Title
Using Competitive Swarm Optimizer with Mutated Agents to Find Optimal Experimental Designs

Author
Zhang, Zizhao

Publication Date
2020

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

Using Competitive Swarm Optimizer with Mutated Agents to Find Optimal Experimental Designs

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Biostatistics

by

Zizhao Zhang

2020

ABSTRACT OF THE DISSERTATION

Using Competitive Swarm Optimizer with Mutated Agents to Find Optimal Experimental Designs

by

Zizhao Zhang
Doctor of Philosophy in Biostatistics
University of California, Los Angeles, 2020
Professor Weng Kee Wong, Chair

Implementing optimal design can provide the most accurate statistical inference with minimal cost. However, optimal designs for high-dimensional models or complicated nonlinear models can be hard to find. I propose a novel swarm algorithm, called competitive swarm optimizer with mutated agents (CSO-MA), to search for optimal designs for high-dimensional and complicated nonlinear models that are useful for biomedical studies. They include logistic models, Poisson-type models with multiple interacting covariates and some factors may have correlated random effects. I first show the proposed algorithm outperforms several state-of-the-art algorithms using benchmark functions commonly used in the engineering literature. I then show it can either perform as efficiently as some current algorithms used in statistics for finding optimal designs or outperform several of its competitors. Additionally, I find some of the claimed optimal designs in the literature are not optimal by showing CSO-MA-generated designs have higher statistical efficiency. Since the bulk of design work in the literature concerns low-dimensional models, my work has the potential to break new ground, especially in the era of big data, where, increasingly, it is more realistic to use more complex models to reflect reality.

The proposed algorithm is a general-purpose optimization algorithm, so it is flexible and can find exact and approximate designs, with and without constraints. In particular, it can efficiently search for different types of optimal designs, including Bayesian optimal

designs, which are especially challenging to find when there are multiple factors and there are multi-dimensional integrals involved in the optimization problem. The results from my research will provide new, more realistic and better quality statistical experimental designs for biomedical researchers at a minimal cost.

The dissertation of Zizhao Zhang is approved.

Thomas R. Belin

Hongquan Xu

Hua Zhou

Weng Kee Wong, Committee Chair

University of California, Los Angeles

2020

To my mother, Xia Wu, and my father, Yue Zhang, who love me unconditionally and encourage me to be a better man;

to Lu Yu, the girl I loved;

to all my friends and family, particularly to Xi Chen and Qinghua Wang, who helped me through countless stress, comforted me when I was close to giving up and cared for me when I felt unprecedented loneliness, especially in the past few months.

Without you, I would never be who I am today.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, I would like to express my sincere gratitude to my advisor, Dr. Weng Kee Wong. Encouraged by his understanding, patience, enthusiasm and wisdom, I have reached farther than I thought I could go. Dr. Wong gave me guidance on my academic work and taught me how to complete every task with efficiency, effectiveness and responsibility. I am also profoundly grateful to the other three professors in my dissertation committee, Dr. Thomas R. Belin, Dr. Hongquan Xu and Dr. Hua Zhou, whose insightful suggestions were essential to improve my research. In addition, I want to appreciate Dr. Kay Chen Tan, who provided valuable instructions on improving our Memetic Computing paper; Dr. Xin Liu and Dr. Rongxian Yue, who led the project of constructing G-optimal designs for hierarchical linear models. At last, I present my acknowledgments to the Biostatistics department of UCLA because they offered me the great opportunity to have my Ph.D. career here and financially supported me.

VITA

2010–2014 B.S. (Mathematics), Shandong University, China.

2014–2016 M.S. (Biostatistics), University of California, Los Angeles, California.

2016–present Teaching Assistant and Graduate Student Researcher, Department of Biostatistics, University of California, Los Angeles, California.

PUBLICATIONS

Shi, Y., Zhang, Z., and Wong, W. K. (2019). Particle swarm based algorithms for finding locally and Bayesian D-optimal designs. *Journal of Statistical Distributions and Applications*, 6(1), 3.

Zhang, Z., Wong, W. K., and Tan, K. C. (2020). Competitive swarm optimizer with mutated agents for finding optimal designs for nonlinear regression models with multiple interacting factors. *Memetic Computing*, 12(3), 219-233.

Liu, X., Yue, R., Zhang, Z., and Wong, W. K. (2020). G-optimal designs for hierarchical linear models: an equivalence theorem and a nature-inspired metaheuristic algorithm. *Journal of Soft Computing*, under review.

Zhang, Z. and Wong, W. K. (2020). Constructing optimal designs for nonlinear mixed-effects models using competitive swarm optimizer with mutated agents. *Biometrics*, under review.

Zhang, Z., Wells, L. K., Huang, S. Z., Hansen, L. P., Roof, E., Holland, A., Port, A. M., Gur, R. C., and Bearden, C. E. (2020). Neurobehavioral dimensions of Prader Willi

Syndrome: Relationships between sleep disturbance and psychotic experiences. *Journal of Neurodevelopmental Disorders*, under review.

Zhang, Z. and Wong, W. K. (2020). A MATLAB package "CSOMA" for optimization and statistical computation. *Journal of Statistical Software*, under preparation.

PREAMBLE

Motivation and Research Scope

This preamble presents the motivation of my dissertation work and describes the scope of my research and the organization of this dissertation.

Motivation

There are a number of factors that motivate the research in my dissertation. First, optimal design ideas continue to gain popularity in many disciplines, especially in the biomedical sciences. Using an optimal design instead of a non-optimal design can greatly enhance the quality of the statistical inference and save costs at the same time. For example, uniform designs that take an equal number of observations at equal intervals across the dose range in a dose-response study may be intuitively appealing, but indiscriminate use of such a design can be very inefficient under some criteria or models. In some research areas, such as in toxicology, there is an increasing mandate to use a minimal number of animals in the laboratory. Practitioners who adopted optimal design ideas in their work can be rewarded as can be seen from the following direct remarks by Verotta et al. (1988), who applied a D-optimal design to estimate binding saturation curves of an enkephalin analog in rat brain:

"The results presented in this paper show that a D-optimal design can be used in binding experiments to estimate binding parameters accurately with a much smaller number of experimental points than needed by traditional designs. D-optimal designs need only a small number of support points, generally equal to the number of unknown parameters (three in our model) for a correct estimate of parameter values. Traditional saturation curves for binding experiments call for many more points requiring large amounts of tissue, materials and work."

In my work, I use several examples to show that an improper choice of experimental design can seriously harm the efficiency of a study in terms of cost or sample size. However, finding optimal design is not an easy task, especially for high-dimensional models with many factors. One traditional approach is to use theoretical methods to find the optimal design analytically, but this is only possible for relatively simple scenarios. More generally, in Chapter 2, I argue that an analytical approach can be very limiting because most theoretical results no longer hold when the model is slightly changed, or one of the assumptions is slightly violated. This has also resulted in the bulk of papers in the optimal design literature only concerned with models with a small number of factors. I am convinced that an algorithmic approach for finding optimal design is both more practical and useful, and this goal has motivated me to find effective and flexible algorithms that can find all types of optimal designs, especially for high-dimensional models.

Researchers in engineering and computer science have been using swarm-based and other nature-inspired metaheuristic algorithms to solve all kinds of optimization problems successfully. Their optimization problems all seem much more complex and they can have mixed types of variables (continuous, discrete). The number of variables to optimize can be in the hundreds or more. I find their algorithmic approaches very refreshing, exciting and inspiring. Interestingly, such algorithms are rarely used to find optimal designs in statistics and the mainstream researchers in statistics seem to have no or limited knowledge of nature-inspired metaheuristic algorithms and modern evolutionary algorithms. Examples of such modern algorithms are particle swarm, differential evolutionary, bat, ant colony, and cuckoo. To date, the use of such optimization techniques and research in the area is still very intense and rapidly evolving, so it is necessary to constantly keep abreast of their development and the new ones that are continuously created. Since all these algorithms are general-purpose optimization algorithms, I am interested in finding out how well these algorithms can find all types of optimal designs for different types of statistical models, including high-dimensional models with many interacting variables. The latter issue is particularly pertinent in the era of big data, where statistical models tend to have more factors/variables to enhance their prediction ability. My primary goal then is to learn nature-inspired metaheuristic algorithms

and create one that is effective for finding all types of optimal designs for a wide variety of models with multiple factors and interaction terms.

There are many algorithms for finding optimal designs in the statistical literature, but many are limiting in scope and some, even popular ones, are no longer effective in the era of big data when models are increasingly complex and high-dimensional with many factors to optimize. In my work, I have developed a state-of-the-art algorithm named CSO-MA and showed that it either outperforms or performs as well as many current algorithms. In particular, I used it and showed that several so-called optimal designs in the literature are actually not optimal. For example, in Chapters 4, Section 4.1, Chapter 5, Section 5.3, and Chapter 6, Section 6.2, I show that many claimed-optimal designs in the literature are in fact not optimal because the optimizers used to find them were based on traditional algorithms or modified algorithms that were not good enough to find the true optimal designs. Additional examples include the optimal exact designs found in Lall et al. (2018) using the popular Fedorov exchange algorithm are not D-optimal as reported. I found that designs generated by popular commercial software, such as JMP, are also not always optimal as claimed. In particular, I show some of the designs generated by JMP are less efficient relative to those found by CSO-MA. Additionally, I show my proposed algorithm is flexible and able to find various types of optimal designs for different types of models that contain multiple interacting variables, whether they are linear models or not and whether they contain random effects or not. For instance, in Chapter 5, I use CSO-MA to search for a Bayesian hierarchical model with a couple of random-effects terms for studying HIV dynamics. Both the model and optimality criterion function are complicated and the computational procedure becomes complex and challenging because it involves MCMC sampling to find the optimal design.

The major contributions in my dissertation work are the introduction (or re-introduction) of modern metaheuristics to search for optimal designs and a new and effective algorithm for finding various types of optimal designs for high-dimensional models. In addition, I have demonstrated that the algorithm is flexible. It either performs better than other metaheuristic algorithms and current algorithms in statistics, or it can find the optimal design when traditional algorithms cannot. My algorithm is useful because it is a general-purpose algo-

rithm and the bulk of the reported optimal designs in the literature to date are only for models with only a couple of variables, whose algorithms are usually specific.

Aims and Scope of the Research

My dissertation aims to develop effective methods that can solve challenging optimal design problems in biomedical and public health studies. These problems not only involve advanced statistical models but also relate to complicated design criteria. Existing methods may not be powerful enough to provide solutions for a part of difficult problems. Once a better design or an optimal design is found, it can instantly help to organize more effective experiments with great control of the experimental resources and a guarantee of high experimental efficiency.

In the following chapters, I propose a new optimization algorithm, competitive swarm optimizer with mutated agents (CSO-MA). A series of benchmark tests show that CSO-MA is a very effective optimizer. Compared to many existing metaheuristic optimization algorithms, it can find higher quality solutions without requiring a longer runtime. Unlike the bulk of the work in optimal design literature that concerns low-dimensional models, I apply CSO-MA to search for different types of optimal designs for high-dimensional models, i.e., specifically, use CSO-MA to find

(i). locally D and c-optimal approximate designs for logistic, Poisson, and negative binomial models with up to five factors and all pairwise interactions;

(ii). locally D and c-optimal approximate designs for longitudinal fractional polynomial mixed models, logistic mixed models, Poisson mixed models, and negative binomial mixed models;

(iii). locally G and extended D-optimal approximate designs for fractional polynomial mixed models;

(iv). locally D-optimal approximate designs for high-dimensional logistic models applied to a car refueling experiment;

(v). locally D-optimal approximate designs for high-dimensional Poisson models applied to measure the retention factor of the drug Sulindac;

(vi). fully Bayesian optimal designs for HIV dynamics models;

(vii). fully Bayesian optimal design for a hierarchical logistic model applied to a heart defibrillator energy experiment.

Results (i)-(iii) are entirely new and represent a new ground of finding locally optimal approximate designs. Prior related work on these models is limited in low-dimensional cases, such as one or two factors. Results (iv) - (vii) are developed from real optimal design applications. Compared to the original ones, I either find designs with better criterion values, which means previous experimental strategies should be replaced, or extend the work to models with more factors or interactions, providing a more precise estimation solution.

All these models have broad applications in biomedical studies and public health domains. For instance, Poisson mixed models are useful for modeling patient recruitment in multiple clinical centers or analyzing the number of days of hospital stays by patients or length of stay for maternity needs (Wang et al., 2002; Anisimov, 2008); longitudinal models for analyzing clinical data over time or tracking the rate of development of depression and psychological distress (Ormel and Wohlfarth, 1991; Gross et al., 1994). However, compared with analysis issues, design issues for such models are seriously under-addressed and are generally more difficult to address because

- these models can be nonlinear and can have many interacting factors;
- design criteria may not be differentiable;
- some models have a very complicated Bayesian framework;
- these models may contain random effects, etc.

In my doctoral dissertation, I will show that CSO-MA is a useful algorithm for tackling complicated optimal design issues and can produce new optimal designs for more realistic models commonly used in biomedical experiments, clinical trials and in public health studies.

Organization of my Dissertation

Chapter 1 introduces the fundamentals of optimal experimental design, including the definitions of different design criteria, the use of equivalence theorem to confirm a design's optimality, and some important theoretical background. Chapter 2 reviews traditional algorithms for finding various types of optimal designs, including observations that inspire my doctoral research work. Chapter 3 proposes a new nature-inspired metaheuristic algorithm called CSO-MA with details and shows it outperforms or performs just as well as other competitors using many benchmark functions commonly used in engineering and computer science. The performance measures include implementation speed, frequency of success and quality of the solution. Chapter 4 applies CSO-MA to find locally optimal designs for high-dimensional nonlinear models with biomedical applications. Chapter 5 uses CSO-MA to find various types of optimal designs for more complicated nonlinear mixed models, including very challenging Bayesian optimal designs. I also demonstrate some optimal designs reported in the literature found by other algorithms are not optimal because CSO-MA can find designs with better criterion values. Chapter 6 demonstrates CSO-MA's flexibility by showing it can search for locally optimal approximate designs for a variety of criteria, such as G-optimality, which is not differentiable, and it can also find optimal exact designs more effectively compared to other commonly-used methods. The conclusion of this dissertation is given at the end of Chapter 6.

CHAPTER 1

Basic Theory of Constructing Optimal Experimental Designs for Statistical Models

This chapter reviews the background and fundamental theory of constructing different types of optimal approximate designs and analytic tools to confirm the optimality of a design. I use a couple of examples to illustrate how theory can be applied to find optimal designs for a few commonly-used nonlinear models in the medical sciences and the advantages that come with the use of an optimal design in practice.

1.1 Introduction

Optimal experimental designs are increasingly used in practice to rein in rising experimental costs, especially in clinical and biomedical trials, industrial applications and beyond. For instance, Berger and Wong (2009) provided a monograph that documents different applications of optimal designs to real problems. The problems range from biomedical studies to social sciences, including one that determines the optimal allocation of water wells in the Los Angeles basin.

Optimal design construction typically requires a design criterion and a fully parametric model defined on a compact space, where observations can be taken subject to a pre-specified fixed amount of budget, which usually translates to a fixed number of observations to be taken. The scientific design questions are the optimal choice of locations or time points for taking measurements from the design space, whether replications are needed and if so, how many or how often subject to the budget restriction? These issues are important because a poorly designed study can affect the efficiency of a design for making statistical inferences.

For instance, researchers might collect white blood cell data from patients in a biomedical study every hour for convenience. However, this strategy might not be the most efficient for estimating the cell's growth or decay rate. I provide examples in my dissertation to show that the frequently used uniform designs, while appealing, should not be blindly implemented because they can be inefficient. This means that they can require more observations and more design points and still provide substantially less accurate inference than a more appropriately constructed design.

Nonlinear models are broadly applied in biomedical or public health studies to describe the relationship between the response variable and one or more independent factors. The parameters enter the model nonlinearly and so the Fisher Information matrix depends on the parameters that we wish to estimate. Since the design criterion is formulated in terms of the information matrix, the optimization problem depends on the unknown parameters. A simple approach to handle the problem is to find locally optimal designs that assume nominal values of the model parameters are available. The unknown model parameters are then replaced by the nominal values and the design variables are optimized. Chernoff (1953) was the first to propose locally optimal designs with the understanding that such optimal designs can crucially depend on the accuracy of the nominal values. In practice, a robustness study must be carried out to ascertain the optimal design sensitivity to the nominal values (Kiefer and Wolfowitz, 1959). Analytical derivation of the optimal designs for most models are not possible because of the complexity of constrained optimization problems, unless the model is relatively simple. For instance, closed-form descriptions of the D-optimal designs for homoscedastic polynomial models are available, but the same assertion is no longer true for fractional polynomial models.

Algorithms for finding different types of optimal designs are continuously proposed in the literature and I review some of them later. Some can be shown to converge to the optimum and many do not. Some are for solving only a certain type of design problem and others are more flexible. Many require technical assumptions and some also require the design space to be discretized. Almost all are developed for linear models and therefore, it is not surprising when they fail to find an optimal design for a nonlinear model, especially when it is high

dimensional. It is thus desirable to find algorithms that work well for finding different types of optimal designs for a given model and also require minimal technical assumptions. The main aim of my dissertation is to develop such an algorithm and use it to solve more challenging optimal design problems of various types, including finding fully Bayesian optimal designs for nonlinear mixed models. Optimal designs can help improve the quality of statistical inference at a minimal cost. Biomedical studies are particularly expensive and so can benefit substantially from a well-designed study.

1.2 Approximate Design

Throughout, we assume that a predetermined fixed number of observations N is to be taken for the study. There are two types of designs: approximate and exact designs. I describe a k-point approximate design first and denote a generic approximate design $\boldsymbol{\eta}$ by

$$\boldsymbol{\eta} = \begin{pmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_k \\ w_1 & w_2 & \cdots & w_k \end{pmatrix}.$$

Here $\mathbf{x}_i$ is a design point of $\boldsymbol{\eta}$, which is defined on a given compact design space $\mathcal{X}$. The dimension of $\mathbf{x}_i$ depends on the number of factors in the model, and w_i is the proportion of observations to be taken at $\mathbf{x}_i$ and subject to the constraint $\sum_{i=1}^{k} w_i = 1$. Thus approximate designs are essentially probability measures defined on $\mathcal{X}$. Such an approximate design is implemented by taking $[Nw_i]$ observations at $\mathbf{x}_i$, where $[Nw_i]$ is the nearest integer to Nw_i and subject to $[Nw_1] + \cdots + [Nw_k] = N$. For example, if an approximate design in a drug study has two design points at $x_1 = 0.6/g$ and $x_2 = 0.9/g$ and the two proportions are 0.4 and 0.6, respectively, then we assign 40% of the patients to the dose $x_1 = 0.6/g$ and 60% of the patients to the dose $x_1 = 0.9/g$. This means that if $N = 100$, the design has 40 patients receive the dose $0.6/g$ and 60 patients receive the dose $0.9/g$.

In contrast, exact designs work directly with the number of replications at each dose. The proportion w_i is replaced by n_i, the number of replications at dose x_i, $i = 1, \ldots, k$, subject to $n_1 + \cdots + n_k = N$. While these two optimization problems appear similar, technically,

there is a big difference. Given a design criterion, finding the optimal approximate design is a convex optimization problem and finding the optimal exact design is not. We can use convex analysis tools to find and confirm the optimality of an approximate design but no analytical tool is available to confirm the optimality of an exact design; see details later in the chapter. In what is to follow, I focus on approximate designs but show in Chapter 6, my proposed algorithm is flexible and can also find a variety of optimal exact designs.

Throughout I assume to have a univariate outcome y and there is a fully specified regression model where the mean response is $\mathbf{E}(y) = f(\mathbf{x}, \boldsymbol{\theta})$ and defined on a user-specified compact space $\mathcal{X}$. The parameter $\boldsymbol{\theta} \in \mathbb{R}^p$ is an unknown p-dimensional model parameter vector and a common goal is to find a design that estimates the parameter as accurately as possible. The optimal approximate design for estimating the parameter $\boldsymbol{\theta}$ can be found under a unified framework if the design criterion is a convex or concave function.

1.2.1 Optimality Criteria

The statistical worth of a design is usually measured by the Fisher information matrix $\boldsymbol{M}(\boldsymbol{\eta}, \boldsymbol{\theta})$, which is constructed from assumed model $f(\mathbf{x}, \boldsymbol{\theta})$, along with the assumed error distribution. This matrix is proportional to the negative expectation of the observed Hessian matrix obtained by differentiating the log-likelihood function twice with respect to $\boldsymbol{\theta}$. The optimality criterion is often formulated as a concave/convex function of the information matrix, which I will assume to be non-singular for the purpose of this dissertation. Let $\boldsymbol{\Omega}$ be the set of all possible designs on space $\mathcal{X}$, which is also a convex set. Below are a few commonly-used design criteria in practice.

- A-optimality

  This criterion minimizes the average variance of the estimates of the model parameters:

$$\min_{\boldsymbol{\eta} \in \boldsymbol{\Omega}} trace[\boldsymbol{M}^{-1}(\boldsymbol{\eta}, \boldsymbol{\theta})].$$

- D-optimality

This criterion minimizes the volume of the confidence ellipsoid for all model parameters:

$$\max_{\boldsymbol{\eta}\in\boldsymbol{\Omega}} \log\{\det[\boldsymbol{M}(\boldsymbol{\eta},\boldsymbol{\theta})]\}.$$

- c-optimality

This criterion minimizes the asymptotic variance of a user-selected linear unbiased estimator of a predetermined linear combination of model parameters $\mathbf{c}^T\theta$:

$$\min_{\boldsymbol{\eta}\in\boldsymbol{\Omega}} \mathbf{c}^T\boldsymbol{M}^{-1}(\boldsymbol{\eta},\boldsymbol{\theta})\mathbf{c}.$$

For linear regression models, it is easy to show $\boldsymbol{M}(\boldsymbol{\eta},\boldsymbol{\theta}) = \sigma^{-2}(\mathbf{X}^T\mathbf{X})$, where $\mathbf{X}$ is the design matrix and σ^2 is the variance of the error term. For nonlinear models, their information matrices depend on the unknown parameters in $\boldsymbol{\theta}$ that require estimation. The simplest way to overcome this problem is to assume nominal values for the unknown parameters are available. Typically, the nominal values come from similar studies, pilot studies, or experts' opinions. Because such optimal designs rely on the nominal values, they are called locally optimal designs. Instead of using a single best guess of the values of the unknown parameters, which might be risky if the guesses are inaccurate, a robust option is preferable.

One option is to find a pseudo-Bayesian optimal design and assume that $\boldsymbol{\theta}$ follows a user-selected prior distribution $\pi(\boldsymbol{\theta})$. A pseudo-Bayesian D-optimal design then maximizes

$$\int_{\boldsymbol{\theta}\in\Xi^p} \log\{\det[\boldsymbol{M}(\boldsymbol{\eta},\boldsymbol{\theta})]\}\pi(\boldsymbol{\theta})d\boldsymbol{\theta},$$

where $\Xi^p \subseteq \mathbb{R}^p$ is a known set of all plausible values of $\boldsymbol{\theta}$. Pseudo-Bayesian optimal designs for other alphabetic optimality can be formulated similarly. Searching for pseudo-Bayesian optimal designs is more complicated than finding locally optimal designs because the high-dimensional integration has to be carried out numerically and efficiently before optimization can take place.

When there is good prior information on the model, another option is to implement a

fully Bayesian optimal design to overcome the dependence of the information matrix on the unknown parameters. This approach is gaining popularity and assumes that the user has in mind a specific utility function $U(\boldsymbol{\eta})$ appropriate for the problem. The Bayesian optimal design, $\boldsymbol{\eta}^*$, maximizes the expectation of the utility function $U(\boldsymbol{\eta})$ with respect to the future observation $\mathbf{y}$ and model parameter $\boldsymbol{\theta}$

$$\begin{aligned}\boldsymbol{\eta}^* &= \underset{\boldsymbol{\eta}\in\boldsymbol{\Omega}}{\operatorname{argmax}}\ \mathbf{E}[U(\boldsymbol{\eta},\mathbf{y},\boldsymbol{\theta})] \\ &= \underset{\boldsymbol{\eta}\in\boldsymbol{\Omega}}{\operatorname{argmax}}\ \int_{\mathbf{y}}\int_{\boldsymbol{\theta}} U(\boldsymbol{\eta},\mathbf{y},\boldsymbol{\theta})\pi(\mathbf{y},\boldsymbol{\theta})d\boldsymbol{\theta}d\mathbf{y}.\end{aligned}$$

The difficulty with this approach is that the integration can be complicated, high-dimensional and there is no closed-form description of the Bayesian optimal design. It is also not clear which numerical methods are best for tackling such optimization problems.

1.2.2 Examples

In this subsection, I use three simple examples to illustrate how an optimal experimental design is determined.

Example 1. Consider a simple linear regression model

$$y_i = \theta_0 + \theta_1 x_i + \epsilon_i,\ \epsilon_i \overset{i.i.d}{\sim} \mathcal{N}(0,\sigma^2),\ x_i \in \mathcal{X} = [-1,1].$$

If the aim is to find the locally D-optimal approximate design and I assume it has two design points x_1 with weight w and x_2 with weight $1-w$, the information matrix is proportional to

$$\begin{pmatrix} 1 & wx_1+(1-w)x_2 \\ wx_1+(1-w)x_2 & wx_1^2+(1-w)x_2^2 \end{pmatrix}.$$

A direct calculation shows the determinant is

$$w(1-w)(x_1-x_2)^2.$$

Since the design space is [-1, 1], it is easy to see that the determinant (or log-determinant) is maximized by setting $x_1 = -1.000$, $x_2 = 1.000$ and $w = 0.500$. Therefore, the two-point D-optimal approximate design is

$$\boldsymbol{\eta}_1 = \begin{pmatrix} -1.000 & 1.000 \\ 0.500 & 0.500 \end{pmatrix},$$

and the log-determinant is 0.000.

Example 2. Consider a Poisson regression model

$$\log[\mathbf{E}(y_i)] = \theta_0 + \theta_1 x_i,\ x_i \in \mathcal{X} = [-1, 1].$$

Suppose I am interested to estimate θ_1 in the model, which means the aim is to find the locally c-optimal approximate design with $\mathbf{c}^T = (0, 1)$. Since this is a nonlinear model, nominal parameter values are required to find the optimal design and I set $\theta_0 = 3.1$, $\theta_1 = 0.7$. I continue to assume a two-point locally c-optimal approximate design exists. Let the two design points be x_1 and x_2 and let their corresponding weights be w and $1 - w$, respectively. A direct calculation shows the information matrix is

$$\boldsymbol{M}(\boldsymbol{\eta}, \boldsymbol{\theta}) = \begin{pmatrix} we^{\theta_0+\theta_1 x_1} + (1-w)e^{\theta_0+\theta_1 x_2} & we^{\theta_0+\theta_1 x_1}x_1 + (1-w)e^{\theta_0+\theta_1 x_2}x_2 \\ we^{\theta_0+\theta_1 x_1}x_1 + (1-w)e^{\theta_0+\theta_1 x_2}x_2 & we^{\theta_0+\theta_1 x_1}x_1^2 + (1-w)e^{\theta_0+\theta_1 x_2}x_2^2 \end{pmatrix},$$

and further calculation shows the asymptotic variance of the estimated parameter of interest is proportional to

$$\mathbf{c}^T \boldsymbol{M}^{-1}(\boldsymbol{\eta}, \boldsymbol{\theta})\mathbf{c} = \frac{h_1}{h_2}, \tag{1.1}$$

where

$$h_1 = we^{\theta_0+\theta_1 x_1} + (1-w)e^{\theta_0+\theta_1 x_2},$$

$$h_2 = [we^{\theta_0+\theta_1 x_1} + (1-w)e^{\theta_0+\theta_1 x_2}][we^{\theta_0+\theta_1 x_1}x_1^2 + (1-w)e^{\theta_0+\theta_1 x_2}x_2^2] - [we^{\theta_0+\theta_1 x_1}x_1 + (1-w)e^{\theta_0+\theta_1 x_2}x_2]^2.$$

To minimize the variance, I take the partial derivatives of function (1.1) with respect to the three variables x_1, x_2, w and set them to equal to zero. The solutions are $x_1 = -1.000$, $x_2 = 1.000$ and $w = 0.668$. Therefore, the two-point c-optimal approximate design for estimating θ_1 is

$$\boldsymbol{\eta}_2 = \begin{pmatrix} -1.000 & 1.000 \\ 0.668 & 0.332 \end{pmatrix},$$

and the criterion value $\mathbf{c}^T \boldsymbol{M}^{-1}(\boldsymbol{\eta}, \boldsymbol{\theta})\mathbf{c}$ is 0.051.

Example 3. Atkinson et al. (2007) considered a quadratic regression model without an intercept given by

$$y_i = \theta_1 x_i + \theta_2 x_i^2 + \epsilon_i,\ \epsilon_i \overset{i.i.d}{\sim} \mathcal{N}(0, \sigma^2),,\ x_i \in \mathcal{X} = [0, 1].$$

The purpose of the study was to estimate θ_2 with minimum variance. This is also a c-optimal design problem with $\mathbf{c}^T = (0, 1)$.

I still assume that a two-point optimal approximate design exists with design points at x_1 and x_2 ($x_1 < x_2$) and their weights are w and $1 - w$, respectively. It is straightforward to show that the information matrix is proportional to

$$\begin{pmatrix} wx_1^2 + (1-w)x_2^2 & wx_1^3 + (1-w)x_2^3 \\ wx_1^3 + (1-w)x_2^3 & wx_1^4 + (1-w)x_2^4 \end{pmatrix}.$$

It is easy to note that x_2 must be 1. Otherwise, a scaling constant $1/x_2$ can be multiplied into the above matrix to make x_2 equal to 1 without affecting the choice of w. Therefore, the estimated variance for θ_2 is proportional to

$$\frac{1 - w + wx_1^2}{w(1-w)x_1^2(1-x_1)^2}. \tag{1.2}$$

Because the aim is to estimate θ_2 with minimum variance, I minimize function (1.2) with respect to x_1 and w. Taking the partial derivatives of function (1.2) with respect to the two

variables and then setting them equal to zero, I need to solve the following two equations

$$(x_1^2 - 1)w(1-w)x_1^2(1-x_1)^2 - x_1^2(1-x_1)^2(1-2w) = 0,$$
$$w^2(1-w)x_1^2(1-x_1)^2 - (1-w+wx_1^2)w(1-w)(2x_1 - 6x_1^2 + 4x_1^3) = 0.$$

It can be verified that the optimal approximate design is

$$\boldsymbol{\eta}_3 = \begin{pmatrix} 0.414 & 1.000 \\ 0.707 & 0.293 \end{pmatrix}.$$

1.3 Equivalence Theorem

In the last section, I show three simple examples for finding optimal designs within the class of two-point design points. Are the resulting optimal designs still optimal among all designs in $\boldsymbol{\Omega}$? Kiefer and Wolfowitz (1960) provided us with an equivalence theorem that can be used to confirm whether an approximate design is optimal among all designs in $\boldsymbol{\Omega}$ for linear models. White (1973) extended the theorem to the case when we have nonlinear models.

Let ϕ be a concave optimality criterion formulated as a function of an approximate design $\boldsymbol{\eta}$ and we want to maximize ϕ among all designs $\boldsymbol{\eta}$ in $\boldsymbol{\Omega}$. Given a statistical nonlinear model with regression function $f(\mathbf{x}, \boldsymbol{\theta})$, the Fréchet derivative of ϕ evaluated at information matrix $\boldsymbol{M}(\boldsymbol{\eta}, \boldsymbol{\theta})$ in the direction of a degenerate design $\mathbf{x} \in \boldsymbol{\Omega}$ is

$$F_\phi(\boldsymbol{M}, \boldsymbol{f}'^T\boldsymbol{f}') = \lim_{\epsilon \to 0^+} \frac{1}{\epsilon}\{\phi[(1-\epsilon)\boldsymbol{M} + \epsilon \boldsymbol{f}'^T\boldsymbol{f}'] - \phi(\boldsymbol{M})\},$$

where I use $\boldsymbol{M} = \boldsymbol{M}(\boldsymbol{\eta}, \boldsymbol{\theta})$ and $\boldsymbol{f}' = \frac{\partial f(\mathbf{x}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$ for simplicity.

1.3.1 D-optimality

The criterion function of the local D-optimality $\phi = \log\det$ is concave with respect to the information matrix (Pázman, 1980). It follows that

$$\begin{aligned}
F_\phi(\boldsymbol{M}, \boldsymbol{f}'^T\boldsymbol{f}') &= \lim_{\epsilon\to 0^+}\frac{1}{\epsilon}\{\phi[(1-\epsilon)\boldsymbol{M} + \epsilon\boldsymbol{f}'^T\boldsymbol{f}'] - \phi(\boldsymbol{M})\} \\
&= \lim_{\epsilon\to 0^+}\frac{1}{\epsilon}\{\log\det[(1-\epsilon)\boldsymbol{M} + \epsilon\boldsymbol{f}'^T\boldsymbol{f}'] - \log\det(\boldsymbol{M})\} \\
&= \lim_{\epsilon\to 0^+}\frac{1}{\epsilon}\{\log\frac{\det[(1-\epsilon)\boldsymbol{M} + \epsilon\boldsymbol{f}'^T\boldsymbol{f}']}{\det(\boldsymbol{M})}\} \\
&= \lim_{\epsilon\to 0^+}\frac{1}{\epsilon}\{\log\det[(1-\epsilon)\mathbf{I} + \epsilon\boldsymbol{f}'^T\boldsymbol{f}'\boldsymbol{M}^{-1}]\} \\
&= \lim_{\epsilon\to 0^+}\frac{1}{\epsilon}\{\log[(1-\epsilon)^p\det(\mathbf{I} + \frac{\epsilon}{1-\epsilon}\boldsymbol{f}'^T\boldsymbol{f}'\boldsymbol{M}^{-1})]\} \\
&= \lim_{\epsilon\to 0^+}\frac{1}{\epsilon}\{\log[(1-\epsilon)^p(1 + \frac{\epsilon}{1-\epsilon}trace(\boldsymbol{f}'^T\boldsymbol{f}'\boldsymbol{M}^{-1}) + \mathcal{O}(\epsilon^2))]\} \\
&= \lim_{\epsilon\to 0^+}\frac{1}{\epsilon}[p\log(1-\epsilon) + \frac{\epsilon}{1-\epsilon}trace(\boldsymbol{f}'^T\boldsymbol{f}'\boldsymbol{M}^{-1}) + \mathcal{O}(\epsilon^2)] \\
&= \boldsymbol{f}'^T\boldsymbol{M}^{-1}\boldsymbol{f}' - p.
\end{aligned}$$

Here $\mathbf{I}$ denotes the $p \times p$ identity matrix and $\boldsymbol{M}(\boldsymbol{\eta}, \boldsymbol{\theta})$ is the $p \times p$ is information matrix for the $p \times 1$ vector of parameters $\boldsymbol{\theta}$. By the Carathéodory theorem, every $\boldsymbol{M}(\boldsymbol{\eta}, \boldsymbol{\theta})$ can be expressed as a convex combination of no more than $p(p+1)/2$ elements of the form $\boldsymbol{f}'^T\boldsymbol{f}'$, which provides the evidence that only derivative in the direction of matrices with the form $\boldsymbol{f}'^T\boldsymbol{f}'$ should be considered.

The sensitivity function $S(\mathbf{x}, \boldsymbol{\eta})$ of the approximate design $\boldsymbol{\eta}$ is the directional derivative of the D-optimality criterion evaluated at $\boldsymbol{\eta}$ in the direction of a degenerate design at $\mathbf{x}$ is

$$S(\mathbf{x}, \boldsymbol{\eta}) = \frac{\partial f(\mathbf{x}, \boldsymbol{\theta})^T}{\partial \boldsymbol{\theta}}\boldsymbol{M}(\boldsymbol{\eta}, \boldsymbol{\theta})^{-1}\frac{\partial f(\mathbf{x}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} - p.$$

The equivalence theorem for D-optimality states the following statements are equivalent:

1. Design $\boldsymbol{\eta}^*$ is locally D-optimal;

2. $S(\mathbf{x}, \boldsymbol{\eta}^*) \leq 0, \forall \mathbf{x} \in \boldsymbol{\Omega}$, with equality at all design points in $\boldsymbol{\eta}^*$.

Pázman (1986) provided us with an efficiency lower bound for evaluating a design's proximity to the D-optimal design without knowing the optimum. Suppose the maximum value of a design's sensitivity function over the design space is α. The D-efficiency lower bound for this design is $e^{-\frac{\alpha}{p}}$. This can be argued from the equivalence theorem; see Pázman (1986) for details and proof of the equivalence theorem.

Two designs $\boldsymbol{\eta}_1$ and $\boldsymbol{\eta}_2$ can be compared using their relative D-efficiency. For local D-optimality, the ratio

$$RE = \left\{ \frac{\det[\boldsymbol{M}(\boldsymbol{\eta}_1, \boldsymbol{\theta})]}{\det[\boldsymbol{M}(\boldsymbol{\eta}_2, \boldsymbol{\theta})]} \right\}^{1/p}$$

measures how well design $\boldsymbol{\eta}_1$ does relative to design $\boldsymbol{\eta}_2$. If $\boldsymbol{\eta}_2$ is the D-optimal design, RE is between 0 and 1. For instance, if $RE = 0.5$, this means the design $\boldsymbol{\eta}_1$ has to be replicated twice to do as well as $\boldsymbol{\eta}_2$.

1.3.2 c-optimality

In some studies, the primary interest is not to estimate model parameters or some of them, but the main interest is to estimate a function of the model parameters. As a simple example, there may be a specific interest to estimate one of the several parameters, i.e., the goal is to estimate, say, θ_i in $\boldsymbol{\theta}$ rather than estimating them all. One can similarly derive the corresponding sensitivity function for c-optimality and verified that it is given by

$$\mathbf{S}(\boldsymbol{\eta}, \mathbf{x}) = [\frac{\partial f(\mathbf{x}, \boldsymbol{\theta})^T}{\partial \boldsymbol{\theta}} \boldsymbol{M}^{-1}(\boldsymbol{\eta}, \boldsymbol{\theta})\mathbf{c}]^2 - \mathbf{c}^T \boldsymbol{M}^{-1}(\boldsymbol{\eta}, \boldsymbol{\theta})\mathbf{c}.$$

A similar equivalence theorem for c-optimality can be similarly derived: a design $\boldsymbol{\eta}^*$ is locally c-optimal if and only if the curve of the sensitivity function is below the zero horizontal line and touches zero at all design points of $\boldsymbol{\eta}^*$. The c-efficiency of design $\boldsymbol{\eta}_1$ relative to $\boldsymbol{\eta}_2$ is $\frac{\mathbf{c}^T \boldsymbol{M}^{-1}(\boldsymbol{\eta}_2, \boldsymbol{\theta})\mathbf{c}}{\mathbf{c}^T \boldsymbol{M}^{-1}(\boldsymbol{\eta}_1, \boldsymbol{\theta})\mathbf{c}}$.

1.3.3 Examples

Following the three examples shown in Section 1.2.2, I continue to display the sensitivity function plots of three designs $\boldsymbol{\eta}_1$, $\boldsymbol{\eta}_2$ and $\boldsymbol{\eta}_3$ in Figure 1.1 and they confirm each design's optimality based on the equivalence theorem.

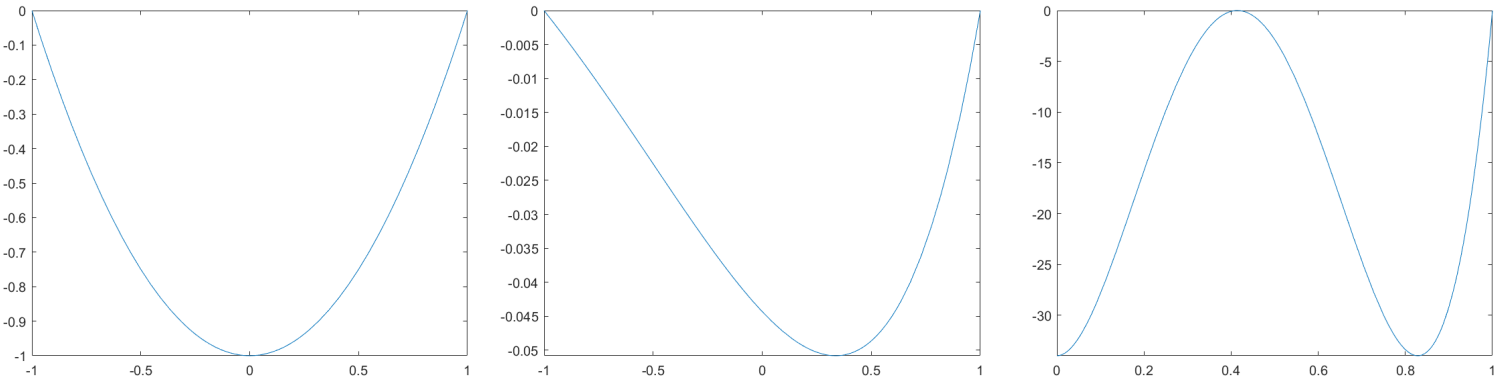

Figure 1.1: Left: the D-sensitivity function of design $\boldsymbol{\eta}_1$ in *Example 1* on the design space [-1, 1]; middle: the c-sensitivity function of design $\boldsymbol{\eta}_2$ in *Example 2* on the design space [-1, 1]; right: the c-sensitivity function of design $\boldsymbol{\eta}_3$ in *Example 3* on the design space [0, 1].

Suppose I propose to use a uniform design $\boldsymbol{\eta}_4$ for the two models in *Example 1* and *Example 2* (because they are defined on the same design space $[-1, 1]$), which is

$$\boldsymbol{\eta}_4 = \begin{pmatrix} -1.000 & -0.500 & 0.000 & 0.500 & 1.000 \\ 0.200 & 0.200 & 0.200 & 0.200 & 0.200 \end{pmatrix}.$$

This is a popular design strategy in many studies due to its ease of use. I can calculate its D-efficiency relative to $\boldsymbol{\eta}_1$ and its c-efficiency relative to $\boldsymbol{\eta}_2$, which are 70.7% and 10.2%, respectively. Therefore, we should seriously consider the design for each experiment and remind ourselves that uniform designs may not be the best choice.

The D and c-sensitivity functions of design $\boldsymbol{\eta}_4$ are shown in Figure 1.2, which confirm that $\boldsymbol{\eta}_4$ is not optimal for the two models.

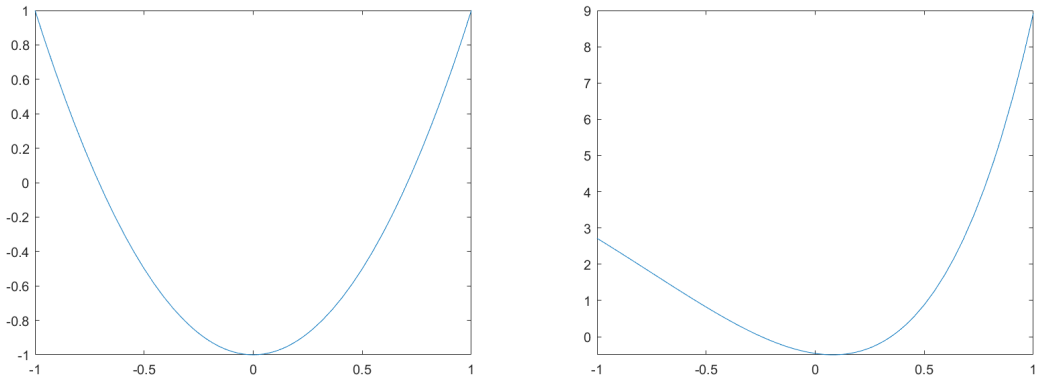

Figure 1.2: Left: the D-sensitivity function of design $\boldsymbol{\eta}_4$ for the linear regression model in *Example 1* on the design space [-1, 1]; right: the c-sensitivity function of design $\boldsymbol{\eta}_4$ for the Poisson regression model in *Example 2* on the design space [-1, 1].

1.4 Nonlinear Models for Biomedical Applications

I am particularly attracted and interested to work on few types of nonlinear models. They are the logistic regression model, Poisson regression model and the negative binomial regression model. They are generalized linear models and have a broad range of applications across various domains. The logistic model is often used for modeling binary response variable and the other two are especially useful for modeling count data.

The logistic regression model has a binary response variable y, taking values 0 or 1 with one or more factors $\mathbf{x}$. The mean response function is

$$\mathbf{E}(y) = \frac{1}{1 + \exp(-\boldsymbol{\theta}^T \mathbf{x})} = f(\mathbf{x}, \boldsymbol{\theta}), \tag{1.3}$$

where $\boldsymbol{\theta} \in \mathbb{R}^p$ is the parameter vector of interest. Applications of the logistic model and its generalization to 3 or 4-parameter models are plentiful across disciplines; see, for example, Murrough et al. (2013); Stein et al. (2013), and Ribba et al. (2014). Specific examples include analyzes of soybean growth data using a three-parameter logistic model in Davidian and Giltinan (1993) and predicting coronary artery disease in Kurt et al. (2008).

The Poisson regression model is commonly used to study the count data and ratio data.

It assumes the response variable follows a Poisson distribution with the mean equal to

$$\mathbf{E}(y) = \exp(\boldsymbol{\theta}^T \mathbf{x}) = f(\mathbf{x}, \boldsymbol{\theta}). \tag{1.4}$$

An interesting application of an extension of the Poisson regression model is illustrated in Lambert (1992), where it was used for detecting defects in the manufacturing process. There are many other applications and generalizations of the Poisson to interesting and diverse applications; see, for instance, Hu et al. (2012, 2007) and Kauhl et al. (2015).

The negative binomial regression model extends the Poisson model by accommodating for over-dispersed or under-dispersed data. We model the mean of a count outcome y and its relationship with a vector of independent variables $\mathbf{x}$ as follows

$$\mathbf{E}(y) = \exp(\boldsymbol{\theta}^T \mathbf{x}) = f(\mathbf{x}, \boldsymbol{\theta}) \quad \text{and} \quad \mathbf{Var}(y) = \exp(\boldsymbol{\theta}^T \mathbf{x})[1 + a \exp(\boldsymbol{\theta}^T \mathbf{x})]. \tag{1.5}$$

Here a is the dispersion parameter; if $a > 0$, the variance exceeds the mean and the data is over-dispersed and if $a < 0$, the data is under-dispersed.

A direct calculation shows that the Fisher information matrix $\boldsymbol{M}(\boldsymbol{\eta}, \boldsymbol{\theta})$ for a logistic model or a Poisson model with respect to a k-point approximate design $\boldsymbol{\eta}$ is proportional to

$$\sum_{i=1}^{k} w_i \lambda_i \mathbf{x}_i \mathbf{x}_i^T, \tag{1.6}$$

where $\lambda_i = f(\mathbf{x}_i, \boldsymbol{\theta})[1 - f(\mathbf{x}_i, \boldsymbol{\theta})]$ for the logistic regression model and $\lambda_i = f(\mathbf{x}_i, \boldsymbol{\theta})$ for the Poisson regression model. The construction of the information matrix for a negative binomial model can be found in Rodríguez-Torreblanca and Rodríguez-Díaz (2007). It's easy to see that, here and elsewhere, $\boldsymbol{M}(\boldsymbol{\eta}, \boldsymbol{\theta})$ becomes singular if $k < p$. In my dissertation, I assume $k \geq p$ throughout the following illustration.

In addition to the primary form of these models, my work involves their more complex forms and a wide range of applications.

1.5 A Brief Review of Theoretical Developments for Linear Models

Theoretical interests in optimal experimental design have a long history. Smith (1918) was probably among the earliest to investigate optimal design issues for constructing optimal experimental designs for linear regression problems using a criterion function. Wald (1943) was the first one who proposed to maximize the determinant of $\mathbf{X}^T\mathbf{X}$ and led Kiefer and Wolfowitz (1959) to investigate the problem. The term D-optimality was coined with D standing for determinant. Intense theoretical research was followed by De la Garza et al. (1954) and Kiefer and Wolfowitz (1959) in several of his later papers mentioned below. Hoel (1961) began to apply this theory to polynomial linear regression models. Other criterion functions had also been discussed; see, for instance, Elfving et al. (1952) among many others. Chernoff (1953) was among the first to consider optimal design issues for nonlinear models and coined the term locally optimal design, when the optimal design is constructed based on a nominal value of the unknown parameter in the nonlinear model. Ehrenfeld (1955) introduced A and E-optimality design criteria for estimating parameters in the linear models using different measures of goodness of the estimates.

A series of important contributions were made in Kiefer and Wolfowitz (1959); Kiefer et al. (1959); Kiefer and Wolfowitz (1960); Kiefer et al. (1961); Kiefer (1961), and Kiefer et al. (1962). Some key results are the introduction of approximate designs and a unified approach to find and confirm the optimality of an approximate design via equivalence theorems. In particular, the equivalence of D and G-optimal designs for two very different criteria was considered a landmark result that inspired much design research after its publication in 1959. A great appeal of working with approximate designs is applicable to solving all kind of design problems so long as the criterion is a convex function of the information matrix. The research for optimal designs for linear models is now quite well developed. Some interesting results include Cook and Nachtsheim (1982), who used polynomial regression models to approximate a general mean response function and then found the optimal designs based on the polynomial approximation; Wong and Cook (1993), who found optimal designs under multiple criteria with unequal interests; Atkinson and Cook (1995), who constructed optimal

designs for heteroscedastic models under a non-differentiable criterion; and Chaloner et al. (1984); DasGupta et al. (1991), who found Bayesian optimal designs under various setups and investigated their robustness properties. A more thorough review of the development of optimal designs is available in John and Draper (1975); Nguyen and Miller (1992); Cook and Wong (1994); Chaloner and Verdinelli (1995); Ryan et al. (2016), and Ranga et al. (2014).

1.6 A Brief Review of Theoretical Developments for Nonlinear Models

Compared to linear models, finding optimal designs for nonlinear models faces more difficulties. Different from linear models, constructing the information matrix for a nonlinear model often requires some of the model parameters to be nominally known before an optimal design can be constructed.

Generalized linear models (GLMs) have an extensive range of applications in biomedical sciences and design issues for them have been quite well studied. For example, Ford and Silvey (1980) and Wu (1985) used theory and constructed optimal designs for several GLMs; White (1975) and Sebastiani and Settimi (1997) theoretically found locally D-optimal design for one-factor logistic regression models; Chaloner and Larntz (1989) and Chaloner (1993) also provided a couple of pseudo-Bayesian D-optimal designs for GLMs. Additional theoretical work was done by Sitter and Wu (1993), who found the theoretical locally A and D-optimal designs for some GLMs Sitter and Torsney (1995); Heise and Myers (1996); Jia and Myers (2001). This was followed by much work over a long period of time to find the theoretical D-optimal designs for the logistic model with two or three factors with and without interaction terms (Haines et al., 2007; Haines and Kabera, 2018; Li and Majumdar, 2008; Haines et al., 2007; Haines and Kabera, 2018). Optimal issues other than D-optimality, for other types of GLMs received less attention; an exception is Rodríguez-Torreblanca and Rodríguez-Díaz (2007) who developed locally c-optimal designs for Poisson and negative binomial models with two factors. A commonality in the optimal design literature to date is that most work tends to focus on D-optimality and assumes that the models have one or two

factors. If there are multiple factors, they invariably assume that the factors are additive to simplify the analytic derivations of the optimal design.

Outside of GLMs, there is also much work to address theoretical optimal issues and applications for nonlinear models. The emphasis is on the latter since design issues arise in many disciplines in practice. For example, Han and Chaloner (2003) found D and c-optimal designs for exponential regression models used in viral dynamics; Ogungbenro et al. (2009) had a few applications of optimal designs to clinical pharmacology experiments, and Puškaš and Miljić (2012) built D-optimal design to study the red wine ageing process.

The results obtained from the above analytical approaches depend sensitively on every aspect of the model. A slight violation of the model assumptions will invalidate the proof. Frequently, they demand technical conditions that may not be realistic. For example, Huang et al. (2019) derived locally D-optimal designs for a series of logistic models under the conditions that all model parameters have to be non-negative and there must be one and only one categorical factor. Further, the proof cannot be amended if the model is slightly changed. The analytical derivation of the optimal design becomes invalid when we have a cubic term in the logistic model, or we have a different link function from the GLM family. The upshot is that solely relying on a theoretical approach to find optimal designs has notable limitations.

1.7 A Brief Review of Theoretical Developments for Mixed Models

Mixed models are gaining increasing attention because they incorporate inter or intra-subject variabilities and covariate effects. This allows for greater interpretability of the results compared to working with fixed effects models. However, this added level of model intricacy makes finding optimal designs for these models a lot more challenging.

Work to date on finding optimal design for mixed models is relatively little compared to those for fixed models. I list a few here. Cheng (1995); Liu et al. (2019) derived results for linear regression models with random effects; Tan and Berger (1999) found optimal designs for linear and quadratic regression models with a random intercept term; Berger and Tan

(2004) used Brute-Force enumeration to find maximin D-optimal designs for linear mixed models after the design space was sufficiently discretized to be small enough, Tekle et al. (2008a) considered longitudinal responses from a logistic mixed model and found maximin D-optimal designs, Bogacka et al. (2017) discussed how to find optimal approximate designs for the Michaelis–Menten models with one covariate, and Tekle et al. (2008b) found D-optimal designs for a clinical trial with different numbers of independent cohorts, comprising the number of repeated measurements per subject over time. Most recently, Zhou et al. (2018) found robust population designs for longitudinal linear mixed models. More recently, Jiang et al. (2019) employed an approximation method to search for locally D-optimal designs and Bayesian D-optimal designs for logistic mixed models with a single covariate.

The above work mainly assumes a longitudinal linear or simple nonlinear mixed models with a couple of factors. The results are helpful and the models are commonly used in practice. However, the models are relatively simple and the methodology or their algorithms may not work well in a more complicated situation when the model is high dimensional or the mean structure is hierarchical. Because of the limitations of an analytic approach mentioned earlier, I believe an algorithmic approach is a more helpful and practical way to find optimal designs for any given model and given design criterion. A major goal of my dissertation work is to develop a more effective algorithm to find all types of optimal designs for realistic models that involve multiple interacting factors and may have a more complex mean response.

CHAPTER 2

Review of Algorithmic Approaches

This chapter provides a selective literature review on algorithms traditionally and currently used for finding optimal designs.

2.1 Exchange Algorithm

Exchange algorithms play an important role in searching for optimal designs. The ideas behind these algorithms are similar: they start with a randomly-generated design or a user-specified design and then work iteratively to replace a current design point from a set of candidate design points by identifying a more promising point. For example, to obtain a D-optimal design, the algorithm adds a new design point $\mathbf{x}_i$ to the current design matrix $\mathbf{X}$ such that its addition will maximally increase the determinant of the information matrix, and simultaneously delete the design point $\mathbf{x}_j$ that results in the minimum decrease of the determinant. Some well-known algorithms that operate on such a principle or some slight variations thereof, include the Fedorov exchange algorithm (Fedorov, 1972), a modification of the Fedorov exchange algorithm (Cook and Nachtrheim, 1980) and the KL-exchange algorithm (Atkinson and Donev, 1989). One significant shortcoming for these methods is that the establishment of an accurate candidate set from prior knowledge, which is very difficult when the number of factors is large or the design region is highly constrained (Jones and Goos, 2007). Meyer and Nachtsheim (1995) developed a cyclic coordinate exchange algorithm for constructing D-optimal exact designs. Some extension of this method can be found in Yang et al. (2013), Sambo et al. (2014) and Overstall and Woods (2017).

Here I show the pseudo-code of using the classic Fedorov exchange algorithm to find

locally D-optimal approximate design.

Algorithm 1 The Pseudo-code for the Fedorov Exchange Algorithm to Find D-optimal Design

Choose a candidate design point set and select a starting design.
Define the Fedorov delta function between the design point $\mathbf{x}_i$ and $\mathbf{x}_j$:
$\boldsymbol{\Delta}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i' \boldsymbol{M}^{-1} \mathbf{x}_i - \mathbf{x}_j' \boldsymbol{M}^{-1} \mathbf{x}_j + (\mathbf{x}_i' \boldsymbol{M}^{-1} \mathbf{x}_j)^2 - (\mathbf{x}_i' \boldsymbol{M}^{-1} \mathbf{x}_i)(\mathbf{x}_j' \boldsymbol{M}^{-1} \mathbf{x}_j)$.
while exchanges are still beneficial **do**
  Seek the pair $(\mathbf{x}_i, \mathbf{x}_j)$ of one candidate design point and one from the current design that maximizes the delta function.
  Exchange $\mathbf{x}_i$ and $\mathbf{x}_j$.
end while

2.2 Multiplicative Algorithm

The multiplicative algorithm was created by Titterington (1976, 1978) to search for locally D-optimal approximate design on a discrete design space. For instance, we assume the space has n design points $\boldsymbol{\Omega} = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n)$ and the aim is to determine the weight vector $\boldsymbol{w} = (w_1, \cdots, w_n)$ that maximizes D-criterion function. The algorithm works by iteratively updating the weight vector. At each iteration, every weight element is adjusted by a multiplicative factor, which is determined by the magnitude of the derivative of the criterion function at this point, i.e.,

$$w_i^{(t+1)} \propto w_i^{(t)} \left(\frac{\partial \phi(\mathbf{x}_1, \cdots, \mathbf{x}_n, \boldsymbol{w}^{(t)})}{\partial w_i}\right)^{\rho},$$

where ρ is a pre-determined scaling factor. One problem for the multiplicative algorithm is that computing the derivative of the criterion function with respect to each weight element might be expensive if the model is complicated.

Martín and Gutiérrez (2015) combined the idea of the an exchange algorithm and the multiplicative algorithm and proposed a new algorithm for finding locally D-optimal designs. Since the exchange algorithms aim at finding more promising design points and the multiplicative algorithm focuses on updating the weights of current design points, this new algorithm deals with these two issues simultaneously in every iteration. However, it also requires a prior determination of the design point pool.

2.3 Vertex Direction Method

The vertex direction method was proposed by Fedorov (1972) and it also iteratively update the weight vector given a discrete design space. Following the basic setup introduced in last section, we continue to define the directional derivative by

$$d(i, \boldsymbol{w}) = \frac{\partial \phi((1-\delta)\boldsymbol{w} + \delta \mathbf{e}_i)}{\partial \delta|_{\delta=0+}},$$

where $\mathbf{e}_i$ is the vector that assigns all the mass to the i-th design point. The vertex direction method updates $\boldsymbol{w}^{(t)}$ to $\boldsymbol{w}^{(t+1)}$ by first identifying the index i which maximizes $d(i, \boldsymbol{w})$ and then setting $\boldsymbol{w}^{(t+1)}$ as the maximizer of the criterion function along the direction $\boldsymbol{w} = (1-\delta)\boldsymbol{w}^{(t)} + \delta \mathbf{e}_{i_{\max}}$, where δ is a rate parameter between 0 to 1.

2.4 Nearest Neighbor Exchange Algorithm

The nearest neighbor exchange algorithm, proposed by Böhning (1986), also has an iterative updating procedure. At every iteration, for each design point $\mathbf{x}_i$ in a discrete design space, we find the other design point $\mathbf{x}_j$ that has the smallest Euclidean distance to $\mathbf{x}_i$. Then a weight updating process is implemented between these two design points. More specifically, at iteration $t+1$, we have

$$w_i^{(t+1)} = w_i^{(t)} - \tau, \; w_j^{(t+1)} = w_i^{(t)} + \tau,$$

where $\tau = \min\{w_i^{(t)}, \max\{-w_j^{(t)}, \tau^*(i,j)\}\}$, and $\tau^*(i,j)$ is decided by the criterion function. For instance, to find locally D-optimal design, we set

$$\tau^*(i,j) = \frac{\boldsymbol{f}'(\mathbf{x}_j)\boldsymbol{M}^{-1}\boldsymbol{f}(\mathbf{x}_j) - \boldsymbol{f}'(\mathbf{x}_i)\boldsymbol{M}^{-1}\boldsymbol{f}(\mathbf{x}_i)}{2\{[\boldsymbol{f}'(\mathbf{x}_i)\boldsymbol{M}^{-1}\boldsymbol{f}(\mathbf{x}_i)][\boldsymbol{f}'(\mathbf{x}_j)\boldsymbol{M}^{-1}\boldsymbol{f}(\mathbf{x}_j)] - [\boldsymbol{f}'(\mathbf{x}_j)\boldsymbol{M}^{-1}\boldsymbol{f}(\mathbf{x}_i)]^2\}}.$$

2.5 Cocktail Algorithm

Yu (2011) proposed the cocktail algorithm for searching locally D-optimal approximate designs, which was a combination of the multiplicative algorithm, the vertex direction method and the nearest neighbor exchange algorithm. Each iteration of the cocktail algorithm contains one iteration of the vertex direction method, one iteration of the multiplicative algorithm and one iteration of the nearest neighbor exchange algorithm. It has been proved that the cocktail algorithm has a monotonic convergence property. That is, if we let the cocktail algorithm run forever, it will eventually converge to the locally D-optimal approximate design.

One shortcoming of the above four algorithms is that they originally target the D-optimality criterion. Although some modifications are now available for other concave/-convex criteria, such as c-optimality criterion, it would be problematic to make it applicable for more complicated situations, such as when we have a non-differentiable utility function to construct a fully Bayesian design; see Chapter 5. Another disadvantage of these algorithms is that they require the design space to be sufficiently discretized to find the optimal design points accurately. This implies that with more factors in the model, the model becomes high-dimensional, and there are more variables to optimize. The implications are that the construction of the design space become time-consuming and tackling the optimal design problem can become very challenging.

2.6 Mathematical Programming Algorithms

Mathematical programming approaches are recently more used as well to find optimal experimental designs. It appears that statisticians are less trained in this subfield to use such methods but some recent publications suggest that the trend may be changing. For example, Ouwens et al. (2006); Tekle et al. (2008b) and Abebe et al. (2014) utilized the Broyden–Fletcher–Goldfarb–Shanno algorithm to search optimal designs for two-factor linear mixed models and two-factor logistic models. Duarte et al. (2018) applied a semidefinite

programming technique to find optimal designs, including Bayesian optimal designs. These methods are very efficient but require that the optimization problems have to be in a certain form. This in turn puts restrictions on the design criterion and the types of models in the problem. A serious limitation of mathematical programming methods is that its capability is very much restricted by the solver. If there is no solver powerful enough to handle a high-dimensional optimization problem, then the method will not be able to find the optimum.

2.7 Metaheuristics

Another recent and increasingly used optimization methodology is to employ metaheuristics. These are general optimization strategies and can solve nearly all types of problems in principle. A particularly interesting and powerful class of metaheuristic algorithms is those inspired by nature, and often referred to as nature-inspired metaheuristic algorithms. Sometimes they are called global optimizers even though none guarantees that it will converge to the global optimum. In practice, they frequently do and quickly, even for high-dimensional optimization problems, which explain why they are widely used in engineering and computer science applications. They do not require technical assumptions and all have tuning parameters and stochastic components in them. By incorporating random components, these algorithms can extricate from local optima and add diversity to the search process. Due to this reason, the optimized trajectory of these algorithms cannot be fully tracked and it is possible that they converge to a different solution in each run.

Such tools seem relatively under-used in mainstream statistical research. Genetic algorithms and simulated annealing fall into these classes of algorithms and are likely the ones that statisticians are most familiar with. For example, Broudiscou et al. (1996) and Heredia-Langner et al. (2003) were some of the early ones to employ a genetic algorithm to search for D-optimal designs. However, genetic algorithms and simulated annealing are relatively dated algorithms and there are modern nature-inspired metaheuristic algorithms that have been shown to outperform them. Recently, more modern algorithms have been employed to find optimal designs in more challenging problems. For instance, Qiu et al. (2014) and Chen

et al. (2015) utilized particle swarm optimization to find optimal designs of various types, including problems with a non-differentiable criterion for finding minimax optimal designs discussed in King and Wong (2000) and their extensions thereof.

Compared to the aforementioned exchange algorithms or the multiplicative algorithms, mathematical programming algorithms, and especially metaheuristics, can solve a wider range of optimal design problems. The increasingly widespread use of these algorithms suggests that the current trend is in this direction. In the next chapter, I propose a new nature-inspired metaheuristic algorithm and show it is effective for finding a wide range of optimal designs in challenging problems and that it frequently outperforms current algorithms, including several of the state-of-the-art metaheuristic competitors.

CHAPTER 3

Competitive Swarm Optimizer with Mutated Agents

In this chapter, I propose a novel swarm-based algorithm called competitive swarm optimizer with mutated agents and abbreviated as CSO-MA. I use it to optimize several benchmark functions commonly used in the engineering literature and show that CSO-MA can either efficiently find better-quality solutions or similar solutions found by other state-of-the-art algorithms. I implement a simulation study to compare the performance of various algorithms, report properties of CSO-MA and performance measures I employ to evaluate the comparison.

3.1 Introduction

Swarm-based and evolutionary algorithms are increasingly used in various disciplines to find solutions to different types of optimization problems. They are generally assumption-free, easy to implement and often able to find good quality solutions for complex or high dimensional optimization problems. For example, the objective function can be non-differentiable or non-separable. The flexibility of these algorithms enables them to tackle different types of real-world optimization problems in engineering and computer science, and increasingly, in other disciplines as well; see Yang et al. (2019), for example, and the many citations below.

Most swarm-based and evolutionary algorithms are initialized by generating particles at random in a user-selected search space $\mathbf{\Omega}$. They represent candidate solutions for the problem and at each iteration, they interact with one another and update their positions according to the rules of the algorithm. For instance, in the commonly used particle swarm optimization, each particle has a local best position representing where the particle believes

the global optimum is, but at each iteration, the particles communicate among themselves and arrive at a global optimum (global best). At the next iteration, each particle then moves towards the global best position but also somewhat in the direction of its local best position. All algorithms have stochastic components and tuning parameters and they vary by numbers from algorithm to algorithm. Some algorithms are sensitive to the choice of tuning parameters and some are sensitive. A commonality of these algorithms is that they are motivated by nature, their codes are widely available and they tend to get to the proximity of the optimum quickly. However, they do not guarantee convergence or converge to the global optimum, although they frequently do. Because there have no or minimal technical assumptions, rigorous proofs of convergence of these algorithms to the global optimum are rarely available, even though there are many pseudo or incomplete proofs. Another feature of such algorithms is that there are commonly many modified versions of the first proposed algorithm, where each modified algorithm seeks to improve one or more aspects of the original or last modified algorithm. Many modified versions are also motivated by the need to solve specific types of optimization problems.

Before I present my nature-inspired metaheuristic algorithm, I describe a good representative, particle swarm optimization (PSO), in some detail. Then I describe a modified version of PSO for improved performance, which is competitive swarm optimizer (CSO). Throughout this chapter, I assume that the goal is to minimize a given objective function $f(\mathbf{x})$ over a given compact space $\mathbf{\Omega} \subset \mathbb{R}^D$, i.e.,

$$\min_{\mathbf{x} \in \mathbf{\Omega}} f(\mathbf{x}),$$

and D is the number of variables to optimize in the problem.

3.2 Swarm Optimization

3.2.1 Particle Swarm Optimization

Particle swarm optimization (PSO), proposed in Eberhart and Kennedy (1995), is one of the most famous swarm algorithms inspired by nature. This simple algorithm is initiated by generating a swarm of particles (candidate solutions) in the user-defined search space, which is assumed to be a compact set. Particles coordinate and move to regions near the perceived optimum iteratively based on each particle's historical pathway and trajectory of the whole swarm. PSO has been applied successfully in many fields, for example, in blind signal separation, power dispatch and model variable selection (Ghamisi and Benediktsson, 2015; Sun et al., 2014; Ishaque and Salam, 2013; Ishaque et al., 2012; Taormina and Chau, 2015). The dimensions of these problems range from 5 to 30.

In classic PSO, each particle $\mathbf{x}_i$ updates itself using its historical movement information and temporary global best solution. Every particle is assigned with a velocity vector $\mathbf{v}$, which is also randomly generated upon initialization. At iteration t, the particle i will move to a new position $\mathbf{x}_i^{t+1}$ using velocity $\mathbf{v}_i^{t+1}$ given by

$$\mathbf{v}_i^{t+1} = \omega \mathbf{v}_i^t + \beta_1 \mathbf{R}_1 \otimes (pbest_i^t - \mathbf{x}_i^t) + \beta_2 \mathbf{R}_2 \otimes (gbest^t - \mathbf{x}_i^t) \tag{3.1}$$

$$\text{and } \mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \mathbf{v}_i^{t+1}, \tag{3.2}$$

where $\mathbf{R}_1, \mathbf{R}_2$ are random vectors whose elements are independent draws from the uniform distribution $U(0,1)$. The operation $\otimes$ means element-wise multiplication and the tuning parameters in PSO are $\omega,\ \beta_1,\ \beta_2$ reflecting the characteristics of the flock with ω representing the inertia of the flock and the β's representing the communicative nature and cognitive ability of the flock. The best position that each particle has visited till iteration t is the personal best $pbest_i^t,\ i = 1, \cdots, n$, and the best position that the swarm of particles has ever reached till iteration t is the global best $gbest^t$. The term best refers to positions where the value of the objective function or its fitness value is smallest. These two centers influence

every particle's movement. By adding stochastic components $\mathbf{R}_1$ and $\mathbf{R}_2$ into the algorithm, these particles have a chance to explore unseen areas where they might be able to capture better solutions.

A drawback of metaheuristic algorithms is that the tuning parameters are generally influential and can seriously affect the performance of the algorithms. There are recommendations on how to tune the parameters in PSO for a more effective search; some examples are Eberhart and Shi (2000); Shi and Eberhart (1998), and Carlisle and Dozier (2000). Some proposed that these parameters be modeled as a function of the iteration number t, temporary best objective values or as topological distances among the particles (Moore and Chapman, 1999; Eberhart and Shi, 1998). Others argued for constant parameters, such as $\omega \in [0.8, 1.2]$, $\beta_1 = \beta_2 = 2$ and showed that they frequently worked well (Eberhart and Kennedy, 1995). This is a particularly attractive feature of PSO, compared with others, such as genetic algorithm or simulated annealing, which are well-known to be hypersensitive to choices in the tuning parameters.

A limitation of PSO is that it can prematurely converge to a local optimum without adequately exploring the space (Bansal et al., 2011; Xinchao, 2010; Meng et al., 2010; Gang et al., 2012; Liu et al., 2014). An effective algorithm explores the space sufficiently and has good exploitation properties to locate the optimum when it is in the proximity of the optimum. The latter means that once the flock is near the neighborhood of the global optimum, it can quickly determine the optimum precisely, rather than lingers around it for a period of time. Frequently, a trade-off between the two competing objectives is required because algorithms that are good at space exploration have limited resources to sufficiently exploit promising areas where the optimum is and algorithms with aggressive exploitation strategy can easily get stuck at a local optimum. Typically, PSO does the exploration in a few iterations and then proceeds to exploit (Nakisa et al., 2014), resulting in a decrease of the solution quality. This premature convergence phenomenon is likely due to its strong connection with the two centers *pbest* and *gbest*, which may be exerting undue influence and not changing frequently enough during iterations, see, for example, Cheng and Jin (2015); Nezami et al. (2013), and Xu et al. (2015).

Because PSO is a very successful algorithm, there are many modified versions with different strategies for improving various aspects of its performance to tackle complicated or high-dimensional optimization problems. They include parameter adaptation (Shi and Eberhart, 1998, 2001; Ratnaweera et al., 2004; Trelea, 2003; Campos et al., 2014), hybridization with other optimization methods (Higashi and Iba, 2003; Robinson et al., 2002; Liu et al., 2005; Pehlivanoglu, 2013) and swarm topological redesign (Suganthan, 1999; Kennedy, 1999; Kennedy and Mendes, 2002; Yang et al., 2019). Simulations have shown that these amended PSO algorithms, among others, perform better than the original version. One of the most effective enhancements is competitive swarm optimizer (CSO), proposed by Cheng et al. (2015), to address the premature convergence issues in PSO. CSO adopts a pairwise competition mechanism to update particles at every iteration. Compared to PSO and most of its variants, CSO has a simpler structure and its updating strategy has been shown to more effective. In particular, many simulations using tests on a series of benchmark functions have shown that CSO can find significantly better solutions than other state-of-the-art EAs for different types of problems up to solving optimization problems with 5000 dimensions (Cheng and Jin, 2015; Sun et al., 2016; Mohapatra et al., 2017; Zhang et al., 2016).

3.2.2 Competitive Swarm Optimizer

Cheng and Jin (2015) proposed CSO to tackle the premature convergence issue by recasting the updating formulas. Like PSO, CSO first generates a swarm of n particles at positions $\mathbf{x}_1, \cdots, \mathbf{x}_n$ with random velocities $\mathbf{v}_1, \cdots, \mathbf{v}_n$ in $\mathbf{\Omega}$. In each iteration, CSO randomly divides them into $\lfloor \frac{n}{2} \rfloor$ pairs and compares their objective function values. The algorithm then identifies $\mathbf{x}_i^t$ as the winner and $\mathbf{x}_j^t$ as the loser if these two are competed at iteration t and $f(\mathbf{x}_i^t) < f(\mathbf{x}_j^t)$. The winner retains the status quo and the loser learns from the winner. The two updating equations for CSO are

$$\mathbf{v}_j^{t+1} = \mathbf{R}_1 \otimes \mathbf{v}_j^t + \mathbf{R}_2 \otimes (\mathbf{x}_i^t - \mathbf{x}_j^t) + \phi \mathbf{R}_3 \otimes (\bar{\mathbf{x}}^t - \mathbf{x}_j^t) \tag{3.3}$$

$$\text{and } \mathbf{x}_j^{t+1} = \mathbf{x}_j^t + \mathbf{v}_j^{t+1}, \tag{3.4}$$

where $\mathbf{R}_1,\ \mathbf{R}_2,\ \mathbf{R}_3$ are all random vectors whose elements are drawn from $U(0,1)$. Similar to PSO, the operation $\otimes$ represents element-wise multiplication and the vector $\bar{\mathbf{x}}^t$ represents the swarm center at iteration t. The parameter ϕ is the social factor that controls the influence of the neighboring particles to the loser and a large value of ϕ is helpful for enhancing swarm diversity (but possibly impacts convergence rate). This process iterates until some stopping criteria are met.

There are three tuning parameters $\omega,\ \beta_1,\ \beta_2$ in Equation (3.1) and Equation (3.3) for PSO and only one parameter ϕ in Equation (3.3) for CSO, suggesting that it is simpler to tune CSO. Further, the transitory data PSO needs to keep track of are stored in a $n \times D$ matrix $\mathbf{x}$, a $n \times D$ matrix $\mathbf{v}$ and a $n \times D$ matrix *pbest* whereas CSO needs two of these implying that a smaller memory space is required to run CSO.

Simulation results have shown that CSO either outperforms or is competitive with many state-of-the-art swarm algorithms, such as PSO with constriction factor (PSO-CO), gaussian bare bones PSO (GBBPSO), or quantum PSO (QPSO). This conclusion was arrived at after comparing CSO performance with state-of-the-art swarm algorithms using a variety of benchmark functions with dimensions up to 5000 (Cheng and Jin, 2015; Sun et al., 2016; Mohapatra et al., 2017; Zhang et al., 2017, 2016; Zhou et al., 2016). They showed that CSO was frequently not only the winner but also required significantly shorter runtime.

CSO is relatively new but has many exciting applications. For example, Gu et al. (2018) applied CSO to select variables for high-dimensional classification models; Xiong and Shi (2018) used CSO to study a power system economic dispatch, which is typically a complex nonlinear multivariable strongly coupled optimization problem with equality and inequality constraints, and Kumarappan and Arulraj (2016) employed CSO to find the optimal installation of multiple distributed generation units in radial distribution network.

3.3 Competitive Swarm Optimizer with Mutated Agents

3.3.1 Motivation

My experience with optimal designs is that it is a common phenomenon that many optimal designs have some design points at the boundary of the design space, regardless of the models. An intuitive explanation is that plugging extreme values into the model can help estimate the lower and upper bound of the factor effect. Therefore, this motivates me to develop a nature-inspired metaheuristic algorithm that facilitates particles to search more intensively at the boundary of the design space, believing that once some design points are found, this would hasten the search process and arrive at the optimum faster.

3.3.2 Development

The genetic algorithm (GA) is an important algorithm and represents one of the earliest evolutionary algorithms (EAs) that gain broad attention. To date, I believe it is the most well-known and popular evolutionary algorithm among statisticians, even though it is dated and many more modern EAs are known to outperform GA in many ways. For example, the differential evolutionary (DE) algorithm is an advanced version of GA that is now commonly used among computer scientists and engineers for general optimization purposes. GA and all its modified versions are based on biological-inspired behaviors observed in genetic studies and they include operations, such as mutation, crossover and selection to evolve better solutions from generation to generation (Whitley, 1994). There have been a lot of inspiring GAs that performed surprisingly well in various fields; see, for instance, Deb et al. (2002); Morris et al. (1998); Anderson-Cook (2005), and Leung and Wang (2001). A main limitation of GA, as pointed out earlier, is the difficulty of specifying good tuning parameters for such algorithms to work well.

In the field of metaheuristics, it is common to hybridize two or more such algorithms to maximally take advantage of the especially useful features in each of the algorithms. Which ones to hybridize with is an open question and requires a broad knowledge of the

many metaheuristic algorithms available and their properties. A guiding principle is that the hybridized algorithm should perform better than each of the individual algorithm. To this end, I incorporate ideas from the genetic algorithm to enrich CSO and call the enhanced version of CSO as competitive swarm optimizer with mutated agents or, in short, CSO-MA. After pairing up the swarm in groups of two at each iteration, I randomly choose a loser particle p as an agent, randomly pick a variable indexed as q and then randomly change the value of $\mathbf{x}_{pq}$ to either $\mathbf{xmax}_q$ or $\mathbf{xmin}_q$, where $\mathbf{xmax}_q$ and $\mathbf{xmin}_q$ represent, respectively, the upper bound and lower bound of the q-th variable. This change is similar to the "mutation" step in GA. A conservative mutation strategy is to randomly reassign each loser particle to a random position on the boundary. If the current optimal value is already close to the global optimum, this change will not hurt since this mutation is implemented on a loser particle, which is not leading the movement of the whole swarm; otherwise, this chosen agent restarts a journey from the boundary and has a chance to escape from a local optimum.

I apply CSO-MA to test its ability to minimize several benchmark functions commonly used to test algorithms in the engineering literature. These multidimensional functions have different shapes, not necessarily separable, differentiable or convex and they may have multiple local optima. The computational complexity of CSO is $\mathcal{O}(nD)$, where n is the swarm size and D is the dimension of the problem to be optimized. Since the modification only adds one coordinate mutation operation to one loser particle, its computational complexity is the same as that of CSO. Algorithm 2 below displays the pseudo-code of CSO-MA.

3.3.3 Parameter Tuning

Tuning parameters is a perennial and critical issue for meta-heuristic algorithms because a poor choice for them can result in very poor performance. Most of these algorithms have at least two or three parameters, which makes a systematic understanding on how they interact to impact the algorithm's performance tricky.

I experimented with varying the values of the tuning parameters in CSO-MA to study its impact on its performance. Here the tuning parameters are ϕ and the swarm size n. I

Algorithm 2 The Pseudo-code for CSO-MA

A swarm of n particles.
$\mathbf{x} \leftarrow$ Randomly assign initial positions in space to particles.
$\mathbf{v} \leftarrow$ Randomly assign initial velocities to particles.
while not stopping criteria **do**
  Randomly divide the swarm into $\lfloor \frac{n}{2} \rfloor$ pairs.
  for each pair **do**
    Compare their objective function values and set the one with smaller value as the winner and the other as the loser.
    Update loser particles.
    if $\mathbf{x}_{loser}$ out of searching space **then**
      $\mathbf{x}_{loser} \leftarrow$ position at boundary.
    end if
    Randomly choose a loser $\mathbf{x}_p$ and a coordinate index q.
    Randomly change q-th variable of $\mathbf{x}_p$ to either $\mathbf{xmax}_q$ or $\mathbf{xmin}_q$, where $\mathbf{xmax}_q$, $\mathbf{xmin}_q$ represent the upper bound and the lower bound of q-th variable.
  end for
end while

found that the original default values for the tuning parameters in CSO for ϕ and n can be reliably transferred to CSO-MA and they are provided later on. I also recommend that when a parallel-computing program or machine is available to run the algorithm, a large value of n should be used. In the next section, I provide details and also discuss whether it is helpful to have the number of agents that mutate at each iteration as an additional parameter in CSO-MA and whether it has an impact on the solution quality.

3.4 Benchmark Comparisons

I use simulation to compare the performance of CSO-MA with a few state-of-the-art swarm-based competitors using several benchmark functions commonly used in the engineering literature (Tian et al., 2008; Yang et al., 2008a,b). I also include a non-swarm-based algorithm, cuckoo search, in the comparison. I choose cuckoo search because it seems to be one of the most competitive algorithms used by engineers today for general optimization purposes (Gandomi et al., 2013; Yang and Deb, 2014).

I use eight benchmark functions with different mathematical properties and consider cases when they have dimensions $D = 100,\ 500$ and 1000. These functions are frequently used for testing or comparing different global optimization algorithms; see, for example, Yang et al.

(2008b); Tian et al. (2008), and Yang et al. (2008a). CSO has already been shown to perform favorably compared with many advanced swarm algorithms for minimizing these functions and in many cases, CSO also performed the best; see the list in Section 3.2.2. Thus it follows that if CSO-MA outperforms CSO, it also outperforms other advanced swarm-based algorithms. (Cheng and Jin, 2015; Zhang et al., 2016; Zhou et al., 2016; Sun et al., 2016; Mohapatra et al., 2017; Zhang et al., 2017). Table 3.1 lists the benchmark functions, their domains, characteristics and their formulas are also given near Table 3.1.

	Function	Characteristics	Hypercube
f_1	Schwefel N.2.21	ND, NS	$[-100, 100]^D$
f_2	Rosenbrock	NS	$[-100, 100]^D$
f_3	Sphere	-	$[-100, 100]^D$
f_4	Rastrigin	MLM	$[-5, 5]^D$
f_5	Schwefel	NC, MLM	$[-500, 500]^D$
f_6	Gramacy & Lee	NC, MLM	$[0.5, 2.5]^D$
f_7	Griewank	NC, NS, MLM	$[-600, 600]^D$
f_8	Ackley	NC, MLM	$[-32, 32]^D$

Table 3.1: Information of the benchmark functions. MLM: multi local minima; NC: non-convex; ND: non-differentiable; NS: non-separable. The last column exhibits the hypercubes for evaluating these functions. Superscript D indicates the problem dimensionality.

Schwefel N.2.21:

$$f(\mathbf{x}) = f(x_1, x_2, \cdots, x_D) = \max_{i=1,\cdots,D} |x_i|;$$

global minimum: $f(\mathbf{x}^*) = 0$, $\mathbf{x}^* = (0, 0, \cdots, 0)$.

Rosenbrock:

$$f(\mathbf{x}) = f(x_1, x_2, \cdots, x_D) = \sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2];$$

global minimum: $f(\mathbf{x}^*) = 0$, $\mathbf{x}^* = (1, 1, \cdots, 1)$.

Sphere:

$$f(\mathbf{x}) = f(x_1, x_2, \cdots, x_D) = \sum_{i=1}^{D} x_i^2;$$

global minimum: $f(\mathbf{x}^*) = 0$, $\mathbf{x}^* = (0, 0, \cdots, 0)$.

Rastrigin:

$$f(\mathbf{x}) = f(x_1, x_2, \cdots, x_D) = 10D + \sum_{i=1}^{D}[x_i^2 - 10cos(2\pi x_i)];$$

global minimum: $f(\mathbf{x}^*) = 0$, $\mathbf{x}^* = (0, 0, \cdots, 0)$.

Schwefel:

$$f(\mathbf{x}) = f(x_1, x_2, \cdots, x_D) = 418.9829D - \sum_{i=1}^{D} x_i sin(\sqrt{|x_i|});$$

global minimum: $f(\mathbf{x}^*) = 0$, $\mathbf{x}^* = (420.9687, 420.9687, \cdots, 420.9687)$.

Gramacy & Lee:

$$f(\mathbf{x}) = f(x_1, x_2, \cdots, x_D) = \sum_{i=1}^{D}[\frac{sin(10\pi x_i)}{2x_i} + (x_i - 1)^4];$$

global minimum: $f(\mathbf{x}^*) = -0.8690D$, $\mathbf{x}^* = (0.5486, 0.5486, \cdots, 0.5486)$.

Griewank:

$$f(\mathbf{x}) = f(x_1, x_2, \cdots, x_D) = 1 + \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} cos(\frac{x_i}{\sqrt{i}});$$

global minimum: $f(\mathbf{x}^*) = 0$, $\mathbf{x}^* = (0, 0, \cdots, 0)$.

Ackley:

$$f(\mathbf{x}) = f(x_1, x_2, \cdots, x_D) = -20e^{-0.2\sqrt{\frac{\sum_{i=1}^{D} x_i^2}{D}}} + e - e^{\frac{\sum_{i=1}^{D} cos(2\pi x_i)}{D}};$$

global minimum: $f(\mathbf{x}^*) = 0$, $\mathbf{x}^* = (0, 0, \cdots, 0)$.

In addition to CSO, I compare CSO-MA algorithm with the following algorithms: (i) a modified CSO (MCSO) algorithm, which replaces the CSO's pairwise competition strategy by a triplet competition and is recognized as an improved CSO (Mohapatra et al., 2017), (ii) the cooperatively coevolving PSO 2 (CCPSO2) algorithm that uses a Cauchy and a Gaussian distribution for sampling next-generation particles, respectively, at *pbest* and *gbest* (Li

and Yao, 2012), (iii) the multilevel cooperative coevolution (MLCC) designed to conduct a self-adaptive neighborhood search for promising particles (Yang et al., 2008a), (iv) the separable covariance matrix adaptation evolution strategy (SEP-CMA-ES), which generates new candidate solutions by sampling around old particles and the sampling covariance matrix is constructed by incorporating information from the current solution (Ros and Hansen, 2008), (v) the efficient population utilization strategy for PSO (EPUS-PSO), which adjusts the population size according to the search results and (vi) the dynamic multi-swarm (DMS-PSO) that adopts a dynamically changing neighborhood structure for each particle (Hsieh et al., 2008; Liang and Suganthan, 2005). The last algorithm that I have included for comparison is the cuckoo search algorithm, which uses Levy flights and random walk to update new solutions (Yang and Deb, 2009). So it operates quite differently and is also motivated very differently from those for swarm-based algorithms and evolutionary algorithms.

For parameter tuning, I follow the recommendations given in Cheng and Jin (2015), which were based on a series of tests. Specifically, when optimizing $100D$ problems, I set $n = 100, \phi = 0$. For higher-dimensional optimization problems, they recommended the choice for these tuning parameters should depend on whether the objective function is separable or not. Specifically, for $500D$ problems, they suggested $n = 250, \phi = 0.1$ for separable functions and $n = 250$ and $\phi = 0.05$ for non-separable functions; for $1000D$ problems, they suggested $n = 500$ and $\phi = 0.15$ for separable functions and $n = 500$ and $\phi = 0.10$ for non-separable functions. Since CSO-MA inherits the same particle updating strategy from CSO, I follow the tuning formula for CSO and show that under the same parameter setup, the optimization performance of CSO-MA is improved. For MCSO, the tuning values of the parameters come from Table 3 of Mohapatra et al. (2017). For other algorithms, similar simulations have been carried out in Cheng and Jin (2015), Mohapatra et al. (2017) and I adopt the same parameter tuning strategy there (I refer to the parameter setup for cuckoo search algorithm suggested in Yang and Deb (2009)). For all the algorithms, I stop running them after $5000D$ function evaluations for each benchmark function, which was the guideline proposed in Tang et al. (2007).

Algorithms and tests are written and implemented using C++ on Xcode 9.0.1 and com-

		f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	w/t/l
CSO	Mean	1.37E+01	1.18E+02	2.05E-72	5.29E+01	6.30E+03	-7.15E+01	**2.22E-16**	**4.44E-15**	5/2/1
	Std dev	2.25E+00	1.03E+01	2.42E-73	1.78E+00	1.31E+02	1.07E+00	**0.00E+00**	**0.00E+00**	
CSO-MA	Mean	**8.67E-03**	9.05E+01	1.88E-33	5.33E-06	**8.15E+02**	**-8.69E+01**	**2.22E-16**	**4.44E-15**	–
	Std dev	**2.08E-04**	3.60E+00	3.57E-35	1.52E-06	**1.59E+01**	**2.27E-01**	**0.00E+00**	**0.00E+00**	
MCSO	Mean	5.22E+00	8.97E+01	**7.11E-78**	9.28E+01	7.14E+03	-7.03E+01	**2.22E-16**	**4.44E-15**	5/2/1
	Std dev	9.07E-01	2.33E+00	**3.56E-78**	8.01E-01	2.05E+02	1.33E+00	**0.00E+00**	**0.00E+00**	
CCPSO2	Mean	7.11E+00	4.21E+02	7.56E-14	3.88E-02	3.62E+03	-6.25E+01	3.41E-03	1.61E-13	8/0/0
	Std dev	7.68E+00	8.72E+01	3.41E-14	1.98E-01	4.19E+02	6.68E+00	1.42E-02	5.20E-12	
MLCC	Mean	3.44E+01	1.52E+02	5.29E-14	**4.65E-13**	1.12E+03	-8.07E+01	1.59E-12	1.06E-12	7/0/1
	Std dev	8.70E+00	5.34E+01	2.35E-14	**9.15E-14**	8.36E+01	3.72E-01	7.77E-13	9.24E-15	
SEP-CMA-ES	Mean	5.15E+01	**4.88E+00**	7.44E-14	2.93E+02	2.65E+03	-7.88E+01	3.50E-03	2.06E+01	6/0/2
	Std dev	1.91E+01	**1.53E+00**	9.06E-15	4.76E+01	2.49E+02	3.02E+00	1.71E-02	8.53E-03	
EPUS-PSO	Mean	2.24E+01	4.75E+03	9.02E-01	4.55E+02	5.79E+03	-6.74E+01	2.99E-01	2.05E+00	8/0/0
	Std dev	1.11E+00	3.80E+02	8.29E-02	1.04E+01	9.53E+01	3.21E-01	2.30E-02	2.20E-01	
DMS-PSO	Mean	6.24E+00	2.86E+02	1.05E-20	1.73E+02	2.66E+03	-7.21E+01	6.52E-10	5.49E-13	8/0/0
	Std dev	5.22E-01	3.18E+01	6.61E-22	3.52E+01	1.66E+02	1.98E+00	2.21E-11	9.86E-14	
Cuckoo	Mean	3.45E+01	6.57E+02	8.54E-01	4.22E+02	4.13E+03	-4.74E+01	3.62E-01	7.62E+00	8/0/0
	Std dev	1.17E+00	4.95E+01	1.06E-02	1.42E+01	1.53E+03	4.33E+00	7.72E-04	3.29E-01	

Table 3.2: The performance of the nine algorithms minimizing eight $100D$ benchmark functions. The values in the last column "**w/t/l**" shows the number of times CSO-MA wins (significantly better), ties (insignificant difference) and losses (significantly worse) to other algorithms using the Wilcoxon rank test at the 0.05 significance level. The bold numbers indicate the best performing algorithm among the nine for minimizing each of the benchmark functions.

plied by GCC 7.2.0. All tests are run on Hoffman2 shared cluster housed at the University of California, Los Angeles. For each function with a specific dimension, I run it ten times and average out the outcomes due to the randomness generated by stochastic components in these algorithms (Lampinen, 2002; Fan, 2002; Ugolotti et al., 2013). For each run, I request the Hoffman2 shared cluster to use a 2.2 GHz Intel Xeon E5-2650v4 CPU and 8 GB memory.

3.4.1 Simulation Results

The simulation results are shown in Table 3.2, Table 3.3 and Table 3.4, where the means and standard deviations of the results are given. The bold numbers in each column represent the best performing algorithm among the nine algorithms for minimizing each of the eight benchmark functions. The Wilcoxon rank test, which is a non-parametric method to test whether there is a significant difference between two sets of measurements, is used to compare

		f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	w/t/l
CSO	Mean	4.58E+01	4.80E+02	1.92E-66	1.58E+02	4.94E+04	-2.89E+02	**4.44E-16**	**8.88E-15**	4/0/4
	Std dev	9.47E+00	5.29E+00	3.99E-67	8.31E+00	7.53E+02	6.33E+00	**0.00E+00**	**0.00E+00**	
CSO-MA	Mean	**6.88E+00**	6.99E+02	5.34E-03	3.84E+01	**2.10E+04**	**-3.50E+02**	6.88E-05	4.40E-04	-
	Std dev	**9.41E-01**	4.52E+00	7.11E-04	2.07E+00	**4.25E+02**	**1.62E+00**	8.73E-07	2.12E-05	
MCSO	Mean	8.34E+01	9.35E+02	**1.42E-83**	8.57E+02	5.67E+04	-2.43E+02	1.23E-02	1.53E-14	6/0/2
	Std dev	4.42E+00	1.30E+01	**7.62E-85**	1.19E+01	4.07E+03	8.11E+00	3.24E-03	5.69E-15	
CCPSO2	Mean	6.32E+01	7.55E+02	6.19E-11	4.04E+00	4.65E+04	-3.15E+02	1.06E-03	4.32E-13	5/0/3
	Std dev	5.22E+00	4.57E+00	3.69E-12	5.29E-01	7.66E+02	6.03E+00	2.16E-03	5.57E-14	
MLCC	Mean	7.05E+01	9.14E+02	3.64E-13	**2.02E-11**	4.77E+04	-2.69E+02	2.15E-13	4.21E-13	4/0/4
	Std dev	5.82E+00	7.61E+01	6.28E-14	**3.05E-11**	2.78E+03	1.01E+01	2.45E-13	3.94E-13	
SEP-CMA-ES	Mean	6.05E+01	**2.87E+02**	2.33E-14	2.22E+03	3.54E+04	-2.73E+02	8.06E-04	3.00E+01	6/0/2
	Std dev	1.00E+00	**2.75E+01**	3.28E-15	1.57E+02	4.29E+02	2.10E+00	2.90E-03	4.31E-01	
EPUS-PSO	Mean	4.40E+01	5.63E+04	8.22E+00	4.03E+03	7.62E+04	-2.85E+02	5.95E-02	5.56E-01	8/0/0
	Std dev	5.51E-01	4.14E+03	2.01E+00	1.12E+02	1.62E+03	3.07E+00	3.99E-03	2.04E-02	
DMS-PSO	Mean	7.35E+01	2.85E+04	5.27E-06	4.29E+03	4.30E+04	-2.88E+02	1.57E-05	8.59E+00	6/0/2
	Std dev	4.00E+00	9.14E+02	8.86E-08	7.02E+01	9.35E+02	7.00E+00	2.46E-06	4.33E-01	
Cuckoo	Mean	6.03E+01	4.27E+04	3.27E+00	6.67E+02	6.09E+04	-1.66E+02	7.02E+00	2.54E+01	8/0/0
	Std dev	2.47E+00	1.03E+03	1.02E+00	2.11E+01	1.44E+02	3.57E+00	4.19E-01	1.16E+00	

Table 3.3: The performance of the nine algorithms minimizing eight $500D$ benchmark functions. The values in the last column "**w/t/l**" shows the number of times CSO-MA wins (significantly better), ties (insignificant difference) and losses (significantly worse) to other algorithms using the Wilcoxon rank test at the 0.05 significance level. The bold numbers indicate the best performing algorithm among the nine for minimizing each of the benchmark functions.

the performance of CSO-MA with other algorithms. At the 0.05 significance level, all results from the tests are significant in all three tables, suggesting the algorithm with the bold value finds a smaller objective function value than each of the other algorithms. The last column in each table with the heading "**w/t/l**" displays the number of times CSO-MA wins, ties and losses to the corresponding algorithm.

There is a celebrated "No Free Lunch" rule that says no algorithm can outperform all other algorithms in all situations (Wolpert and Macready, 1997). An interesting interpretation of this theorem is recently available in McDermott (2020). In Table 3.5, I rank each algorithm's performance optimizing the benchmark functions. For instance, when minimizing functions with $D = 100$, CSO-MA has 3 times defeating over all other eight algorithms, 3 times over other seven algorithms, 2 time over other six algorithms, etc., and I record such result as (3, 3, 2, ..., 0) corresponding to the header "Rank", "Rank2", etc. A smaller rank indicates

		f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	w/t/l
CSO	Mean	2.80E+01	1.45E+03	7.50E-02	1.60E+02	2.38E+05	-3.69E+02	1.30E-02	4.90E-01	8/0/0
	Std dev	3.62E+00	1.94E+02	2.09E-05	4.16E+01	3.81E+03	3.54E+00	2.64E-04	6.68E-03	
CSO-MA	Mean	**2.01E+01**	**1.19E+03**	4.70E-03	1.53E+02	**9.95E+04**	**-4.98E+02**	1.24E-02	3.01E-03	-
	Std dev	**8.27E-01**	**9.12E+01**	5.90E-04	6.31E+00	**1.88E+03**	**9.84E-01**	2.62E-04	9.40E-05	
MCSO	Mean	8.17E+01	2.03E+03	**2.97E-66**	2.34E+03	1.35E+05	-4.07E+02	**2.85E-15**	1.24E+00	6/0/2
	Std dev	3.59E-01	3.02E+01	**9.94E-68**	2.64E+01	1.11E+04	6.00E+00	**1.61E-05**	3.72E-01	
CCPSO2	Mean	7.45E+01	1.33E+03	5.29E-13	3.05E-01	2.45E+05	-3.57E+02	3.00E+00	**1.06E-12**	5/0/3
	Std dev	3.98E+00	1.17E+02	9.54E-14	2.60E-01	9.33E+02	1.07E+00	8.22E-01	**3.77E-13**	
MLCC	Mean	8.99E+01	1.82E+03	8.45E-13	**3.66E-10**	1.88E+05	-3.46E+02	4.18E-07	1.06E-12	4/0/4
	Std dev	2.65E+00	1.53E+02	4.67E-14	**4.92E-11**	4.87E+03	9.06E+00	2.47E-13	4.82E-13	
SEP-CMA-ES	Mean	4.22E+01	2.12E+03	5.92E-11	5.60E+03	2.25E+05	-3.11E+02	3.66E-04	3.42E+01	6/0/2
	Std dev	5.07E+00	7.93E+01	4.41E-13	2.17E+02	9.45E+03	3.77E+00	1.08E-05	2.26E+00	
EPUS-PSO	Mean	5.13E+01	9.66E+04	3.98E+02	4.57E+03	6.60E+05	-2.56E+02	7.44E+00	1.56E+01	8/0/0
	Std dev	1.07E+00	1.08E+03	1.77E+01	1.49E+02	1.15E+04	3.03E+00	9.62E-01	1.07E+00	
DMS-PSO	Mean	9.15E+01	5.74E+04	3.29E-03	3.83E+03	7.75E+05	-3.03E+02	4.11E+00	1.10E+01	7/0/1
	Std dev	3.44E-01	1.55E+03	3.12E-05	9.54E+01	9.29E+03	8.07E+00	5.50E-01	4.82E-01	
Cuckoo	Mean	8.26E+01	9.02E+04	2.34E+01	2.54E+03	8.66E+05	-1.47E+02	1.62E+01	4.60E+01	8/0/0
	Std dev	2.06E+00	2.63E+02	5.52E-01	3.06E+01	1.11E+04	2.98E+00	5.10E-01	1.66E+00	

Table 3.4: The performance of the nine algorithms minimizing eight $1000D$ benchmark functions . The values in the last column "**w/t/l**" shows the number of times CSO-MA wins (significantly better), ties (insignificant difference) and losses (significantly worse) to other algorithms using the Wilcoxon rank test at the 0.05 significance level. The bold numbers indicate the best performing algorithm among the nine for minimizing each of the benchmark functions.

	Number of times	Rank 1	Rank 2	Rank 3	Rank 4	Rank 5	Rank 6	Rank 7	Rank 8	Rank 9	Average Rank
$D = 100$	CSO	0	3	0	2	2	0	0	1	0	3
	CSO-MA	3	3	2	0	0	0	0	0	0	**1**
	MCSO	1	4	0	0	1	1	0	0	1	2
	CCPSO2	0	0	1	2	1	1	2	1	0	6
	MLCC	1	2	0	1	2	1	1	0	0	3
	SEP-CMA-ES	1	0	2	0	0	1	2	0	2	7
	EPUS-PSO	0	0	0	0	0	1	3	1	3	8
	DMS-PSO	0	0	1	3	2	2	0	0	0	5
	Cuckoo	0	0	0	0	0	1	0	5	2	9
$D = 500$	CSO	2	2	2	1	0	1	0	0	0	**1**
	CSO-MA	3	0	2	1	1	0	1	0	0	**1**
	MCSO	1	1	0	0	0	2	2	1	1	6
	CCPSO2	0	2	0	3	1	2	0	0	0	3
	MLCC	1	1	1	1	2	0	2	0	0	4
	SEP-CMA-ES	1	1	1	0	2	1	1	0	1	5
	EPUS-PSO	0	1	0	0	1	1	0	2	3	8
	DMS-PSO	0	0	2	1	0	1	2	1	1	7
	Cuckoo	0	0	0	1	1	0	0	4	2	9
$D = 1000$	CSO	0	1	2	2	2	0	1	0	0	5
	CSO-MA	4	0	2	1	0	1	0	0	0	**1**
	MCSO	2	2	0	0	3	1	0	0	0	2
	CCPSO2	1	3	0	1	1	2	0	0	0	4
	MLCC	2	1	2	1	1	0	0	1	0	2
	SEP-CMA-ES	0	0	2	2	0	2	0	1	1	6
	EPUS-PSO	0	0	0	1	0	0	2	3	2	8
	DMS-PSO	0	0	0	0	1	1	4	1	1	7
	Cuckoo	0	0	0	0	0	1	1	2	4	9

Table 3.5: Algorithms' rankings of minimizing f_1 to f_8 ($D = 100,\ 500,\ 1000$). A smaller ranking value indicates a better algorithm performance.

that the algorithm has a better minimization performance. The last column in the table displays the "Average Rank" and so tells whether the algorithm can stably solve different optimization tasks.

From the above tables, CSO-MA outperforms the other algorithms for minimizing functions f_1, f_5 and f_6 regardless of the dimension of the problem. For functions f_2 and f_4, CSO-MA provides competitive results among all the algorithms. Although CSO-MA does relatively poor minimizing functions f_3, f_7 and f_8, its results are acceptable because these solutions are within 10^{-3} units from the true optimum. One possible explanation is that for these functions, CSO-MA sacrifices its ability to exploit at the expense of having the mutated agents do more space exploration. An overall observation is that CSO-MA is the most consistent optimizer among these algorithms since, on average, it has the best performance in terms of minimizing the objective functions regardless of the dimension of the problem, which is also confirmed by its average rank for minimizing the functions by the various algorithms.

Other algorithms perform differently for each benchmark function, which again confirms the "No Free Lunch" rule. CSO, MLCC and CCPSO2 can consistently provide intermediate results while MCSO's outputs are not stable. EPUS-PSO, DMS-PSO and Cuckoo have a relatively poor ability optimizing these commonly-used functions.

If I set 10^{-3} as the tolerance level, so solutions that are within $\pm 10^{-3}$ from the true "optimal" value are deemed optimal. Under this rounding setup, CSO-MA's overall performance relative to the other seven algorithms for optimizing the 24 benchmark functions becomes more impressive with 131 wins, 37 ties, 0 losses compared to the earlier more stringent criterion with 130 wins, 4 ties, 34 losses.

3.4.2 More Mutated Agents?

The change I make in CSO-MA algorithm is to randomly select an agent from the loser list at every iteration and reassign it at random to a point on the boundary. My results have shown that this is an effective strategy. A natural question to ask is whether having more mutated agents at every iteration will further enhance the performance of CSO-MA.

To address this question, I keep the benchmark test configurations fixed and compare CSO-MA results when, the number of agents, $m = 2, \ldots, 10$ versus the case when $m = 1$. The histogram in Figure 3.1 shows the number of times significantly improved results are obtained via the Wilcoxon test when a larger value of m is used versus $m = 1$. The top histogram (a) shows different values of m and the bottom histogram (b) shows corresponding results when m is expressed as a percentage of n. From the two histograms, I observe that a larger value of m tends to decrease the algorithm's effectiveness. One explanation is that when m increases, there is less balance between exploration and exploitation. In particular, a larger value of m encourages the swarm to explore a larger area since more particles are assigned to random positions on the boundary and so more likely to find a better solution. This follows from the fact that for some optimization problems, like finding D-optimal designs to be discussed later, design points tend to be at the boundary of the search space. However, with a larger value of m, more particles mutate and this may make the swarm more difficult

to exploit the current promising area.

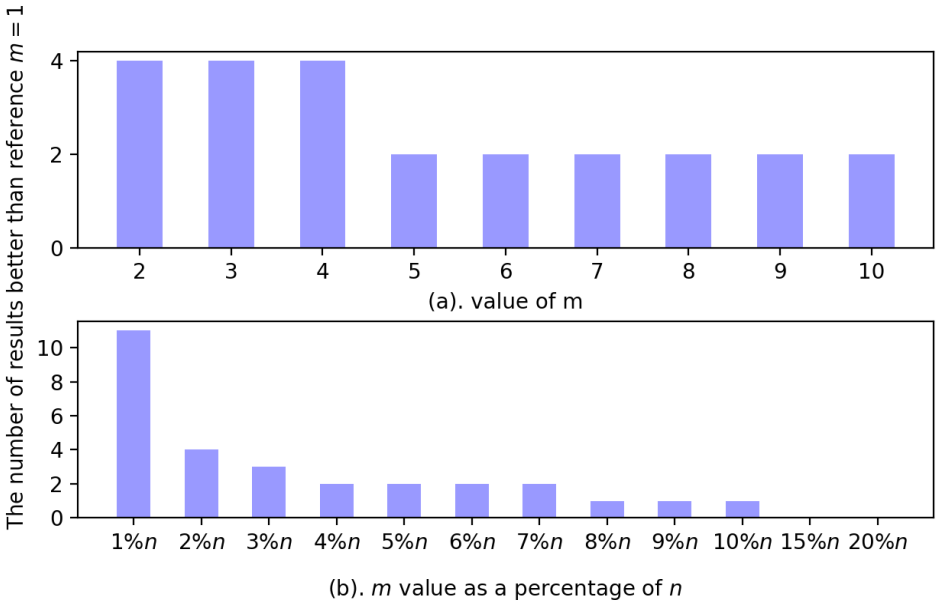

Figure 3.1: The number of significantly better results found by the algorithm using different m values compared to using $m = 1$ for optimizing the 24 benchmark functions when the swarm size is n.

The analysis of the differences between GA and CSO-MA is that the latter has in-built features that likely explain its out-performance when compared with its other competitors. For example, GA requires that a part of offspring chromosomes to mutate and for CSO-MA, there is only one mutation per iteration. This means that CSO-MA requires fewer number of computational operations and so saves time. Further, if the size of the cohort/swarm or the number of chromosomes is fixed, GA replaces existing "bad" chromosomes with newly-mutated offspring chromosomes that may not identify more promising solutions, and also loses all the information provided by the previous "bad" chromosomes. In contrast, the amount of information lost by mutating particles, i.e., one particle and one coordinate per iteration, in CSO-MA is relatively trivial. The upshot is that in CSO-MA, loser particles do not lead the swarm movement, delay the movement speed, able to inform others of unpromising areas and allow CSO-MA to explore new areas more effectively.

3.4.3 Swarm Diversity

I have claimed that the advantage of CSO-MA over CSO is that CSO-MA realizes a more diverse swarm. To show that, I use the swarm diameter as an index proposed by Olorunda and Engelbrecht (2008) to measure the diversity of a swarm. This index is defined as

$$Dia = \max_{(i \neq j)} \sqrt{\sum_{k=1}^{D} (\mathbf{x}_{ik} - \mathbf{x}_{jk})^2}.$$

When the swarm diversity index drops to zero, this implies that the search has ended and all particles have converged to a single point. The swarm cannot revive and find a better solution. On the contrary, as long as the swarm diversity is above a specific level, the swarm has a chance to explore other areas of the space.

Table 3.6 shows the swarm diameter Dia for minimizing functions f_4 to f_8 with CSO-MA having 10 wins, 3 ties and 2 losses versus CSO. These values are measured at the start and end of each search, plus one measured at the mid-point during each search. I note that these four functions all have a lot of local minima and their global minima are not located at or around the boundary of the search space. This means that merely having a common practice to send particles search at or near the boundary to search for the optimum is not helpful and these four functions are very hard to optimize. It appears that CSO-MA's success in finding better solutions for these functions than CSO is due to its having a more diverse swarm during the search process. CSO also seems to run out of energy midway during its search, whereas CSO-MA always keeps a dynamic and diversified swarm and enables it to jump out of local optima.

CSO-MA's enhanced performance is not necessarily limited to optimizing multimodal functions. I observe that the results from CSO-MA for optimizing unimodal functions f_1 to f_3 are comparable to those from CSO. For the space consideration, I do not display the Dia patterns for optimizing the other functions, but note that they share a very similar pattern. In real applications, the CPU time required to find the optimum is unknown and so it is common to employ longer runs. The implication is that when a longer runtime is allowed,

Function	Dimension	CSO's *Dia*	CSO's Results	CSO-MA's *Dia*	CSO-MA's Results
f_4	$D = 100$	(50.40, 0.00, 0.00)	5.29E+01	(52.61, 6.85, 5.30)	5.33E-06*
	$D = 500$	(103.73, 0.00, 0.00)	1.58E+02	(103.76, 5.21, 5.26)	3.84E+01*
	$D = 1000$	(143.18, 0.00, 0.00)	1.60E+02	(143.92, 3.48, 6.52)	4.16E+01*
f_5	$D = 100$	(4878.05, 0.00, 0.00)	6.30E+03	(4871.30, 914.08, 965.59)	1.31E+02*
	$D = 500$	(10069.29, 0.00, 0.00)	4.94E+04	(10057.30, 905.36, 1061.222)	7.53E+02*
	$D = 1000$	(14047.72, 0.00, 0.00)	2.38E+05	(14051.95, 604.02, 798.37)	3.81E+03*
f_6	$D = 100$	(9.87, 0.00, 0.00)	-7.15E+01	(9.75, 0.63, 1.97)	-8.69E+01*
	$D = 500$	(19.98, 0.00, 0.00)	-2.89E+02	(19.62, 1.37, 0.67)	-3.50E+02*
	$D = 1000$	(27.84, 0.00, 0.00)	-3.69E+02	(28.11, 1.12, 1.09)	-4.98E+02*
f_7	$D = 100$	(5829.34, 0.00, 0.00)	2.22E-16	(5816.03, 745.05, 611.29)	2.22E-16
	$D = 500$	(12196.04, 0.00, 0.00)	4.44E-16*	(12230.41, 708.34, 715.51)	6.88E-05
	$D = 1000$	(16750.82, 0.00, 0.00)	1.30E-02	(16779.22, 681.06, 720.33)	1.24E-02
f_8	$D = 100$	(315.39, 0.00, 0.00)	4.44E-15	(318.09, 44.81, 44.29)	4.44E-15
	$D = 500$	(655.34, 0.00, 0.00)	8.88E-15*	(639.74, 39.53, 41.36)	4.40E-04
	$D = 1000$	(893.27, 0.00, 0.00)	4.90E-01	(885.87, 43.27, 43.41)	3.01E-03*

Table 3.6: The average swarm diameter measured at the 1-st function evaluation, the $2500D$-th function evaluation and at the $5000D$-th function evaluation when CSO and CSO-MA are applied to minimize the benchmark functions f_4 to f_8, which all have many local minima. The rightmost column displays the difference in the optimal values of the function found by CSO and CSO-MA at the termination with an asterisk if the mean difference is found to be significantly different from 0.

CSO-MA is likely to find a better solution than CSO.

Chi et al. (2012) proposed a mutating-to-the-boundary strategy in an improved PSO algorithm called elastic boundary for particle swarm optimization (EBPSO). At each iteration of EBPSO, it defines an elastic region given the current global value. Then each particle is examined by a criterion to determine whether it needs to fly to a boundary area of the elastic region according to an updating function for space exploration. Compared to their algorithm design, CSO-MA has a dominant advantage that at each iteration, only one particle needs to be mutated (calculation complexity of the mutation step $\mathcal{O}(1)$), while for EBPSO, all particles have to be examined and some have to be mutated (calculation complexity of the mutation step at least $\mathcal{O}(nD)$). In Table 2 of Chi et al. (2012), the mean results EBPSO obtained for minimizing four benchmark functions are 5.05E-31, 5.26E-03, 3.98E-01 and 2.85E+00. Under the same testing setup, CSO-MA's mean results are 7.03E-55, 4.92E-07, 2.17E-06 and 1.90E-03. These results show that CSO-MA's mutation strategy is more effective than EBPSO.

3.4.4 Algorithm Speed

CSO-MA only adds a mutation operation on one particle per iteration and so the algorithmic complexity does not change compared to the original CSO. Does CSO-MA require more running time to find the optimum? To this end, I record the average running time for CSO and CSO-MA to minimize each function in Table 3.7. The table shows no significant efficiency gap between them because, for the same function, both algorithms require very similar CPU time.

	$100D$		$500D$		$1000D$	
	CSO	CSO-MA	CSO	CSO-MA	CSO	CSO-MA
f_1	3.0s	3.1s	94.5s	92.6s	412.3s	417.1s
f_2	6.6s	6.6s	180.4s	181.9s	746.2s	758.6s
f_3	3.0s	2.9s	92.9s	86.3s	411.7s	420.2s
f_4	9.2s	9.1s	247.8s	249.0s	998.0s	1025.5s
f_5	6.2s	6.2s	165.6s	165.0s	723.5s	734.6s
f_6	9.7s	9.7s	245.2s	260.1s	1040.5s	1052.1s
f_7	10.1s	10.0s	265.0s	271.4s	1144.2s	1150.3s
f_8	9.5s	9.9s	257.7s	255.2s	1053.1s	1064.7s

Table 3.7: The runtime for CSO and CSO-MA completing $5000D$ function evaluations on each benchmark function. Average results are given based on ten independent runs for each function.

CHAPTER 4

Optimal Designs for Nonlinear Fixed-effects Models and Applications

In this chapter, I apply CSO-MA to find optimal designs for different nonlinear fixed models, including high-dimensional logistic, Poisson and negative binomial models. The results show that, compared to other commonly-used algorithms, CSO-MA frequently outperforms them and is also more likely to find the complicated optimal designs. I provide three applications of CSO-MA and show the generated optimal designs guarantee the best statistical inference at minimal cost for the given amount of resources.

In Section 4.1, I use CSO-MA to find locally D-optimal designs for two-factor logistic models and compare the results with the literature. A brief comparison with a modified Fedorov exchange algorithm is also provided. In Section 4.2, I apply CSO-MA to search locally D-optimal designs for high-dimensional logistic and Poisson models and compare its performance with the other four algorithms. Section 4.3 covers a couple of experimental design applications, including estimating the gender effect on the frequency of hospitalization by acute stroke patients, testing a vision-based car refueling system, and measuring the retention factor of the drug Sulindac.

All calculations in the following chapters are done on a Windows PC with 3.20GHz Intel i7-8700 CPU, 32GB DDR4 2666MHz memory and 512G SSD storage. The programming platform is MATLAB 2018a.

4.1 Low-dimensional Models

Models with two independent factors and one interaction term are widely used in many applications. A recent theoretical result for logistic models is presented in Haines and Kabera (2018) and D-optimal designs were constructed subject to all the model parameters, except the intercept θ_0, were non-negative. This is a common observation where artificial conditions are imposed so that the theoretical optimal design can be found. For Poisson models, analytical results are only available for very simple models and for multiple factors, the factors are assumed to be additive to obtain some partial theoretical results.

In this subsection, I apply CSO-MA to find locally D-optimal designs for three two-factor logistic models with an interaction term using nominal parameters listed in the left column of Table 4.1. I observe that CSO-MA can find exactly the same locally D-optimal designs theoretically derived in Haines and Kabera (2018) in less than 1 second. I then choose additional nominal values so that the technical conditions required in Haines and Kabera (2018) are violated and test whether CSO-MA is able to find the locally D-optimal designs for the two models with and without the interaction term. Accordingly, I choose some parameter values to be negative and list them in the right column of Table 4.1.

Since CSO-MA has stochastic components in the algorithm, it can produce a different result for each run. Most of the time, the results are close, although they can be quite different occasionally. I decide to run it five times for each model and average the outputs. I stop each searching process if the criterion function value change between two successive iterations is less than 10^{-5}. I use a swarm of 50 particles and choose the value of ϕ to be any value between $[0.05, 0.20]$ following the suggestion in Cheng and Jin (2015).

After plugging the new parameters listed in the last column in Table 4.1 into the logistic models (denote them by $L1$, $L2$, $L3$ corresponding to the model index in Table 4.1 and the Poisson models (denote them by $P1$, $P2$, $P3$ corresponding to the model index in Table 4.1), I find that CSO-MA still finds the locally D-optimal designs for the models in 1 second on the design space $\mathcal{X} = [-1, 1]^2$ and these designs are shown below.

Index	In Haines and Kabera (2018)	In my simulation
1	$\theta_0 = -1,\ \theta_1 = 1,\ \theta_2 = 2,\ \theta_3 = 2$	$\theta_0 = -1,\ \theta_1 = 1,\ \theta_2 = -2,\ \theta_3 = 2$
2	$\theta_0 = -1.7,\ \theta_1 = 1,\ \theta_2 = 2,\ \theta_3 = 1$	$\theta_0 = -1.7,\ \theta_1 = -1,\ \theta_2 = 2,\ \theta_3 = -1$
3	$\theta_0 = -3,\ \theta_1 = 2,\ \theta_2 = 3,\ \theta_3 = 1$	$\theta_0 = -3,\ \theta_1 = -2,\ \theta_2 = 3,\ \theta_3 = 1$

Table 4.1: The parameters for the models having two factors and one interaction.

$$\boldsymbol{\eta}_{L1} = \begin{pmatrix} -1.000 & -0.774 & 0.108 & 1.000 & 1.000 \\ 1.000 & -1.000 & -1.000 & -0.331 & 0.378 \\ 0.250 & 0.233 & 0.142 & 0.142 & 0.233 \end{pmatrix}, \ \boldsymbol{\eta}_{P1} = \begin{pmatrix} -1.000 & -0.333 & 1.000 & 1.000 \\ 1.000 & -1.000 & -1.000 & -0.500 \\ 0.250 & 0.250 & 0.250 & 0.250 \end{pmatrix},$$

$$\boldsymbol{\eta}_{L2} = \begin{pmatrix} -1.000 & -1.000 & -0.569 & 0.869 & 1.000 \\ -0.246 & 0.713 & 1.000 & 1.000 & -1.000 \\ 0.247 & 0.128 & 0.128 & 0.247 & 0.250 \end{pmatrix}, \ \boldsymbol{\eta}_{P2} = \begin{pmatrix} -1.000 & -1.000 & 0.000 & 1.000 \\ 0.333 & 1.000 & 1.000 & -1.000 \\ 0.250 & 0.250 & 0.250 & 0.250 \end{pmatrix},$$

$$\boldsymbol{\eta}_{L3} = \begin{pmatrix} -1.000 & -1.000 & 0.366 & 1.000 \\ -0.398 & 1.000 & 0.317 & 1.000 \\ 0.250 & 0.250 & 0.250 & 0.250 \end{pmatrix}, \ \boldsymbol{\eta}_{P3} = \begin{pmatrix} -1.000 & -1.000 & 0.236 & 1.000 \\ 0.000 & 1.000 & 0.382 & 1.000 \\ 0.250 & 0.250 & 0.250 & 0.250 \end{pmatrix}.$$

The criterion values corresponding to $L1$, $P1$, $L2$, $P2$, $L3$ and $P3$ are -9.962, 1.030, -10.920, -4.384, -11.783 and -7.933.

After finding these designs, I plot their sensitivity functions in Figure 4.1 and they confirm the D-optimality of the generated designs. In summary, CSO-MA finds these locally D-optimal designs successfully and each can be done within 1 second of CPU time. My experience is that with a total of thirty runs with five runs for each model, CSO-MA produces very stable results and finds the D-optimal design in every run.

In the appendices, I provide the MATLAB codes of using CSO-MA to find locally D-optimal approximate design for a two-factor additive logistic model.

4.1.1 Comparison with Exchange Algorithm

The Fedorov exchange algorithm and its variants are well-known and widely used for constructing D-optimal designs in the literature. One problem using these algorithms is that

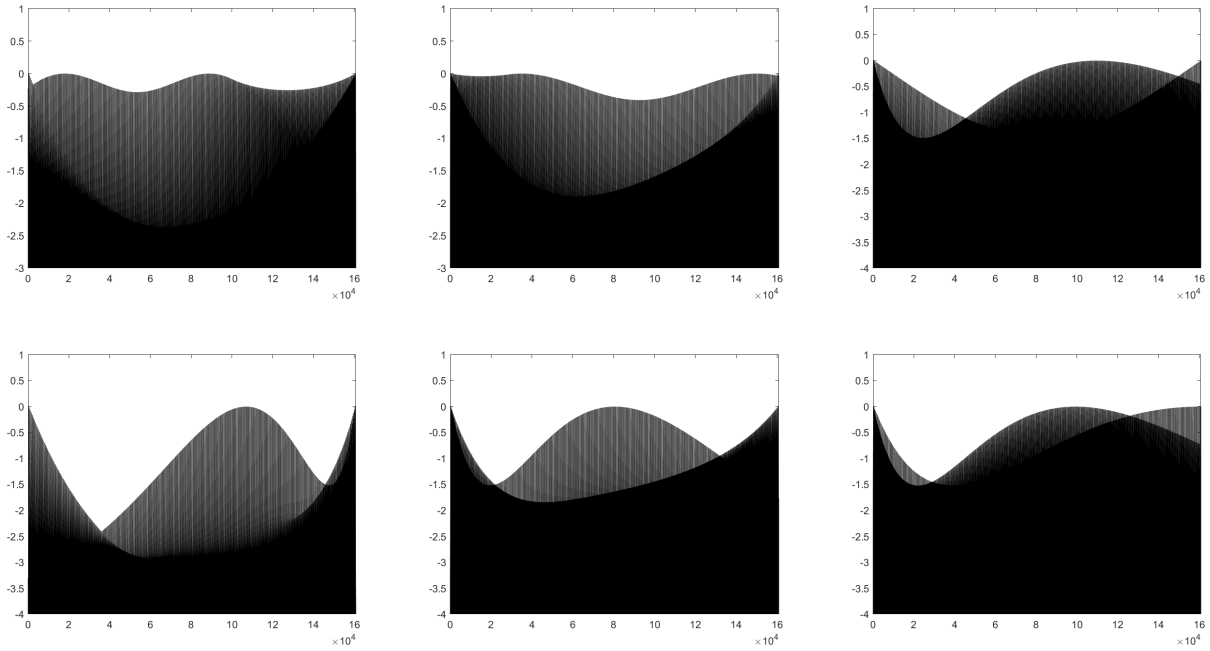

Figure 4.1: The sensitivity functions for the six designs that confirm their D-optimality. First row (left to right): $L1$, $L2$, and $L3$; second row (left to right): $P1$, $P2$, and $P3$.

practitioners have to provide a set of candidate design points and then try to find the optimal design whose design points must be found among the points in the candidate set, which may not contain all the true design points of the optimal design.

Al Labadi et al. (2015) proposed a modified Fedorov exchange algorithm for constructing locally D-optimal approximate designs by adding or exchanging two or more points simultaneously at each step and used the below example to show his proposed modified algorithm is more efficiently than the standard exchange algorithm that replaces one design point one at a time.

The regression model discussed in Al Labadi et al. (2015) is given by

$$\mathbf{E}(y) = \beta_1 + \frac{\beta_2}{1-0.2x} + \frac{\beta_3}{1+0.2x} + \frac{\beta_4}{1-0.4x} + \frac{\beta_5}{1+0.4x} + \frac{\beta_6}{1-0.6x} + \frac{\beta_7}{1+0.6x} + \frac{\beta_8}{1-0.8x} + \frac{\beta_9}{1+0.8x},$$

and the design space is $[-1, 1]$. To find the D-optimal design for estimating the parameters in the above model, Al Labadi et al. (2015) used a discretized design space with candidate

design points given by $x_i = -1 + 2i/99,\ i = 0, \cdots, 99$. Since the model contains only one variable x, dividing the design space uniformly into 100 pieces using a step size of 0.02 seems like a reasonable strategy to maintain a high level of precision. However, neither the modified Fedorov exchange algorithm nor the standard one succeeded to find the locally D-optimal approximate design, which is

$$\boldsymbol{\eta}_r = \begin{pmatrix} -1.000 & -0.934 & -0.754 & -0.433 & 0.000 & 0.433 & 0.754 & 0.934 & 1.000 \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} & \frac{1}{9} & \frac{1}{9} & \frac{1}{9} & \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{pmatrix}.$$

CSO-MA finds $\boldsymbol{\eta}_r$ in 0.2 CPU seconds, which is faster than the two exchange algorithms. The design found by the modified Fedorov algorithm is 99.8% D-efficient and the one found by the standard Fedorov algorithm is 95.6% D-efficient. A possible reason is that the design space is discretized in this problem and this means that the support points of the generated optimal design must be among the grid points, which may not be true, especially if the grid is not fine enough. However, having a very fine grid to search for the optimal design for models with many factors also requires more time to generate the grid set and requires a more powerful solver when mathematical programming methods are used. For the problem at hand, it is not surprising that the modified Fedorov algorithm finds a more efficient design than the one found by the standard Fedorov algorithm since the former is an enhancement of the latter. Until recently, exchange-type algorithms also require the space to be discretized; my experience is that the difficulty of using exchange algorithms to find optimal designs would rise dramatically when the model is nonlinear and has many factors. In the next section, I show that CSO-MA is also useful for finding various types of optimal designs for different types of high-dimensional models.

4.2 High-dimensional Models

To further test CSO-MA's capability, I apply it to search locally D-optimal designs for more complicated models. Specifically, logistic and Poisson regression models now contain five factors and all pairwise interactions. This means there are 16 parameters in each model,

implying that optimal designs for these models have to have at least 16 design points; otherwise, the Fisher information matrices are singular. As in the previous section, the design space is $[-1, 1]^5$. If we expect the optimal design has k ($\geq$ 16) design points, there are $k - 1$ weights and $5k$ components in the design points to optimize. If the optimal design is minimally supported, i.e., $k = 16$, the number of variables to optimize is at least 95. If $k = 25$, for instance, this number becomes 149 and so the problem becomes high-dimensional rapidly.

I also compare the performance of CSO-MA with four popular stochastic algorithms, namely particle swarm optimization (PSO), genetic algorithm (GA) (Miller et al., 1995), cuckoo search (CS) (Yang and Deb, 2009) and CSO. They all have many interesting real-world applications and their general effectiveness is widely documented; see, for example, Yoshida et al. (2000); Valian et al. (2011); Yang and Deb (2014), and Syahputra (2017).

The nominal parameters for logistic models are generated randomly from $U(-1, 1)$ and the nominal parameters for Poisson models are generated randomly from $U(-3, 3)$. For each set of the nominal values listed in Table 4.2, I run each algorithm for each model ten independent times and record the mean criterion values, their standard deviations and CPU time as outputs.

Model index	$\theta_0, \theta_1, \cdots, \theta_{15}$
1 (logistic)	[0.72, -0.25, 0.11, 0.91, 0.47, 0.63, -0.80, 0.86 0.22, 0.19, -0.82, -0.31, 0.33, -0.12, 0.10, 0.41]
2 (logistic)	[-0.50, -0.10, -0.18, -0.48, 0.74, -0.63, -0.96, 0.90 0.36, -0.03, -0.93, -0.21, -0.84, -0.30, -0.67, 0.97]
3 (Poisson)	[0.54, -2.70, 0.37, 1.60, 2.47, -2.44, 2.42, -0.23 -0.29, 3.00, -2.03, 1.26, -2.04, -1.86, -2.79, 0.21]
4 (Poisson)	[0.17, -1.01, -0.88, -2.53, 0.34, -2.01, -1.23, 2.04 -0.82, -0.96, 1.26, -2.81, -0.17, 1.39, 1.64, -1.55]

Table 4.2: The parameter values for four simulated models: two logistic and two Poisson models containing five factors and all pairwise interactions. The notations θ_0, $\theta_1, \cdots, \theta_{15}$ are the simulated parameters; for the logistic models, they are generated randomly from $U(-1, 1)$ and, for the Poisson models, they are generated randomly from $U(-3, 3)$.

The average criterion values and standard deviations (in the parentheses) for these four

Model index	PSO	GA	CS	CSO	CSO-MA
1	-31.05(1.51)	-29.54(0.83)	-46.27(3.65)	-28.80(0.37)	-28.45(0.11)
2	-30.78(1.07)	-29.76(1.12)	-48.39(2.93)	-28.91(0.54)	-28.37(0.26)
3	163.11(0.92)	167.30(1.31)	64.09(0.65)	169.04(1.24)	169.88(0.09)
4	93.23(1.40)	100.14(1.71)	49.24(3.27)	100.35(0.64)	101.17(0.34)
Average runtime	64.5s	95.2s	33.6s	42.3s	43.9s

Table 4.3: The average criterion values of the generated designs found by the other four algorithms for the four models with five factors and all pairwise interaction terms. Their standard deviations are in parentheses and the last row reports the average CPU time for each algorithm. The corresponding results from CSO-MA are in the last column.

models obtained by the five algorithms are summarized in Table 4.3. I observe that CSO-MA has the best and most stable performance for searching D-optimal designs for the simulated models. Finding the D-optimal designs for these models is not a trivial task but most of the designs found by CSO-MA here can be deemed as highly efficient. CSO's results are closest to CSO-MA. PSO's outcomes are worse. GA can find high-quality designs if it is allowed to run longer. If we compare results obtained by GA and CSO-MA at around 40s CPU runtime, CSO-MA could easily beat GA. CS gives the worst results.

As an illustration, I exhibit a CSO-MA-generated design for model 4 in Table 4.4 which has a D-efficiency lower bound of 99%. I also present the plot for its sensitivity function over a fine grid with uniformly spaced points.

x_1	x_2	x_3	x_4	x_5	w
1.000	-1.000	-1.000	-1.000	1.000	0.054
1.000	-1.000	-1.000	1.000	1.000	0.043
1.000	-1.000	1.000	-1.000	-1.000	0.053
1.000	-1.000	1.000	1.000	-1.000	0.037
1.000	1.000	-1.000	1.000	-1.000	0.054
1.000	1.000	-1.000	1.000	1.000	0.050
1.000	1.000	1.000	-1.000	-1.000	0.042
1.000	1.000	1.000	-1.000	1.000	0.052
1.000	1.000	1.000	1.000	1.000	0.048
-1.000	-1.000	-1.000	1.000	1.000	0.049
-1.000	-1.000	1.000	1.000	-1.000	0.049
-1.000	-0.259	1.000	1.000	1.000	0.051
-1.000	0.608	1.000	-1.000	1.000	0.046
-1.000	1.000	-1.000	-1.000	1.000	0.053
-1.000	1.000	-1.000	1.000	0.298	0.048
-1.000	1.000	1.000	-1.000	-1.000	0.043
-1.000	1.000	1.000	1.000	-1.000	0.049
-0.827	1.000	-1.000	-1.000	-1.000	0.046
-0.239	-1.000	-1.000	-1.000	-1.000	0.052
0.673	1.000	-1.000	-1.000	-1.000	0.035
0.752	-1.000	-1.000	1.000	-1.000	0.046

Table 4.4: A 99% locally D-efficient design for model 4 in Table 4.3.

4.3 Applications

4.3.1 Locally c-optimal Design for Estimating the Gender Effect on the Frequency of Hospitalization by Acute Stroke Patients

Acute stroke hospitalization is among the most expensive of any types of hospitalization which mainly consists of accurate diagnosis, therapeutic treatments to minimize stroke recurrence or occurrence of other vascular problems, provision of nursing care and early rehabilitation during the acute phase, and coordination of discharge planning. For instance, stroke ranks eighth in the Australian health system in terms of total financial burden, with the greater component of the overall cost associated with institutionalization (Anderson et al., 1994; Jørgensen et al., 1997; Diringer et al., 1999; Rundek et al., 2000). Lee et al. (2003) applied a negative binomial model to assess the association between the number of hospi-

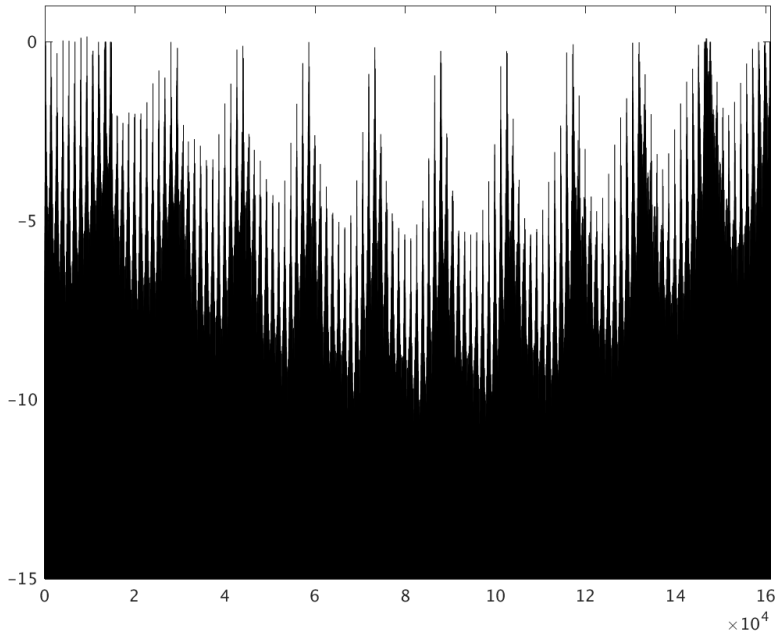

Figure 4.2: The sensitivity function plot for the design in Table 4.4.

talizations for patients sustaining ischaemic stroke and their demographic characteristics, health-related factors and medical history to plan discharge strategies, along with appropriate rehabilitation and to efficiently manage the cost of acute care. The factors are listed in Table 4.5.

Factor	Value
Intercept	–
Age (x_1)	[20, 90]
Gender (x_2)	Male: 1, female: 0
Indigenous status (x_3)	Aboriginal: 1, otherwise: 0
Area of residence (x_4)	Rural: 1, remote: 0
Presence of hypertension (x_5)	Yes: 1, no: 0
Presence of diabetes (x_6)	Yes: 1, no: 0
Presence of trial fibrillation (x_7)	Yes: 1, no: 0
Presence of transient ischaemic attack (x_8)	Yes: 1, no: 0
Presence of hypercholesterolaemia (x_9)	Yes: 1, no: 0
Presence of urinary incontinence (x_{10})	Yes: 1, no: 0
Presence of carotid endarterectomy (x_{11})	Yes: 1, no: 0

Table 4.5: The explanatory factors used in the negative binomial model for the acute stroke hospitalization study (Lee et al., 2003).

The parameter vector $\boldsymbol{\theta}$ was estimated in Lee et al. (2003) to be $(-1.232, -0.006, 0.060, 0.045, 0.573, 0.111, 0.288, -0.203, 0.151, 0.406, 0.550, 1.345)^T$ and $a = 3$. I am especially interested in estimating the effect of gender on the number of acute stroke hospitalizations. To this end, I want to find a c-optimal design to estimate the appropriate coefficient in the model as accurately as possible and this can be accomplished by taking $\mathbf{c} = (0, 0, 1, 0, \cdots, 0)^T$. I use CSO-MA to find the c-optimal design and the value of the optimality criterion is 4.16 and the optimizer, on average, requires 57.1s to find the c-optimal design shown in Table 4.6. This optimal design theoretically provides the most accurate inference for ascertaining gender's effect on the number of days of hospital stay among acute stroke patients. The c-optimal design requires many 20-year-old subjects (x_1) and so the design does not require patients' ages to be well spread out over the range $[20, 90]$. Similar observations apply to binary factors such as x_4, x_9, x_{10} and x_{11}. This finding also shows that optimal designs providing the most accurate inference based on technical considerations may be non-intuitive and undesirable from a practical viewpoint.

In practice, optimal designs should be amended based on practical considerations before implementation. The guiding principle is to modify the optimal design to meet the practical demands to the extent possible without sacrificing too much statistical efficiency. For this reason, optimal designs are more appropriately called calibration designs and used as benchmarks to measure the quality of the implemented design. In practice, the implemented design may stray from the optimum to meet practical demands, provided the loss in efficiency is not too large.

For the application at hand, it is likely desirable to have a more diverse age group of patients in the study. A direct calculation shows that the efficiency of the same design after I replace half the patients aged 20 by, say, patients aged 60, has a criterion value of 4.23, which is close to the criterion value of 4.20 of the optimal design. This implies that the modified design has a c-efficiency of $4.20/4.23 = 0.992$ or 99.2% and so also works quite well as the optimum and it has more diverse age groups of subjects in the study. Further, if more diverse and equally spread age groups are sought, I may consider replacing values in the first column of Table 4.6 by an equal number of patients with ages $(20, 25, 30, 35, \cdots, 80, 85)$

and keeping other factor settings in the design unchanged. A direct calculation shows the criterion value of this new design is 4.26 and is 98% efficient compared to the c-optimal design. Similar calculations apply if I wish to investigate changes in other settings of the optimal design.

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	w
20	0	0	1	0	1	0	0	1	1	1	0.06
20	0	1	1	0	0	1	0	1	1	1	0.02
20	1	0	1	0	1	0	1	1	1	1	0.19
20	0	0	1	1	1	0	1	1	1	1	0.09
27	0	1	1	0	1	0	0	1	1	1	0.06
20	1	1	1	1	1	0	0	1	1	1	0.14
20	1	0	1	1	1	0	1	1	1	1	0.01
20	0	0	1	0	1	0	1	1	1	1	0.07
31	1	1	1	0	1	1	1	1	1	1	0.02
20	1	1	1	1	1	0	1	1	1	1	0.10
28	1	1	1	1	1	0	1	1	1	1	0.02
20	0	0	1	0	1	0	1	1	1	1	0.01
20	1	0	1	1	0	0	1	1	1	1	0.02
20	0	1	1	1	1	0	1	1	1	1	0.19

Table 4.6: The c-optimal design for estimating the gender effect on the frequency of hospitalization by acute stroke patients.

4.3.2 Car Refueling Experiment

Grimshaw et al. (2001) described an experiment, based on a logistic model, for testing a vision-based car refueling system with the question that whether a computer-controlled nozzle was able to insert itself into the gas pipe correctly or not (Lukemire et al., 2018). The experiment includes four binary explanatory factors ($x_1 \sim x_4$ numerically taking -1 or 1): ring type (white paper or reflective), lighting (room lighting or two flood lights and room lights), sharpening (without or with), smoothing (without or with); six continuous factors ($x_5 \sim x_{10}$): lightning angle (50 to 90 degrees), gas-cap angle 1 (30 to 55 degrees), gas-cap angle 2 (0 to 10 degrees), can distance (18 to 48 inches), reflective ring thickness (0.125 to 0.425 inches) and threshold step vale (5 to 15). Moreover, three potential pairwise interaction terms are considered to be included, which are the interaction between ring type and reflective

ring thickness, the interaction between lighting and lighting angle, the interaction between smoothing and car distance. To fully test CSO-MA's potential for finding the D-optimal design for estimating all parameters in the mean function of a complex and high-dimensional model, I include two three-factor interaction terms. The physical interpretations of the explanatory factors are described in the accompanying table.

Variable	Notation	Type	Range
Ring type	x_1	Binary	-1 or 1
Lightning	x_2	Binary	-1 or 1
Sharpening	x_3	Binary	-1 or 1
Smoothing	x_4	Binary	-1 or 1
Lightning Angle	x_5	Continuous	[50, 90]
Gas-cap Angle 1	x_6	Continuous	[30, 55]
Gas-cap Angle 2	x_7	Continuous	[0, 10]
Can Distance	x_8	Continuous	[18, 48]
Reflective Ring Thickness	x_9	Continuous	[0.125, 0.425]
Threshold Step Value	x_{10}	Continuous	[5, 15]
P-Interaction 1	x_1x_9	-	-
P-Interaction 2	x_2x_5	-	-
P-Interaction 3	x_4x_8	-	-
T-Interaction 1	$x_6x_7x_8$	-	-
T-Interaction 2	$x_3x_4x_{10}$	-	-

Table 4.7: Variable information for the car refueling experiment.

The full model contains 10 factors and 16 parameters. To find locally D-optimal design, a set of parameter values is proposed: $\boldsymbol{\theta} = (3.00, 0.50, 0.75, 1.25, 0.80, 0.50, 0.80, -0.40, -1.00, 2.65, 0.65, 1.10, -0.20, 0.90, -0.36, 1.07)$. If I only consider the additive linear part without interaction terms, corresponding locally D-optimal design had already been found in Lukemire et al. (2018) by employing another evolutionary algorithm called "Quantum-behaved PSO" and its average running time touching the optimal design was 140 seconds.

To initialize CSO-MA and search for locally D-optimal designs for both models (with and without interaction terms), I set $k = 20$, which is the initial guess on the number of design points and $n = 200$, the number of particles. Since evolutionary algorithms work by incorporating random factors, I run the algorithm ten independent times. On average, CSO-MA spends 24 seconds to find the optimal design for the no-interaction model and

about 400 seconds for the model with all pairwise interactions.

Table 4.8 displays the locally 17-point D-optimal design for the full model and its criterion value is -7.256.

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	w
1.000	-1.000	-1.000	-1.000	50.000	30.000	0.026	31.494	0.125	5.000	0.062
1.000	-1.000	1.000	-1.000	90.000	30.000	0.285	18.000	0.425	5.000	0.063
1.000	-1.000	1.000	-1.000	90.000	37.342	0.000	47.999	0.425	15.000	0.061
1.000	-1.000	1.000	1.000	68.511	55.000	0.209	29.239	0.425	15.000	0.062
1.000	1.000	-1.000	-1.000	90.000	30.000	0.085	28.026	0.125	15.000	0.062
1.000	1.000	-1.000	-1.000	90.000	31.591	0.000	34.269	0.425	5.000	0.062
1.000	1.000	1.000	-1.000	50.000	55.000	0.000	33.014	0.125	5.000	0.062
-1.000	-1.000	-1.000	-1.000	50.000	36.649	0.000	48.000	0.425	15.000	0.061
-1.000	-1.000	-1.000	-1.000	90.000	55.000	0.025	48.000	0.425	5.000	0.062
-1.000	-1.000	-1.000	-1.000	90.000	55.000	0.091	36.073	0.125	15.000	0.061
-1.000	-1.000	-1.000	1.000	75.860	30.000	0.363	18.000	0.125	15.000	0.063
-1.000	-1.000	1.000	-1.000	50.000	55.000	0.007	36.516	0.125	15.000	0.062
-1.000	-1.000	1.000	-1.000	90.000	30.000	0.029	38.137	0.425	15.000	0.020
-1.000	-1.000	1.000	-1.000	90.000	30.000	0.000	45.986	0.125	5.000	0.060
-1.000	1.000	-1.000	-1.000	50.000	30.000	0.000	34.471	0.125	15.000	0.057
-1.000	1.000	-1.000	1.000	67.477	30.000	0.070	48.000	0.125	15.000	0.063
-1.000	1.000	1.000	-1.000	50.000	30.000	0.011	18.361	0.425	15.000	0.056

Table 4.8: The locally D-optimal design for the full model of the car refueling experiment.

4.3.3 Optimal Design for Measuring the Retention Factor of the Drug Sulindac

Sulindac is an anti-inflammatory drug used to reduce pain, swelling, and joint stiffness for arthritis patients. One special interest in Sulindac is to measure its retention factor $\mathcal{R}$ by chromatography technique, which can separate Sulindac from some of its impurities. The retention factor is the ratio of the distance that the interested material moves above the origin to the distance that the solvent front moves above the origin when using chromatography. Different compounds have different retention factors when in different solution systems; thus, they can be separated. Knowing a material's retention factor, especially when it often appears in the form of a mixture, can further help researchers learn its concentration or other interesting properties.

Such an experiment was described by Krier et al. (2011) where the retention factor $\mathcal{R}$

was linked with four factors that they had an interest in:

1. the gradient elution time (x_1, minutes);
2. the percentage of acetonitrile at the beginning of the gradient elution (x_2);
3. the percentage of acetonitrile at the end of the gradient elution (x_3);
4. whether proceed isocratic elution procedure (x_4).

Factors x_1, x_2, x_3 are continuous with range $[1, 5]$, $[15\%, 55\%]$, $[55\%, 65\%]$, respectively. Factor x_4 is binary taking value 0 or 1 corresponding to without and with isocratic elution procedure. A Poisson regression model was proposed in the literature and is shown below, including some quadratic and interaction terms

$$\begin{aligned}\log \mathcal{R} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_4 + \theta_5 x_1^2 + \theta_6 x_2^2 + \theta_7 x_3^2 + \theta_8 x_1 x_2 + \theta_9 x_1 x_3 \\ + \theta_{10} x_1 x_4 + \theta_{11} x_2 x_3 + \theta_{12} x_2 x_4 + \theta_{13} x_3 x_4.\end{aligned}$$

In addition, I add all three-way interaction terms ($\theta_{14} x_1 x_2 x_3$, $\theta_{15} x_1 x_2 x_4$, $\theta_{16} x_1 x_3 x_4$, $\theta_{17} x_2 x_3 x_4$) to the model so that higher-order interactive effects can be studied as well.

For illustrative purposes, I take a set of nominal values for the model parameters to be: $\boldsymbol{\theta} = (1.0, -0.5, 0.7, 1.2, 0.8, 0.5, 0.8, -0.4, -1.0, 2.7, 0.6, 1.7, 2.4, -1.1, 0.3, 0.6, -0.4, -0.2)$. CSO-MA takes around 12 seconds to find corresponding locally D-optimal design $\boldsymbol{\eta}_1$ for the full model (including all the three-way interactions) and 10 seconds to find D-optimal design $\boldsymbol{\eta}_2$ for the model without any three-way interaction terms. These two optimal designs are shown in Table 4.9 and Table 4.10.

I have not seen any D-optimal design for a four-factor Poisson model with all three-way interaction terms and so such a design is new. Admittedly, such models may not be common in practice, but it is not clear if it is because such optimal designs were not available before. I demonstrated that my methodology can find such optimal designs if they are needed.

x_1	x_2	x_3	x_4	w	x_1	x_2	x_3	x_4	w
4.172	0.550	0.650	1.000	0.056	4.635	0.550	0.650	0.000	0.053
4.641	0.150	0.550	0.000	0.051	4.655	0.150	0.650	0.000	0.054
4.661	0.550	0.550	1.000	0.055	4.667	0.150	0.550	1.000	0.055
4.668	0.550	0.550	0.000	0.011	4.680	0.150	0.650	1.000	0.055
4.790	0.550	0.650	1.000	0.056	5.000	0.150	0.550	0.000	0.055
5.000	0.150	0.550	1.000	0.054	5.000	0.150	0.650	0.000	0.055
5.000	0.150	0.650	1.000	0.055	5.000	0.389	0.650	1.000	0.045
5.000	0.392	0.609	1.000	0.025	5.000	0.550	0.550	0.000	0.054
5.000	0.550	0.550	1.000	0.055	5.000	0.550	0.609	1.000	0.046
5.000	0.550	0.650	0.000	0.055					

Table 4.9: Locally D-optimal design $\boldsymbol{\eta}_1$ for the Poisson model including all three-way interactions.

x_1	x_2	x_3	x_4	w	x_1	x_2	x_3	x_4	w
4.179	0.550	0.650	1.000	0.056	4.325	0.150	0.650	1.000	0.031
4.653	0.150	0.650	0.000	0.071	4.680	0.550	0.550	1.000	0.043
4.683	0.150	0.550	1.000	0.045	4.765	0.150	0.650	1.000	0.057
4.777	0.550	0.650	1.000	0.062	5.000	0.150	0.550	0.000	0.071
5.000	0.150	0.550	1.000	0.064	5.000	0.150	0.609	1.000	0.039
5.000	0.150	0.650	0.000	0.071	5.000	0.150	0.650	1.000	0.064
5.000	0.347	0.611	1.000	0.035	5.000	0.347	0.650	1.000	0.059
5.000	0.550	0.550	1.000	0.063	5.000	0.550	0.609	1.000	0.032
5.000	0.550	0.650	0.000	0.071	5.000	0.550	0.650	1.000	0.064

Table 4.10: Locally D-optimal design $\boldsymbol{\eta}_2$ for the Poisson model without any three-way interactions.

CHAPTER 5

Optimal Designs for Mixed-effects Models and Bayesian Optimal Designs

This chapter applies CSO-MA to search for optimal designs for nonlinear mixed models. These models are widely used in biomedical studies, for instance, in longitudinal studies to monitor patients' reactions to intervention over time. I also use CSO-MA to construct Bayesian optimal designs, which can be challenging to find. My results show that CSO-MA can efficiently find different types of optimal designs for models with one or more factors for implementation or use them to calibrate other experiments.

In Sections 5.1 and 5.2, I show that CSO-MA can find optimal designs for various mixed models, including logistic model, Poisson model, and negative binomial model. These models can contain multiple correlated random effects, making searching for optimal designs a challenging task. Some novel and interesting questions are addressed; for instance, unlike most of the literature treating time as a continuous factor, I propose to choose a finite number of time points from a given candidate set that optimizes the criterion function. A couple of real applications of Bayesian optimal designs are also presented in Sections 5.3 and 5.4, including measuring the HIV dynamics model and heart defibrillator energy level. The Bayesian optimal designs found by CSO-MA greatly enhance experimental efficiency.

5.1 Longitudinal Mixed Models

Mixed-effects models are increasingly used in various fields, particularly in biomedical and public health studies. For example, longitudinal studies are increasingly analyzed using mixed-effects models. Another example is in pharmacokinetic and pharmacodynamic studies,

where data are invariably analyzed using nonlinear mixed models. In longitudinal studies, design issues may involve choosing the number of time points, the locations of the time points and how many subjects to assign to each of the time points. Additional constraints such as the varying cost of a study at various levels or ensuring a minimum spread between the time points can complicate the construction of an optimal design. On the other hand, sometimes, the time points are pre-selected for practical or medical reasons, and finding an appropriate design becomes simpler because there are fewer variables to optimize.

5.1.1 Fractional Polynomial Models

Suppose I wish to monitor the pulmonary function of the lungs of patients periodically in a clinical trial. A common measure is forced vital capacity, which is a continuous variable. Polynomial models are traditionally used to model the outcome over time, but increasingly, fractional polynomial models are used because they are more versatile and provide more flexibility to curve fitting, especially when the true relationship between the mean response and the explanatory factors is less smooth or can experience an abrupt but smooth change. (Royston and Wright, 1998; Long and Ryoo, 2010). Fractional polynomials are polynomials but are allowed to have positive and negative fractions in the power for each nominal. Royston and Altman (1994) proposed such models and suggested that it is adequate to select powers from the set $\mathcal{S} = \{-2, -1, -0.5, 0, 0.5, 1, 2, 3\}$. For this problem, I assume each time point can be chosen from a pre-determined discrete set $\mathcal{T} = \{1, 2, \cdots, T_{max}\}$, where T_{max} is given and I want to determine the number and sampling time points for each subject in an optimal way.

Consider a linear fractional polynomial model with a subject random effect given by

$$\begin{gathered} y_{ij} = \theta_0 + \theta_1 t_{ij}^{-2} + \theta_2 t_{ij}^{-1} + \theta_3 t_{ij}^{-1/2} + \theta_4 t_{ij}^{1/2} + \theta_5 t_{ij} + \theta_6 t_{ij}^2 + \theta_7 t_{ij}^3 + b_i + e_{ij}, \\ b_i \sim \mathcal{N}(0, \sigma_b^2),\ e_{ij} \sim \mathcal{N}(0, \sigma_e^2). \end{gathered} \tag{5.1}$$

Here t_{ij} refers to the time point of the j-th visit/measurement for individual i; b_i is the random intercept term assumed to have a normal distribution with zero mean and variance

σ_b^2; e_{ij} is an error term following a normal distribution with zero mean and variance σ_e^2. All random effects and error terms are assumed to be mutually independent of one another. My interest is to find a locally D-optimal design to estimate the model parameters $\boldsymbol{\theta} = (\theta_0, \cdots, \theta_7)$ by selecting the optimal subset of time points from $\mathcal{T}$ among all possible subsets from the pre-selected and discretized design space. Thus unlike all previous discussions, the design space is not an interval and consists of all possible subsets of $\mathcal{T}$. Consequently, the sensitivity plots displayed below for an approximate design are not the usual ones seen in a typical design paper or monograph.

Suppose $\boldsymbol{\eta}$ is a design with n design points $\eta_1, \cdots, \eta_n$ and each design point η_k has n_k time points $(t_{k1}, \cdots, t_{kn_k})$. Let the weight for each subset of design points from $\mathcal{T}$ for the design η_k be w_j with $\sum_{j=1}^{n} w_j = 1$. Let $\mathbf{J}_{n_k}$ be a $n_k \times n_k$ square matrix where every element is equal to one; let $\mathbf{I}_{n_k}$ be an identity matrix of dimension n_k, $SR = \sigma_b^2/\sigma_e^2$ and let

$$\mathbf{T}_k = \begin{pmatrix} 1 & t_{k1}^{-2} & t_{k1}^{-1} & \cdots & t_{k1}^{3} \\ 1 & t_{k2}^{-2} & t_{k2}^{-1} & \cdots & t_{k2}^{3} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & t_{kn_k}^{-2} & t_{kn_k}^{-1} & \cdots & t_{kn_k}^{3} \end{pmatrix}.$$

A direct calculation shows

$$\begin{aligned} \mathbf{M}(\eta_k) &= \mathbf{T}_k^T \boldsymbol{\Sigma}_k^{-1} \mathbf{T}_k \\ &= \frac{1}{\sigma_e^2} \mathbf{T}_k^T (\mathbf{I}_{n_k} + \frac{\sigma_b^2}{\sigma_e^2} \mathbf{J}_{n_k})^{-1} \mathbf{T}_k \\ &= \frac{1}{\sigma_e^2} \mathbf{T}_k^T (\mathbf{I}_{n_k} - \frac{1}{1 + n_k \frac{\sigma_b^2}{\sigma_e^2}} \frac{\sigma_b^2}{\sigma_e^2} \mathbf{J}_{n_k}) \mathbf{T}_k \\ &= \frac{1}{\sigma_e^2} \mathbf{T}_k^T (\mathbf{I}_{n_k} - \frac{SR}{1 + n_k SR} \mathbf{J}_{n_k}) \mathbf{T}_k. \end{aligned}$$

In practice, costs are incurred when we observe a measurement from the study and they may vary depending on which time point the observation is taken. More generally, suppose the cost of implementing design η_k is given by a known function $c(\eta_k)$ apart from parameters. Then I normalize the information matrix by its cost function and work with the normalized

information matrix $\mathbf{M}^*(\boldsymbol{\eta})$ and denote it by

$$\mathbf{M}^*(\boldsymbol{\eta}) = \sum_{k=1}^{n} w_k \frac{\mathbf{M}(\eta_k)}{c(\eta_k)}.$$

The advantages of incorporating cost into the design have been discussed in Gagnon and Leonov (2004) and Tekle et al. (2008b) and some commonly-used cost functions were discussed in Zhou et al. (2018). For example, one may consider a linear cost function given by

$$c(\eta_k) = \alpha_1 + \alpha_2 n_k,$$

where the user-defined parameters α_1, α_2 influence the cost of having n_k visits or measurements. The normalized information matrix is reminiscent of the common case seen in optimal design literature when errors are heteroscedastic and the inverse of the error variance at a point is represented similarly in the normalized matrix. When the cost structure is linear, it is straightforward to observe that the optimal design depends on the ratio of the two parameters $r = \alpha_1/\alpha_2$ and not on the values of α_1 and α_2. In the following subsections, I apply CSO-MA to generate designs for estimating parameters in various mixed models and some include cost considerations. All generated approximate designs have been verified to be optimal and for space consideration, I only show some of their sensitivity plots, which confirm the optimality of the CSO-MA-generated designs. I use 256 particles and set $\phi = 0.1$ for the following five examples.

Example 1. If $SR = 2$, $r = 0.5$, and $T_{max} = 10$, CSO-MA-generated design $\boldsymbol{\eta}_1$ is

$$\boldsymbol{\eta}_1 = \begin{pmatrix} \{1, 2, 3, 5, 8, 10\} & 0.832 \\ \{1, 2, 3, 6, 8, 10\} & 0.168 \end{pmatrix}.$$

It has a D-criterion value of -33.200 and CSO-MA takes 2.0 seconds to find it. This generated design has two design points, requiring about 83% of subjects have six measurements at times 1, 2, 3, 5, 8, 10 and the other 17% of subjects have six measurements at times 1, 2, 3, 6, 8, 10. This design strategy guarantees that statistical inference for all the fixed

parameters are estimated with maximum efficiency.

Example 2. If $SR = 2$, $r = 5$, and $T_{max} = 10$, CSO-MA-generated design $\boldsymbol{\eta}_2$ is

$$\boldsymbol{\eta}_2 = \begin{pmatrix} \{1,2,4,6,8,9,10\} & 0.100 \\ \{1,2,3,5,6,8,9,10\} & 0.260 \\ \{1,2,3,4,6,8,10\} & 0.275 \\ \{1,2,3,5,6,8,9,10\} & 0.365 \end{pmatrix},$$

which can be interpreted the same way as the first case. It has a D-criterion value of -36.141 and CSO-MA takes 2.0 seconds to find it.

These two examples seem to suggest that a larger value of r requires subjects to be observed more times.

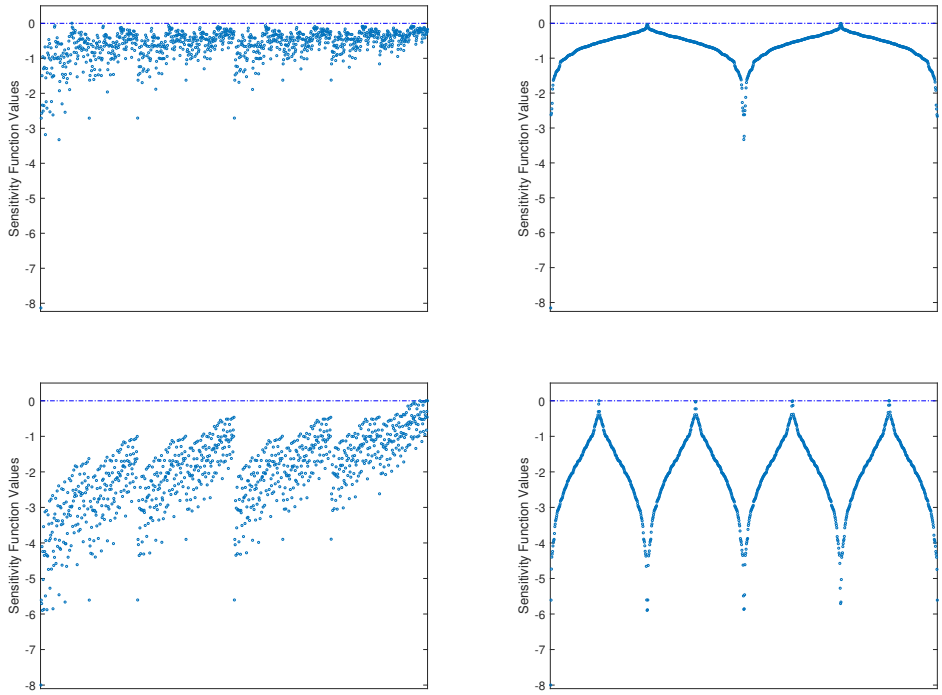

Figure 5.1: The sensitivity functions of the CSO-MA-generated designs in *Example 1* (first row) and *Example 2* (second row) versus the design space comprising all subsets of possible time points when they are appropriately ordered (right) and when they are not (left).

The sensitivity functions of the two CSO-MA-generated designs for the two examples are displayed in Figure 5.1. Suppose the time unit is an hour and since the time interval is

divided into $\{1, 2, \cdots, 10\}$, there are 1023 possible sets of time points to observe a subject over the 10-hour period, i.e., $\{1\}$, $\{2\}$, $\{3\}$, $\cdots$, $\{10\}$, $\{1,2\}$, $\{1,3\}$, $\cdots$, $\{1,2,\cdots,10\}$. This means, for example, that the first choice requires the subject to be observed once at the end of the first 1 hour, and the last choice requires that the subject to be observed every hour for 10 hours. The optimal approximate design selects what percentage of subjects to be observed at different sets of time points and what the sets of time points are. To construct the plot, I first order the sets of time points according to their values of the sensitivity function and then plot the function across the ordered 1023 sets of time points so that the pattern becomes visibly clear. Two plots confirm both designs' D-optimality.

Example 3. Fractional Polynomial Models with Correlated Random Coefficients

I show that CSO-MA can generate optimal designs for models with multiple correlated random effects. The example is illustrative in that the nominal model parameters and the covariance matrix of the random effects are arbitrarily selected. Different from the previous examples, I assume observations for each subject can only be taken in a continuous time interval $t \in [1, 10]$ and this study requires each subject to have exactly four observations/measurements.

This model contains more random effects and is given by

$$\begin{aligned} y_{ij} &= \theta_0 + \theta_1 t_{ij}^{-1/3} + \theta_2 t_{ij}^{-1/2} + \theta_3 t_{ij}^{1/3} + \theta_4 t_{ij}^{1/2} + \theta_5 t_{ij} + \theta_6 t_{ij}^2 + e_{ij}, \\ (\theta_0, \cdots, \theta_6)' &\sim \mathcal{N}(\mathbf{0}, \boldsymbol{D}),\ e_{ij} \sim \mathcal{N}(0, \sigma_e^2), \end{aligned} \tag{5.2}$$

where

$$\boldsymbol{D} = Blockdiag(\boldsymbol{D}_1, \boldsymbol{D}_2),$$

$$\boldsymbol{D}_1 = \begin{pmatrix} 1.0 & 0.8 & 0.4 \\ 0.8 & 1.2 & 0,5 \\ 0.4 & 0.5 & 1.9 \end{pmatrix}, \boldsymbol{D}_2 = \begin{pmatrix} 1.3 & 0.6 & 0.6 & 0.3 \\ 0.6 & 1.2 & 0,7 & 0.4 \\ 0.6 & 0.7 & 1.3 & 0.3 \\ 0.3 & 0.4 & 0.3 & 1.0 \end{pmatrix}.$$

The information matrix for this model can be derived in a similar manner as in Schmelter et al. (2007). Since σ_e^2 does not affect the optimization process, I assume it equal to 1. The D-criterion value of the CSO-MA-generated locally design is -60.583 and the design is

$$\boldsymbol{\eta}_3 = \begin{pmatrix} \{1.878, 4.224, 8.297, 10.000\} & 0.347 \\ \{1.000, 1.254, 2.015, 3.910\} & 0.311 \\ \{1.282, 3.000, 6.070, 9.053\} & 0.189 \\ \{1.000, 3.116, 6.132, 9.114\} & 0.153 \end{pmatrix}.$$

This design requires that the total sample of patients be divided unequally into four groups, each with a different time schedule. For example, one group has 34.7% of the patients and each patient is observed at the time points $1.878, 4.224, 8.297$ and 10.000. The other groups are similarly interpreted for the design. When implemented, this design maximizes the D-efficiency for estimating model parameters.

5.1.2 Logistic Regression Models

Binary outcomes are ubiquitous and a logistic mixed model is commonly used to model the binary longitudinal data. One example is when we take longitudinal measurements on some physiological outcomes, such as examining subjects' muscular strength and endurance by examining whether the subject passes a push-up test. The probability of passing the test for individual i at the time point t_{ij} can be described by a logistic model with a random intercept term. As before, I assume each time point should be chosen from a given discrete set $\mathcal{T} = \{1, 2, \cdots, T_{max}\}$.

Suppose that i-th subject has n_i measurements at $t_{i1}, \cdots, t_{in_i}$ in the study and p_{ij} is the probability at time t_{ij}, the model is

$$p_{ij} = \frac{\exp(\theta_0 + \theta_1 t_{ij} + b_i)}{1 + \exp(\theta_0 + \theta_1 t_{ij} + b_i)},\ b_i \sim \mathcal{N}(0, \sigma_b^2),\ j = 1, \cdots, n_i,$$

where b_i is the random intercept term and is normally distributed. The research question

is to find optimal time points for each patient to estimate the parameters θ_0 and θ_1 as accurately as possible.

The log-likelihood function of the above model does not have a closed form and the information matrix thus cannot be derived analytically. One solution is to use a first-order penalized quasi-likelihood (PQL1) for approximating the true likelihood function (Breslow and Clayton, 1993; Jang and Lim, 2009; Abebe et al., 2014). By using the PQL1 method and assuming a n-point design $\boldsymbol{\eta}$ with each design point η_i having n_i time points/measurements, the information matrix can be approximated as

$$\mathbf{M} \approx \sum_{i=1}^{n} \mathbf{T}_i^T \mathbf{V}_i^{-1} \mathbf{T}_i, \ \mathbf{V}_i \approx \mathbf{W}_i^{-1} + \sigma_b^2 \mathbf{J}_{n_i},$$

where $\mathbf{T}_i$ is the i-th design matrix; $\mathbf{J}_{n_i}$ is a n_i-dimensional square matrix where every element is equal to one; $\mathbf{W}_i$ is a diagonal matrix given by

$$\mathbf{W}_i = diag[\mathbf{Var}(y_{i1}|b_i), \mathbf{Var}(y_{i2}|b_i), \cdots, \mathbf{Var}(y_{in_i}|b_i)].$$

Example 4. Logistic Mixed Models with Fractional Polynomials

The linear predictor is now a fractional polynomial in the logistic model and is given by

$$p_{ij} = \frac{\exp(\theta_0 + \theta_1 t_{ij}^{-2} + \theta_2 t_{ij}^{-1} + \theta_3 t_{ij}^{-1/2} + \theta_4 t_{ij} + b_i)}{1 + \exp(\theta_0 + \theta_1 t_{ij}^{-2} + \theta_2 t_{ij}^{-1} + \theta_3 t_{ij}^{-1/2} + \theta_4 t_{ij} + b_i)}, \ b_i \sim \mathcal{N}(0, \sigma_b^2).$$

After the information matrix is approximated by PQL1 method, CSO-MA can search for the optimal design. For example, I assume the nominal parameter values are: $\sigma_b^2 = 0.2$ and $\boldsymbol{\theta} = (1.0, 0.2, -3.0, 0.5, -1.2)$. Two relative cost coefficients are considered, which are $r = 6$ and $r = 0.3$. Moreover, time points are chosen from the set $\{1, 2, \cdots, 6\}$. D-optimal design $\boldsymbol{\eta}_{4\text{-}1}$ found by CSO-MA with $r = 6$ is

$$\boldsymbol{\eta}_{4\text{-}1} = \begin{pmatrix} \{1, 2, 3, 4, 6\} & 0.500 \\ \{1, 2, 3, 4, 5, 6\} & 0.500 \end{pmatrix},$$

and its criterion value is -49.381. The second D-optimal design $\boldsymbol{\eta}_{4\text{-}2}$ with $r = 0.3$ is

$$\boldsymbol{\eta}_{4\text{-}2} = \begin{pmatrix} \{1, 2, 3, 4, 6\} & 1.000 \end{pmatrix},$$

and its criterion value is -45.876.

These two optimal designs also suggest that when r is large, each design point is more likely to include more time points.

5.1.3 Negative Binomial Regression Models

To demonstrate the flexibility of my approach, I now design for a count model with mixed effects in a longitudinal study. The negative binomial regression model is a flexible model as it can be used to model over-dispersed or under-dispersed data in a clinical trial. The count variable can be the number of new flares in Scleroderma patients or the number of new lesions in patients after a new treatment regimen over a period of time. Interestingly, Healy et al. (2010) found an optimal design for a phase I/II clinical trial for treating multiple sclerosis with gadolinium-enhanced lesions as the endpoint. However, their approach is devoid of optimal design theory and the design was selected among a few candidate designs based entirely on simulated error rates.

Unlike previous examples, I add a constraint that all subjects need to be observed T times and T is user-selected. Such a constraint can arise in situations when, for example, taking observations are either laborious or expensive or even risky in pediatric trials where only a limited number of measurements are allowed for the young subjects. In what is to follow, I show CSO-MA can directly accommodate such a constraint without difficulties.

In the two-drug trial, I denote the combinations of the drug treatments by $\mathbf{x}_i$ randomly assigned to the i-th subject and x_{ik} is the dose level of drug k. I assume all the drug levels have been normalized to $[-1, 1]$ and once a drug level is determined for a patient, it remains unchanged throughout the trial. For administration purposes, each patient is required to observe T times given the space $\{1, 2, 3, \cdots, T_{max}\}$. A negative binomial regression model is

used to study the effects of explanatory factors on the count outcome. The count outcome may be the number of allergy reactions after each treatment or the number of new lesions occurring after each treatment as in Healy et al. (2010). I choose the negative binomial regression model over the commonly-used Poisson regression model because the former is flexible and can capture under or over-dispersion.

The model of interest is an additive negative binomial mixed model with three variables including a time trend variable

$$\begin{gathered}\log \mu_{ij} = \theta_0 + \theta_1 x_{i1} + \theta_2 x_{i2} + \theta_3 t_{ij} + b_i,\ b_i \sim \mathcal{N}(0, \sigma_b^2),\ j = 1, \cdots, T, \\ \mathbf{E}(y_{ij}) = \mu_{ij},\ \mathbf{Var}(y_{ij}) = \mu_{ij} + a\mu_{ij}.\end{gathered} \tag{5.3}$$

Here the parameter a is the dispersion factor and if it is positive, it suggests that the data is over-dispersed which is usually the case for real data. The information matrix of model (5.3) can also be approximated by the PQL1 method.

Example 5. c-optimal Approximate Design for a Negative Binomial Mixed Model

Each subject in the clinical trial receives a combination dose from the two drugs and each subject is observed $T = 3$ times. The goals are to find an optimal design to determine what combination doses and which 3 time points are best for answering the question: are the two drugs equally effective if the same dosage levels are given to the patients?

I find the locally c-optimal approximate design with $\mathbf{c} = (0, 1, -1, 0)$. For illustrative purposes, I assume the nominal model parameters are $(\theta_0, \theta_1, \theta_2) = (0.3, 1.0, -0.5)$, $\theta_3 = 1.1$, $a = 1.2$, $\sigma_b^2 = 0.5$, $T_{max} = 6$ and $T = 3$. The ranges of values for the two independent variables are $x_1 \in [-1, 1]$, $x_2 \in [-1, 1]$, after their dosage levels are appropriately scaled. The c-optimal design found by CSO-MA is

$$\boldsymbol{\eta}_5 = \begin{pmatrix} x_1 & x_2 & t_1 & t_2 & t_3 & w \\ -1.000 & 1.000 & 2 & 5 & 6 & 0.500 \\ 1.000 & -1.000 & 1 & 2 & 6 & 0.500 \end{pmatrix},$$

and its criterion value is 0.501.

CSO-MA-generated design $\boldsymbol{\eta}_5$ randomly assigns an equal number of patients to two-dose combinations of the two drugs after their dosages have been normalized to [-1, 1]. Suppose now the time unit is a week, responses from each patient are observed 3 times, once per week, out of the 6 possible weeks. The first group received a combination of the extreme doses from the two drugs and the second group receives the other set of extreme doses from the two drugs. One group is observed at the 1-st, 2-nd and 6-th week, and the other group is observed at the 2-nd, 5-th and 6-th week.

5.2 *D*-optimal Designs for Poisson Regression Models with Random Coefficients

Naderi et al. (2018) discussed finding locally D-optimal designs for two-factor Poisson regression models with random coefficients. Both the cases with and without an interaction term were discussed. These models are

$$Y_{ij} \sim \mathbf{P}(\lambda_{ij}),\ \lambda_{ij} = \exp(\theta_{0j} + \theta_{1j}x_{i1} + \theta_{2j}x_{i2}), \tag{5.4}$$

$$\text{and } Y_{ij} \sim \mathbf{P}(\lambda_{ij}),\ \lambda_{ij} = \exp(\theta_{0j} + \theta_{1j}x_{i1} + \theta_{2j}x_{i2} + \theta_{3j}x_{i1}x_{i2}), \tag{5.5}$$

$$i = 1, \cdots, n;\ j = 1, \cdots, m,$$

where Y_{ij} is the j-th replication for the individual i at the experimental setting $\mathbf{x}_i$; θ_{0j} and $\theta_{1j}, \theta_{2j}, \theta_{3j}$ are random effects from a multivariate normal distribution whose mean vector is $\boldsymbol{\theta}$ and covariance matrix is $\boldsymbol{\Sigma}$. It can be shown that the Fisher information matrix for above models based on a k-point design $\boldsymbol{\eta}$ can be expressed as

$$\boldsymbol{M}(\boldsymbol{\eta}) = \boldsymbol{F}_{\boldsymbol{\eta}}^T(\boldsymbol{A}_{\boldsymbol{\eta}}^{-1} + \boldsymbol{C}_{\boldsymbol{\eta}})^{-1}\boldsymbol{F}_{\boldsymbol{\eta}}, \tag{5.6}$$

where

$$\boldsymbol{F}_{\boldsymbol{\eta}} = \begin{pmatrix} \boldsymbol{F}_{\boldsymbol{\eta},1} \\ \vdots \\ \boldsymbol{F}_{\boldsymbol{\eta},k} \end{pmatrix}, \ \boldsymbol{F}_{\boldsymbol{\eta},i} = \underbrace{\begin{pmatrix} 1 & \mathbf{x}_i \\ 1 & \mathbf{x}_i \\ \vdots & \vdots \\ 1 & \mathbf{x}_i \end{pmatrix}}_{m \text{ rows}}$$

$$\boldsymbol{A}_{\boldsymbol{\eta}} = Blockdiag[w_1\lambda_1 \exp(\frac{1}{2}\boldsymbol{F}_{\boldsymbol{\eta},1}\boldsymbol{\Sigma}\boldsymbol{F}_{\boldsymbol{\eta},1}^T), \cdots, w_k\lambda_k \exp(\frac{1}{2}\boldsymbol{F}_{\boldsymbol{\eta},k}\boldsymbol{\Sigma}\boldsymbol{F}_{\boldsymbol{\eta},k}^T)],$$

$$\lambda_i = \exp(\boldsymbol{\theta}^T\mathbf{x}_i) \ \text{ and } \ \boldsymbol{C}_{\boldsymbol{\eta}} = \exp(\boldsymbol{F}_{\boldsymbol{\eta}}\boldsymbol{\Sigma}\boldsymbol{F}_{\boldsymbol{\eta}}^T) - 1.$$

Naderi et al. (2018) found the D-optimal designs on a selected design space, along with some technical restrictions that may not apply in practice. It is not clear if his/her methodology still works well if the design space is scaled differently or the restrictions are modified or removed. For example, when deriving Theorem 3.1, they supposed that $q_{1ij} = \exp(\theta_{1j}x_{i1})$, $q_{2ij} = \exp(\theta_{2j}x_{i2})$ and imposed constraints that $0 \leq q_{1ij} \leq 1$, $0 \leq q_{2ij} \leq 1$. These are strong conditions for the results to hold.

If we look at the D-optimal design in the fourth row of Table 1 in Naderi et al. (2018), it satisfies that $q_{2ij} = 0.113$. If $\theta_{2j} = 0.1$, we have $x_{i2} = -21.778$ (the second coordinate of the first support point), which means the design space along x_2 axis should cover -21.778. If, unluckily, the real design space for x_2, say, is equal to $[-5, 0]$, then the results from Theorem 3.1 are no longer valid. Therefore, this is a limitation of their result.

CSO-MA can quickly find D-optimal designs for the above models with and without constraints on covariance matrix or on the design space. First, I use CSO-MA to verify the results provided in Naderi et al. (2018) (Table 5.1). Additionally, I apply CSO-MA and generate the D-optimal designs for models with independent or correlated random effects. They are shown in Table 5.2.

The take home message from this subsection is that CSO-MA is a flexible algorithm and can find different types of optimal designs quickly. It does not require technical assumptions or limiting restrictions on optimization problems for it to work well.

Model	$\boldsymbol{\theta}$	$\boldsymbol{\Sigma}$	Design space	D-optimal design	$\log \lvert \boldsymbol{M} \rvert$
(5.4)	$(\log \frac{0.5}{e-1} - 0.5,\ 2,\ 3)$	$diag(1, 0, 0)$	$[-4, 0]^2$	$\begin{pmatrix} -1.019 & 0.000 & 0.000 \\ 0.000 & -0.679 & 0.000 \\ 0.346 & 0.346 & 308 \end{pmatrix}$	-11.992
(5.4)	$(\log \frac{10}{e-1} - 0.5,\ 2,\ 3)$	$diag(1, 0, 0)$	$[-5, 0]^2$	$\begin{pmatrix} -1.120 & 0.000 & 0.000 \\ 0.000 & -0.747 & 0.000 \\ 0.396 & 0.396 & 208 \end{pmatrix}$	-4.343

Table 5.1: The D-optimal designs for Poisson mixed models shown in Naderi et al. (2018).

Model	$\boldsymbol{\theta}$	$\boldsymbol{\Sigma}$	Design space	D-optimal design	$\log \lvert \boldsymbol{M} \rvert$
(5.4)	$(\log \frac{5}{e-1} - 0.5,\ 1,\ 1)$	$diag(1, 0, 0)$	$[-1, 1]^2$	$\begin{pmatrix} -1.000 & 1.000 & 1.000 \\ 1.000 & -1.000 & 1.000 \\ 0.399 & 0.399 & 0.202 \end{pmatrix}$	2.018
(5.4)	(-1, 2, -3)	$\begin{pmatrix} 1.0 & 0.2 & 0.0 \\ 0.2 & 0.3 & 0.0 \\ 0.0 & 0.0 & 0.0 \end{pmatrix}$	$[-1, 1]^2$	$\begin{pmatrix} -0.441 & 1.000 & 1.000 \\ -1.000 & -0.153 & -1.000 \\ 0.311 & 0.536 & 0.153 \end{pmatrix}$	-0.058
(5.5)	(-0.5, 0.2, -0.3, 0.4)	$\begin{pmatrix} 0.4 & 0.2 & 0.1 & 0.0 \\ 0.2 & 1.1 & 0.6 & 0.0 \\ 0.1 & 0.6 & 1.2 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 \end{pmatrix}$	$[-1, 1]^2$	$\begin{pmatrix} -1.000 & -1.000 & -0.568 & 0.632 & 1.000 \\ -0.464 & 1.000 & -0.825 & 0.544 & -1.000 \\ 0.075 & 0.365 & 0.135 & 0.205 & 0.220 \end{pmatrix}$	4.237
(5.5)	(-0.5, 0.2, -0.3, 0.4)	$diag(0.3, 0.2, 0.5, 0.8)$	$[-1, 1]^2$	$\begin{pmatrix} -1.000 & -1.000 & -0.879 & 1.000 & 1.000 \\ -0.870 & 0.969 & -1.000 & -0.879 & 0.869 \\ 0.163 & 0.300 & 0.057 & 0.245 & 0.235 \end{pmatrix}$	-2.573

Table 5.2: The D-optimal designs for Poisson mixed models given different sets of model parameters.

5.3 Bayesian Optimal Design for Nonlinear Mixed Models Applied to HIV Dynamics

Finding Bayesian optimal designs for real applications are challenging because (i) the model is often more complex, (ii) the design criterion is expressed in terms of multiple integrals, which may require approximations using a random sampling scheme, and (iii) there is a general lack of an effective algorithm to optimize the design criterion.

In this subsection, I apply CSO-MA to search for Bayesian optimal exact designs for estimating two selected parameters in a nonlinear mixed model for an HIV study (Han et al., 2002). Eight candidate designs were available for implementation and all have 16 points. The goal was to determine which one of the eight designs has the best design criterion value for two prior distributions. Because the focus was on finding the best design among the eight candidate designs, they did not find the Bayesian optimal exact design.

Han et al. (2002) used a Bayesian approach to study plasma concentration in HIV patients under protease inhibitor monotherapy. The model is a nonlinear mixed model and the natural logarithm of the plasma concentration from subject i at time t_j after the pharmacologic delay

of the drug effect is y_{ij}, given by

$$y_{ij}|\boldsymbol{\theta}_i, t_j, \sigma^2 \sim \mathcal{N}(s(V_{0i}, c_i, \delta_i, t_j), \sigma^2),\ j = 1, \cdots, T,$$

$$\boldsymbol{\theta}_i = (\log V_{0i}, \log c_i, \log \delta_i)^T | V_0, c, \delta, \boldsymbol{\Sigma} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}),$$

where $\boldsymbol{\mu} = (\mu_1, \mu_2, \mu_3) = (\log V_0, \log c, \log \delta)^T$ and

$$s(V_{0i}, c_i, \delta_i, t_j) = \log V_{0i} + \log[\frac{c_i^2}{(c_i - \delta_i)^2} e^{-\delta_i t_j} - \frac{c_i^2 - (c_i\delta_i)^2}{(c_i - \delta_i)^2} e^{-c_i t_j} - \frac{c_i \delta_i}{c_i - \delta_i} t_j e^{-c_i t_j}].$$

Here V_{0i} is the plasma concentration of HIV particles at treatment initiation; c_i is the virion clearance rate and δ_i is the rate at which the infected CD4 cells die (Han and Chaloner, 2004). The expression of $s(V_{0i}, c_i, \delta_i, t_j)$ solves a system of differential equations that describe the transactions among virus particles, target cells, and infected cells (Nowak and May, 2000).

Han and Chaloner (2004) proposed a more complicated Bayesian model for studying HIV dynamics and employed it to evaluate candidate designs without finding the Bayesian optimal design. The model structure is

$$y_{ij}|\boldsymbol{\theta}_i, t_j, \sigma^2 \sim \mathcal{N}(s(\boldsymbol{\theta}_i, t_j), \sigma^2),\ j = 1, \cdots, T,$$

$$\boldsymbol{\theta}_i | \boldsymbol{\mu}, \boldsymbol{\Sigma} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}),\ i = 1, \cdots, n,$$

$$\sigma^{-2} \sim \mathcal{G}(\alpha, \theta),$$

$$\boldsymbol{\mu} \sim \mathcal{N}(\boldsymbol{\zeta}, \boldsymbol{\Lambda}),$$

$$\boldsymbol{\Sigma}^{-1} \sim \mathcal{W}(\boldsymbol{\Phi}, \gamma),$$

where $(\alpha, \theta, \boldsymbol{\zeta}, \boldsymbol{\Lambda}, \boldsymbol{\Phi}, \gamma)$ are hyperparameters. A total of T measurements are to be taken from each subject and I want to determine the optimal time points $t_1, \cdots, t_T$. The $\boldsymbol{\theta}_i$'s are random effects generated from $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Other prior distributions are defined accordingly.

The two main goals of the study are to estimate $\log c$ and $\log \delta$. To incorporate prior

information on the parameters, Bayesian design criteria were used to find time points $\mathbf{t} = (t_1, \cdots, t_T)$ that minimize $\mathbf{E}[\mathbf{Var}(\mu_2|\mathbf{y})]$ and $\mathbf{E}[\mathbf{Var}(\mu_3|\mathbf{y})]$, or equivalently, maximize the the utility functions, i.e., $-\mathbf{E}[\mathbf{Var}(\mu_2|\mathbf{y})]$ and $-\mathbf{E}[\mathbf{Var}(\mu_3|\mathbf{y})]$. Han and Chaloner (2004) compared eight 16-point candidate designs and their performances are reported in Table 1 of Han and Chaloner (2004). All design points are in $[0, 7]$, implying that the all measurements for the study have to be taken within seven days if the unit is a day.

Finding the Bayesian optimal design is not a trivial problem because we need to draw samples from the posterior distribution, which does not have an explicit form. The distribution of $\boldsymbol{\mu}|\mathbf{y}$ in this application cannot be obtained analytically and I resort to MCMC method to approximate the posterior distribution. Noting that

$$f(\boldsymbol{\mu}, \mathbf{y}) \propto \int f(\mathbf{y}|\boldsymbol{\mu}, \boldsymbol{\Sigma}^{-1}, \sigma^{-2}) f(\boldsymbol{\mu}) f(\boldsymbol{\Sigma}^{-1}) f(\sigma^{-2}) d\boldsymbol{\Sigma}^{-1} d\sigma^{-2}, \tag{5.7}$$

where $f(\mathbf{y}|\boldsymbol{\mu}, \boldsymbol{\Sigma}^{-1}, \sigma^{-2}) = \int f(\mathbf{y}|\boldsymbol{\theta}, \mathbf{t}) f(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma}^{-1}, \sigma^{-2}) d\boldsymbol{\theta}$. I use the Metropolis-Hasting algorithm to draw random samples from $f(\boldsymbol{\mu}|\mathbf{y})$ and to estimate $\mathbf{E}[\mathbf{Var}(\mu_2|\mathbf{y})]$ and $\mathbf{E}[\mathbf{Var}(\mu_3|\mathbf{y})]$. The algorithm for searching the Bayesian optimal design for this HIV dynamics model can be summarized as follows:

1. Given an input $\mathbf{t}$ and hyper parameters $(\alpha, \theta, \boldsymbol{\zeta}, \boldsymbol{\Lambda}, \boldsymbol{\Phi}, \gamma)$, I hierarchically draw random samples for $\boldsymbol{\mu}$, $\boldsymbol{\theta}$ and $\mathbf{y}$.

2. Given an observation $\mathbf{y}$, I use formula (5.7) and apply the M-H algorithm to estimate the posterior variance of $\boldsymbol{\mu}|\mathbf{y}$. A multivariate normal distribution is used as the proposal distribution, which is also the proposal distribution used in Han and Chaloner (2004).

3. I apply CSO-MA to search for the optimal design on the design space of $\mathbf{t}$.

In the original work, two prior distributions π_1, π_2 were used with π_1 defined by $\sigma^{-2} \sim \mathcal{G}(4.5, 9.0)$, $\boldsymbol{\mu} \sim \mathcal{N}((11.0, 1.1, -1.0)', \text{diag}(6.0, 0.1, 0.01))$ and $\boldsymbol{\Sigma}^{-1} \sim \mathcal{W}(\text{diag}(0.26, 2.5, 2.5), 3.0)$. The prior π_2 is the same as π_1 except that the prior variance for μ_3 is 0.001 instead of 0.01.

Table 5.3 displays the results by using the above method to calculate candidate designs' criterion values and they agree with the results reported in Table 2 of Han and Chaloner

(2004). For space considerations, I only report the results for candidate designs 1-4. The two CSO-MA-generated 16-point Bayesian exact designs under two prior distributions are shown in Table 5.4 and they have smaller criterion values than the proposed candidate designs.

	π_1		π_2	
	$\hat{\mathbf{E}}[\mathbf{Var}(\mu_2\|\mathbf{y})]$	**SD**	$\hat{\mathbf{E}}[\mathbf{Var}(\mu_2\|\mathbf{y})]$	**SD**
Candidate design 1	0.02915	0.0029	0.02968	0.0014
Candidate design 2	0.02882	0.0017	0.02874	0.0019
Candidate design 3	0.03257	0.0031	0.03315	0.0023
Candidate design 4	0.03003	0.0026	0.02946	0.0018
	$\hat{\mathbf{E}}[\mathbf{Var}(\mu_3\|\mathbf{y})]$	**SD**	$\hat{\mathbf{E}}[\mathbf{Var}(\mu_3\|\mathbf{y})]$	**SD**
Candidate design 1	0.005172	0.00030	0.01117	0.0014
Candidate design 2	0.005298	0.00014	0.01120	0.0019
Candidate design 3	0.005367	0.00012	0.01103	0.0023
Candidate design 4	0.005330	0.00009	0.01089	0.0011

Table 5.3: The values of $\mathbf{E}[\mathbf{Var}(\mu_2|\mathbf{y})]$ and $\mathbf{E}[\mathbf{Var}(\mu_3|\mathbf{y})]$ for the candidate designs 1-4 produced from the above three-step algorithm. The corresponding standard deviation (**SD**) of the criterion value averaged over ten repetitions is also reported. The results from the three-step algorithm are very close to the results in Han and Chaloner (2004).

Objective	Prior	Design								Estimate (**SD**)
$\mathbf{E}[\mathbf{Var}(\mu_2\|\mathbf{y})]$	π_1	1.580	2.297	2.500	2.678	2.855	2.932	3.011	3.594	0.0197 (0.0013)
		4.028	4.115	4.146	4.155	4.200	4.591	7.000	7.000	
$\mathbf{E}[\mathbf{Var}(\mu_2\|\mathbf{y})]$	π_2	0.000	0.017	0.473	1.224	1.490	1.901	1.997	2.400	0.0214 (0.0011)
		3.278	4.109	5.280	5.635	5.761	6.363	7.000	7.000	
$\mathbf{E}[\mathbf{Var}(\mu_3\|\mathbf{y})]$	π_1	0.404	0.542	0.810	1.008	2.899	3.125	3.220	3.979	0.0023 (0.0006)
		4.336	4.379	4.389	4.485	5.236	5.554	5.706	7.0000	
$\mathbf{E}[\mathbf{Var}(\mu_3\|\mathbf{y})]$	π_2	0.000	0.212	1.078	1.588	1.792	2.711	2.901	2.933	0.0026 (0.0004)
		3.072	3.583	3.600	3.818	3.950	4.062	4.561	6.575	

Table 5.4: The Bayesian optimal designs found by CSO-MA for minimizing $\mathbf{E}[\mathbf{Var}(\mu_2|\mathbf{y})]$ and $\mathbf{E}[\mathbf{Var}(\mu_3|\mathbf{y})]$ under prior π_1 and π_2, along with their estimated criterion values in the last column.

It is interesting to note that when I use the MATLAB function "fmincon" to search for these Bayesian optimal designs, I find that the function does not perform well for optimizing the problems at hand. For instance, when the "fmincon" function was called to find the Bayesian optimal design that minimizes $\mathbf{E}[\mathbf{Var}(\mu_3|\mathbf{y})]$ under π_1 and invoked sequential quadratic programming (SQP) algorithm, I observe that the SQP-generated design repeatedly has a significantly worse criterion value than CSO-MA-generated designs. This suggests

that nature-inspired metaheuristic algorithms provide a useful option when the mathematical programming approach fails.

5.4 Bayesian Optimal Design for a Hierarchical Logistic Model Applied to Heart Defibrillator Energy Level

Clyde et al. (1995) wanted to design a study to estimate the effective heart defibrillator energy level necessary for implanting defibrillation, which is used for restoring a normal heart rhythm. Proper choice of the energy level is crucial since it might cause serious injury or death to the recipient. On the other hand, having a too weak energy level for the defibrillation would also result in failure to restore the heart rhythm to the normal level. For this study, the range of energy levels was from 0.001 joules to 32 joules and the researchers were interested in having an eight-point design. A total of 900 patients were available for the study and each patient was tested at energy level $x_1, x_2, \cdots, x_8$. The binary variable y_{ij} for patient in the logistic model takes on the value 1 if the defibrillation was successful at test level x_i for patient j or the value 0 if it is not successful. A hierarchical model was proposed

$$
\begin{aligned}
p(y_{ij} = 1|\theta_j, \lambda_j) &= \frac{1}{1 + \exp[-\theta_j(x_{ij} - \lambda_j) - \log(0.95/0.05)]}, \\
\phi_j = (\log\theta_j, \log\lambda_j)|\boldsymbol{\mu}, \mathbf{V} &\sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{V}), \\
\boldsymbol{\mu} &\sim \mathcal{N}(\mathbf{m}, \mathbf{B}), \\
\mathbf{V}^{-1} &\sim \mathbf{W}(q, (q\mathbf{Q})^{-1}),
\end{aligned}
$$

where θ_j, λ_j are unknown parameters in the logistic model for patient j; parameter λ_j is the ED_{95}, the effective energy level that defibrillates 95% of the time, for patient j. The aim of this study was to find an eight-point design that minimizes the posterior variance of the estimate for ED_{95} under the squared error loss. The utility function was

$$
\int_{\mathbf{y}} \int_{\boldsymbol{\phi}} -\{\log(\lambda) - \mathbf{E}[\log(\lambda)|\mathbf{y}, \boldsymbol{\eta}]\}^2 p(\boldsymbol{\phi}, \mathbf{y}|\boldsymbol{\eta}) d\boldsymbol{\phi} d\mathbf{y}.
$$

Given the hyperparameters q, $\mathbf{Q}$, $\boldsymbol{\mu}$, $\mathbf{B}$, the optimal design found in Clyde et al. (1995) had an posterior variance 0.095. Following their modeling setup and using a similar sampling strategy I show in the last section, the design found by CSO-MA has a posterior variance 0.077 with design points 2.196, 3.620, 4.270, 5.288, 6.508, 7.550, 8.759 and 11.572. Therefore, the design found by CSO-MA can provide a more accurate estimation for the ED_{95} in this case compared to Clyde's design.

CHAPTER 6

Extensions and Conclusions

This chapter discusses two additional frontiers of research for addressing challenging design issues. In Section 6.1, I provide the theory of G-optimal design for estimating the entire response surface over the design space, extended D-optimal designs and show how CSO-MA can find them for hierarchical linear polynomial or fractional polynomial models. In Section 6.2, I use a few examples and demonstrate that CSO-MA is also effective for finding optimal exact designs. In addition, I mention that the routines available in commercial statistical packages, such as JMP or SAS, are limited in scope and they can under-perform CSO-MA in some situations. Section 6.3 concludes with some ongoing work and a future research plan.

6.1 G and Extended D-optimal Designs for Hierarchical Linear Models

Multi-level linear models or hierarchical linear model are frequently used in the educational arena and in the biomedical and life sciences. For example, a typical application is to evaluate the effectiveness of a new teaching method versus a conventional method and the intervention is delivered at the classroom level so that all students in the class receive the same treatment. Likewise, Bastani et al. (2007) carried out a two-arm cluster randomized cancer control and prevention trial to assess whether an intervention method that provides information of Hepatitis B to the subjects versus health information in the other group was effective in getting subjects tested six months after the intervention. The outcome is binary, whether a subject received the test or not. For a continuous outcome measurement observed

over time, y_{ij}, a common hierarchical linear model to study its changes over time is

$$y_{ij} = f^T(x_j)\boldsymbol{\theta}_i + \epsilon_{ij}. \tag{6.1}$$

Here the linear regression function $f(x)$ is user-selected; $\mathbf{E}(\boldsymbol{\theta}_i) = \boldsymbol{\theta}$, $\mathbf{Cov}(\boldsymbol{\theta}_i) = \sigma^2\mathbf{D}$, $j = 1, \cdots, m$; $i = 1, \cdots, n$; term ϵ_{ij} is a normally distributed error with mean 0 and standard deviation σ.

The quality of the estimate quality from design $\boldsymbol{\eta}$ can be measured by the mean squared error matrix of $(\hat{\boldsymbol{\theta}}_1, \cdots, \hat{\boldsymbol{\theta}}_n)$

$$MSE(\boldsymbol{\eta}) = \frac{\sigma^2}{m}\{\frac{1}{n}\mathbf{J}_n \otimes \boldsymbol{M}^{-1}(\boldsymbol{\eta}) + (\mathbf{I}_n - \frac{1}{n}\mathbf{J}_n) \otimes [\boldsymbol{\Delta} - \boldsymbol{\Delta}(\boldsymbol{M}^{-1}(\boldsymbol{\eta}) + \boldsymbol{\Delta})^{-1}\boldsymbol{\Delta}]\},$$

where $\boldsymbol{M}$ is the information matrix; $\mathbf{J}_n$ is a n-dimensional square matrix where every element is equal to one and $\mathbf{I}_n$ is a n-dimensional identity matrix. Prus and Schwabe (2016) proposed to search for the extended D-optimal design which minimizes the determinant of the MSE matrix. An equivalence theorem was also derived to confirm a design's extended D-optimality. Let $\boldsymbol{\Delta} = m\boldsymbol{D}$ and define

$$\phi(x, \boldsymbol{\eta}) = f^T(x)\boldsymbol{M}^{-1}(\boldsymbol{\eta})f(x) + (n-1)f^T(x)[\boldsymbol{\Delta} - \boldsymbol{\Delta}(\boldsymbol{M}^{-1}(\boldsymbol{\eta}) + \boldsymbol{\Delta})^{-1}\boldsymbol{\Delta}]f(x). \tag{6.2}$$

Prus and Schwabe (2016) showed a design $\boldsymbol{\eta}^*$ is extended D-optimal if and only if

$$\max_x \phi(x, \boldsymbol{\eta}^*) = p + (n-1)trace\{[\boldsymbol{M}^{-1}(\boldsymbol{\eta}^*) + \boldsymbol{\Delta}]^{-1}\},$$

where the the maximum is achieved at every support points of $\boldsymbol{\eta}^*$.

In addition, one can search for the G-optimal design that minimizes the criterion function $\max_x \phi(x, \boldsymbol{\eta})$. Wong and Cook (1993) provided an equivalence theorem for G-optimal design to minimize the maximal variance of the fitted response over the design space when there is heteroscedasticity in a linear model and it is more complicated than that for D-optimality. Here, an equivalence theorem can also be obtained as follows. First, define the answering

set for a design $\boldsymbol{\eta}$ by

$$\mathcal{A}(\boldsymbol{\eta}) = \left\{ x | \phi(x, \boldsymbol{\eta}) = \max_{z} \phi(z, \boldsymbol{\eta}) \right\},$$

and let

$$\phi_G(x, \boldsymbol{\eta}) = f^T(x)\boldsymbol{M}^{-1}(\boldsymbol{\eta})\boldsymbol{M}_{\mathcal{A}}(\mu)\boldsymbol{M}^{-1}(\boldsymbol{\eta})f(x) + (n-1)f^T(x)\boldsymbol{N}(\boldsymbol{\eta}, \boldsymbol{\Delta})\boldsymbol{M}_{\mathcal{A}}(\mu)\boldsymbol{N}(\boldsymbol{\eta}, \boldsymbol{\Delta})f(x),$$

where $\boldsymbol{N}(\boldsymbol{\eta}, \boldsymbol{\Delta}) = \boldsymbol{\Delta} - \boldsymbol{\Delta}(\boldsymbol{M}^{-1}(\boldsymbol{\eta}) + \boldsymbol{\Delta})^{-1}\boldsymbol{\Delta}$, μ is a probability measure on $\mathcal{A}(\boldsymbol{\eta})$ and

$$\boldsymbol{M}_{\mathcal{A}}(\mu) = \int_{\mathcal{A}(\boldsymbol{\eta})} f(x)f^T(x)d\mu.$$

Then a design $\boldsymbol{\eta}^*$ is G-optimal if and only if there exists a probability measure μ^* on $\mathcal{A}(\boldsymbol{\eta}^*)$ such that

$$\max_{x} \ \phi_G(x, \boldsymbol{\eta}^*) = trace[\boldsymbol{M}_{\mathcal{A}}(\mu^*)\boldsymbol{M}^{-1}(\boldsymbol{\eta}^*) + (n-1)\boldsymbol{M}(\boldsymbol{\eta}^*)\boldsymbol{N}(\boldsymbol{\eta}^*, \boldsymbol{\Delta})\boldsymbol{M}_{\mathcal{A}}(\mu^*)\boldsymbol{N}(\boldsymbol{\eta}^*, \boldsymbol{\Delta})].$$

Moreover, the maximum is achieved at every design point of $\boldsymbol{\eta}^*$.

Compared to finding a traditional D-optimal design, finding the extended D-optimal design is computationally similar except that the information matrix is replaced by the MSE matrix. However, finding the G-optimal design is more challenging because there is more than one layer of optimization involved.

To search for a G-optimal design, the inner optimization step is a low-dimensional maximization problem represented in Equation (6.2) and the outer loop is to minimize the obtained maximum. Accordingly, In the CSO-MA algorithm, I set $\phi = 0$, 32 particles, 128 iterations for the inner optimization and $\phi = 0$, 128 particles and 200 iterations for the outer optimization task.

After finding a design $\boldsymbol{\eta}^*$, I determine its answering set $\mathcal{A}(\boldsymbol{\eta}^*)$ and the probability measure μ^* that meets the conditions in the equivalence theorem. To efficiently find all x's that maximize $\phi(x, \boldsymbol{\eta}^*)$, I split the design space into two or more subspaces and then run CSO-

MA on each subspace to search for all x's that maximize $\phi(x, \boldsymbol{\eta}^*)$. For instance, if the design space $\mathcal{X} = [0, 1]$, I split it to $[0, 0.5)$ and $[0.5, 1]$. There are no firm rules for the number of subspaces and my suggestion is to first try with two subspaces, then aggregate all such points to obtain $\mathcal{A}(\boldsymbol{\eta}^*)$. I then sequentially increase the number of subspaces and stop the process when splitting the design space into more subspace does not enlarge the size of $\mathcal{A}(\boldsymbol{\eta}^*)$.

To find the probability measure μ^* that meets the equivalence theorem conditions, I have to determine the values of $k,\ x_1,\ \cdots,\ x_k$ and the corresponding weight w_j at each x_j in μ^* subject to $w_1 + \cdots + x_k = 1$. To this end, remembering that CSO-MA is a general purpose optimization algorithm, I apply CSO-MA to minimize the following function with respect to the variable weights w_i in μ^*

$$\begin{aligned}&\{\max_x\ \phi_G(x, \boldsymbol{\eta}^*) - trace[\boldsymbol{M}_{\mathcal{A}}(\mu^*)\boldsymbol{M}^{-1}(\boldsymbol{\eta}^*) + (n-1)\boldsymbol{N}(\boldsymbol{\eta}^*, \boldsymbol{\Delta})\mathbf{M}_{\mathcal{A}}(\mu^*)\boldsymbol{N}(\boldsymbol{\eta}^*, \boldsymbol{\Delta})]\}^2 \\ &+ \sum_{j=1}^{k} w_j\{\phi_G(x_j, \boldsymbol{\eta}^*)\ - trace[\mathbf{M}_{\mathcal{A}}(\mu^*)\boldsymbol{M}^{-1}(\boldsymbol{\eta}^*) + (n-1)\boldsymbol{N}(\boldsymbol{\eta}^*, \boldsymbol{\Delta})\mathbf{M}_{\mathcal{A}}(\mu^*)\boldsymbol{N}(\boldsymbol{\eta}^*, \boldsymbol{\Delta})]\}^2.\end{aligned} \tag{6.3}$$

One problem using Equation (6.3) is that for each design $\boldsymbol{\eta}^*$, I still have to find x that maximizes $\phi_G(x, \boldsymbol{\eta}^*)$. This two-layer optimization problem can also be tackled using CSO-MA again. My experience after lots of experiments with solving the problem is that it seems helpful to first minimize the second term of Equation (6.3). Frequently, this simplified approach suffices to obtain the desired μ^* and also reduce the computational time noticeably. Specifically, this means that I find μ^* by minimizing

$$\sum_{j=1}^{k} w_j\{\phi_G(x_j, \boldsymbol{\eta}^*)\ - trace[\mathbf{M}_{\mathcal{A}}(\mu^*)\boldsymbol{M}^{-1}(\boldsymbol{\eta}^*) + (n-1)\boldsymbol{N}(\boldsymbol{\eta}^*, \boldsymbol{\Delta})\mathbf{M}_{\mathcal{A}}(\mu^*)\boldsymbol{N}(\boldsymbol{\eta}^*, \boldsymbol{\Delta})]\}^2. \tag{6.4}$$

If optimizing Equation (6.4), as a first step, does not produce the desired μ^*, I would then proceed to optimize Equation (6.3). At this step, I set $\phi = 0.05$, use 64 particles and 1200 iterations to run CSO-MA and suggest that the user can stop running the algorithm at

any time once the objective value is smaller than, say, 10^{-5}. This process can usually be completed in 5 seconds.

The next few tables list G-optimal designs for various linear mixed models with different assumptions on the design space and the structure of the random effects, along with their G-optimality criterion value and a figure showing the plot of the sensitivity function of the reported design across the design space. In the examples, all parameters such as entries in the covariance matrix $\boldsymbol{D}$, n and m are chosen arbitrarily and for illustrative purposes. Since σ^2 does not affect the optimization process, I set $\sigma^2 = 1$.

Model	$\mathrm{y}_{ij} = \theta_{i0} + \theta_{i1} x_j^{1/2} + \theta_{i2} x_j + \theta_{i3} x_j^2 + \epsilon_{ij}$
$\boldsymbol{D}$	$diag(0.3, 0.5, 0.8, 0.2)$
Design space	$[1, 3]$
(n, m)	(10, 5)
G-optimal design	$\begin{pmatrix} 1.000 & 1.440 & 2.342 & 3.000 \\ 0.186 & 0.190 & 0.120 & 0.504 \end{pmatrix}$
G-criterion value	17.939
μ^* on $A(\boldsymbol{\eta})$	$\begin{pmatrix} 1.000 & 1.431 & 2.338 & 3.000 \\ 0.233 & 0.224 & 0.100 & 0.443 \end{pmatrix}$
G-sensitivity function	Figure 6.1

Table 6.1: The G-optimal design found by CSO-MA for a fractional polynomial mixed model with terms $x^{1/2}$, x, x^2 and a diagonal covariance matrix on the design space $[1, 3]$.

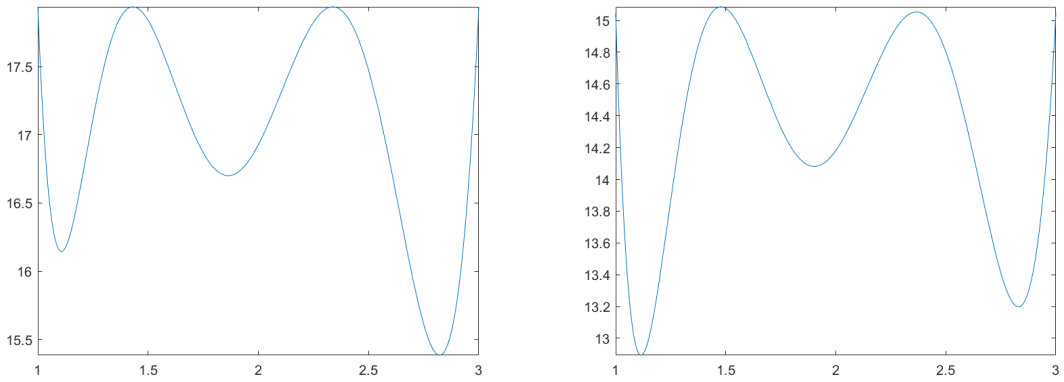

Figure 6.1: $\phi(x, \boldsymbol{\eta})$ (left) and $\phi_G(x, \boldsymbol{\eta})$ (right) of the G-optimal design in Table 6.1.

Model	$\mathrm{y}_{ij} = \theta_{i0} + \theta_{i1} x_j^{1/2} + \theta_{i2} x_j + \theta_{i3} x_j^2 + \epsilon_{ij}$
$\boldsymbol{D}$	$\begin{pmatrix} 0.80 & 0.30 & 0.10 & 0.05 \\ 0.30 & 0.50 & 0.08 & 0.04 \\ 0.10 & 0.08 & 0.40 & 0.02 \\ 0.05 & 0.04 & 0.02 & 0.30 \end{pmatrix}$
Design space	$[1, 3]$
(n, m)	(8, 4)
G-optimal design	$\begin{pmatrix} 1.000 & 1.419 & 2.372 & 3.000 \\ 0.230 & 0.186 & 0.122 & 0.462 \end{pmatrix}$
G-criterion value	15.546
μ^* on $A(\boldsymbol{\eta})$	$\begin{pmatrix} 1.000 & 1.410 & 2.360 & 3.000 \\ 0.274 & 0.198 & 0.106 & 0.422 \end{pmatrix}$
G-sensitivity function	Figure 6.2

Table 6.2: The G-optimal design found by CSO-MA for a fractional polynomial mixed model with terms $x^{1/2}$, x, x^2 and a non-diagonal covariance matrix on the design space $[1, 3]$.

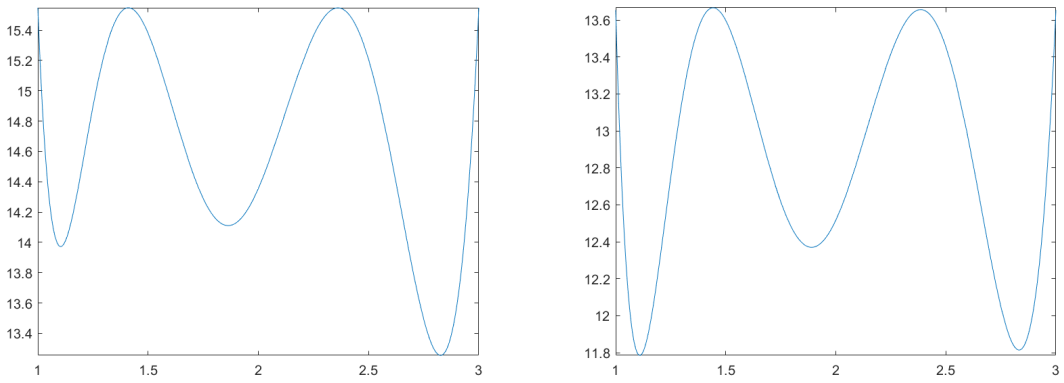

Figure 6.2: $\phi(x, \boldsymbol{\eta})$ (left) and $\phi_G(x, \boldsymbol{\eta})$ (right) of the G-optimal design in Table 6.2.

Model	$\mathrm{y}_{ij} = \theta_{i0} + \theta_{i1} x_j^{1/3} + \theta_{i2} x_j^{1/2} + \theta_{i3} x_j + \theta_{i4} x_j^2 + \epsilon_{ij}$
$\boldsymbol{D}$	$diag(0.1, 2.0, 3.0, 3.0, 2.0)$
Design space	$[2, 6]$
(n, m)	(8, 4)
G-optimal design	$\begin{pmatrix} 2.000 & 2.492 & 3.555 & 5.138 & 6.000 \\ 0.175 & 0.180 & 0.160 & 0.105 & 0.380 \end{pmatrix}$
G-criterion value	19.081
μ^* on $A(\boldsymbol{\eta})$	$\begin{pmatrix} 2.000 & 2.485 & 3.570 & 5.134 & 6.000 \\ 0.188 & 0.172 & 0.168 & 0.105 & 0.367 \end{pmatrix}$
G-sensitivity function	Figure 6.3

Table 6.3: The G-optimal design found by CSO-MA for a fractional polynomial mixed model with terms $x^{1/3}$, $x^{1/2}$, x, x^2 and a diagonal covariance matrix on the design space $[2, 6]$.

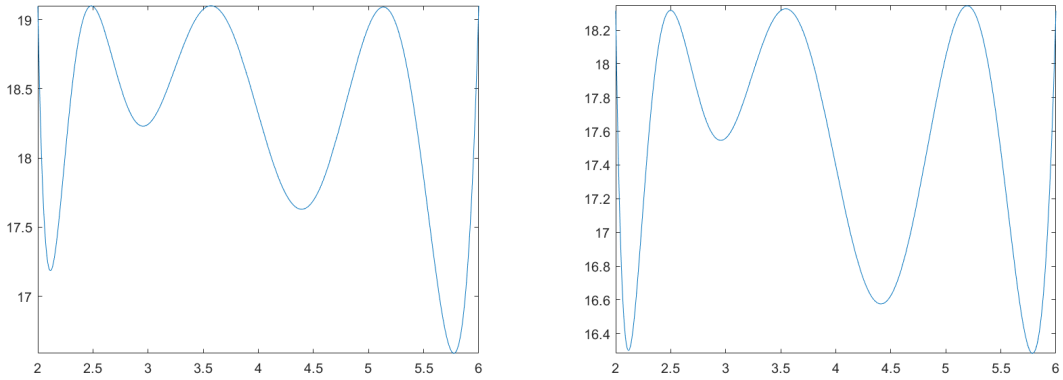

Figure 6.3: $\phi(x, \boldsymbol{\eta})$ (left) and $\phi_G(x, \boldsymbol{\eta})$ (right) of the G-optimal design in Table 6.3.

Model	$\mathrm{y}_{ij} = \theta_{i0} + \theta_{i1} x_j^{1/3} + \theta_{i2} x_j^{1/2} + \theta_{i3} x_j + \theta_{i4} x_j^2 + \epsilon_{ij}$
$\boldsymbol{D}$	$\begin{pmatrix} 0.7 & 0.3 & 0.2 & 0.2 & 0.1 \\ 0.3 & 0.6 & 0.6 & 0.9 & 0.2 \\ 0.2 & 0.6 & 1.4 & 0.8 & 0.8 \\ 0.2 & 0.9 & 0.8 & 2.0 & 0.9 \\ 0.1 & 0.2 & 0.8 & 0.9 & 2.0 \end{pmatrix}$
Design space	$[2, 5]$
(n, m)	$(10,\ 5)$
G-optimal design	$\begin{pmatrix} 2.000 & 2.364 & 3.190 & 4.427 & 5.000 \\ 0.210 & 0.186 & 0.114 & 0.093 & 0.397 \end{pmatrix}$
G-criterion value	22.589
μ^* on $A(\boldsymbol{\eta})$	$\begin{pmatrix} 2.000 & 2.359 & 3.183 & 4.407 & 5.000 \\ 0.219 & 0.185 & 0.117 & 0.091 & 0.388 \end{pmatrix}$
G-sensitivity function	Figure 6.4

Table 6.4: The G-optimal design found by CSO-MA for a fractional polynomial mixed model with terms $x^{1/3}$, $x^{1/2}$, x, x^2 and a non-diagonal covariance matrix on the design space $[2, 6]$.

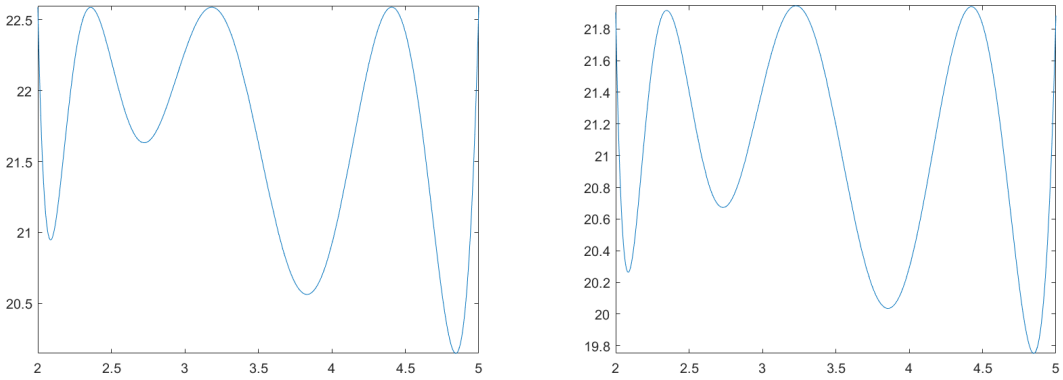

Figure 6.4: $\phi(x, \boldsymbol{\eta})$ (left) and $\phi_G(x, \boldsymbol{\eta})$ (right) of the G-optimal design in Table 6.4.

Model	$\mathrm{y}_{ij} = \theta_{i0} + \theta_{i1}x_j^{-1/2} + \theta_{i2}x_j^{1/3} + \theta_{i3}x_j^{1/2} + \theta_{i4}x_j + \theta_{i5}x_j^2 + \epsilon_{ij}$
$\boldsymbol{D}$	$diag(0.3, 0.2, 0.3, 0.6, 0.2, 0.2)$
Design space	$[1, 3]$
(n, m)	(6, 2)
G-optimal design	$\begin{pmatrix} 1.000 & 1.155 & 1.535 & 2.119 & 2.728 & 3.000 \\ 0.133 & 0.130 & 0.122 & 0.117 & 0.165 & 0.333 \end{pmatrix}$
G-criterion value	12.772
μ^* on $A(\boldsymbol{\eta})$	$\begin{pmatrix} 1.000 & 1.153 & 1.531 & 2.109 & 2.722 & 3.000 \\ 0.154 & 0.147 & 0.128 & 0.108 & 0.139 & 0.324 \end{pmatrix}$
G-sensitivity function	Figure 6.5

Table 6.5: The G-optimal design found by CSO-MA for a fractional polynomial mixed model with terms $x^{-1/2}$, $x^{1/3}$, $x^{1/2}$, x, x^2 and a diagonal covariance matrix on the design space $[1, 3]$.

Model	$\mathrm{y}_{ij} = \theta_{i0} + \theta_{i1}x_j^{-1/2} + \theta_{i2}x_j^{1/3} + \theta_{i3}x_j^{1/2} + \theta_{i4}x_j + \theta_{i5}x_j^2 + \epsilon_{ij}$
$\boldsymbol{D}$	$\begin{pmatrix} 0.80 & 0.06 & 0.05 & 0.07 & 0.06 & 0.04 \\ 0.06 & 1.10 & 0.30 & 0.20 & 0.50 & 0.09 \\ 0.05 & 0.30 & 0.90 & 0.50 & 0.60 & 0.03 \\ 0.07 & 0.20 & 0.50 & 1.40 & 0.40 & 0.07 \\ 0.06 & 0.50 & 0.60 & 0.40 & 1.10 & 0.02 \\ 0.04 & 0.09 & 0.03 & 0.07 & 0.02 & 0.90 \end{pmatrix}$
Design space	$[1, 3]$
(n, m)	(6, 2)
G-optimal design	$\begin{pmatrix} 1.000 & 1.150 & 1.512 & 2.085 & 2.715 & 3.000 \\ 0.162 & 0.089 & 0.131 & 0.107 & 0.134 & 0.377 \end{pmatrix}$
G-criterion value	14.959
μ^* on $A(\boldsymbol{\eta})$	$\begin{pmatrix} 1.000 & 1.148 & 1.510 & 2.087 & 2.712 & 3.000 \\ 0.176 & 0.159 & 0.133 & 0.103 & 0.124 & 0.305 \end{pmatrix}$
G-sensitivity function	Figure 6.5

Table 6.6: The G-optimal design found by CSO-MA for a fractional polynomial mixed model with terms $x^{-1/2}$, $x^{1/3}$, $x^{1/2}$, x, x^2 and a diagonal covariance matrix on the design space $[1, 3]$.

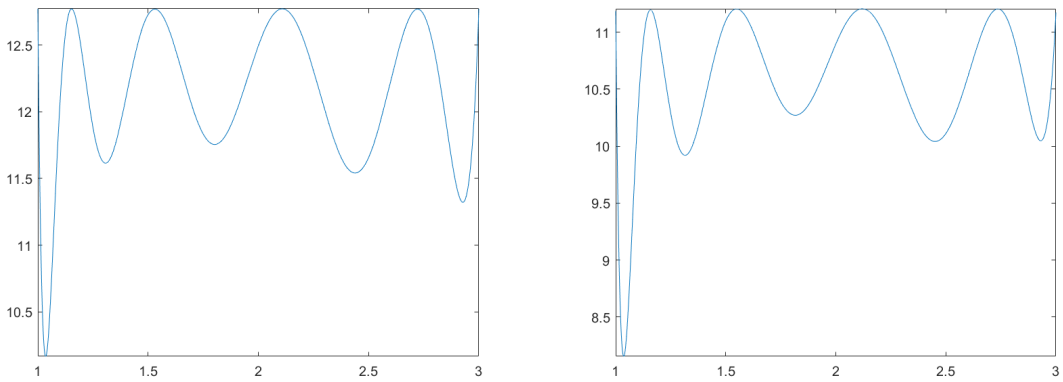

Figure 6.5: $\phi(x, \boldsymbol{\eta})$ (left) and $\phi_G(x, \boldsymbol{\eta})$ (right) of the G-optimal design in Table 6.5.

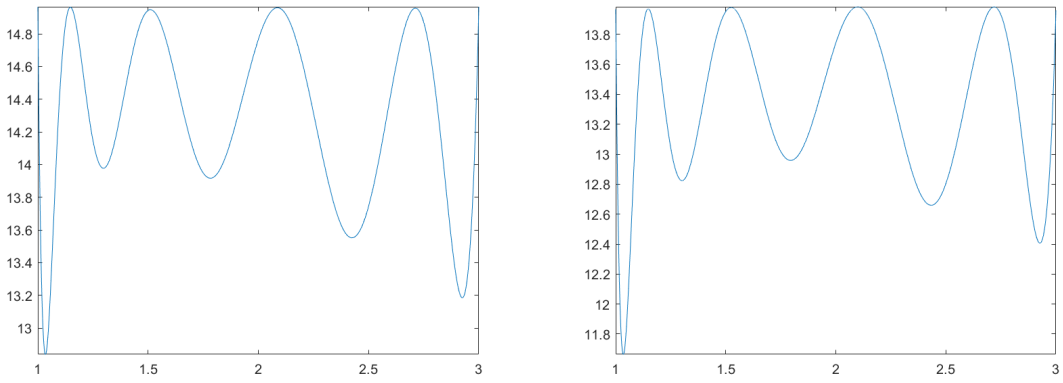

Figure 6.6: $\phi(x, \boldsymbol{\eta})$ (left) and $\phi_G(x, \boldsymbol{\eta})$ (right) of the G-optimal design in Table 6.6.

In the next three tables, I not only show G-optimal designs but also extended D-optimal designs for linear mixed models given the same model parameters and design spaces. Their sensitivity functions are also plotted and they confirm the optimality of the CSO-MA-generated designs. The tables also reveal the two types of designs are very close to one another, and the G-efficiencies of the extended D-optimal designs are 99%.

Model	$y_{ij} = \theta_{i0} + \theta_{i1}x_j + \theta_{i2}x_j^2 + \epsilon_{ij}$
$\boldsymbol{D}$	$diag(0.2, 0.2, 0.3)$
Design space	$[0, 2]$
(n, m)	(10, 5)
G-optimal design	$\begin{pmatrix} 0.000 & 0.966 & 2.000 \\ 0.146 & 0.140 & 0.714 \end{pmatrix}$
G-criterion value	13.480
Extended D-efficiency	99%
Extended D-optimal design	$\begin{pmatrix} 0.000 & 0.946 & 2.000 \\ 0.157 & 0.140 & 0.703 \end{pmatrix}$
Extended D-criterion value	-71.615
G-efficiency	99%
Sensitivity functions	Figure 6.7

Table 6.7: The G-optimal design and its corresponding extended D-optimal design found by CSO-MA for a fractional polynomial mixed model with terms x, x^2 and a diagonal covariance matrix on the design space $[0, 2]$. Their relative efficiencies are reported.

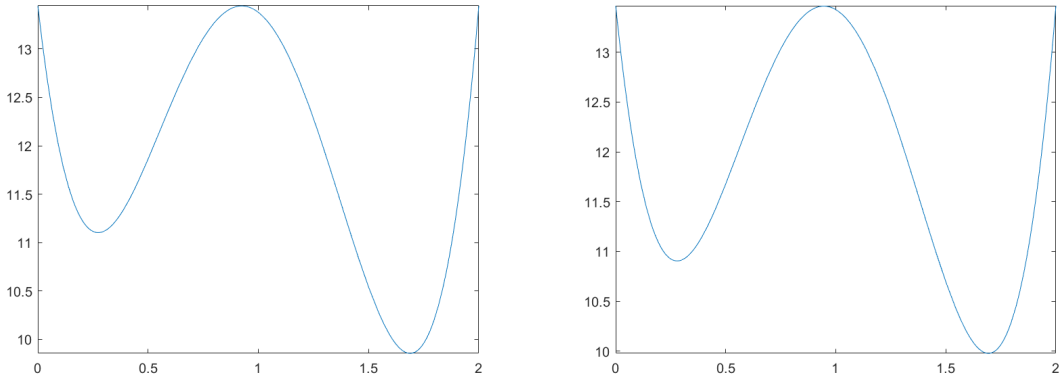

Figure 6.7: The sensitivity functions for the G-optimal design (left) and the extended D-optimal design (right) in Table 6.7.

Model	$\mathrm{y}_{ij} = \theta_{i0} + \theta_{i1}x_j + \theta_{i2}x_j^2 + \epsilon_{ij}$
$\boldsymbol{D}$	$\begin{pmatrix} 0.80 & 0.30 & 0.10 \\ 0.30 & 0.50 & 0.08 \\ 0.10 & 0.08 & 0.40 \end{pmatrix}$
Design space	$[0, 3]$
(n, m)	(11, 4)
G-optimal design	$\begin{pmatrix} 0.000 & 1.274 & 3.000 \\ 0.165 & 0.270 & 0.565 \end{pmatrix}$
G-criterion value	19.000
Extended D-efficiency	99%
Extended D-optimal design	$\begin{pmatrix} 0.000 & 1.239 & 3.000 \\ 0.163 & 0.272 & 0.565 \end{pmatrix}$
Extended D-criterion value	-80.940
G-efficiency	99%
Sensitivity functions	Figure 6.8

Table 6.8: The G-optimal design and its corresponding extended D-optimal design found by CSO-MA for a fractional polynomial mixed model with terms x, x^2 and a non-diagonal covariance matrix on the design space $[0, 3]$. Their relative efficiencies are reported.

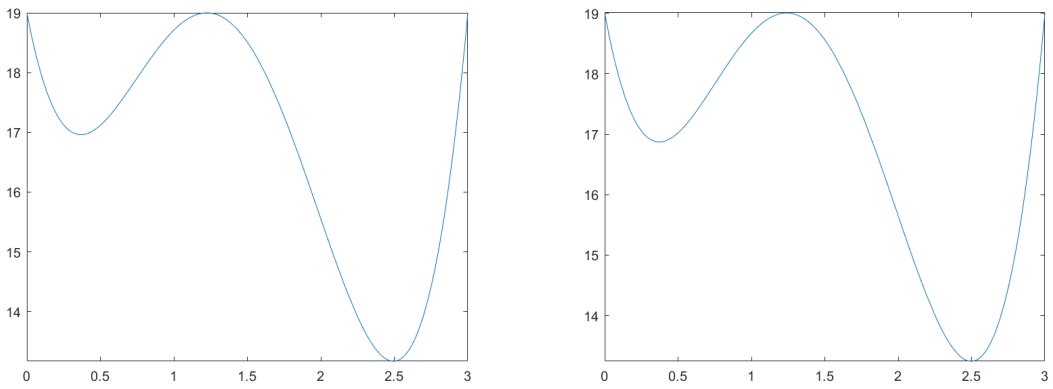

Figure 6.8: The sensitivity functions for the G-optimal design (left) and the extended D-optimal design (right) in Table 6.8.

Model	$\mathrm{y}_{ij} = \theta_{i0} + \theta_{i1} x_j^{1/2} + \theta_{i2} x_j + \theta_{i3} x_j^2 + \epsilon_{ij}$
$\boldsymbol{D}$	$diag(0.3, 0.5, 0.8, 0.2)$
Design space	$[1, 3]$
(n, m)	(10, 5)
G-optimal design	$\begin{pmatrix} 1.000 & 1.440 & 2.342 & 3.000 \\ 0.186 & 0.190 & 0.120 & 0.504 \end{pmatrix}$
G-criterion value	17.939
Extended D-efficiency	99%
Extended D-optimal design	$\begin{pmatrix} 1.000 & 1.412 & 2.361 & 3.000 \\ 0.230 & 0.186 & 0.122 & 0.462 \end{pmatrix}$
Extended D-criterion value	-71.700
G-efficiency	99%
Sensitivity functions	Figure 6.9

Table 6.9: The G-optimal design and its corresponding extended D-optimal design found by CSO-MA for a fractional polynomial mixed model with terms $x^{1/2}$, x, x^2 and a diagonal covariance matrix on the design space $[1, 3]$.. Their relative efficiencies are reported.

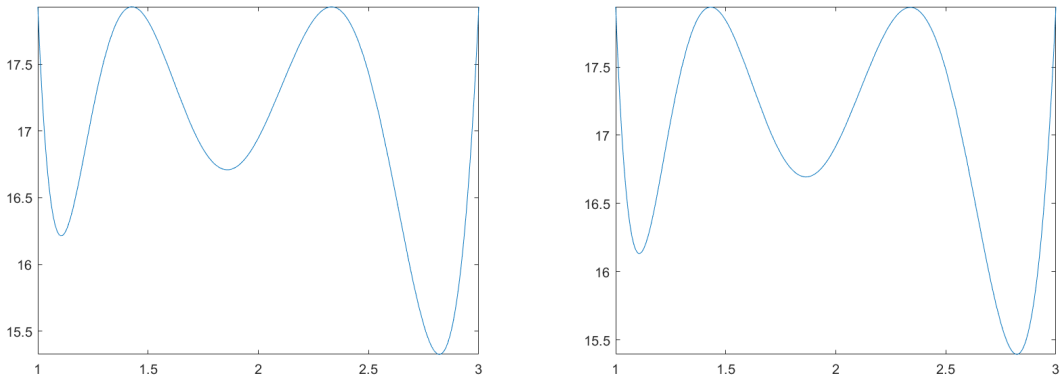

Figure 6.9: The sensitivity functions for the G-optimal design (left) and the extended D-optimal design (right) in Table 6.9.

6.2 Optimal Exact Designs

I have focused on constructing optimal approximate design problems in the previous chapters of my dissertation. Approximate designs provide useful guidance on how to design the study and study properties of the design. This subsection shows CSO-MA can also find optimal exact designs that require direct optimization of the number of replicates at each design point. Optimal exact design problems are very tough to solve because there is no unified framework to solve them generally; in particular, there is no effective algorithm for finding them and no equivalence theorem for confirming their optimality. Each optimal exact design problem has to be tackled individually because each has its own mathematical complexity, which is often intractable. Usually, esoteric number theory is often necessary to bring about an analytic solution and so this can be limiting in practice.

Examples of optimal exact designs are available in Dette et al. (2008) and Antognini and Zagoraiou (2010). There are few closed-form descriptions of optimal exact designs because they are available only for relatively simple settings. Many are found numerically using many different types of algorithms. However, after a series of experiments, I find that some designs which were claimed as "optimal" may not be optimal since the CSO-generated exact designs have higher efficiency. In what is to follow, I discuss three types of exchange algorithms and show that CSO-MA can find more efficient designs than the claimed optimal designs found by the algorithms. I now describe the three popular algorithms.

6.2.1 Coordinate Exchange Algorithm

JMP is a suite of computer programs for statistical analysis developed by the JMP business unit of SAS Institute (https://www.jmp.com/en_us/home.html). The JMP DOE (Design of Experiment) platforms offer solutions for finding different optimal designs and deemed as a standard tool in this field. According to the "JMP Design of Experiments Guide" (version 14, 2018), JMP uses the coordinate exchange algorithm for finding optimal designs.

Here I discuss three examples of locally D-optimal exact designs found by JMP for additive logistic regression models with five factors and the parameters in the models are randomly generated. For each model, I run the algorithm from JMP ten times and compare their designs with the CSO-MA-generated designs. Then I rank the twenty designs according to their criterion values and select the one with the best criterion value as the optimal design (a reference for calculating other designs' efficiency). Minimally supported locally D-optimal exact designs are sought, so there are six support points for each generated design with $N = 6$. For these three examples, our results show that the designs found by CSO-MA have higher D-efficiencies than those reported in JMP.

(i). The model parameters are set to $(\theta_0, \cdots, \theta_5) = (-3, -2, -1, 1, 2, 3)$. The design space is $[-1, 1]^5$ and the number of runs is 6. The designs found by CSO-MA have an average criterion value -2.08 (maximize the log-determinant of the information matrix), an average efficiency 99%. The designs found by JMP DOE have an average criterion value -2.35, average efficiency 98%. The best designs found by CSO-MA (100% efficient) and JMP DOE (98% efficient) are displayed in Table 6.10.

CSO-MA					JMP DOE				
x_{i1}	x_{i2}	x_{i3}	x_{i4}	x_{i5}	x_{i1}	x_{i2}	x_{i3}	x_{i4}	x_{i5}
-1.000	-1.000	-1.000	-1.000	0.953	-1.000	1.000	1.000	-1.000	0.792
-1.000	1.000	-1.000	1.000	0.771	-0.112	1.000	-1.000	1.000	1.000
-1.000	1.000	1.000	-1.000	0.953	-1.000	-1.000	-1.000	1.000	-0.913
1.000	-1.000	1.000	0.395	1.000	-1.000	-1.000	-1.000	-0.927	1.000
0.787	1.000	-1.000	1.000	1.000	1.000	-1.000	1.000	-0.203	1.000
-0.924	-1.000	1.000	1.000	-1.000	-1.000	-1.000	1.000	1.000	-0.688

Table 6.10: The best designs found by CSO-MA (100% efficient as the reference) and JMP DOE (98% efficient) for model (i).

(ii). The model parameters are set to $(\theta_0, \cdots, \theta_5) = (0.1, -0.5, 1.9, 3.3, -4.0, -0.4)$. The design space is $[-1, 1]^5$ and the number of runs is 6. The designs found by CSO-MA have an average criterion value -2.44, an average efficiency 99%. The designs found by JMP DOE have an average criterion value -2.66, an average efficiency 98%. The best designs found by CSO-MA (100% efficient) and JMP DOE (98% efficient) are displayed in Table 6.11.

(iii). The model parameters are set to $(\theta_0, \cdots, \theta_5) = (0.1, -0.5, 1.9, 3.3, -4.0, -0.4)$. The

CSO-MA					JMP DOE				
x_{i1}	x_{i2}	x_{i3}	x_{i4}	x_{i5}	x_{i1}	x_{i2}	x_{i3}	x_{i4}	x_{i5}
1.000	1.000	-1.000	-0.327	1.000	1.000	1.000	-1.000	-0.696	1.000
-1.000	-1.000	-0.655	-1.000	1.000	-1.000	-1.000	1.000	0.227	1.000
1.000	1.000	-1.000	-0.727	-1.000	1.000	1.000	0.389	1.000	-1.000
-1.000	1.000	0.828	1.000	1.000	1.000	-1.000	-0.515	-1.000	-1.000
1.000	-1.000	1.000	0.462	-1.000	-1.000	1.000	-1.000	-0.241	-1.000
-1.000	1.000	-1.000	0.123	-1.000	-1.000	-1.000	-1.000	-0.894	1.000

Table 6.11: The best designs found by CSO-MA (100% efficient as the reference) and JMP DOE (98% efficient) for model (ii).

design space is $[-1, 1]^5$ and the number of runs is 6. The designs found by CSO-MA have an average criterion value -3.22, an average efficiency 99%. The designs found by JMP DOE have an average criterion value -3.33, an average efficiency 98%. The best designs found by CSO-MA (100% efficient) and JMP DOE (98% efficient) are displayed in Table 6.12.

x_{i1}	x_{i2}	x_{i3}	x_{i4}	x_{i5}	x_{i1}	x_{i2}	x_{i3}	x_{i4}	x_{i5}
1.000	-1.000	0.997	1.000	1.000	1.000	1.000	0.990	-1.000	-1.000
1.000	-1.000	0.515	-1.000	1.000	1.000	-1.000	1.000	1.000	-0.526
-1.000	1.000	0.256	-1.000	-1.000	-1.000	-1.000	0.669	1.000	-1.000
1.000	1.000	1.000	1.000	-1.000	-1.000	1.000	-0.585	-1.000	1.000
-1.000	1.000	-0.389	1.000	1.000	1.000	1.000	0.576	1.000	1.000
-1.000	-1.000	0.496	1.000	-1.000	1.000	-1.000	0.719	-1.000	1.000

Table 6.12: The best designs found by CSO-MA (100% efficient as the reference) and JMP DOE (98% efficient) for model (iii).

When N is large, say, $N = 20$ or 30 for the above models, CSO-MA can also find more highly efficient exact designs than those reported in JMP, but their differences in their D-efficiencies are trivial. For instance, if the aim is to find the 20-point optimal exact design for model (i), the exact designs found by CSO-MA have an average criterion value -5.418 and the corresponding average criterion value of JMP-generated designs is -5.428. When N becomes larger, JMP becomes advantageous over CSO-MA in terms of CPU time needed to find a highly D-efficient exact design. However, CSO-MA is much more flexible than the algorithms available in JMP since we demonstrated CSO-MA can be used to find other types of optimal designs, such as c and G-optimal designs. Overall, I find that CSO-MA is a very promising method for searching optimal exact or approximate designs, especially for more

complicated models than many current algorithms in the statistical literature.

6.2.2 Fedorov Exchange Algorithm

Lall et al. (2018) used Fedorov exchange algorithm to find the 6-point locally D-optimal exact design on the design space $[-1, 1]^2$ for a logistic model with two factors, two second-order terms and one interaction. The nominal values of the parameters were $\theta_0 = -1,\ \theta_1 = 2,\ \theta_2 = 0.5,\ \theta_3 = 2,\ \theta_4 = 0.1$ and $\theta_5 = 0.01$.

The design found by Lall et al. (2018) is

$$\boldsymbol{\eta} = \begin{pmatrix} -1.000 & 1.000 & -1.000 & 0.057 & 1.000 & 0.143 \\ 1.000 & -1.000 & -0.700 & 0.066 & -0.026 & 1.000 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix},$$

where the first two rows indicate the 2-dimensional design points and the last row is the number of replicates at each design point. The criterion value of $\boldsymbol{\eta}$ is -7.455. The CSO-MA-generated design $\boldsymbol{\eta}^*$ CSO-MA is

$$\boldsymbol{\eta}^* = \begin{pmatrix} -1.000 & -1.000 & -0.063 & 0.366 & 0.613 & 0.712 \\ 1.000 & -1.000 & -1.000 & 1.000 & 1.000 & 0.000 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix},$$

and the criterion value of $\boldsymbol{\eta}^*$ is -5.520. A direct calculation shows that design $\boldsymbol{\eta}$ is 95% D-efficient relative to $\boldsymbol{\eta}^*$. Therefore, CSO-MA can outperform the exchange algorithm and find a more efficient design.

6.2.3 Approximate Coordinate Exchange Algorithm

Overstall et al. (2017) used the approximate coordinate exchange (ACE) algorithm to minimize the expected loss function for finding Bayesian optimal design. In the R package "acebayes" (title: Optimal Bayesian Experimental Design Using the ACE Algorithm), the ACE algorithm was applied to find the pseudo-Bayesian D-optimal exact design for a four-factor

additive logistic model. The prior distributions for θ_0 to θ_4 are all uniform with corresponding lower bounds (-3, 4, 5, -6, -2.5) and upper bounds (3, 10, 11, 0, 3.5). Finally, a 6-point exact design is provided by ACE on the design space $[-1, 1]^4$. Under the same model setup, the CSO-MA-generated 6-point pseudo-Bayesian exact design is more efficient. The relative efficiency of the ACE-generated design to the CSO-MA-generated design is around 78%. Table 6.13 lists the two designs.

ACE				CSO-MA			
x_1	x_2	x_3	x_4	x_1	x_2	x_3	x_4
-0.357	0.161	-0.613	0.928	-0.590	0.996	1.000	0.936
-0.917	0.914	0.698	0.261	-0.803	0.412	-1.000	-1.000
-0.884	0.429	-1.000	-0.968	0.804	-1.000	-0.999	0.893
0.370	-0.271	0.653	0.185	0.630	-0.413	1.000	-0.796
0.717	-0.347	-0.060	-0.659	-0.581	0.038	-0.433	0.866
0.747	0.059	1.000	-0.174	0.625	-0.089	0.491	-0.899

Table 6.13: The 6-point pseudo-Bayesian exact designs for a four-factor additive logistic model with uniform priors found by ACE and CSO-MA.

I also test the reliability of using two algorithms to find the 10-point pseudo-Bayesian D-optimal exact design under a multivariate normal prior whose mean vector is $(-7.0, 1.2, 2.5, -0.3, 16.0)$ and the covariance matrix Σ is

$$\mathbf{\Sigma} = \begin{pmatrix} 2.66 & -1.03 & -0.87 & 0.24 & -0.05 \\ -1.03 & 2.91 & -0.52 & -0.19 & 0.18 \\ -0.87 & -0.52 & 3.73 & 1.19 & 0.52 \\ 0.24 & -0.19 & 1.19 & 3.53 & -0.47 \\ -0.05 & 0.18 & 0.52 & -0.47 & 2.14 \end{pmatrix}. \tag{6.5}$$

A direct calculation shows the design found by ACE is around 73% D-efficient relative to the CSO-MA-generated design. Table 6.14 lists the two designs.

ACE				CSO-MA			
x_1	x_2	x_3	x_4	x_1	x_2	x_3	x_4
-1.000	-0.288	0.046	0.533	1.000	0.571	-1.000	0.375
-0.071	1.000	-1.000	0.281	1.000	1.000	1.000	0.104
-1.000	-0.953	-0.708	0.648	-1.000	-0.909	-1.000	0.499
-0.689	0.176	-0.134	0.394	-1.000	1.000	-0.147	0.452
1.000	0.014	0.930	0.417	1.000	0.994	-1.000	0.074
0.759	-0.580	0.452	0.458	-1.000	-1.000	1.000	0.809
1.000	1.000	-0.971	0.173	1.000	1.000	1.000	0.416
0.095	-0.467	-0.447	0.678	0.892	-1.000	1.000	0.368
-0.609	0.932	1.000	0.428	-1.000	1.000	0.108	0.182
1.000	1.000	-0.825	-0.008	1.000	-1.000	0.794	0.638

Table 6.14: The 10-point pseudo-Bayesian exact designs for a four-factor additive logistic model with a multivariate normal prior found by ACE and CSO-MA.

6.3 Conclusions

This dissertation proposes a novel optimization algorithm, CSO-MA, for finding various types of optimal experimental designs for nonlinear models with or without random effects. I also demonstrate it can be used to find Bayesian optimal designs and also for finding optimal approximate or exact designs. Many of the optimal designs are new and they are found by solving more challenging optimization problems. Unlike the statistical models typically reported in the literature, the statistical models in my dissertation have many more factors than those in the literature and they allow for interacting factors as well. My work shows that CSO-MA is a flexible and powerful algorithm and can perform as well as other state-of-the-art methods, and frequently outperforms them. For instance, many reported "optimal" designs in the literature have less optimal criterion values than those found by CSO-MA. For those design problems where the generated designs can be verified to be optimal via an equivalence theorem, it is reassuring to know that all CSO-MA-generated designs have been verified to be optimal.

There are ongoing work and future work that I plan to pursue. First, I plan to organize and publish the CSO-MA codes, including finding optimal designs for different types of models, either in the Journal of Statistical Software or in the R Journal. Some of these codes are

in the appendices. They will be made freely available for users interested to use CSO-MA to find efficient designs for their studies. My immediate future work is to investigate the capability of CSO-MA to solve other statistical problems, such as finding more efficient estimates for model parameters. This is possible since CSO-MA is a general-purpose optimization algorithm. Already, there is some preliminary work that shows PSO can outperform maximum likelihood estimates for complicated likelihood functions obtained from commercial software packages. My interest is to investigate whether CSO-MA can similarly outperform results from SAS or STATA for obtaining more precise estimates when we have complicated likelihood functions. These estimates include $L1$-estimates, ridge regression estimates and the like.

Appendix A

MATLAB Codes – Competitive Swarm Optimizer with Mutated Agents

```
%% Function ''csoma'' can minimize a user-defined objective function.
%% Inputs:
%%          obj_fun   -- a user-defined objective function;
%%          lb        -- a vector of searching space lower bounds;
%%          ub        -- a vector of searching space upper bounds;
%%          swarmsize -- the number of particles;
%%          phi       -- parameter φ;
%%          maxiter   -- the number of iterations.
%% Outputs:
%%          minf      -- the minimum of the objective function;
%%          minx      -- the solution that minimizes the objective function.
function [minf, minx] = csoma(obj_fun, lb, ub, swarmsize, phi, maxiter)
    % check whether the dimension of lower bounds matches the dimension of
    % upper bounds
    assert(length(lb) == length(ub), 'Not equal length of bounds');
    % check whether inputted upper bounds are greater than lower bounds
    if all(ub - lb <= 0) > 0
        error('Error. \n Upper bound must be greater than lower bound.')
    end
    % simplify notations
    S = swarmsize;
    % D is the dimension of the problem
    D = length(ub);
    % randomly initalize all particles
```

```
x = rand(S, D);
x = bsxfun(@plus, lb, bsxfun(@times, ub-lb, x));
v = zeros([S D]); % set initial velocities to 0
iter = 0;
pairnum_1 = floor(S / 2);
losers = 1:S;
fx = arrayfun(@(K) obj_fun(x(K, :)), 1:S);
randperm_index = randperm(S);
while iter <= maxiter
    % update losers' function values
    fx(losers) = arrayfun(@(K) obj_fun(x(K, :)), losers);
    % calculate the swarm center
    swarm_center = mean(x);
    % randomly permuate all particle indexes
    randperm_index = randperm(S);
    % random pairing and compare
    rpairs = [randperm_index(1:pairnum_1); randperm_index(S-pairnum_1+1:S)]';
    cmask= (fx(rpairs(:, 1)) > fx(rpairs(:, 2)))';
    % losers who with larger values
    losers = bsxfun(@times, cmask, rpairs(:, 1))  ...
        + bsxfun(@times, ~cmask, rpairs(:, 2));
    % winners who with smaller values
    winners = bsxfun(@times, ~cmask, rpairs(:, 1)) ...
        + bsxfun(@times, cmask, rpairs(:, 2));
    R1 = rand(pairnum_1, D);
    R2 = rand(pairnum_1, D);
    R3 = rand(pairnum_1, D);
    % update losers
    v(losers, :) = bsxfun(@times, R1, v(losers, :)) ...
        + bsxfun(@times, R2, x(winners, :) - x(losers, :)) ...
        + phi * bsxfun(@times, R3, bsxfun(@minus, swarm_center, x(losers, :)));
    x(losers, :) = x(losers, :) + v(losers, :);
    % check whether new particles are out of bounds.
    maskl = bsxfun(@lt, x(losers, :), lb);
    masku = bsxfun(@gt, x(losers, :), ub);
```

```
        mask = bsxfun(@lt, x(losers, :), lb) | bsxfun(@gt, x(losers, :), ub);
        x(losers, :) = bsxfun(@times, ~mask, x(losers, :)) ...
            + bsxfun(@times, lb, maskl) + bsxfun(@times, ub, masku);
        % mutation
        mutation_idx = randsample(losers, 1);
        mutation_dim = randsample(1:D, 1);
        x(mutation_idx, mutation_dim) = randsample([lb(mutation_dim) ...
        ub(mutation_dim)], 1);
        iter = iter + 1;
        fprintf('Iter: %d\n', iter);
        fprintf('Best fitness: %e\n', min(fx));
    end
    fprintf('Best fitness: %e\n', min(fx));
    [minf, min_index] = min(fx);
    minx = x(min_index, :);
end
```

Appendix B

MATLAB Codes – Locally D-optimal Design for a Two-factor Additive Logistic Model

```
%% Function ''LogisticDOD2'' is the criterion function of finding locally
%% D-optimal approximate design for a two-factor additive logistic model.
%% Inputs:
%%         x      -- a vectorized design;
%%         theta  -- model parameter θ including three nominal values;
%%         k      -- the number of design points in the design.
%% Outputs:
%%         dcri       -- the negative D-criterion value.
function dcri = LogisticDOD2(x, theta, k)
    % x(1:k): x_11, x_12, ..., x_1k
    % x(k+1:2*k): x_21, x_22, ..., x_2k
    % x(2*k+1:3*k): w_1, w_2, ..., w_k
    X = cat(2, ones(k, 1), reshape(x, [k, 3]));
    % calculate the information matrix
    Info_mat = zeros(3, 3);
    total_weight = sum(X(:, 4));
    X(:, 4) = X(:, 4) ./ total_weight;
    linear_part = X(:, [1:3]) * theta;
    exp_part = exp(linear_part)./ (1 + exp(linear_part)).^2;
    for i=1:k
        Info_mat = Info_mat + X(i, 4) * exp_part(i) * X(i, 1:3)' * X(i, 1:3);
    end
    dcri = -log(det(Info_mat));
end
```

```
%% Function ''SF2'' is the D-sensitivity function of a given design
%% for a two-factor additive logistic model.
%% Inputs:
%%         genD   -- a degenerate design;
%%         design -- a design obtained by optimizing the criterion function;
%%         theta  -- model parameter θ including three nominal values;
%%         k      -- the number of design points in the design.
%% Outputs:
%%         alpha  -- the maximum of the D-sensitivity function.
function alpha = SF2(genD, design, theta, k)
    % design(1:k): x_11, x_12, ..., x_1k
    % design(k+1:2*k): x_21, x_22, ..., x_2k
    % design(2*k+1:3*k): w_1, w_2, ..., w_k
    X = cat(2, ones(k, 1), reshape(design, [k, 3]));
    % calculate the information matrix
    Info_mat = zeros(3, 3);
    total_weight = sum(X(:, 4));
    X(:, 4) = X(:, 4) ./ total_weight;
    linear_part = X(:, [1:3]) * theta;
    exp_part = exp(linear_part)./ (1 + exp(linear_part)).^2;
    for i=1:k
        Info_mat = Info_mat + X(i, 4) * exp_part(i) * X(i, 1:3)' * X(i, 1:3);
    end
    d = [1 genD];
    alpha = 3 - exp(d * theta) / (1 + exp(d * theta))^2 * d / Info_mat * d';
end

%% To find the D-optimal approximate design, implement the following commands
theta = [-0.1, 1.5, 4.0]
k = 5;
Model = @(x)LogisticDOD2(x, theta', k);
% design space [-1, 1]^2
lb = cat(2, -1 * ones(1, 2*k), zeros(1, k));
ub = cat(2, 1 * ones(1, 2*k), ones(1, k));
```

```
swarmsize = 128;
phi = 0.1;
maxiter = 300;
[dcri, Ddesign] = csoma(Model, lb, ub, swarmsize, phi, maxiter);

%% To confirm the design's D-optimality, implement the following commands
SF = @(genD)SF2(genD, Ddesign, theta', k);
lbsf = [-1 -1];
ubsf = [1 1];
swarmsize = 64;
phi = 0;
maxiter = 200;
[alpha alpha_x] = csoma(SF, lbsf, ubsf, swarmsize, phi, maxiter).
```

Bibliography

Abebe, H. T., Tan, F. E., Van Breukelen, G. J., and Berger, M. P. (2014). Bayesian *D*-optimal designs for the two parameter logistic mixed effects model. *Computational Statistics & Data Analysis*, 71:1066–1076.

Al Labadi, L. et al. (2015). Some refinements on fedorov's algorithms for constructing *D*-optimal designs. *Brazilian Journal of Probability and Statistics*, 29(1):53–70.

Anderson, C. S., Jamrozik, K. D., and Stewart-Wynne, E. G. (1994). Patterns of acute hospital care, rehabilitation, and discharge disposition after acute stroke: the perth community stroke study 1989–1990. *Cerebrovascular Diseases*, 4(5):344–353.

Anderson-Cook, C. M. (2005). Practical genetic algorithms.

Anisimov, V. V. (2008). Using mixed poisson models in patient recruitment in multicentre clinical trials. In *Proceedings of the World Congress on Engineering*, volume 2, pages 1046–1049.

Antognini, A. B. and Zagoraiou, M. (2010). Exact optimal designs for computer experiments via kriging metamodelling. *Journal of Statistical Planning and Inference*, 140(9):2607–2617.

Atkinson, A. and Cook, R. (1995). *D*-optimum designs for heteroscedastic linear models. *Journal of the American Statistical Association*, 90(429):204–212.

Atkinson, A., Donev, A., Tobias, R., et al. (2007). *Optimum Experimental Designs, with SAS*, volume 34. Oxford University Press.

Atkinson, A. C. and Donev, A. N. (1989). The construction of exact *D*-optimum experimental designs with application to blocking response surface designs. *Biometrika*, 76(3):515–526.

Bansal, J. C., Singh, P., Saraswat, M., Verma, A., Jadon, S. S., and Abraham, A. (2011). Inertia weight strategies in particle swarm optimization. In *Nature and Biologically Inspired Computing (NaBIC), 2011 Third World Congress*, pages 633–640. IEEE.

Bastani, R., Glenn, B. A., Maxwell, A. E., and Jo, A. M. (2007). Hepatitis b testing for liver cancer control among korean americans. *Ethnicity and Disease*, 17(2):365.

Berger, M. P. and Tan, F. E. (2004). Robust designs for linear mixed effects models. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 53(4):569–581.

Berger, M. P. and Wong, W.-K. (2009). *An introduction to optimal designs for social and biomedical research*, volume 83. John Wiley & Sons.

Bogacka, B., Latif, M. A., Gilmour, S. G., and Youdim, K. (2017). Optimum designs for non-linear mixed effects models in the presence of covariates. *Biometrics*, 73(3):927–937.

Böhning, D. (1986). A vertex-exchange-method in d-optimal design theory. *Metrika*, 33(1):337–347.

Breslow, N. E. and Clayton, D. G. (1993). Approximate inference in generalized linear mixed models. *Journal of the American statistical Association*, 88(421):9–25.

Broudiscou, A., Leardi, R., and Phan-Tan-Luu, R. (1996). Genetic algorithm as a tool for selection of *D*-optimal design. *Chemometrics and Intelligent Laboratory Systems*, 35(1):105–116.

Campos, M., Krohling, R. A., and Enriquez, I. (2014). Bare bones particle swarm optimization with scale matrix adaptation. *IEEE Transactions on Cybernetics*, 44(9):1567–1578.

Carlisle, A. and Dozier, G. (2000). Adapting particle swarm optimization to dynamic environments. In *International Conference on Artificial Intelligence*, volume 1, pages 429–434.

Chaloner, K. (1993). A note on optimal bayesian design for nonlinear problems. *Journal of Statistical Planning and Inference*, 37(2):229–235.

Chaloner, K. et al. (1984). Optimal bayesian experimental design for linear models. *The Annals of Statistics*, 12(1):283–300.

Chaloner, K. and Larntz, K. (1989). Optimal bayesian design applied to logistic regression experiments. *Journal of Statistical Planning and Inference*, 21(2):191–208.

Chaloner, K. and Verdinelli, I. (1995). Bayesian experimental design: A review. *Statistical Science*, pages 273–304.

Chen, R.-B., Chang, S.-P., Wang, W., Tung, H.-C., and Wong, W. K. (2015). Minimax optimal designs via particle swarm optimization methods. *Statistics and Computing*, 25(5):975–988.

Cheng, C.-S. (1995). Optimal regression designs under random block-effects models. *Statistica Sinica*, pages 485–497.

Cheng, R. and Jin, Y. (2015). A competitive swarm optimizer for large scale optimization. *IEEE Transactions on Cybernetics*, 45(2):191–204.

Chernoff, H. (1953). Locally optimal designs for estimating parameters. *The Annals of Mathematical Statistics*, pages 586–602.

Chi, Y., Sun, F., Jiang, L., Yu, C., and Zhang, P. (2012). Elastic boundary for particle swarm optimization. In *International Conference in Swarm Intelligence*, pages 125–132. Springer.

Clyde, M., Müller, P., and Parmigiani, G. (1995). Optimal design for heart defibrillators. In *Case Studies in Bayesian Statistics, Volume II*, pages 278–292. Springer.

Cook, R. D. and Nachtrheim, C. J. (1980). A comparison of algorithms for constructing exact D-optimal designs. *Technometrics*, 22(3):315–324.

Cook, R. D. and Nachtsheim, C. J. (1982). Model robust, linear-optimal designs. *Technometrics*, 24(1):49–54.

Cook, R. D. and Wong, W. K. (1994). On the equivalence of constrained and compound optimal designs. *Journal of the American Statistical Association*, 89(426):687–692.

DasGupta, A., Studden, W., et al. (1991). Robust bayesian experimental designs in normal linear models. *The Annals of Statistics*, 19(3):1244–1256.

Davidian, M. and Giltinan, D. M. (1993). Some general estimation methods for nonlinear mixed-effects model. *Journal of Biopharmaceutical Statistics*, 3(1):23–55.

De la Garza, A. et al. (1954). Spacing of information in polynomial regression. *The Annals of Mathematical Statistics*, 25(1):123–130.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197.

Dette, H., Kunert, J., and Pepelyshev, A. (2008). Exact optimal designs for weighted least squares analysis with correlated errors. *Statistica Sinica*, pages 135–154.

Diringer, M., Edwards, D., Mattson, D., Akins, P., Sheedy, C., Hsu, C., and Dromerick, A. (1999). Predictors of acute hospital costs for treatment of ischemic stroke in an academic center. *Stroke*, 30(4):724–728.

Duarte, B. P., Sagnol, G., and Wong, W. K. (2018). An algorithm based on semidefinite programming for finding minimax optimal designs. *Computational Statistics & Data Analysis*, 119:99–117.

Eberhart, R. C. and Kennedy, J. (1995). Particle swarm optimization. In *Proceedings of the IEEE International Conference On Neural Networks*, volume 4, pages 1942–1948.

Eberhart, R. C. and Shi, Y. (1998). Comparison between genetic algorithms and particle swarm optimization. In *International Conference on Evolutionary Programming*, pages 611–616. Springer.

Eberhart, R. C. and Shi, Y. (2000). Comparing inertia weights and constriction factors in particle swarm optimization. In *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*, volume 1, pages 84–88. IEEE.

Ehrenfeld, S. (1955). On the efficiency of experimental designs. *The Annals of Mathematical Statistics*, 26(2):247–255.

Elfving, G. et al. (1952). Optimum allocation in linear regression theory. *The Annals of Mathematical Statistics*, 23(2):255–262.

Fan, H. (2002). A modification to particle swarm optimization algorithm. *Engineering Computations*, 19(8):970–989.

Fedorov, V. V. (1972). *Theory of optimal experiments*. Elsevier.

Ford, I. and Silvey, S. (1980). A sequentially constructed design for estimating a nonlinear parametric function. *Biometrika*, 67(2):381–388.

Gagnon, R. and Leonov, S. (2004). Optimal population designs for PK models with serial sampling. *Journal of Biopharmaceutical Statistics*, 15(1):143–163.

Gandomi, A. H., Yang, X.-S., and Alavi, A. H. (2013). Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. *Engineering with computers*, 29(1):17–35.

Gang, M., Wei, Z., and Xiaolin, C. (2012). A novel particle swarm optimization algorithm based on particle migration. *Applied Mathematics and Computation*, 218(11):6620–6626.

Ghamisi, P. and Benediktsson, J. A. (2015). Feature selection based on hybridization of genetic algorithm and particle swarm optimization. *IEEE Geoscience and Remote Sensing Letters*, 12(2):309–313.

Grimshaw, S. D., Collings, B. J., Larsen, W. A., and Hurt, C. R. (2001). Eliciting factor importance in a designed experiment. *Technometrics*, 43(2):133–146.

Gross, D., Conrad, B., Fogg, L., and Wothke, W. (1994). A longitudinal model of maternal self-efficacy, depression, and difficult temperament during toddlerhood. *Research in Nursing & Health*, 17(3):207–215.

Gu, S., Cheng, R., and Jin, Y. (2018). Feature selection for high-dimensional classification using a competitive swarm optimizer. *Soft Computing*, 22(3):811–822.

Haines, L. M., Kabera, G., and O'Brien, T. E. (2007). *D*-optimal designs for logistic regression in two variables. In *mODa 8-Advances in Model-Oriented Design and Analysis*, pages 91–98. Springer.

Haines, L. M. and Kabera, G. M. (2018). *D*-optimal designs for the two-variable binary logistic regression model with interaction. *Journal of Statistical Planning and Inference*, 193:136–150.

Han, C. and Chaloner, K. (2003). *D*-and *c*-optimal designs for exponential regression models used in viral dynamics and other applications. *Journal of Statistical Planning and Inference*, 115(2):585–601.

Han, C. and Chaloner, K. (2004). Bayesian experimental design for nonlinear mixed-effects models with application to hiv dynamics. *Biometrics*, 60(1):25–33.

Han, C., Chaloner, K., and Perelson, A. S. (2002). Bayesian analysis of a population hiv dynamic model. In *Case Studies in Bayesian Statistics*, pages 223–237. Springer.

Healy, B. C., Ikle, D., Macklin, E. A., and Cutter, G. (2010). Optimal design and analysis of phase I/II clinical trials in multiple sclerosis with gadolinium-enhanced lesions as the endpoint. *Multiple Sclerosis*, pages 840–847.

Heise, M. A. and Myers, R. H. (1996). Optimal designs for bivariate logistic regression. *Biometrics*, pages 613–624.

Heredia-Langner, A., Carlyle, W. M., Montgomery, D. C., Borror, C. M., and Runger, G. C. (2003). Genetic algorithms for the construction of *D*-optimal designs. *Journal of Quality Technology*, 35(1):28–46.

Higashi, N. and Iba, H. (2003). Particle swarm optimization with gaussian mutation. In *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE*, pages 72–79. IEEE.

Hoel, P. G. (1961). Some properties of optimal spacing in polynomial estimation. *Annals of the Institute of Statistical Mathematics*, 13(1):1–8.

Hsieh, S.-T., Sun, T.-Y., Liu, C.-C., and Tsai, S.-J. (2008). Solving large scale global optimization using improved particle swarm optimizer. In *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress*, pages 1777–1784. IEEE.

Hu, M., Deng, K., Selvaraj, S., Qin, Z., Ren, B., and Liu, J. S. (2012). HiCNorm: removing biases in Hi-C data via poisson regression. *Bioinformatics*, 28(23):3131–3133.

Hu, W., Tong, S., Mengersen, K., and Connell, D. (2007). Weather variability and the incidence of cryptosporidiosis: comparison of time series poisson regression and sarima models. *Annals of Epidemiology*, 17(9):679–688.

Huang, S.-H., Huang, M.-N. L., and Lin, C.-W. (2019). Optimal designs for binary response models with multiple nonnegative variables. *Journal of Statistical Planning and Inference*.

Ishaque, K. and Salam, Z. (2013). A deterministic particle swarm optimization maximum power point tracker for photovoltaic system under partial shading condition. *IEEE Transactions on Industrial Electronics*, 60(8):3195–3206.

Ishaque, K., Salam, Z., Amjad, M., and Mekhilef, S. (2012). An improved particle swarm optimization (pso)–based mppt for pv with reduced steady-state oscillation. *IEEE Transactions on Power Electronics*, 27(8):3627–3638.

Jang, W. and Lim, J. (2009). A numerical study of pql estimation biases in generalized linear mixed models under heterogeneity of random effects. *Communications in Statistics-Simulation and Computation*, 38(4):692–702.

Jia, Y. and Myers, R. (2001). Optimal experimental designs for two-variable logistic models.

Jiang, H.-Y., Yue, R.-X., and Zhou, X.-D. (2019). Optimal designs for multivariate logistic mixed models with longitudinal data. *Communications in Statistics-Theory and Methods*, 48(4):850–864.

John, R. S. and Draper, N. R. (1975). D-optimality for regression designs: a review. *Technometrics*, 17(1):15–23.

Jones, B. and Goos, P. (2007). A candidate-set-free algorithm for generating D-optimal split-plot designs. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 56(3):347–364.

Jørgensen, H. S., Nakayama, H., Raaschou, H. O., and Olsen, T. S. (1997). Acute stroke care and rehabilitation: an analysis of the direct cost and its clinical and social determinants: the copenhagen stroke study. *Stroke*, 28(6):1138–1141.

Kauhl, B., Heil, J., Hoebe, C. J., Schweikart, J., Krafft, T., and Dukers-Muijrers, N. H. (2015). The spatial distribution of hepatitis c virus infections and associated determinants—an application of a geographically weighted poisson regression for evidence-based screening interventions in hotspots. *PloS one*, 10(9):e0135656.

Kennedy, J. (1999). Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, volume 3, pages 1931–1938. IEEE.

Kennedy, J. and Mendes, R. (2002). Population structure and particle swarm performance. In *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, volume 2, pages 1671–1676. IEEE.

Kiefer, J. (1961). Optimum experimental designs v, with applications to systematic and rotatable designs. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 381–405. Univ of California Press.

Kiefer, J. et al. (1961). Optimum designs in regression problems, ii. *The Annals of Mathematical Statistics*, 32(1):298–325.

Kiefer, J. et al. (1962). Two more criteria equivalent to D-optimality of designs. *The Annals of Mathematical Statistics*, 33(2):792–796.

Kiefer, J. and Wolfowitz, J. (1959). Optimum designs in regression problems. *The Annals of Mathematical Statistics*, pages 271–294.

Kiefer, J. and Wolfowitz, J. (1960). The equivalence of two extremum problems. *Canadian Journal of Mathematics*, 12(363-366):234.

Kiefer, J., Wolfowitz, J., et al. (1959). Optimum designs in regression problems. *The Annals of Mathematical Statistics*, 30(2):271–294.

King, J. and Wong, W.-K. (2000). Minimax D-optimal designs for the logistic model. *Biometrics*, 56(4):1263–1267.

Krier, F., Brion, M., Debrus, B., Lebrun, P., Driesen, A., Ziemons, E., Evrard, B., and Hubert, P. (2011). Optimisation and validation of a fast hplc method for the quantification of sulindac and its related impurities. *Journal of Pharmaceutical and Biomedical Analysis*, 54(4):694–700.

Kumarappan, N. and Arulraj, R. (2016). Optimal installation of multiple dg units using competitive swarm optimizer (cso) algorithm. In *Evolutionary Computation (CEC), 2016 IEEE Congress on*, pages 3955–3960. IEEE.

Kurt, I., Ture, M., and Kurum, A. T. (2008). Comparing performances of logistic regression, classification and regression tree, and neural networks for predicting coronary artery disease. *Expert Systems with Applications*, 34(1):366–374.

Lall, S., Jaggi, S., Varghese, E., Varghese, C., and Bhowmik, A. (2018). An algorithmic approach to construct D-optimal saturated designs for logistic model. *Journal of Statistical Computation and Simulation*, 88(6):1191–1199.

Lambert, D. (1992). Zero-inflated poisson regression, with an application to defects in manufacturing. *Technometrics*, 34(1):1–14.

Lampinen, J. (2002). A constraint handling approach for the differential evolution algorithm. In *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, volume 2, pages 1468–1473. IEEE.

Lee, A. H., Wang, K., Yau, K. K., and Somerford, P. J. (2003). Truncated negative binomial mixed regression modelling of ischaemic stroke hospitalizations. *Statistics in Medicine*, 22(7):1129–1139.

Leung, Y.-W. and Wang, Y. (2001). An orthogonal genetic algorithm with quantization for global numerical optimization. *IEEE Transactions on Evolutionary Computation*, 5(1):41–53.

Li, G. and Majumdar, D. (2008). *D*-optimal designs for logistic models with three and four parameters. *Journal of Statistical Planning and Inference*, 138(7):1950–1959.

Li, X. and Yao, X. (2012). Cooperatively coevolving particle swarms for large scale optimization. *IEEE Transactions on Evolutionary Computation*, 16(2):210–224.

Liang, J.-J. and Suganthan, P. N. (2005). Dynamic multi-swarm particle swarm optimizer with local search. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, volume 1, pages 522–528. Ieee.

Liu, B., Wang, L., Jin, Y.-H., Tang, F., and Huang, D.-X. (2005). Improved particle swarm optimization combined with chaos. *Chaos, Solitons & Fractals*, 25(5):1261–1271.

Liu, C., Du, W.-B., and Wang, W.-X. (2014). Particle swarm optimization with scale-free interactions. *PloS One*, 9(5):e97822.

Liu, X., Yue, R.-X., and Wong, W. K. (2019). *D*-optimal designs for multi-response linear mixed models. *Metrika*, 82(1):87–98.

Long, J. and Ryoo, J. (2010). Using fractional polynomials to model non-linear trends in longitudinal data. *British Journal of Mathematical and Statistical Psychology*, 63(1):177–203.

Lukemire, J., Mandal, A., and Wong, W. K. (2018). d-qpso: A quantum-behaved particle swarm technique for finding *D*-optimal designs with discrete and continuous factors and a binary response. *Technometrics*, (just-accepted):1–27.

Martín, R. M. and Gutiérrez, I. G. C. (2015). Combined algorithm to compute D-optimal designs. *Journal of Computational and Applied Mathematics*, 278:248–257.

McDermott, J. (2020). When and why metaheuristics researchers can ignore "no free lunch" theorems. *SN Computer Science*, page In press.

Meng, K., Wang, H. G., Dong, Z., and Wong, K. P. (2010). Quantum-inspired particle swarm optimization for valve-point economic load dispatch. *IEEE Transactions on Power Systems*, 25(1):215–222.

Meyer, R. K. and Nachtsheim, C. J. (1995). The coordinate-exchange algorithm for constructing exact optimal experimental designs. *Technometrics*, 37(1):60–69.

Miller, B. L., Goldberg, D. E., et al. (1995). Genetic algorithms, tournament selection, and the effects of noise. *Complex Systems*, 9(3):193–212.

Mohapatra, P., Das, K. N., and Roy, S. (2017). A modified competitive swarm optimizer for large scale optimization problems. *Applied Soft Computing*, 59:340–362.

Moore, J. and Chapman, R. (1999). Application of particle swarm to multiobjective optimization. *Department of Computer Science and Software Engineering, Auburn University*, 32.

Morris, G. M., Goodsell, D. S., Halliday, R. S., Huey, R., Hart, W. E., Belew, R. K., Olson, A. J., et al. (1998). Automated docking using a lamarckian genetic algorithm and an empirical binding free energy function. *Journal of Computational Chemistry*, 19(14):1639–1662.

Murrough, J. W., Iosifescu, D. V., Chang, L. C., Al Jurdi, R. K., Green, C. E., Perez, A. M., Iqbal, S., Pillemer, S., Foulkes, A., Shah, A., et al. (2013). Antidepressant efficacy of ketamine in treatment-resistant major depression: a two-site randomized controlled trial. *American Journal of Psychiatry*, 170(10):1134–1142.

Naderi, D., Niaparast, M., and Zangenehmehr, A. (2018). *D*-optimal designs for multiple poisson regression model with random coefficients. *Asian Research Journal of Mathematics*, pages 1–11.

Nakisa, B., Rastgoo, M. N., Norodin, M. J., et al. (2014). Balancing exploration and exploitation in particle swarm optimization on search tasking. *Research Journal of Applied Sciences, Engineering and Technology*, 8(12):1429–1434.

Nezami, O. M., Bahrampour, A., and Jamshidlou, P. (2013). Dynamic diversity enhancement in particle swarm optimization (ddepso) algorithm for preventing from premature convergence. *Procedia Computer Science*, 24:54–65.

Nguyen, N.-K. and Miller, A. J. (1992). A review of some exchange algorithms for constructing discrete *D*-optimal designs. *Computational Statistics & Data Analysis*, 14(4):489–498.

Nowak, M. and May, R. M. (2000). *Virus dynamics: mathematical principles of immunology and virology: mathematical principles of immunology and virology*. Oxford University Press, UK.

Ogungbenro, K., Dokoumetzidis, A., and Aarons, L. (2009). Application of optimal design methodologies in clinical pharmacology experiments. *Pharmaceutical Statistics: The Journal of Applied Statistics in the Pharmaceutical Industry*, 8(3):239–252.

Olorunda, O. and Engelbrecht, A. P. (2008). Measuring exploration/exploitation in particle swarms using swarm diversity. In *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, pages 1128–1134. IEEE.

Ormel, J. and Wohlfarth, T. (1991). How neuroticism, long-term difficulties, and life situation change influence psychological distress: a longitudinal model. *Journal of Personality and Social Psychology*, 60(5):744.

Ouwens, M. J., Tan, F. E., and Berger, M. P. (2006). A maximin criterion for the logistic random intercept model with covariates. *Journal of Statistical Planning and Inference*, 136(3):962–981.

Overstall, A. M., McGree, J. M., and Drovandi, C. C. (2017). An approach for finding fully bayesian optimal designs using normal-based approximations to loss functions. *Statistics and Computing*, pages 1–16.

Overstall, A. M. and Woods, D. C. (2017). Bayesian design of experiments using approximate coordinate exchange. *Technometrics*, 59(4):458–470.

Pázman, A. (1980). Some features of the optimal design theory–a survey. *Statistics: A Journal of Theoretical and Applied Statistics*, 11(3):415–446.

Pázman, A. (1986). *Foundations of optimum experimental design*, volume 14. Springer.

Pehlivanoglu, Y. V. (2013). A new particle swarm optimization method enhanced with a periodic mutation strategy and neural networks. *IEEE Transactions on Evolutionary Computation*, 17(3):436–452.

Prus, M. and Schwabe, R. (2016). Optimal designs for the prediction of individual parameters in hierarchical models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(1):175–191.

Puškaš, V. and Miljić, U. (2012). The application of D-optimal design for modelling the red wine ageing process. *Food Control*, 28(2):362–367.

Qiu, J., Chen, R.-B., Wang, W., and Wong, W. K. (2014). Using animal instincts to design efficient biomedical studies via particle swarm optimization. *Swarm and Evolutionary Computation*, 18:1–10.

Ranga, S., Jaimini, M., Sharma, S. K., Chauhan, B. S., and Kumar, A. (2014). A review on design of experiments (doe). *Int. J. Pharm. Chem. Sci*, 3(1):216–24.

Ratnaweera, A., Halgamuge, S. K., and Watson, H. C. (2004). Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. *IEEE Transactions on Evolutionary Computation*, 8(3):240–255.

Ribba, B., Holford, N. H., Magni, P., Trocóniz, I., Gueorguieva, I., Girard, P., Sarr, C., Elishmereni, M., Kloft, C., and Friberg, L. E. (2014). A review of mixed-effects models of tumor growth and effects of anticancer drug treatment used in population analysis. *CPT: Pharmacometrics & Systems Pharmacology*, 3(5):1–10.

Robinson, J., Sinton, S., and Rahmat-Samii, Y. (2002). Particle swarm, genetic algorithm, and their hybrids: optimization of a profiled corrugated horn antenna. In *Antennas and Propagation Society International Symposium, 2002. IEEE*, volume 1, pages 314–317. IEEE.

Rodríguez-Torreblanca, C. and Rodríguez-Díaz, J. (2007). Locally D-and c-optimal designs for poisson and negative binomial regression models. *Metrika*, 66(2):161–172.

Ros, R. and Hansen, N. (2008). A simple modification in cma-es achieving linear time and space complexity. In *International Conference on Parallel Problem Solving from Nature*, pages 296–305. Springer.

Royston, P. and Altman, D. G. (1994). Regression using fractional polynomials of continuous covariates: parsimonious parametric modelling. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 43(3):429–453.

Royston, P. and Wright, E. M. (1998). A method for estimating age-specific reference intervals ('normal ranges') based on fractional polynomials and exponential transformation. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 161(1):79–101.

Rundek, T., Mast, H., Hartmann, A., Boden-Albala, B., Lennihan, L., Lin, I.-F., Paik, M., and Sacco, R. (2000). Predictors of resource use after acute hospitalization: the northern manhattan stroke study. *Neurology*, 55(8):1180–1187.

Ryan, E. G., Drovandi, C. C., McGree, J. M., and Pettitt, A. N. (2016). A review of modern computational algorithms for bayesian optimal design. *International Statistical Review*, 84(1):128–154.

Sambo, F., Borrotti, M., and Mylona, K. (2014). A coordinate-exchange two-phase local search algorithm for the D-and i-optimal designs of split-plot experiments. *Computational Statistics & Data Analysis*, 71:1193–1207.

Schmelter, T., Benda, N., and Schwabe, R. (2007). Some curiosities in optimal designs for random slopes. In *mODa 8-Advances in Model-Oriented Design and Analysis*, pages 189–195. Springer.

Sebastiani, P. and Settimi, R. (1997). A note on D-optimal designs for a logistic regression model. *Journal of Statistical Planning and Inference*, 59(2):359–368.

Shi, Y. and Eberhart, R. C. (1998). Parameter selection in particle swarm optimization. In *International Conference on Evolutionary Programming*, pages 591–600. Springer.

Shi, Y. and Eberhart, R. C. (2001). Fuzzy adaptive particle swarm optimization. In *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, volume 1, pages 101–106. IEEE.

Sitter, R. and Wu, C. (1993). Optimal designs for binary response experiments: Fieller, d, and a criteria. *Scandinavian Journal of Statistics*, pages 329–341.

Sitter, R. R. and Torsney, B. (1995). D-optimal designs for generalized linear models. In *MODA4—Advances in Model-Oriented Data Analysis*, pages 87–102. Springer.

Smith, K. (1918). On the standard deviations of adjusted and interpolated values of an observed polynomial function and its constants and the guidance they give towards a proper choice of the distribution of observations. *Biometrika*, 12(1/2):1–85.

Stein, L. H., Berger, J., Tranquilli, M., and Elefteraides, J. A. (2013). Effect of statin drugs on thoracic aortic aneurysms. *The American Journal of Cardiology*, 112(8):1240–1245.

Suganthan, P. N. (1999). Particle swarm optimiser with neighbourhood operator. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, volume 3, pages 1958–1962. IEEE.

Sun, C., Ding, J., Zeng, J., and Jin, Y. (2016). A fitness approximation assisted competitive swarm optimizer for large scale expensive optimization problems. *Memetic Computing*, pages 1–12.

Sun, J., Palade, V., Wu, X.-J., Fang, W., and Wang, Z. (2014). Solving the power economic dispatch problem with generator constraints by random drift particle swarm optimization. *IEEE Transactions on Industrial Informatics*, 10(1):222–232.

Syahputra, R. (2017). Distribution network optimization based on genetic algorithm. *Journal of Electrical Technology UMY*, 1(1):1–9.

Tan, F. E. and Berger, M. P. (1999). Optimal allocation of time points for the random effects model. *Communications in Statistics-Simulation and Computation*, 28(2):517–540.

Tang, K., Yáo, X., Suganthan, P. N., MacNish, C., Chen, Y.-P., Chen, C.-M., and Yang, Z. (2007). Benchmark functions for the cec'2008 special session and competition on large scale global optimization. *Nature Inspired Computation and Applications Laboratory, USTC, China*, 24.

Taormina, R. and Chau, K.-W. (2015). Data-driven input variable selection for rainfall–runoff modeling using binary-coded particle swarm optimization and extreme learning machines. *Journal of Hydrology*, 529:1617–1632.

Tekle, F. B., Tan, F. E., and Berger, M. P. (2008a). Maximin D-optimal designs for binary longitudinal responses. *Computational Statistics & Data Analysis*, 52(12):5253–5262.

Tekle, F. B., Tan, F. E., and Berger, M. P. (2008b). D-optimal cohort designs for linear mixed-effects models. *Statistics in Medicine*, 27(14):2586–2600.

Tian, J., Yu, W., and Xie, S. (2008). An ant colony optimization algorithm for image edge detection. In *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*, pages 751–756. IEEE.

Titterington, D. (1976). Algorithms for computing D-optimal designs on a finite design space. In *Proc. of the 1976 Conf. on Information Science and Systems, John Hopkins University*, volume 3, pages 213–216.

Titterington, D. (1978). Estimation of correlation coefficients by ellipsoidal trimming. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 27(3):227–234.

Trelea, I. C. (2003). The particle swarm optimization algorithm: convergence analysis and parameter selection. *Information Processing Letters*, 85(6):317–325.

Ugolotti, R., Nashed, Y. S., Mesejo, P., Ivekovič, Š., Mussi, L., and Cagnoni, S. (2013). Particle swarm optimization and differential evolution for model-based object detection. *Applied Soft Computing*, 13(6):3092–3105.

Valian, E., Mohanna, S., and Tavakoli, S. (2011). Improved cuckoo search algorithm for feedforward neural network training. *International Journal of Artificial Intelligence & Applications*, 2(3):36–43.

Verotta, D., Petrillo, P., La Regina, A., Rocchetti, M., and Tavani, A. (1988). D-optimal design applied to binding saturation curves of an enkephalin analog in rat brain. *Life Sciences*, 42(6):735–743.

Wald, A. (1943). On the efficient design of statistical investigations. *The Annals of Mathematical Statistics*, 14(2):134–140.

Wang, K., Yau, K. K., and Lee, A. H. (2002). A hierarchical poisson mixture regression model to analyse maternity length of hospital stay. *Statistics in Medicine*, 21(23):3639–3654.

White, L. V. (1973). An extension of the general equivalence theorem to nonlinear models. *Biometrika*, 60(2):345–348.

White, L. V. (1975). The optimal design of experiments for estimation in non-linear models.

Whitley, D. (1994). A genetic algorithm tutorial. *Statistics and Computing*, 4(2):65–85.

Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82.

Wong, W. K. and Cook, R. D. (1993). Heteroscedastic G-optimal designs. *Journal of the Royal Statistical Society: Series B (Methodological)*, 55(4):871–880.

Wu, C. (1985). Asymptotic inference from sequential design in a nonlinear situation. *Biometrika*, 72(3):553–558.

Xinchao, Z. (2010). A perturbed particle swarm algorithm for numerical optimization. *Applied Soft Computing*, 10(1):119–124.

Xiong, G. and Shi, D. (2018). Orthogonal learning competitive swarm optimizer for economic dispatch problems. *Applied Soft Computing*.

Xu, G., Wu, Z.-H., and Jiang, M.-Z. (2015). Premature convergence of standard particle swarm optimisation algorithm based on markov chain analysis. *International Journal of Wireless and Mobile Computing*, 9(4):377–382.

Yang, M., Biedermann, S., and Tang, E. (2013). On optimal designs for nonlinear models: a general and efficient algorithm. *Journal of the American Statistical Association*, 108(504):1411–1420.

Yang, Q., Chen, W.-N., Gu, T., Zhang, H., Yuan, H., Kwong, S., and Zhang, J. (2019). A distributed swarm optimizer with adaptive communication for large-scale optimization. *IEEE Transactions on Cybernetics*.

Yang, X.-S. and Deb, S. (2009). Cuckoo search via lévy flights. In *Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on*, pages 210–214. IEEE.

Yang, X.-S. and Deb, S. (2014). Cuckoo search: recent advances and applications. *Neural Computing and Applications*, 24(1):169–174.

Yang, Z., Tang, K., and Yao, X. (2008a). Multilevel cooperative coevolution for large scale optimization. In *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*, pages 1663–1670. IEEE.

Yang, Z., Tang, K., and Yao, X. (2008b). Self-adaptive differential evolution with neighborhood search. In *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*, pages 1110–1116. IEEE.

Yoshida, H., Kawata, K., Fukuyama, Y., Takayama, S., and Nakanishi, Y. (2000). A particle swarm optimization for reactive power and voltage control considering voltage security assessment. *IEEE Transactions on Power Systems*, 15(4):1232–1239.

Yu, Y. (2011). *D*-optimal designs via a cocktail algorithm. *Statistics and Computing*, 21(4):475–481.

Zhang, Q., Cheng, H., Ye, Z., and Wang, Z. (2017). A competitive swarm optimizer integrated with cauchy and gaussian mutation for large scale optimization. In *Control Conference (CCC), 2017 36th Chinese*, pages 9829–9834. IEEE.

Zhang, W.-X., Chen, W.-N., and Zhang, J. (2016). A dynamic competitive swarm optimizer based-on entropy for large scale optimization. In *Advanced Computational Intelligence (ICACI), 2016 Eighth International Conference on*, pages 365–371. IEEE.

Zhou, J., Fang, W., Wu, X., Sun, J., and Cheng, S. (2016). An opposition-based learning competitive particle swarm optimizer. In *Evolutionary Computation (CEC), 2016 IEEE Congress on*, pages 515–521. IEEE.

Zhou, X.-D., Wang, Y.-J., and Yue, R.-X. (2018). Robust population designs for longitudinal linear regression model with a random intercept. *Computational Statistics*, 33(2):903–931.