On optimal designs for nonlinear models: a general and efficient algorithm

Min Yang
University of Missouri
Department of Statistics
146 Middlebush Hall
Columbia, MO 65211-6100, USA
email: YangMi@Missouri.edu

Stefanie Biedermann
University of Southampton
School of Mathematics
Highfield
Southampton, SO17 1BJ, UK
email: S.Biedermann@southampton.ac.uk

Abstract

Deriving optimal designs for nonlinear models is challenging in general. Although some recent results allow us to focus on a simple subclass of designs for most problems, deriving a specific optimal design mainly depends on algorithmic approaches. There is need of a general and efficient algorithm which is more broadly applicable than the current state of the art methods. We present a new algorithm that can be used to find optimal designs with respect to a broad class of optimality criteria, when the model parameters or functions thereof are of interest, and for both locally optimal and multi-stage design strategies. We prove convergence to the desired optimal design, and show that the new algorithm outperforms the best available algorithm in various examples.

Keywords and Phrases: Convergence; Locally optimal design; Multi-stage design; Φ_p-optimality

1 Introduction

Designing experiments is an integral part of the scientific process, both for discovery and verification. Resources are almost always scarce, and judicious use of the limited resources is essential. Identifying efficient and optimal designs for data collection is therefore paramount. With the development of computation technology, nonlinear models have become more useful and popular. However, how to select good designs in practical studies is far behind the pace of application of nonlinear models.

A major complication in studying optimal designs for nonlinear models is that information matrices and thus optimal designs depend on the unknown model parameters, held in the vector $\theta = (\theta_1, \ldots, \theta_k)'$. A typical approach is to use a locally optimal design, which is based on a "best guess" of the unknown parameters (Chernoff, 1953). However, locally optimal designs may be poor if the selected values for $\theta_1, \ldots, \theta_k$ are far from the true values. A practical way to get a reliable "best guess" is to employ multi-stage designs (adaptive designs), which have gained a lot of popularity in practice. An initial experiment, typically a robust design, is conducted to get a better idea about the unknown parameters. The estimate for θ from these data is then used as the "best guess" on which the design for the next stage can be based. At the second stage, the question becomes how to add more design points, such that the combination of the existing design and the newly added design is optimal/efficient with respect to some optimality criterion. The observations from both the initial and the second stage design are subsequently used to obtain new estimates for the parameters. If a third stage design is needed, these will serve as the "best guess", and so on.

Recently, Yang and Stufken (2009), Yang (2010), and Dette and Melas (2011) convincingly demonstrated that unifying results for multiple models, multiple optimality criteria and multiple objectives can be obtained in the context of nonlinear models. They show that we can focus on a subclass of designs with a simple form, no matter what type of optimal designs we are looking for, including optimal multi-stage designs. While these results are big steps towards solving optimal design problems for nonlinear models in general, a research gap still exists. For specific design problems, except for some special cases, it is impossible to find an optimal design analytically, and we have to rely on a numerical solution. While we can focus on a subset of designs of a simple form, the numerical computation may still be problematic. For example, suppose we can focus on designs with at most five support points. Then we still have nine variables (support points plus their weights) to be determined. Obviously, a full grid search is not feasible in this situation, and efficient algorithms are needed.

There are a few algorithms available: W-algorithm (Wynn, 1970), V-algorithm (Fedorov, 1972), and Multiplicative algorithm (Silvey, Titterington and Torsney, 1978). While, in theory, these algorithms can be applied to general optimality problems, they have been criticized for slow convergence (Silvey, 1980). As a result, many different modifications of these algorithms have been proposed, for example, the Vertex exchange method (Böhning, 1986), which converge considerably faster. However, these modifications mainly focus on one specific design problem: D-optimality, when all parameters are of interest. Chernoff (1999) has some concerns about the concentration on D-optimality, and prefers optimality results under different criteria. The selection of an appropriate optimality criterion depends on the main objective of the experiment. For example, if there are some nuisance parameters in the model, we may want to choose a design which is optimal, with respect to some optimality criterion, for the parameters of interest only. In addition, and perhaps even more important, these algorithms cannot be used for deriving optimal multi-stage designs.

Recently, Yu (2011) proposed a new algorithm named "Cocktail algorithm", which leads to dramatically improved speed compared with the existing algorithms, sometimes by several orders of magnitude. The Cocktail algorithm addresses the speed issue nicely. However, it is still restricted to D-optimal designs for the full parameter vector θ, and may not be directly applied to find multi-stage designs. These are major obstacles to wider use of the optimal design approach by practitioners.

The purpose of this paper is to develop a general yet efficient algorithm which can address this gap in the literature. We propose a new algorithm which works not just for D-optimality, but also for a large class of optimality criteria; not just for the case where all parameters are of interest, but also for any subsets or (differentiable) functions of parameters; not just for locally optimal designs, but also for multi-stage designs.

Convergence is the foundation of an algorithm. We investigate the theoretical properties of the algorithm, and prove convergence in many practical situations. Silvey (1980) notes "What is important about an algorithm is not whether it converges, but whether it is effective in the sense that it guarantees arbitrary close approach to the optimum; and how fast this approach is". We show that the new algorithm is highly efficient for all different optimality problems. In fact, for those optimality problems to which the current algorithms can be applied, the new algorithm outperforms the best available algorithm by a large scale.

In addition, we investigate how to select a grid to substitute a continuous design space. It is common practice that we consider a design set with grid points spread equidistantly in each variable. The finer the grid, the better the design we obtain. But this also increases the computational burden, especially in higher dimensions. We derive a lower bound for the efficiency of a design which is optimal on a grid, relative to the corresponding optimal design on the continuous design space. This helps us to determine how fine the grid should be in order to avoid unnecessary computational effort.

This paper is organized as follows. In Section 2, we introduce the necessary notations. The main results including convergence properties, implemention of the algorithm, as well as efficiency considerations in continuous space are presented in Section 3. Applications to many commonly studied nonlinear models, and comparisons with the current state of the art algorithms are shown in Section 4. Section 5 provides a brief discussion, followed by an appendix containing the proofs.

2 Set up and Notations

Suppose we have a nonlinear regression model for which at each point $\mathbf{x}$ the experimenter observes a response Y. Here $\mathbf{x}$ could be a vector, and we assume that the Y's are independent and follow some distribution from the exponential family with mean $\eta(\mathbf{x}, \theta)$, where θ is the $(k \times 1)$-parameter vector. Typically, approximate designs are studied, i.e., instead of the exact sample sizes for each support point, design weights are used. Let $\mathcal{X}$ represent the set of all possible design points. For a numerical study, while the original design space $\mathcal{C}$ is continuous, we typically consider $\mathcal{X}$ to be a set of grid points spread equidistantly in each variable. For example, suppose the design space is $\mathcal{C} = [0, 1]$, and we consider

$\mathcal{X} = \{x_i = i/10000, i = 0, 1, \ldots, 10000\}$. We then have $N = 10001$ design points in our design set. We will later study the efficiency of the optimal design based on $\mathcal{X}$ in the continuous space for different values of N.

In the multi-stage design context, let ξ_0 denote the design we have already carried out, and n_0 be the corresponding sample size. Suppose we can take n samples at the next design stage, and we need to determine the next stage design $\xi = \{(\mathbf{x}_i, \omega_i), i = 1, \ldots, m\}$ where $\mathbf{x}_i \in \mathcal{X}$ with $\omega_i > 0$ and $\sum_{i=1}^m \omega_i = 1$. Note that if $n_0 = 0$, the multi-stage design is a locally optimal design. Denote the information matrix at a single point $\mathbf{x}$ as $I_{\mathbf{x}}$. The information matrix for design ξ can be written as $I_\xi = \sum_{i=1}^m \omega_i I_{\mathbf{x}_i}$. The information matrix for design ξ_0, I_{ξ_0}, can be defined in a similar fashion.

Now the question is how to select the design ξ for the next stage, such that the combined design, $\xi_0 + \xi$, is best with respect to some optimality criterion for the parameters of interest. The information matrix of the combined design $\xi_0 + \xi$ can be written as $I_{\xi_0+\xi} = n_0 I_{\xi_0} + n I_\xi$. (Note that, unlike I_{ξ_0} or I_ξ, the sample sizes are included in $I_{\xi_0+\xi}$, for the purpose of easy presentation.) Throughout this paper, we assume that I_{ξ_0} is non-singular and $n_0 > 0$ unless specified otherwise. Such assumptions are common in design for nonlinear models, especially for an algorithmic approach. In fact, we expect the information matrix of the initial design to be non-singular, since the purpose of an initial design is to obtain estimates for all parameters.

Let $g(\theta) = (g_1(\theta), \ldots, g_v(\theta))^T$, $1 \le v \le k$, be the (possibly vector-valued) differentiable function of the parameters, which is of interest. We can estimate $g(\theta)$ using the maximum likelihood estimator $g(\hat{\theta})$, where $\hat{\theta}$ is the maximum likelihood estimator of θ. The asymptotic variance-covariance matrix of $g(\hat{\theta})$ under design $\xi_0 + \xi$ can be written as

$$\Sigma_{\xi_0+\xi}(g) = \frac{\partial g(\theta)}{\partial \theta^T} I_{\xi_0+\xi}^{-1} \left(\frac{\partial g(\theta)}{\partial \theta^T} \right)^T .$$

The aim is to identify a design ξ, such that the variance-covariance matrix $\Sigma_{\xi_0+\xi}(g)$ is minimized under some optimality criterion.

There are a variety of optimality criteria. The commonly used ones are A-, D-, and E-optimality, which is to minimize $Tr(\Sigma_{\xi_0+\xi}(g))$, $|\Sigma_{\xi_0+\xi}(g)|$, and λ_{max}, respectively, where λ_{max} is the largest eigenvalue of $\Sigma_{\xi_0+\xi}(g)$. These optimality criteria are appealing because of their statistical meaning. For example, an A-optimal design minimizes the sum of the variances of the estimators, a D-optimal design minimizes the volume of a confidence ellipsoid of the estimators, and an E-optimal design protects against the worst case when we make inference. Kiefer (1974), in an effort to unify these criteria, defined the class of functions

$$\Phi_p\left(\Sigma_{\xi_0+\xi}(g)\right) = \left[\frac{1}{v} Tr\left(\Sigma_{\xi_0+\xi}(g)\right)^p\right]^{1/p}, \ 0 \le p < \infty.$$

The case $p=0$ is understood as the limit $\lim_{p\to 0}\Phi_p\left(\Sigma_{\xi_0+\xi}(g)\right) = |\Sigma_{\xi_0+\xi}(g)|^{1/v}$ (D-optimality); for $p=1$, we have A-optimality; $\lim_{p\to\infty}\Phi_p\left(\Sigma_{\xi_0+\xi}(g)\right) = \lambda_{max}$, and we obtain E-optimality for $p\to\infty$. Throughout this paper, we shall consider Φ_p-optimality. Due to technical reason, we restrict to p being nonnegative integer. This restriction has little impact on any practical optimality problem since it is rare to consider Φ_p-optimality for non-integer p. Note that minimizing $\Phi_p\left(\Sigma_{\xi_0+\xi}(g)\right)$ is equivalent to minimizing

$$\widetilde{\Phi}_p\left(\Sigma_{\xi_0+\xi}(g)\right) = \begin{cases} \log|\Sigma_{\xi_0+\xi}(g)|, & \text{if } p=0; \\ Tr\left(\Sigma_{\xi_0+\xi}(g)\right)^p, & \text{if } p>0. \end{cases}$$

3 New algorithm

3.1 Convergence and optimal weights

The new algorithm can be described as follows:

Start with a set of initial points $S^{(0)}$. Let $S^{(t)}$ denote the set of support points at the tth iteration.

$$\begin{aligned} &\xi_{S^{(t)}} \text{ denotes the design with support points } S^{(t)} \text{ with optimal weights,} \\ &S^{(t+1)} = S^{(t)} \bigcup \{\mathbf{x}_t^*\}, \text{ where } \mathbf{x}_t^* = \arg\max_{\mathbf{x}\in\mathcal{X}} d_p(\mathbf{x}, \xi_{S^{(t)}}). \end{aligned} \quad (1)$$

Here, $d_p(\mathbf{x},\xi)$, the sensitivity function, is defined as follows:

$$d_p(\mathbf{x},\xi) = \begin{cases} Tr\left((\Sigma_{\xi_0+\xi}(g))^{-1}A(g,\xi)\right), & \text{if } p=0; \\ \left(\frac{1}{v}\right)^{\frac{1}{p}}\left(Tr(\Sigma_{\xi_0+\xi}(g))^p\right)^{\frac{1}{p}-1} Tr\left((\Sigma_{\xi_0+\xi}(g))^{p-1}A(g,\xi)\right), & \text{if } p>0; \end{cases} \quad (2)$$

where

$$A(g,\xi) = n\left(\frac{\partial g(\theta)}{\partial\theta^T}\right)\left(I_{\xi_0+\xi}\right)^{-1}\left(I_{\mathbf{x}} - I_\xi\right)\left(I_{\xi_0+\xi}\right)^{-1}\left(\frac{\partial g(\theta)}{\partial\theta^T}\right)^T.$$

Theorem 1. *In a multi-stage design, suppose that $n_0 > 0$ and I_{ξ_0} is non-singular. For any set of initial points $S^{(0)}$, $\xi_{S^{(t)}}$ defined in (1), converge to an optimal design which minimizes $\Phi_p\left(\Sigma_{\xi_0+\xi}(g)\right)$, as $t \to \infty$.*

Theorem 1 demonstrates that, regardless of the choice of initial points, the iteration procedure will converge to a Φ_p-optimal design for any parameters of interest as long as $I_{\xi_0} > 0$ and $n_0 > 0$, which generally hold in multi-stage designs. The result also holds when $n_0 = 0$ (locally optimal design) if we have an additional condition.

Theorem 2. *Suppose that $n_0 = 0$ and $\frac{\partial g(\theta)}{\partial \theta^T}$ is a square matrix of full rank. Then, as $t \to \infty$, $\xi_{S^{(t)}}$ defined in (1) converge to an optimal design which minimizes $\Phi_p\left(\Sigma_{\xi_0+\xi}(g)\right)$, provided the initial set $S^{(0)}$ satisfies $I_{\xi_{S^{(0)}}} > 0$.*

If $g(\theta) = \theta$, then $\frac{\partial g(\theta)}{\partial \theta^T}$ is the $(k \times k)$-identity matrix. Theorem 2 can thus be applied to any Φ_p-optimal design for the full parameter vector θ. This includes the well-studied D-optimality problem for all parameters.

The critical step in the new algorithm is to find the optimal weights for given support points. Pukelsheim and Torsney (1991) give an explicit formula for the optimal weights, which - although presented in the context of linear models - can also be applied to nonlinear models. However, there are two limitations. First of all, it requires the linear regression vectors to be independent. As a result, the number of support points cannot be greater than the number of parameters. Secondly, the formula is for one-stage designs only, and it is unlikely that it can be extended to multi-stage designs, at least not directly. The first issue can be remedied in most cases. Yang and Stufken (2009), Yang (2010), and Dette and Melas (2011) have shown that, for a large variety of nonlinear models, we can focus on a subset of designs with the number of support points being less than or equal to the number of parameters. However, the second issue remains, as we are trying to develop a general framework for optimal multi-stage designs.

The multiplicative algorithm has a general iteration formula for deriving the optimal weights. Yu (2010) proved that this algorithm enjoys monotonic convergence for a broad class of optimality criteria. While the multiplicative algorithm is simple, easy to implement and monotonically convergent, the convergence is relatively slow. In addition, it is also limited to one-stage designs.

Can we derive the optimal weights more efficiently in multi-stage designs? To answer this question, we first investigate a property of optimal weights. Let $\omega = (\omega_1, \dots, \omega_{m-1})^T$. Denote $\Omega = \{\omega : \omega_i \geq 0,\ i = 1, \dots, m-1, \sum_{i=1}^{m-1} \omega_i \leq 1\}$.

Theorem 3. *In a multi-stage design, suppose that I_{ξ_0} is non-singular and $n_0 > 0$. For given θ and support points $(\mathbf{x}_1, \dots, \mathbf{x}_m)$ of ξ, $\widetilde{\Phi}_p\left(\Sigma_{\xi_0+\xi}(g)\right)$ is minimized at any critical point in Ω (i.e., the points where $\frac{\partial \widetilde{\Phi}_p\left(\Sigma_{\xi_0+\xi}(g)\right)}{\partial \omega} = 0$, the zero-vector), or at the boundary of Ω, i.e., $\omega_i = 0$ for some $1 \leq i \leq m$. In addition, the Hessian matrix of $\widetilde{\Phi}_p\left(\Sigma_{\xi_0+\xi}(g)\right)$, $\frac{\partial^2 \widetilde{\Phi}_p\left(\Sigma_{\xi_0+\xi}(g)\right)}{\partial \omega \partial \omega^T}$, is a nonnegative definite matrix.*

Note that when $p > 0$, we can obtain a more general result if we replace $\widetilde{\Phi}_p\left(\Sigma_{\xi_0+\xi}(g)\right)$ by $\Phi_p\left(\Sigma_{\xi_0+\xi}(g)\right)$ in Theorem 3. The proof can be derived directly utilizing the convexity of Φ_p. However, we stick to the above version of Theorem 3 since the corresponding Hessian matrix of $\Phi_p\left(\Sigma_{\xi_0+\xi}(g)\right)$ is more cumbersome to handle in computations. The proof of Theorem 3, see appendix, also gives us the gradient and the Hessian matrix of $\widetilde{\Phi}_p\left(\Sigma_{\xi_0+\xi}(g)\right)$, which are needed for numerical search.

Theorem 3 clearly gives us the necessary guidance to find optimal weights for the given points. We need to solve $m-1$ nonlinear equations. Unfortunately, in general there is no closed form solution. We have to rely on a numerical approach, such as Newton's iteration method. Newton's method is well-known for its quadratic convergence rate, which, loosely speaking, means that the number of significant digits doubles after each iteration (See Isaacson and Keller, 1966).

By Theorem 3, the Hessian matrix is nonnegative definite in Ω. This guarantees convergence given the starting point is sufficiently close to the critical point (Kaplan, 1999). Since Ω is a compact set, we can always find the critical points (given there exists a critical point within Ω) if we use sufficiently many different initial points.

If the numerical search leads to a minimum on the boundary, we should remove the design point with zero weight. This reduces the number of support points, and we can search for the optimal weights on that set. This process is repeated until we find weights that satisfy the constraints. Given that the number of support points is finite, we can search each subset of the given support points and eventually we can find the optimal weights.

One may be tempted to apply Theorem 3 directly to the whole design set $\mathcal{X}$. However, this is not feasible in general, unless the size of the design set is very small, say, around the same as the number of parameters. When the number of given support points is even moderate, the computation of the Hessian matrix is time consuming, and there will be too many different boundary situations to be considered.

3.2 Implementation of the algorithm

Theorems 1, 2, and 3 provide the theoretical foundation for the algorithm defined in (1). A step-by-step procedure for its implementation in programming is described in what follows:

(i) Let $S^{(0)}$ be a set of $k+1$ design points uniformly distributed in $\mathcal{X}$ (the initial weights are uniform);

(ii) Derive $\xi_{S^{(t)}}$ using Newton's method;

(iii) Derive $\mathbf{x}_t^* = \arg\max_{\mathbf{x}\in\mathcal{X}} d_p(\mathbf{x}, \xi_{S^{(t)}})$;

(iv) Select a small value $\varepsilon_0 > 0$. If $d_p(\mathbf{x}_t^*, \xi_{S^{(t)}}) \leq \varepsilon_0$, $\xi_{S^{(t)}}$ is the desired design;

(v) Otherwise, set $S^{(t+1)} = S^{(t)} \bigcup \{\mathbf{x}_t^*\}$ and repeat (ii) - (iv). The optimal weights from $\xi_{S^{(t)}}$ (zero weight for $\mathbf{x}_t^*$) serve as initial weights of $S^{(t+1)}$ in step (ii).

Here, Newton's method, for a given set of support points $S^{(t)}$ and the associated initial weights $\omega_0^{(t)}$, updates $\omega_j^{(t)}$, the weights after the jth iteration, as follows (starting with $\alpha = 1$).

(a) $\omega_j^{(t)} = \omega_{j-1}^{(t)} - \alpha \left(\frac{\partial^2 \widetilde{\Phi}_p(\Sigma_{\xi_0+\xi}(g))}{\partial\omega\omega^T}|_{\omega=\omega_{j-1}^{(t)}} \right)^{-1} \frac{\partial \widetilde{\Phi}_p(\Sigma_{\xi_0+\xi}(g))}{\partial\omega}|_{\omega=\omega_{j-1}^{(t)}}$;

(b) Check if there are non-positive components of $\omega_j^{(t)}$. If so, go to step (c2), otherwise proceed to (c1);

(c1) Check whether $||\frac{\partial \widetilde{\Phi}_p(\Sigma_{\xi_0+\xi}(g))}{\partial\omega}|_{\omega=\omega_j^{(t)}}||$ is less than a pre-specified $\tilde{\varepsilon} > 0$. If so, $\omega^{(t+1)}$ is the vector of optimal weights. Otherwise, start the next iteration;

(c2) Reduce α to $\alpha/2$. Repeat (a) and (b) until α reaches a pre-specified value, say 0.00001. Remove the support point with smallest weight. For the new set of support points as well as their weights, go to (a).

Concrete expressions for $\frac{\partial \widetilde{\Phi}_p\left(\Sigma_{\xi_0+\xi}(g)\right)}{\partial \omega}$ and $\frac{\partial^2 \widetilde{\Phi}_p\left(\Sigma_{\xi_0+\xi}(g)\right)}{\partial \omega \omega^T}$, respectively, can be found in the appendix (formulae (26) and (27) for $p=0$, and (28) and (29) for $p>0$).

In what follows, we briefly discuss the practical properties of this algorithmic procedure.

1) Computation time. The computation time of Newton's method cannot be judged by the number of iterations only. Each iteration includes the calculation of the second derivative and evaluation of the Hessian matrix. If the number of given support points in $S^{(t)}$ is large, the computation of the Hessian matrix can be time consuming. The algorithm adds one more point to the existing support in each iteration. However, for a large support at least one of the optimal weights will lie on the boundary, and the support point with zero weight will be removed, thus reducing the size of the support for the next iteration. We know from Caratheodory's theorem that any design is dominated by a design with $k(k+1)/2$ support points, and some recent theoretical results (Yang and Stufken, 2009, Yang, 2010, and Dette and Melas, 2011) show that for a large class of nonlinear models the support size can even be reduced to k. These results give some theoretical justification that, when the iteration progresses towards an optimal design, it is expected that the number of support points is close to k. This has been verified in our numerical studies in the next section.

2) Choice of ε_0. When $d_p(\mathbf{x}_t^*, \xi_{S^{(t)}})=0$, it means $\xi_{S^{(t)}}$ is an optimal design (see Theorem 4 below). In numerical computations, it is rare to achieve the bound. Typically we choose a small positive value, say ε_0, as the cut-off point, which depends on how efficient the derived design should be compared with the true optimal design from theory. From the proof of Theorem 1, we know that

$$\begin{aligned}
&\widetilde{\Phi}_0\left(\Sigma_{\xi_0+\xi_{S^{(t)}}}(g)\right)-\widetilde{\Phi}_0\left(\Sigma_{\xi_0+\xi^*}(g)\right) \leq d_0(\mathbf{x}_t^*, \xi_{S^{(t)}}),\\
&\Phi_p\left(\Sigma_{\xi_0+\xi_{S^{(t)}}}(g)\right)-\Phi_p\left(\Sigma_{\xi_0+\xi^*}(g)\right) \leq d_p(\mathbf{x}_t^*, \xi_{S^{(t)}}) \text{ for } p>0.
\end{aligned} \tag{3}$$

The inequalities in (3) give a lower bound for the efficiency of the derived design, which is $\exp(-\frac{\varepsilon_0}{v})$ for $p=0$ and $1-\frac{\varepsilon_0}{\Phi_p\left(\Sigma_{\xi_0+\xi_{S^{(t)}}}(g)\right)}$ for $p>0$. Here, efficiency of a design ξ is defined as $\frac{\Phi_p\left(\Sigma_{\xi_0+\xi^*}\right)}{\Phi_p\left(\Sigma_{\xi_0+\xi}\right)}$.

3) There is no guarantee that the above Newton iteration procedure always finds the optimal weights. We extend the well-known equivalence theorem for locally Φ_p-optimal designs to Φ_p-optimal multi-stage designs, and use it to check whether the answer obtained

from the algorithm indeed corresponds to an optimal design. In our experience, it virtually always does.

Theorem 4. *In a multi-stage design, suppose that $n_0 > 0$ and I_{ξ_0} is non-singular. Then a design ξ^* is Φ_p-optimal for $g(\theta)$ if and only if, for all $\mathbf{x} \in \mathcal{X}$,*

$$d_p(\mathbf{x}, \xi^*) \leq 0,$$

with equality if $\mathbf{x}$ is any support point of a Φ_p-optimal design. Here $d_p(\mathbf{x}, \xi^*)$ is the sensitivity function defined in (2).

The proof of Theorem 4 is omitted, since it is based on the same standard argument as the proof of the corresponding equivalence theorem for locally optimal designs, which can e.g. be found in Pukelsheim (2006).

4) E-optimality is equivalent to Φ_p-optimality when $p \to \infty$, but we cannot use $p = \infty$ in practice. When we implement the algorithm we may choose a large value of p. Let ξ_p^* be a Φ_p-optimal design, λ_{max} be the largest eigenvalue of $\Sigma_{\xi_0+\xi_p^*}(g)$, ξ_E^* be an E-optimal design, and λ_{max}^E be the largest eigenvalue of $\Sigma_{\xi_0+\xi_E^*}(g)$. Clearly, we have the following inequality,

$$\left(\frac{1}{v}\right)^{\frac{1}{p}} \lambda_{max} \leq \Phi_p\left(\Sigma_{\xi_0+\xi_p^*}(g)\right) \leq \Phi_p\left(\Sigma_{\xi_0+\xi_E^*}(g)\right) \leq \lambda_{max}^E.$$

Then the E-efficiency of ξ_p^* is bounded by

$$\text{eff}_E(\xi_p^*) = \frac{\lambda_{max}^E}{\lambda_{max}} \geq \left(\frac{1}{v}\right)^{\frac{1}{p}},$$

and we choose p such that $\left(\frac{1}{v}\right)^{\frac{1}{p}}$ is sufficiently close to 1.

3.3 Efficiency on continuous design spaces

As is common practice in numerical design search, when the design space $\mathcal{C}$ is continuous, we consider optimal designs on $\mathcal{X}$, a set of grid points spread equidistantly in each variable. The finer the grid the more confident one can be about the optimality of the design derived numerically. However, computation time will quickly increase with grid size, particularly in higher dimensions. For example, if the design space is three dimensional, then taking 100 points in each dimension results in 10^6 design points in $\mathcal{X}$. In order to find a good balance between design performance and computation time, we investigate the relationship between grid size and design efficiency. Let ξ_p^* be a Φ_p-optimal design on $\mathcal{X}$ and ξ_p^c be a Φ_p-optimal design on $\mathcal{C}$ (ξ_p^c is not available in general). Define $\text{eff}_p^c(\xi_p^*)$ to be the Φ_p-efficiency of the design ξ_p^* on $\mathcal{C}$, i.e.,

$$\text{eff}_p^c(\xi_p^*) = \frac{\Phi_p\left(\Sigma_{\xi_0+\xi_p^c}\right)}{\Phi_p\left(\Sigma_{\xi_0+\xi_p^*}\right)}.$$

The following theorem provides a lower bound for $\text{eff}_p^c(\xi_p^*)$.

Theorem 5. *Let $\mathcal{X}$ be a grid on the continuous design space $\mathcal{C} \in \mathcal{R}^r$, with grid points spread equidistantly in each variable with step size ε_j, $j = 1, \ldots, r$. Then $\textit{eff}_p^c(\xi_p^*)$ is bounded from below by*

$$\textit{eff}_p^c(\xi_p^*) \geq \begin{cases} \exp\left(1 - \frac{1}{2v}\max_{\mathbf{c}\in\mathcal{C}}\sum_{j=1}^r B_j(\mathbf{c})\varepsilon_j\right), & \textit{if } p = 0; \\ 1 - \frac{1}{2\Phi_p\left(\Sigma_{\xi_0+\xi_p^*}(g)\right)}\max_{\mathbf{c}\in\mathcal{C}}\sum_{j=1}^r B_j(\mathbf{c})\varepsilon_j, & \textit{if } p > 0. \end{cases} \tag{4}$$

Here,

$$B_j(\mathbf{c}) = \begin{cases} n|Tr\left(M\frac{\partial I_{\mathbf{c}}}{\partial c_j}\right)|, & \textit{if } p = 0; \\ n\left(\frac{1}{v}\right)^{\frac{1}{p}}\left(Tr(\Sigma_{\xi_0+\xi_p^*}(g))^p\right)^{\frac{1}{p}-1}|Tr\left(M\frac{\partial I_{\mathbf{c}}}{\partial c_j}\right)|, & \textit{if } p > 0; \end{cases} \tag{5}$$

where

$$M = \left(I_{\xi_0+\xi_p^*}\right)^{-1}\left(\frac{\partial g(\theta)}{\partial\theta^T}\right)^T\left(\Sigma_{\xi_0+\xi_p^*}(g)\right)^{p-1}\left(\frac{\partial g(\theta)}{\partial\theta^T}\right)\left(I_{\xi_0+\xi_p^*}\right)^{-1}.$$

Each term on the right hand side of (4) can be computed directly from programming except $\frac{\partial I_{\mathbf{c}}}{\partial c_i}$, which we may have to compute by hand. Here, c_i is the ith variable of $\mathbf{c}$. Computing $\max_{\mathbf{c}\in\mathcal{C}}\sum_{j=1}^r B_j(\mathbf{c})\varepsilon_j$ is challenging in general. One suggestion is to consider another grid set on $\mathcal{C}$, say $\mathcal{X}'$, with grid points spread equidistantly in each variable with step size ε_j'. For any $\mathbf{c} \in \mathcal{C}$, there exists a point, say $\mathbf{x}' \in \mathcal{X}'$, such that $|c_j - x_j'| \leq \varepsilon_j'/2$ for $j = 1, \ldots, r$. Here, x_j' is the jth variable of $\mathbf{x}'$. By the mean value theorem and the Cauchy-Schwarz inequality, we can show that

$$|Tr\left(M\frac{\partial I_{\mathbf{c}}}{\partial c_j}\right)| \leq |Tr\left(M\frac{\partial I_{\mathbf{x}'}}{\partial x_j'}\right)| + \frac{1}{2}|M|\max_{\mathbf{c}'\in\mathcal{C}}\sqrt{\sum_{l=1}^r|\frac{\partial^2 I_{\mathbf{c}'}}{\partial c_l'\partial c_j'}|^2}\sqrt{\sum_{l=1}^r(\varepsilon_l')^2}, \tag{6}$$

where $|.|$ is the L_2-norm. By (5) and (6), with some rearranging of terms, we have

$$\max_{\mathbf{c}\in\mathcal{C}}\sum_{j=1}^{r}B_j(\mathbf{c})\varepsilon_j \le \max_{\mathbf{x}'\in\mathcal{X}'}\sum_{j=1}^{r}B_j(\mathbf{x}')\varepsilon_j + \frac{1}{2}nD|M|\sum_{j=1}^{r}\left(\varepsilon_j \max_{\mathbf{c}'\in\mathcal{C}}\sqrt{\sum_{l=1}^{r}|\frac{\partial^2 I_{\mathbf{c}'}}{\partial c'_l\partial c'_j}|^2}\sqrt{\sum_{l=1}^{r}(\varepsilon'_l)^2}\right). \tag{7}$$

Here, $D = 1$ if $p = 0$ and $D = \left(\frac{1}{v}\right)^{\frac{1}{p}}\left(Tr(\Sigma_{\xi_0+\xi_p^*}(g))^p\right)^{\frac{1}{p}-1}$ if $p > 0$. All terms on the right hand side of (7) can be computed by SAS programming except for $\max_{\mathbf{c}'\in\mathcal{C}}\sqrt{\sum_{l=1}^{r}|\frac{\partial^2 I_{\mathbf{c}'}}{\partial c_l\partial c'_j}|^2}$, which requires computation by hand or by symbolic software, such as Maple or Mathematica. We shall illustrate this method through an example in Section 4.

4 Examples

The most important feature of an algorithm is speed. Wu (1978) notes that "Speed of approach, in the sense of computation time required is usually best dealt with empirically." In this section, we shall demonstrate, through several examples, that the algorithm is highly efficient. All coding was done in SAS IML, and computed at a Dell Laptop (2.2GHz and 8Gb RAM). The cut-off value for checking optimality was chosen to be $\varepsilon_0 = 10^{-6}$. All derived optimal designs have been verified through Theorem 4.

As far as we know, the existing algorithms mainly focus on D-optimal design, when all parameters are of interest. Yu (2010) shows that the Cocktail algorithm outperforms all existing algorithms by a large scale in this situation. We therefore compare the new algorithm only with the Cocktail algorithm. We also assess the computation time of the new algorithm for different optimality criteria, different sets or functions of parameters of interest, and for multi-stage designs - scenarios other algorithms may not be directly applied to. Note that for multi-stage designs, in practice, we need to estimate the parameters, then use the estimated parameters to select the design for the next stage. Since the aim here is to demonstrate the performance of the algorithm, we use the true parameters for illustration purposes.

Example 1. Consider the nonlinear model

$$Y \sim \theta_1 e^{-\theta_2 x} + \theta_3 e^{-\theta_4 x} + N(0,\sigma^2), \quad \theta = (\theta_1,\theta_2,\theta_3,\theta_4).$$

Let $(\theta_2, \theta_4) = (1, 2)$ and $\mathcal{X} = \{3i/N, i = 1, \ldots, N\}$. Yu (2010) finds D-optimal designs for θ (Table 1 of Yu, 2010). We code the Cocktail algorithm in SAS IML and compare its computation time with the new algorithm for different grid sizes N in Table 4.1. We can see that, while the Cocktail algorithm performs well, the new algorithm is about twice faster for moderate grid sizes, and even four times faster for finer grids.

Table 4.1: Computation time (seconds) for D-optimal designs for θ

	$N = 500$	$N = 1000$	$N = 5000$	$N = 10000$
Cocktail	0.32	0.46	2.54	5.16
New algorithm	0.14	0.21	0.99	1.26

The new algorithm is also highly efficient for different problems. We consider three different sets of parameters: θ, (θ_1, θ_3), and (θ_2, θ_4) and two different optimality criteria: D- and A-optimality. The computation time is less than 1 second for almost all cases, even if there are 10000 design points in $\mathcal{X}$; see Table 4.2. Note that, although Theorem 2 does not imply that the new algorithm converges when partial parameters are of interest, it seems the new algorithm can still be applied to such problems.

Table 4.2: Computation time (seconds) for different locally optimal designs

	D			A		
	θ	(θ_1, θ_3)	(θ_2, θ_4)	θ	(θ_1, θ_3)	(θ_2, θ_4)
$N = 500$	0.14	0.10	0.10	0.10	0.10	0.10
$N = 1000$	0.21	0.12	0.15	0.11	0.12	0.12
$N = 5000$	0.99	0.32	0.46	0.24	0.28	0.23
$N = 10000$	1.26	0.54	0.85	0.45	0.42	0.45

The new algorithm can also be applied to multi-stage designs. Suppose we have an initial design $\xi_0 = \{(0, 0.25), (1, 0.25), (2, 0.25), (3, 0.25)\}$ with $n_0 = 40$. The problem is how to allocate the next 80 subjects. Again, we consider three sets of parameters, θ, (θ_1, θ_3), and (θ_2, θ_4), and D- and A-optimality. We can see from Table 4.3 that the performance is similar to that for locally optimal designs.

Table 4.3: Computation time (seconds) for different multi-stage optimal designs

	D			A		
	θ	(θ_1, θ_3)	(θ_2, θ_4)	θ	(θ_1, θ_3)	(θ_2, θ_4)
$N = 500$	0.36	0.34	0.32	0.09	0.09	0.10
$N = 1000$	0.42	0.37	0.37	0.10	0.09	0.11
$N = 5000$	0.78	0.57	0.67	0.34	0.29	0.23
$N = 10000$	1.27	0.78	0.98	0.54	0.57	0.40

Example 2. Consider the linear model

$$Y \sim \theta_1 + \theta_2 x_1 + \theta_3 x_1^2 + \theta_4 x_2 + \theta_5 x_1 x_2 + N(0, \sigma^2), \quad \theta = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5), \tag{8}$$

and let $\mathcal{X} = \{(2i/s - 1, j/s), i = 1, \ldots, s, j = 1, \ldots, s\}$. Yu (2010) studies D-optimal design for θ (Table 4 of Yu, 2010).

Table 4.4 shows that the new algorithm is again faster than the Cocktail algorithm. As the grid size increases, this becomes more pronounced. For example, when $N = 200^2$ or $N = 500^2$, the new algorithm is about 5 times faster.

Table 4.4: Computation time (seconds) for D-optimal designs for θ

	$N = 20^2$	$N = 50^2$	$N = 100^2$	$N = 200^2$	$N = 500^2$
Cocktail	0.20	0.82	2.30	8.68	53.69
New algorithm	0.15	0.24	0.51	1.66	11.03

We also assessed the new algorithm on different problems; D- and A-optimality; all or just partial parameters of θ; locally optimal or multi-stage designs. The performance is similar, i.e., the computation time is less than 1 second for most cases, about 1.5 seconds for grid size $N = 200^2$, and about 10 seconds for grid size $N = 500^2$. The tables are not shown here for conciseness.

Suppose the continuous design space is $\mathcal{C} = [-1, 1] \times [0, 1]$. How does $\text{eff}_p^c(\xi_p^*)$, the Φ_p-efficiency of the design ξ_p^* on $\mathcal{C}$, change with N? Suppose we have an initial design $\xi_0 = \{[(0.2, -1), 0.25], [(0.5, 0), 0.25], [(0.8, 1), 0.25], [(0.5, 0.5), 0.25]\}$ with $n_0 = 40$, and that there are 120 further subjects to be allocated. We compute lower bounds for $\text{eff}_p^c(\xi_p^*)$, according to Theorem 5, for a variety of problems.

For Model (8), the information matrix of a single point $\mathbf{x} = (x_1, x_2)$, $I_{\mathbf{x}} = f(\mathbf{x})f(\mathbf{x})^T$, where $f(\mathbf{x}) = (1, x_1, x_1^2, x_2, x_1x_2)^T$. Hence $\frac{\partial I(\mathbf{x})}{\partial x_1} = f_1(\mathbf{x})f(\mathbf{x})^T + f(\mathbf{x})f_1(\mathbf{x})^T$, where $f_1(\mathbf{x}) = (0, 1, 2x_1, 0, x_2)^T$, and $\frac{\partial I(\mathbf{x})}{\partial x_2} = f_2(\mathbf{x})f(\mathbf{x})^T + f(\mathbf{x})f_2(\mathbf{x})^T$, where $f_2(\mathbf{x}) = (0, 0, 0, 1, x_1)^T$. With some algebra, we can show that

$$|\frac{\partial^2 I(\mathbf{x})}{\partial^2 x_1}|^2 + |\frac{\partial^2 I(\mathbf{x})}{\partial x_1 \partial x_2}|^2 \leq 350 \ \text{ and } \ |\frac{\partial^2 I(\mathbf{x})}{\partial x_1 \partial x_2}|^2 + |\frac{\partial^2 I(\mathbf{x})}{\partial^2 x_2}|^2 \leq 78. \tag{9}$$

For given s, ε_1 and ε_2 (defined in Theorem 5) are $\frac{2}{s}$ and $\frac{1}{s}$, respectively. Thus we have $\sqrt{\sum_{j=1}^{2} \varepsilon_j^2} = \sqrt{5}/s$. We take $\mathcal{X}'$, defined in (6), with grid points spread equidistantly in each variable with step size 2/3000 and 1/3000. We consider two different sets of parameters of interest, θ and $(\theta_2, \ldots, \theta_5)$, and three different optimality criteria, D-, A-, and Φ_2-optimality. Applying Theorem 5 and utilizing (9), we obtain the following table of lower bounds for $\text{eff}_p^c(\xi_p^*)$ for different grid sizes N.

Table 4.5: Lower bounds for $\text{eff}_p^c(\xi_p^*)$ for different grid sizes N

	θ			$(\theta_2, \ldots, \theta_5)$		
	D	A	Φ_2	D	A	Φ_2
$N = 20^2$	0.872	0.680	0.662	0.843	0.676	0.662
$N = 50^2$	0.947	0.872	0.865	0.934	0.870	0.865
$N = 100^2$	0.973	0.936	0.932	0.966	0.935	0.932
$N = 200^2$	0.986	0.968	0.966	0.983	0.968	0.966
$N = 500^2$	0.995	0.987	0.986	0.993	0.987	0.986

As N increases, the lower bounds for $\text{eff}_p^c(\xi_p^*)$ increase. For $N = 100^2$, the lower bounds are already relatively high (> 0.93). Note that the true efficiencies could be much higher than the lower bounds. For example, for $N = 20^2$, the A-optimal design for θ gives an optimality value of 0.203818 while the corresponding optimality value for $N = 500^2$ is 0.203797, which implies that the A-optimal design derived for $N = 20^2$ is at least 98.7% efficient. However, we can only find this out by deriving the optimal design for $N = 500^2$ and its lower bound for efficiency.

Example 3. We consider a multinomial model with three different categories, i.e. a response $\mathbf{Y} = (Y_1, Y_2, Y_3)^T$, with $Y_1 + Y_2 + Y_3 = 1$, at experimental condition $\mathbf{x}$ has a multinomial distribution with parameters $\pi_1(\mathbf{x}), \pi_2(\mathbf{x}), 1-\pi_1(\mathbf{x})-\pi_2(\mathbf{x})$, where

$$\pi_i(\mathbf{x}) \;=\; P(Y_i=1|\mathbf{x}) = \frac{e^{g(\mathbf{x})^T\theta_i}}{1+e^{g(\mathbf{x})^T\theta_1}+e^{g(\mathbf{x})^T\theta_2}}, \quad i=1,2. \tag{10}$$

The vectors $g(\mathbf{x})$ usually hold lower order monomials in the covariates $\mathbf{x}=(x_1,\ldots,x_r)^T$ and θ_1 and θ_2 are the corresponding coefficient vectors. The log-likelihood for a single observation and parameter vector $\theta=(\theta_1^T,\theta_2^T)^T \in I\!R^k$ is then

$$l(\theta;Y) = Y_1 \log\pi_1(\mathbf{x}) + Y_2\log\pi_2(\mathbf{x}) + (1-Y_1-Y_2)\log(1-\pi_1(\mathbf{x})-\pi_2(\mathbf{x})),$$

and we obtain the information matrix at a single point $\mathbf{x}$ as

$$I_{\mathbf{x}} \;=\; \left(\begin{array}{c|c} \pi_1(\mathbf{x})(1-\pi_1(\mathbf{x}))\,J(\mathbf{x}) & -\pi_1(\mathbf{x})\pi_2(\mathbf{x})\,J(\mathbf{x}) \\ \hline -\pi_1(\mathbf{x})\pi_2(\mathbf{x})\,J(\mathbf{x}) & \pi_2(\mathbf{x})(1-\pi_2(\mathbf{x}))\,J(\mathbf{x}) \end{array}\right),$$

where $J(\mathbf{x}) = g(\mathbf{x})g^T(\mathbf{x})$. Note that the information matrix $I_{\mathbf{x}}$ at a single point cannot be written in the form $I_{\mathbf{x}} = f(\mathbf{x})f(\mathbf{x})^T$ for any vector function $f(\mathbf{x})$. Consequently, the Cocktail algorithm cannot be applied to this example directly. The new algorithm, however, is not thus restricted. In what follows we consider linear predictors, i.e. $g(\mathbf{x}) = (1,\mathbf{x}^T)^T = (1,x_1,x_2,x_3)^T$ with $\theta_1=(\theta_{10},\theta_{11},\theta_{12},\theta_{13})$ and $\theta_2=(\theta_{20},\theta_{21},\theta_{22},\theta_{23})$.

Optimal designs for model (10) depend on the values of the unknown parameters (Zocchi and Atkinson, 1999). We assume that $\theta_1=(1,1,-1,2)$ and $\theta_2=(-1,2,1,-1)$, and let the design space $\mathcal{X}=\{(6i/s,6j/s,6k/s), i,j,l=0,1,\ldots,s\}$. The number of design points in $\mathcal{X}$ increases rapidly as s increases. We therefore employ the multi-stage search strategy described in Stufken and Yang (2011): start with a coarse grid that is made increasingly finer in later stages; at each stage identify the best design based on the current grid. For the next stage, a finer grid is restricted to neighborhoods of the best support points found at the current stage. The search continues until a specified accuracy for the design points is reached. The last step is to verify optimality through the equivalence theorem. Based on our experience, this strategy can further reduce the computation time, especially for high-dimensional design spaces. Note that we apply this search strategy for this example only.

We consider (i) two different sets of parameters, θ and $\theta'=(\theta_{11},\theta_{12},\theta_{13},\theta_{21},\theta_{22},\theta_{23})$; (ii) both D- and A-optimality; and (iii) locally optimal design and multi-stage design with

$\xi_0 = \{[(1,3,6),0.25],[(4,2,1),0.25],[(0,1,2),0.25],[(2,1,0),0.25]\}$ and $n_0 = 40$. Table 4.6 shows the computation times for different grid sizes.

Table 4.6: Computation time (seconds) for optimal designs

	Locally optimal designs				Multi-stage optimal designs			
	θ		θ'		θ		θ'	
	D	A	D	A	D	A	D	A
$N = 10^3$	0.32	0.32	0.26	0.39	0.29	0.31	0.18	0.23
$N = 20^3$	0.62	1.07	1.15	1.71	0.74	1.73	0.65	1.34
$N = 50^3$	8.14	17.81	17.94	24.52	12.85	16.98	11.29	19.57
$N = 100^3$	54.38	86.92	101.13	169.57	71.73	68.14	71.09	114.64
$N = 200^3$	524.1	653.0	664.4	814.3	531.8	738.7	718.2	853.8

Example 4. Dette, Melas, and Shpilev (2011) studied optimal designs for estimating the derivative of the expected response in nonlinear regression models. The desired optimal designs are determined numerically based on some recursive formulas they derived. They considered two examples to demonstrate their methods:

$$Y \sim \theta_1 e^{\theta_2 x} + \theta_3 e^{\theta_4 x} + N(0,\sigma^2) \tag{11}$$

and

$$Y \sim \frac{\theta_1}{x+\theta_2 x} + \frac{\theta_3}{x+\theta_4 x} + N(0,\sigma^2), \tag{12}$$

with $x \in [0,1]$. The function of parameters of interest, $g(\theta)$, is $\frac{\partial}{\partial x}\left(\theta_1 e^{\theta_2 x} + \theta_3 e^{\theta_4 x}\right)$ for Model (11) and $\frac{\partial}{\partial x}\left(\frac{\theta_1}{x+\theta_2 x} + \frac{\theta_3}{x+\theta_4 x}\right)$ for Model (12). Since $g(\theta)$ is a scalar, this is a c-optimal design problem. With $(\theta_1,\theta_2,\theta_3,\theta_4) = (1,0.5,1,1)$ and $x = 0$ for both models, the optimal designs provided by Dette, Melas, and Shpilev (2011) are
$\{(0,0.3509),(0.3011,0.4438),(0.7926,0.1491),(1,0.0562)\}$ for Model (11) and
$\{(0,0.3509),(0.0952,0.4419),(0.4707,0.1479),(1,0.0597)\}$ for Model (12).

We apply the new algorithm to this problem. Either A- or D-optimality for $g(\theta)$ will give us the desired optimal designs. We use D-optimality here, and consider the discrete design space $\mathcal{X} = \{i/10000, i = 0,1,\ldots,10000\}$. We obtain

$\{(0, 0.3508), (0.3011, 0.4438), (0.7926, 0.1491), (1, 0.0563)\}$ for Model (11) and $\{(0, 0.3504), (0.0952, 0.4415), (0.4705, 0.1480), (1, 0.0601)\}$ for Model (12). The optimal designs provided by the new algorithm are not exactly the same as those found by Dette, Melas, and Shpilev (2011), which may be due to floating errors during the numerical computation. Our designs actually give slightly smaller optimal values if we do not round up to four decimal places. The computation time for deriving these designs was 0.42 seconds for Model (11) and 0.34 seconds for Model (12). We also tested the algorithm under different scenarios, such as different parameter values, different optimality criteria, and locally optimal or multi-stage designs. The computation times were all less than 1 second.

5 Discussion

While the importance of optimal/efficient designs in scientific studies cannot be disputed, their application in practice is not well-established. The main reason is the lack of availablity of efficient designs, caused by the lack of a general and efficient algorithm. The existing algorithms mainly focus on a specific optimality problem: locally D-optimal design for all parameters. The new algorithm outperforms the best available algorithm for that specific problem. Moreover, the new algorithm can be applied to a much broader class of optimality problems: any set of differentiable functions of the parameters of interest; all Φ_p-optimality criteria with p being integer; locally optimal or multi-stage design. For all problems, the new algorithm performs efficiently; for most cases, we get instantaneous results. We believe this can greatly facilitate the application of optimal/efficient designs in practice.

The E-efficiency of a Φ_p-optimal design is bounded from below by $\left(\frac{1}{v}\right)^{\frac{1}{p}}$. We may need a large value of p to obtain a good E-efficiency. For example, if $v = 3$, we need $p = 11$ to have a lower bound of 0.905 for the E-efficiency. This may bring some computational difficulties. Based on our experience, the algorithm does not work well when $p > 6$. The main reason is that the elements of some matrices are huge, resulting in imprecise inverses of those matrices due to rounding errors. Hence extra care must be taken when p is relative large.

Theorems 1 and 2 do not guarantee the convergence in some situations, i.e., singular I_{ξ_0} (for multi-stage designs) or $\frac{\partial g(\theta)}{\partial \theta^T}$ not being a full rank square matrix (for locally optimal designs). However, we experienced convergence in virtually all different situations. It may be worth to study the theoretical properties for these cases. On the other hand, we feel the idea in this paper can be extended to Bayesian optimal design, where numerical approaches are even more important. How well will the idea work there? More research is certainly needed in this direction.

The coding of the new algorithm is more complicated than that of the existing algorithms. However, the main body of the code is the same for all models, with the only part requiring change is the form of the information matrix. The SAS IML codes for all examples in this paper can be downloaded from www.missouri.edu/∼yangmi. These codes can be easily modified for different optimality problems.

6 Appendix

Proof of Theorem 1. We only give the proof for $p > 0$. For $p = 0$, the proof is exactly the same with $\Phi_p\left(\Sigma_{\xi_0+\xi}(g)\right)$ replaced by $\log|\Sigma_{\xi_0+\xi}(g)|$.

Let Ξ be the set of all ξ. Denote ξ^* as an optimal design which minimizes $\Phi_p\left(\Sigma_{\xi_0+\xi}(g)\right)$. Since I_{ξ_0} is nonsingular and $n_0 > 0$, for any $\xi \in \Xi$, we have

$$\Phi_p\left(\Sigma_{\xi_0+\xi^*}(g)\right) \leq \Phi_p\left(\Sigma_{\xi_0+\xi}(g)\right) \leq \Phi_p\left(\Sigma_{\xi_0}(g)\right), \tag{13}$$

where $\Sigma_{\xi_0}(g) = \frac{\partial g(\theta)}{\partial \theta^T}(n_0 I_{\xi_0})^{-1}\{\frac{\partial g(\theta)}{\partial \theta^T}\}^T$. In addition, $I_{\xi_0+\xi}$ is nonsingular regardless of ξ. Thus $\Phi_p\left(\Sigma_{\xi_0+(1-\alpha)\xi_1+\alpha\xi_2}(g)\right)$ is infinitely differentiable with respect to α. Combining this fact with (13), there exists $K < \infty$, such that

$$\sup\{\frac{\partial^2 \Phi_p\left(\Sigma_{\xi_0+(1-\alpha)\xi_1+\alpha\xi_2}(g)\right)}{\partial \alpha^2} : \xi_1 \in \Xi, \xi_2 \in \Xi, \alpha \in [0,1]\} = K. \tag{14}$$

The convergence of $\Phi_p\left(\Sigma_{\xi_0+\xi_{S(t)}}(g)\right)$ is obvious since it is a decreasing nonnegative function. We shall show that

$$\lim_{t\to\infty} \Phi_p\left(\Sigma_{\xi_0+\xi_{S(t)}}(g)\right) = \Phi_p\left(\Sigma_{\xi_0+\xi^*}(g)\right). \tag{15}$$

If (15) does not hold, by the monotonicity of $\Phi_p\left(\Sigma_{\xi_0+\xi_{S(t)}}(g)\right)$, there exists $\delta > 0$, such that

$$\Phi_p\left(\Sigma_{\xi_0+\xi_{S(t)}}(g)\right) - \Phi_p\left(\Sigma_{\xi_0+\xi^*}(g)\right) > \delta$$

for all t. Utilizing the convexity of Φ_p, for any $0 \leq \varepsilon \leq 1$, we have

$$\Phi_p\left(\Sigma_{\xi_0+(1-\varepsilon)\xi_{S^{(t)}}+\varepsilon\xi^*}(g)\right) \leq (1-\varepsilon)\Phi_p\left(\Sigma_{\xi_0+\xi_{S^{(t)}}}(g)\right) + \varepsilon\Phi_p\left(\Sigma_{\xi_0+\xi^*}(g)\right),$$

which implies that

$$\lim_{\varepsilon\downarrow 0}\frac{1}{\varepsilon}\left(\Phi_p\left(\Sigma_{\xi_0+(1-\varepsilon)\xi_{S^{(t)}}+\varepsilon\xi^*}(g)\right) - \Phi_p\left(\Sigma_{\xi_0+\xi_{S^{(t)}}}(g)\right)\right) \leq \Phi_p\left(\Sigma_{\xi_0+\xi^*}(g)\right) - \Phi_p\left(\Sigma_{\xi_0+\xi_{S^{(t)}}}(g)\right). \tag{16}$$

The left hand side of (16) is the first derivative of $\Phi_p\left(\Sigma_{\xi_0+(1-\alpha)\xi_{S^{(t)}}+\alpha\xi^*}(g)\right)$ with respect to α when $\alpha = 0$. By some standard matrix differentiation approach, utilizing (16), (2), and the definition of $\mathbf{x}_t^*$ in Theorem 1, for all t, we have

$$d_p(\mathbf{x}_t^*, \xi_{S^{(t)}}) \geq \delta. \tag{17}$$

Consider a differently updated design $\xi_{S^{(t+1)}}(\alpha) = (1-\alpha)\xi_{S^{(t)}} \bigcup (x_t^*, \alpha)$, where $0 \leq \alpha \leq 1$. By the definition of $S^{(t+1)}$ in Theorem 1, for any $\alpha \in [0,1]$,

$$\Phi_p\left(\Sigma_{\xi_0+\xi_{S^{(t+1)}}}(g)\right) \leq \Phi_p\left(\Sigma_{\xi_0+\xi_{S^{(t+1)}}(\alpha)}(g)\right). \tag{18}$$

Expanding into a Taylor series in α, and applying (14) and (17), we can show that

$$\begin{aligned}\Phi_p\left(\Sigma_{\xi_0+\xi_{S^{(t+1)}}(\alpha)}(g)\right) =& \Phi_p\left(\Sigma_{\xi_0+\xi_{S^{(t)}}}(g)\right) - d_p(x_t^*, \xi_{S^{(t)}})\alpha \\ &+ \frac{1}{2}\alpha^2 \frac{\partial^2 \Phi_p\left(\Sigma_{\xi_0+(1-\alpha)\xi_1+\alpha\xi_2}\right)}{\partial\alpha^2}|_{\alpha=\alpha'} \\ \leq& \Phi_p\left(\Sigma_{\xi_0+\xi_{S^{(t)}}}(g)\right) - \delta\alpha + \frac{1}{2}K\alpha^2,\end{aligned} \tag{19}$$

where $\alpha' \in [0, \alpha]$. If $K > \delta$, let $\alpha = \frac{\delta}{K}$. By (19), we have

$$\Phi_p\left(\Sigma_{\xi_0+\xi_{S^{(t+1)}}(\frac{\delta}{K})}(g)\right) - \Phi_p\left(\Sigma_{\xi_0+\xi_{S^{(t)}}}(g)\right) \leq -\frac{\delta^2}{2K}. \tag{20}$$

By (18) and (20), we have, for all $t \geq 0$,

$$\Phi_p\left(\Sigma_{\xi_0+\xi_{S^{(t+1)}}}(g)\right) - \Phi_p\left(\Sigma_{\xi_0+\xi_{S^{(t)}}}(g)\right) \leq -\frac{\delta^2}{2K}. \tag{21}$$

Inequality (21) implies that $\lim_{t\to\infty}\Phi_p\left(\Sigma_{\xi_0+\xi_{S^{(t)}}}(g)\right) = -\infty$, which contradicts the fact that $\Phi_p\left(\Sigma_{\xi_0+\xi_{S^{(t)}}}(g)\right)$ is a nonnegative function. Similar arguments can be applied to the case when $K \leq \delta$, in which we let $\alpha = 1$. $\square$

Proof of Theorem 2. For the same reason as in the proof of Theorem 1, we only give the proof for $p > 0$.

When $n_0 = 0$, there is no initial design ξ_0, i.e., $\Sigma_{\xi_0+\xi}(g) = \Sigma_\xi(g)$, where

$$\Sigma_\xi(g) = \frac{1}{n}\frac{\partial g(\theta)}{\partial \theta^T} I_\xi^{-1}\{\frac{\partial g(\theta)}{\partial \theta^T}\}^T. \tag{22}$$

Define $\Xi_1 = \{\xi : \Phi_p(\Sigma_\xi(g)) \le \Phi_p\left(2\Sigma_{\xi_{S^{(0)}}}(g)\right)\}$. Clearly $\xi_{S^{(t)}} \in \Xi_1$ since $\Phi_p\left(\Sigma_{\xi_{S^{(t)}}}(g)\right)$ is a decreasing nonnegative function. Consider a differently updated design $\xi_{S^{(t+1)}}(\alpha) = (1-\alpha)\xi_{S^{(t)}} \bigcup(\mathbf{x}_t^*, \alpha)$. For any $\alpha \in [0, \frac{1}{2}]$, we have

$$\Phi_p\left(\Sigma_{\xi_{S^{(t+1)}}(\alpha)}(g)\right) \le \Phi_p\left(2\Sigma_{\xi_{S^{(t)}}}(g)\right). \tag{23}$$

Inequality (23) implies that $\xi_{S^{(t+1)}}(\alpha) \in \Xi_1$ for any $\alpha \in [0, \frac{1}{2}]$.

Since $\frac{\partial g(\theta)}{\partial \theta^T}$ is a full rank square matrix, I_ξ is nonsingular for any $\xi \in \Xi_1$. Thus the function $\Phi_p\left(\Sigma_{(1-\alpha)\xi_1+\alpha\xi_2}(g)\right)$ is infinitely differentiable with respect to α for any $\alpha \in [0, \frac{1}{2}]$. Combining this fact with (22), there exists $M_1 < \infty$, such that

$$\sup\{\frac{\partial^2 \Phi_p\left(\Sigma_{(1-\alpha)\xi_1+\alpha\xi_2}(g)\right)}{\partial \alpha^2} : \xi_1 \in \Xi_1, \xi_2 \in \Xi, \alpha \in [0, \frac{1}{2}]\} = M_1.$$

The rest of the proof is the same as that of Theorem 1 with a minor but obvious modification.

□

Proof of Theorem 3. By the standard theory of multivariate convex functions (cf Kaplan, 1999, Section 1.9), it is sufficient to show that the Hessian of $\widetilde{\Phi}_p(\Sigma_{\xi_0+\xi}(g))$, $\frac{\partial^2 \widetilde{\Phi}_p\left(\Sigma_{\xi_0+\xi}(g)\right)}{\partial \omega \partial \omega^T}$, is a nonnegative definite matrix. Stufken and Yang (2011) prove this for $p = 0, 1$, for one-stage designs. Here we extend their results to nonnegative integers p, for multi-stage designs.

For simplification, we rewrite $\Sigma_{\xi_0+\xi}(g)$ as Σ and $I_{\xi_0+\xi}$ as I. For $i = 1, \ldots, m-1$, define $I^i = n(I_{\mathbf{x}_i} - I_{\mathbf{x}_m})$. Applying matrix differentiation, we have

$$\frac{\partial \Sigma}{\partial \omega_i} = -\frac{\partial g(\theta)}{\partial \theta^T} I^{-1} I^i I^{-1}\{\frac{\partial g(\theta)}{\partial \theta^T}\}^T, \quad i = 1, \ldots, m-1, \tag{24}$$

$$\frac{\partial^2 \Sigma}{\partial \omega_i \omega_j} = \frac{\partial g(\theta)}{\partial \theta^T}\left(I^{-1}I^jI^{-1}I^iI^{-1} + I^{-1}I^iI^{-1}I^jI^{-1}\right)\{\frac{\partial g(\theta)}{\partial \theta^T}\}^T, \quad i, j = 1, \ldots, m-1. \tag{25}$$

Case (i): $p = 0$. Applying matrix differentiation, we obtain

$$\frac{\partial \log|\Sigma|}{\partial \omega_i} = Tr\left(\Sigma^{-1}\frac{\partial \Sigma}{\partial \omega_i}\right), \quad i = 1, \ldots, m-1, \tag{26}$$

$$\frac{\partial^2 \log|\Sigma|}{\partial\omega_i\omega_j} = Tr\left(\Sigma^{-1}\frac{\partial^2\Sigma}{\partial\omega_i\omega_j} - \Sigma^{-1}\frac{\partial\Sigma}{\partial\omega_j}\Sigma^{-1}\frac{\partial\Sigma}{\partial\omega_i}\right), \quad i,j = 1,\ldots,m-1. \tag{27}$$

Utilizing (24) and (25), with some matrix algebra, we can show that

$$\begin{aligned}\frac{\partial^2 \log|\Sigma|}{\partial\omega_i\omega_j} =& Tr\left(\Sigma^{-\frac{1}{2}}\frac{\partial g(\theta)}{\partial\theta^T}I^{-1}I^jI^{-\frac{1}{2}}I^{-\frac{1}{2}}I^iI^{-1}\{\frac{\partial g(\theta)}{\partial\theta^T}\}^T\Sigma^{-\frac{1}{2}}\right)\\ &+ Tr\left(\Sigma^{-\frac{1}{2}}\frac{\partial g(\theta)}{\partial\theta^T}I^{-1}I^jI^{-\frac{1}{2}}P^\perp\left[I^{-\frac{1}{2}}\{\frac{\partial g(\theta)}{\partial\theta^T}\}^T\right]I^{-\frac{1}{2}}I^iI^{-1}\{\frac{\partial g(\theta)}{\partial\theta^T}\}^T\Sigma^{-\frac{1}{2}}\right),\end{aligned}$$

where $P^\perp(X) = I - X(X^TX)^-X^T$ denotes the orthogonal projection matrix onto the orthogonal complement of the column space of X. Thus the Hessian matrix of $\log|\Sigma|$ is nonnegative definite by Proposition 1 of Stufken and Yang (2011).

Case (ii): $p > 0$. Applying matrix differentiation, we have

$$\frac{\partial Tr\,(\Sigma)^p}{\partial\omega_i} = pTr\left(\Sigma^{p-1}\frac{\partial\Sigma}{\partial\omega_i}\right), \quad i = 1,\ldots,m-1, \tag{28}$$

$$\frac{\partial^2 Tr\,(\Sigma)^p}{\partial\omega_i\omega_j} = pTr\left(\Sigma^{p-1}\frac{\partial^2\Sigma}{\partial\omega_i\omega_j} + \sum_{l=0}^{p-2}\Sigma^l\frac{\partial\Sigma}{\partial\omega_j}\Sigma^{p-2-l}\frac{\partial\Sigma}{\partial\omega_i}\right), \quad i,j = 1,\ldots,m-1. \tag{29}$$

Note that when $p = 1$, the second term on the right hand side of (29) vanishes.

Utilizing (25), with some algebra, it can be shown that

$$Tr\left(\Sigma^{p-1}\frac{\partial^2\Sigma}{\partial\omega_i\omega_j}\right) = 2Tr\left(\Sigma^{\frac{p-1}{2}}\frac{\partial g(\theta)}{\partial\theta^T}I^{-1}I^jI^{-\frac{1}{2}}I^{-\frac{1}{2}}I^iI^{-1}\{\frac{\partial g(\theta)}{\partial\theta^T}\}^T\Sigma^{\frac{p-1}{2}}\right) \tag{30}$$

and

$$Tr\left(\Sigma^l\frac{\partial\Sigma}{\partial\omega_j}\Sigma^{p-2-l}\frac{\partial\Sigma}{\partial\omega_i}\right) = Tr\left(\Sigma^{\frac{l}{2}}\frac{\partial\Sigma}{\partial\omega_j}\Sigma^{\frac{p-2-l}{2}}\Sigma^{\frac{p-2-l}{2}}\frac{\partial\Sigma}{\partial\omega_i}\Sigma^{\frac{l}{2}}\right). \tag{31}$$

By (29), (30), and (31), applying Proposition 1 of Stufken and Yang (2011), we conclude that the Hessian matrix of $Tr(\Sigma)^p$ is nonnegative definite. □

Proof of Theorem 5. Let ξ_p^c be a Φ_p-optimal design on $\mathcal{C}$. By the same argument as in the proof of Theorem 1, we have

$$\Phi_p\left(\Sigma_{\xi_0+\xi_p^*}(g)\right) - \Phi_p\left(\Sigma_{\xi_0+\xi_p^c}(g)\right) \le \max_{\mathbf{c}\in\mathcal{C}} d_p(\mathbf{c}, \xi_p^*).$$

By the construction of $\mathcal{X}$, for any $\mathbf{c} \in \mathcal{C}$, there exists $\mathbf{x}^c \in \mathcal{X}$, such that $|c_j - x_j^c| \le \varepsilon_j/2$ for $j = 1,\ldots,r$. Here, c_j and x_j^c are the jth variable of $\mathbf{c}$ and $\mathbf{x}^c$, respectively. From the mean value theorem, there exists a scalar $\alpha \in (0,1)$ such that

$$\begin{aligned} d_p(\mathbf{c}, \xi_p^*) &= d_p(\mathbf{x}^c, \xi_p^*) + \nabla d_p((1-\alpha)\mathbf{c} + \alpha \mathbf{x}^c, \xi_p^*) \cdot (\mathbf{c} - \mathbf{x}^c) \\ &\leq \sum_{j=1}^{r} |\nabla_j d_p((1-\alpha)\mathbf{c} + \alpha \mathbf{x}^c, \xi_p^*)| \varepsilon_j / 2. \end{aligned} \tag{32}$$

where $\cdot$ denotes the Euclidean inner product, ∇ denotes the gradient and ∇_j denotes its jth element. The inequality in (32) follows from the fact that $\max_{x \in \mathcal{X}} d_p(\mathbf{x}, \xi_p^*) = 0$. From the definition of d_p in (2), it is straightforward to show that $|\nabla_j d_p((1-\alpha)\mathbf{c} + \alpha \mathbf{x}^c, \xi_p^*)| = B_j((1-\alpha)\mathbf{c} + \alpha \mathbf{x}^c)$, where B_j is defined in (5). From the definition of efficiency, the conclusion follows. $\square$

References

[1] Böhning, D. (1986), "A vertex-exchange-method in *D*-optimal design theory", *Metrika*, 33, 337-347.

[2] Chernoff, H. (1953). "Locally optimal designs for estimating parameters", *Annals of Mathematical Statistics*, 24, 586-602.

[3] Chernoff, H. (1999), "Gustav Elfving's Impact on Experimental Design", *Statistical Science*, 14, 201-205.

[4] Dette, H. and Melas, V. B. (2011), "A note on the de la Garza phenomenon for locally optimal designs", *The Annals of Statistics*, 39, 1266-1281.

[5] Dette, H., Melas, V. B., and Shpilev, P. (2011), "Optimal designs for estimating the derivative in nonlinear regression", *Statistica Sinica*, 21, 1557-1570.

[6] Fedorov, V.V. (1972). *Theory of optimal experiments (transl and ed by Studden WJ, Klimko EM)*, New York: Academic.

[7] Isaacson, E. and Keller, H. B. (1966). *Analysis of numerical methods*, John Wiley & Sons Inc.

[8] Kiefer, J. (1974), "General equivalence theory for optimum designs (approximate theory) extremum problems", *The Annals of Statistics*, 2, 849-879.

[9] Kaplan, W. (1999). *Maxima and minima with applications*, John Wiley & Sons Inc., New York.

[10] Pukelsheim, F. (2006). *Optimal design of experiments*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA.

[11] Pukelsheim, F., and Torsney, B. (1991), "Optimal weights for experimental designs on linearly independent support points", *The Annals of Statistics*, 19, 1614-1625.

[12] Silvey, S. D. (1980). *Optimal Design: An Introduction to the Theory for Parameter Estimation*. Chapman & Hall, London.

[13] Silvey, S.D., Titterington, D.M., and Torsney, B. (1978), "An algorithm for optimal designs on a finite design space", *Communications in Statistics - Theory and Methods*, 14, 1379-1389.

[14] Stufken, J. and Yang, M. (2011), "On locally optimal designs for generalized linear models with group effects", *Statistica Sinica*, preprint, doi:10.5705/ss.2010.261.

[15] Wu, C. F. J. (1978), "Some iterative procedures for generating nonsingular optimal designs", *Communications in Statistics - Theory and Methods*, 14, 1399-1412.

[16] Wynn, H. P. (1970), "The sequential generation of D-optimal experimental designs", *The Annals of Mathematical Statistics*, 41, 1655-1664.

[17] Yang, M. (2010). "On the de la Garza Phenomenon", *The Annals of Statistics*, 38, 2499-2524.

[18] Yang, M. and Stufken, J. (2009), "Support points of locally optimal designs for nonlinear models with two parameters", *The Annals of Statistics*, 37, 518-541.

[19] Yu, Y. (2010), "Monotonic convergence of a general algorithm for computing optimal designs", *The Annals of Statistics*, 38, 1593-1606.

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

[21] Zocchi, S.S. and Atkinson, A.C. (1999), "Optimum Experimental Designs for Multinomial Logistic Models", *Biometrics*, 55, 437-444.