580
	10	1024	22	−0.0176	8.7205
	12	4096	33	−0.0383	55.6876
	13	8192	45	−0.0465	247.5268
A-Optimality				$\frac{1}{q}\,\mathrm{trace}(\mathbf{I}_{\xi_N^*}^{-1}(\boldsymbol{\theta}))$	
	7	128	10	139.0272	1.0296
	10	1024	31	173.9662	8.9233
	12	4096	43	77.8354	54.2804
	13	8192	57	63.7061	251.7388

TABLE 4: D- and E-optimal designs for the logistic model with $k = 7$, $q = 12$ and $\boldsymbol{\theta} = (1.0, -6.0, 5.79, 0.25, 3.15, -0.9, -1.2, 2.06, -0.5, -1.08, 0.65, 0.01)^{\top}$. The notation λ_{min} denotes the smallest eigenvalue of the matrix in brackets.

Design space S_N	No. of points in ξ_N^*	Objective function	Time (s)
$N = N_1 \times N_2 \times \cdots \times N_7$		D-Optimality $\mathcal{L}'_D(\xi_N^*)$	
$N = 2^7$	21	−0.0905	1.5532
$N = 3^7$	32	−0.1246	11.7313
$N = 5 \times 5 \times 5 \times 2 \times 2 \times 2 \times 3$	37	−0.1254	20.8573
$N = 5 \times 5 \times 5 \times 5 \times 2 \times 2 \times 3$	40	−0.1256	98.1325
		E-Optimality $\lambda_{min}\left(\mathbf{I}_{\xi_N^*}(\boldsymbol{\theta})\right)$	
$N = 2^7$	14	0.0036	1.3104
$N = 3^7$	16	0.0049	7.2696
$N = 5 \times 5 \times 5 \times 2 \times 2 \times 2 \times 3$	22	0.0049	19.6093
$N = 5 \times 5 \times 5 \times 5 \times 2 \times 2 \times 3$	17	0.0049	91.3367

Yang, Zhang & Huang (2011) studied optimal designs based on 2^k support points with each variable taking two points at -1 *and* $+1$. In particular, when $k + 1$ is a multiple of 4, the minimum number of support points can be as small as $k + 1$ and a procedure is proposed to construct such designs. When $k + 1$ is not a multiple of 4, the minimum number of support points can be much larger than $k + 1$. In this case, Algorithm 1 can be directly applied to find optimal designs on S_N including all the 2^k points. For $k = 7$, Algorithm 1 finds a D-optimal design with 8 support points with equal weight, which is the smallest number of support points to make the information matrix non-singular. In the Supplementary Material, a 10-point A-optimal design for $k = 7$ variables is presented. The computation is very fast for $k \leq 13$ even though $N = 2^{13} = 8{,}192$ is quite large. Table 3 lists the number of support points of the A- and D-optimal designs and the computation time.

Example 6. This example computes D- and E-optimal designs for the logistic model with 7 covariates $x_1, \ldots, x_7$ and 4 interaction terms: x_1x_2, x_1x_3, x_1x_4 and x_1x_5. The design space is $S = [-1, +1]^7$ and we discretize the design space of each variable x_j by dividing it into N_j equally spaced grid points in $[-1, +1]$, $j = 1, \ldots, 7$, so that S_N has $N = N_1N_2 \ldots N_7$ points. We have used various combinations of grid points in S_N and apply Algorithm 1 for finding optimal designs. Representative results for D- and E-optimal designs are in Table 4. From Table 4, the optimal designs on S_N with $N = 2^7$ are only 72.6% and 72.1% efficient as those with $N = 3^7$ for D- and E-optimality criteria, respectively. When $N > 3^7$, the efficiency of the optimal designs increases very little in this example. We also notice that the number of support points in the E-optimal designs is smaller than that in the D-optimal designs. When the theoretical results for optimal designs are extremely hard to derive for the logistic model with several covariates and interaction terms, Algorithm 1 can easily find the optimal designs on various S_N.

6. ANOTHER CVX BASED ALGORITHM

When there are many design variables in the model, the number of grid points needed to cover the design space can be huge. If N is huge, say $N > 10,000$, Algorithm 1 may be slow or fail to find the optimal designs. For this situation, we recommend modifying Algorithm 1 to speed up the algorithm. For given $\boldsymbol{\theta}$ and S_N we define function, $\Delta(\mathbf{x}, \xi_N) = \text{trace}\left(\mathbf{I}_{\xi_N}^{-1}(\boldsymbol{\theta})\,\mathbf{I}_{\mathbf{x}}(\boldsymbol{\theta})\right) - q$ for D-optimality, or

$$\Delta(\mathbf{x}, \xi_N) = \text{trace}\left(\mathbf{C}^\top\,\mathbf{I}_{\xi_N}^{-1}(\boldsymbol{\theta})\,\mathbf{I}_{\mathbf{x}}(\boldsymbol{\theta})\mathbf{I}_{\xi_N}^{-1}(\boldsymbol{\theta})\mathbf{C}\right) - \text{trace}\left(\mathbf{C}^\top\,\mathbf{I}_{\xi_N}^{-1}(\boldsymbol{\theta})\mathbf{C}\right)$$

for the criterion in (6). From Lemma 1, ξ_N^* is optimal if and only if $\Delta(\mathbf{x}_i, \xi_N^*) \leq 0$ for all $i = 1, \ldots, N$.

Yang, Biedermann & Tang (2013) proposed a 3-step general algorithm to find ξ_N^* with the following steps:

Step (i): Start with a small subset $S^{(0)} \subset S_N$. Usually N_0 is much smaller than N, but larger than k. Set $t = 0$.
Step (ii): Compute optimal weights for the support points in $S^{(t)}$ and denote the optimal design on $S^{(t)}$ as $\xi^{(t)}$. Update $S^{(t)}$ by removing those points with zero weights.
Step (iii): Find $\mathbf{x}_t^* = arg\,max_{\mathbf{x}\in S_N}\Delta(\mathbf{x}, \xi^{(t)})$. If $\Delta(\mathbf{x}_t^*, \xi^{(t)}) < \delta$ (a pre-specified small positive number), then $\xi^{(t)}$ is the desired optimal design. Otherwise, set $t = t + 1$ and $S^{(t+1)} = S^{(t)} \bigcup \{\mathbf{x}_t^*\}$ and go to Step (ii).

Step (ii) is accomplished by using Newton's iterative method to obtain the optimal weights. There are two practical issues with this algorithm. First, the user needs to feed the first and second derivatives of $\mathcal{L}(\xi_{N_t})$ with respect to the vector of weights $\mathbf{w} = (w_1, \ldots, w_{N_t})^\top$ to the codes for each iteration. Here N_t is the number of design points in $S^{(t)}$, and $w_1, \ldots, w_{N_t}$ are the weights in ξ_{N_t}. This task can be both laborious and complex, especially when the model is complicated. The second problem is that the Newton's iteration procedure does not guarantee to find the optimal weights and further, sometimes the weights can become negative and violate the conditions in Equation (3).

To overcome these issues, we develop a fast CVX based algorithm by combining CVX with the algorithm proposed by Yang, Biedermann & Tang (2013). The key is to recognize that Step (ii) above is similar to solving an optimization problem in Equation (7). Our experience is that if we use CVX instead of the Newton's iterative method in Step (ii), we optimize the weights effectively without the issues related to Newton iteration procedure. The combined algorithm is outlined as follows.

Algorithm 2 (Iterative procedure using CVX).

Step (i): It is the same as in Step (i) of Yang, Biedermann & Tang (2013). N_0 points in $S^{(0)}$ can be randomly selected from S_N or uniformly distributed on S_N.
Step (ii): Use CVX to compute optimal weights for the support points in $S^{(t)}$ and denote the optimal design on $S^{(t)}$ as $\xi^{(t)}$. Update $S^{(t)}$ by removing those points with zero weights.
Step (iii): It is the same as in Step (iii) of Yang, Biedermann & Tang (2013).

We follow Steps (ii) and (iii) of Algorithm 1 to implement Step (ii) of Algorithm 2. Algorithm 2 is very fast and can find optimal designs on S_N with huge N. In Algorithm 1 we use CVX to compute optimal weights for all the points in S_N, which can be slow when we have a high dimensional design problem and N is huge. At each iteration in Algorithm 2, we use CVX to compute the optimal weights for all the points in $S^{(t)}$. Since $S^{(t)}$ usually has a small number of points, this is the main reason that Algorithm 2 can handle a large number of grid points. Using Algorithm 2 we have computed optimal designs on S_N with N as large as 10^6.

TABLE 5: Computation times of ALGI (Algorithm 1) and ALGII (Algorithm 2).

	D-Optimal design		*A*-Optimal design	
Computation time (s)	ALGI	ALGII	ALGI	ALGII
$N = 500$	1.1312	6.0937	1.3740	3.4109
$N = 1{,}000$	2.0932	7.1963	2.1478	3.7916
$N = 5{,}000$	10.9751	8.7841	11.0433	4.2915
$N = 10{,}000$	33.7330	9.7546	34.1228	5.2793
$N = 20{,}000$	123.2608	11.0105	123.3329	5.6068
$N = 50{,}000$		16.9809		8.4723
$N = 100{,}000$		28.6356		15.3243
$N = 1{,}000{,}000$		203.2512		99.5638

Table 5 compares the computation times required by Algorithms 1 and 2 for finding the optimal design in Example 1 in Yang, Biedermann & Tang (2013) using various values of N. When Algorithm 1 takes too long to produce the design, we only report the results for Algorithm 2. From Table 5, we observe that Algorithm 1 is faster than Algorithm 2 when $N \leq 1,000$, and Algorithm 2 is much faster than Algorithm 1 when $N \geq 10,000$. In addition, the ratio of computation time of Algorithm 2 to N decreases as N increases, which makes Algorithm 2 very efficient for large N. We tried to code the algorithm in Yang, Biedermann & Tang (2013) in the software R and its resulting performance was very slower. We recognize that our codes were implemented on different platforms and on computers with different capabilities and so it is problematic to have a truly fair comparison.

7. CONCLUSION

Our article provides theoretical insights into optimal designs found on a discretized design space and the theoretical optimal designs on the continuous design space. We study invariance properties of the optimal designs we found for NMs and GLMs, including relationships between optimal designs found on different design spaces and how an effective transformation of the information matrix can help us find the optimal design. We also recommend strategies when scaling issues arise due to either a large design space or wildly varying magnitudes of the elements in the information matrix that make it ill-conditioned. We emphasize that algorithms are important tools for finding optimal designs for complex models and they should be fast, flexible and easy to use.

We show that CVX based algorithms are very useful tools for finding optimal designs. In particular, they are fast and flexible. Our Algorithms 1 and 2 easily find various types of optimal designs for any model on a user-selected discretized design space S_N. Algorithm 1 is effective and fast for $N < 10{,}000$ and we recommend Algorithm 2 when $N \geq 10{,}000$. Our experience is that we often do not require a large value of N to obtain a highly efficient design and so we believe Algorithm 1 suffices in practice. We close by noting that there is a version of CVX in R called CVXR that solves convex optimization problems. Our experience is that at this time, this program works very slow compared with the MATLAB codes but we note that CVXR is currently pending an update.

Four MATLAB programs are provided for computing D-, c-, A- and E-optimal designs in Examples 1, 3, 5 and 6. We also give the detailed scale transformation of the objective function in Example 3 and an A-optimal design in Example 5 in Supplementary Material.

APPENDIX

Proof of Theorem 1. Let $\mathbf{x}_1^c, \ldots, \mathbf{x}_m^c \in S$ be the support points of ξ^c and let the optimal weights be $w_1^c, \ldots, w_m^c$. From the construction of S_N $(\subset S)$, there exists a sequence of distinct points, $\mathbf{u}_1^{(N)}, \ldots, \mathbf{u}_m^{(N)}$ in S_N, such that $\lim_{N\to\infty} \mathbf{u}_j^{(N)} = \mathbf{x}_j^c$ for $j = 1, \ldots, m$. For each N, define a distribution ξ_N^1 with m support points, $\mathbf{u}_1^{(N)}, \ldots, \mathbf{u}_m^{(N)}$, and weights as, $w_1^c, \ldots, w_m^c$, respectively. Then we have, for each N,

$$\mathcal{L}(\xi^c) \le \mathcal{L}(\xi_N^*) \le \mathcal{L}(\xi_N^1),$$

since ξ^c is the optimal design on S, ξ_N^* is the optimal design on $S_N \subset S$ and ξ_N^1 is a design on S_N. As $N \to \infty$, ξ_N^1 converges to ξ^c in distribution by the definition of ξ_N^1. If all the elements of $\mathbf{I}_\mathbf{x}$ are bounded and continuous functions of $\mathbf{x} \in S$, we have $\mathbf{I}_{\xi_N^1}(\boldsymbol{\theta})$ converges to $\mathbf{I}_{\xi^c}(\boldsymbol{\theta})$ and $\mathcal{L}(\xi_N^1)$ converges to $\mathcal{L}(\xi^c)$, which implies that $\mathcal{L}(\xi_N^*)$ converges to $\mathcal{L}(\xi^c)$.

■

Proof of Theorem 2. From (9) we have

$$\mathbf{I}_{\xi(S_N^V,\boldsymbol{\theta}^V)}(\boldsymbol{\theta}^V) = \sum_{i=1}^N w_i \mathbf{I}_{\mathbf{V}\mathbf{x}_i}(\boldsymbol{\theta}^V) = \mathbf{T}\left(\sum_{i=1}^N w_i \mathbf{I}_{\mathbf{x}_i}(\boldsymbol{\theta})\right)\mathbf{T}^\top = \mathbf{T}\,\mathbf{I}_{\xi(S_N,\boldsymbol{\theta})}(\boldsymbol{\theta})\,\mathbf{T}^\top,$$

so minimizing $-\det\left(\mathbf{I}_{\xi(S_N^V,\boldsymbol{\theta}^V)}(\boldsymbol{\theta}^V)\right)$ over $\mathbf{w}$ is equivalent to minimizing $-\det\left(\mathbf{I}_{\xi(S_N,\boldsymbol{\theta})}(\boldsymbol{\theta})\right)$ over $\mathbf{w}$. This implies that if $\xi^*(S_N,\boldsymbol{\theta}) = \{(\mathbf{x}_i^*, w_i^*),\ i = 1, \ldots, m\}$ is a D-optimal design for parameter $\boldsymbol{\theta}$ on S_N, then $\xi^*(S_N^V, \boldsymbol{\theta}^V) = \{(\mathbf{V}\mathbf{x}_i^*, w_i^*),\ i = 1, \ldots, m\}$ is a D-optimal design for parameter $\boldsymbol{\theta}^V$ on design space S_N^V.

■

Proof of Theorem 3. (i) By considering the directional derivative of $\mathcal{L}_D(\xi_N)$ in (4) at ξ_N^*, we obtain $\sum_{i=1}^N w_i \operatorname{trace}\left(\mathbf{I}_{\xi_N^*}^{-1}(\boldsymbol{\theta})\,\mathbf{I}_{\mathbf{x}_i}(\boldsymbol{\theta}) - I_q\right) \le 0$, for all $\mathbf{w}$, where I_q is the $q \times q$ identity matrix. Applying this result to the objective function $\mathcal{L}_D^\alpha(\xi_N)$, we have, for all $\mathbf{w}$,

$$(1-\alpha)\sum_{i=1}^N w_i \operatorname{trace}\left(\mathbf{I}_{\xi_N^*}^{-1}(\boldsymbol{\theta}^*)\,\mathbf{I}_{\mathbf{x}_i}(\boldsymbol{\theta}^*) - I_q\right) + \alpha\sum_{i=1}^N w_i \operatorname{trace}\left(\mathbf{I}_{\xi_N^*}^{-1}(\boldsymbol{\theta}^{**})\,\mathbf{I}_{\mathbf{x}_i}(\boldsymbol{\theta}^{**}) - I_q\right) \le 0,$$

which leads to

$$\sum_{i=1}^N w_i \left[(1-\alpha)\operatorname{trace}\left(\mathbf{I}_{\xi_N^*}^{-1}(\boldsymbol{\theta}^*)\,\mathbf{I}_{\mathbf{x}_i}(\boldsymbol{\theta}^*)\right) + \alpha\operatorname{trace}\left(\mathbf{I}_{\xi_N^*}^{-1}(\boldsymbol{\theta}^{**})\,\mathbf{I}_{\mathbf{x}_i}(\boldsymbol{\theta}^{**})\right) - q\right] \le 0.$$

This implies the result in part (i). The proof of part (ii) is similar and is omitted.

■

ACKNOWLEDGEMENTS

The authors are grateful to the Editor and referees for their helpful comments and suggestions. This research work is supported by Discovery Grants from the Natural Science and Engineering Research Council of Canada. Wong was partially supported by a grant from the National Institute of General Medical Sciences of the National Institutes of Health. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

BIBLIOGRAPHY

Berger, M. P. F. & Wong, W. K. (2009). *An Introduction to Optimal Designs for Social and Biomedical Research*. Wiley, Chichester, UK.

Biedermann, S., Dette, H., & Zhu, W. (2006). Optimal designs for dose–response models with restricted design spaces. *Journal of the American Statistical Association*, 101, 747–759.

Bose, M. & Mukerjee, R. (2015). Optimal design measures under asymmetric errors, with application to binary design points. *Journal of Statistical Planning and Inference*, 159, 28–36.

Boyd, S. & Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press, New York.

Broudiscou, A., Leardi, R., & Phan-Tan-Luu, R. (1996). Genetic algorithm as a tool for selection of *D*-optimal design. *Chemometrics and Intelligent Laboratory Systems*, 35, 105–116.

Cuervo, D. P., Goos, P., & Sorensen, K. (2016). Optimal design of large-scale screening experiments: a critical look at the coordinate-exchange algorithm. *Statistics and Computing*, 26, 15–28.

Dassel, K. A. & Rawlings, J. O. (1988). Experimental design strategy for the Weibull dose response model. *Environmental Pollution*, 53, 333–349.

Dette, H. & Biedermann, S. (2003). Robust and efficient designs for the Michaelis–Menten model. *Journal of the American Statistical Association*, 98, 679–686.

Dette, H., Melas, V. B., & Wong, W. K. (2006). Locally D-optimal designs for exponential regression models. *Statistica Sinica*, 16, 789–803.

Duarte, B. P. M. & Wong, W. K. (2014). A semi-infinite programming based algorithm for finding minimax optimal designs for nonlinear models. *Statistics and Computing*, 24, 1063–1080.

Duarte, B. P. M., Wong, W. K., & Atkinson, A. C. (2015). A semi-infinite programming based algorithm for determining T-optimum designs for model discrimination. *Journal of Multivariate Analysis*, 135, 11–24.

Duarte, B. P. M., Wong, W. K., & Dette, H. (2018). Adaptive grid semi-definite programming for finding optimum designs. *Statistics and Computing*, 28, 441–460.

Fang, Z., Wiens, D. P., & Wu, Z. (2006). Locally D-optimal designs for multistage models and heteroscedastic polynomial regression models. *Journal of Statistical Planning and Inference*, 136, 4059–4070.

Fedorov, V. V. (1972). *Theory of Optimal Experiments*. Academic Press, New York.

Gao, L. L. & Zhou, J. (2017). D-optimal designs based on the second-order least squares estimator. *Statistical Papers*, 58, 77–94.

Guess, H., Crump, K., & Peto, R. (1977). Uncertainty estimates for low-dose-rate extrapolations of animal carcinogenicity data. *Cancer Research*, 37, 3475–3483.

Grant, M. C. & Boyd, S. P. (2013). *The CVX Users Guide, Release 2.0 (beta)*. CVX Research, Inc., Stanford University, Stanford.

Hoel, P. G. & Jennrich, R. I. (1979). Optimal designs for dose response experiments in cancer research. *Biometrika*, 66, 307–316.

Mandal, A., Wong, W. K., & Yu, Y. (2015). Algorithmic searches for optimal designs. In *Handbook of Design and Analysis of Experiments*, A. Dean, M. Morris, J. Stufken, & D. Bingham, editors Boca Raton, FL: Chapman & Hall/CRC Press; 755–784.

Papp, D. (2012). Optimal designs for rational function regression. *Journal of the American Statistical Association*, 107, 400–411.

Royle, J. A. (2002). Exchange algorithms for constructing large spatial designs. *Journal of Statistical Planning and Inference*, 100, 121–134.

Silvey, S. D. (1980). *Optimal Design*. Chapman and Hall, London.

Wong, W. K., Yin, Y., & Zhou, J. (2017). Using SeDuMi to find various optimal designs for regression models. *Statistical Papers*, 1–21, 10.1007/s00362-017-0887-7, (to appear in print).

Wong, W. K., Yin, Y., & Zhou, J. (2018). Optimal designs for multi-response nonlinear regression models with several factors via semi-definite programming. *Journal of Computational and Graphical Statistics*, 10.1080/10618600.2018.1476250, (to appear in print).

Yang, M., Biedermann, S., & Tang, E. (2013). On optimal designs for nonlinear models: a general and efficient algorithm. *Journal of the American Statistical Association*, 108, 1411–1420.

Yang, M., Zhang, B., & Huang, S. (2011). Optimal designs for generalized linear models with multiple design variables. *Statistica Sinica*, 21, 1415–1430.

Ye, J. J., Zhou, J., & Zhou, W. (2017). Computing A-optimal and E-optimal designs for regression models via semidefinite programming. *Communications in Statistics—Simulation and Computation*, 46, 2011–2024.

Yu, Y. (2011). D-optimal designs via a cocktail algorithm. *Statistics and Computing*, 21, 475–481.

Received 16 February 2018
Revised 07 December 2018
Accepted 23 December 2018