Technometrics

ISSN: 0040-1706 (Print) 1537-2723 (Online) Journal homepage: https://www.tandfonline.com/loi/utch20

d-QPSO: A Quantum-Behaved Particle Swarm Technique for Finding *D*-Optimal Designs With Discrete and Continuous Factors and a Binary Response

Joshua Lukemire, Abhyuday Mandal & Weng Kee Wong

To cite this article: Joshua Lukemire, Abhyuday Mandal & Weng Kee Wong (2019) *d*-QPSO: A Quantum-Behaved Particle Swarm Technique for Finding *D*-Optimal Designs With Discrete and Continuous Factors and a Binary Response, Technometrics, 61:1, 77-87, DOI: 10.1080/00401706.2018.1439405

To link to this article: https://doi.org/10.1080/00401706.2018.1439405

TECHNOMETRICS
2019, VOL. 61, NO. 1, 77–87
https://doi.org/10.1080/00401706.2018.1439405

Taylor & Francis
Taylor & Francis Group

d-QPSO: A Quantum-Behaved Particle Swarm Technique for Finding *D*-Optimal Designs With Discrete and Continuous Factors and a Binary Response

Joshua Lukemire[a], Abhyuday Mandal[b], and Weng Kee Wong[c]

[a]Department of Biostatistics and Bioinformatics, Emory University, Atlanta, GA; [b]Department of Statistics, University of Georgia, Athens, GA; [c]Department of Biostatistics, University of California at Los Angeles, Los Angeles, CA

ABSTRACT

Identifying optimal designs for generalized linear models with a binary response can be a challenging task, especially when there are both discrete and continuous independent factors in the model. Theoretical results rarely exist for such models, and for the handful that do, they usually come with restrictive assumptions. In this article, we propose the *d*-QPSO algorithm, a modified version of quantum-behaved particle swarm optimization, to find a variety of *D*-optimal approximate and exact designs for experiments with discrete and continuous factors and a binary response. We show that the *d*-QPSO algorithm can efficiently find locally *D*-optimal designs even for experiments with a large number of factors and robust pseudo-Bayesian designs when nominal values for the model parameters are not available. Additionally, we investigate robustness properties of the *d*-QPSO algorithm-generated designs to various model assumptions and provide real applications to design a bio-plastics odor removal experiment, an electronic static experiment, and a 10-factor car refueling experiment. Supplementary materials for the article are available online.

ARTICLE HISTORY
Received July 2016
Revised January 2018

KEYWORDS
Approximate design; Design efficiency; Equivalence theorem; Exact design; Pseudo-Bayesian design

1. Introduction

Our work is motivated by an odor removal study (Wang et al. 2016) conducted in the Department of Textiles, Merchandising, and Interiors within the College of Family and Consumer Sciences at the University of Georgia (UGA). In the research, protein-rich algae were used in the creation of bio-plastic. In bio-plastic formulation, algae undergo some chemical processing so that their proteins have desired properties. As a side effect of this chemical processing, algae-based bio-products often have unpleasant odors, which must be removed or at least significantly diminished if the products are to be used for commercial purposes. Wang's experiment was carried out to determine the effect of several processing steps on the presence or absence of an unpleasant odor.

Table 1 displays the factors thought to be relevant to bio-plastic odor. The researchers designed their experiment investigating the four discrete factors and implemented a design assuming a constant temperature of 25°C. There is currently no known optimal design for such an experiment with both discrete and continuous factors, and this may explain why the researchers at UGA decided to fix the temperature at one arbitrary level. The experiment has a binary response Y denoting whether the odor is successfully removed from the bio-plastic. In this work, we revisit this experiment and consider designs incorporating all four discrete factors and storage temperature as a continuous variable. We model μ, the mean response of Y, using logistic regression and find optimal designs for estimating all parameters in the model. We refer to such studies with both discrete and continuous factors as having "mixed factors."

Generalized linear models (GLMs) are widely used to model the mean response of a Bernoulli random variable. Let Y_l be the response corresponding to the lth combination of factor levels, $\boldsymbol{x}_l = (1, x_{l,1}, x_{l,2}, \ldots)$, that may include integration terms among the factors. Without loss of generality, for an experiment with k factors, we assume the first term of $\boldsymbol{x}_l$ corresponds to the intercept, the next k terms to the factor settings, and any remaining terms to interactions among the factors. Accordingly, the 2nd through the $(k+1)$th terms in $\boldsymbol{x}_l$ become a support point of the design and the collection of all such points constitutes the experimental design, $\boldsymbol{\psi}$.

In the GLM, the mean response μ_l of Y_l is related to the linear predictor $\eta_l = \boldsymbol{x}_l^T\boldsymbol{\beta}$ by a monotonic link function $g(\cdot)$ via $g(\mu_l) = \eta_l$, with the logit and probit links being two of the most commonly used when the response is binary. Here, the range of values for each factor is assumed to be known. This implies that we have a known design space where combination levels of the factors can be selected to observe the response.

The exact design problem is to determine the optimal number of support points (L), the support points themselves, and the optimal number of replicates, $n_1, \ldots, n_L$, subject to the constraint $n_1 + \cdots + n_L = N$. The value N is the known total number of observations for the study and is predetermined either by the duration or cost of the study. Alternatively, an approximate design optimizes the proportion of the total number of observations at each support point subject to the constraint that they sum to unity (Kiefer 1959). For such designs, we relax the assumption that each proportion $p_l = n_l/N$ is a ratio of two positive integers between 0 and 1 and implement the approximate

CONTACT Abhyuday Mandal amandal@stat.uga.edu Department of Statistics, University of Georgia, Athens, GA.
Color versions of one or more of the figures in this article are available online at www.tandfonline.com/r/TECH.
Supplementary materials for this article are available online. Please go to http://www.tandfonline.com/r/TECH

Table 1. Factor types and levels for the bio-plastic odor removal experiment.

Type	Factor	Levels	
		−	+
Discrete	Algae	Catfish algae	Solix Microalgae
	Scavenger	Activated carbon	Zeolite
	Resin	Polyethylene	Polypropylene
	Compatibilizer	Absent	Present
Continuous	Temperature	Temperature from 5°C to 35°C	

design by taking roughly Np_l observations at each $\boldsymbol{x}_l$ subject to the requirement that they sum to N and each Np_l is an integer.

Let $\boldsymbol{\psi}$ be a design with support points at $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_L$ for which there are n_l replicates at each $\boldsymbol{x}_l$. A direct calculation shows that the Fisher information matrix is

$$\boldsymbol{I}_{\psi} = \sum_{l=1}^{L} n_l \Upsilon(\eta_l) \boldsymbol{x}_l \boldsymbol{x}_l^T,$$

where $\Upsilon(\eta_l) = \frac{(d\mu_l/d\eta_l)^2}{\mu_l(1-\mu_l)}$. A D-optimal design maximizes the log-determinant of the Fisher information matrix and so it is appropriate for estimating all parameters in the model. Because the Fisher information matrix depends on the model parameters, nominal values of the parameters are required before we can implement the design. Such nominal values typically come from the literature or pilot studies. The resulting optimal designs are therefore locally optimal and are often used as building blocks for constructing more complicated designs (Ford, Torsney, and Wu 1992) or as benchmarks for other designs when a single best guess of the parameters is available (Stufken and Yang 2012).

When the design criterion is a concave function of the Fisher information matrix, such as D-optimality, we verify the optimality of an approximate design among all designs using an equivalence theorem, see, for example, Kiefer and Wolfowitz (1959) or Pukelsheim (1993). For the logistic model with q parameters in the linear predictor, this theorem asserts that the design $\boldsymbol{\psi}^*$ is locally D-optimal among all designs if and only if for all $\boldsymbol{x}$ in the design space,

$$\frac{\exp\{\boldsymbol{\beta}^T\boldsymbol{x}\}}{(1+\exp\{\boldsymbol{\beta}^T\boldsymbol{x}\})^2}\, \boldsymbol{x}^T\, \boldsymbol{I}_{\psi^*}^{-1}\, \boldsymbol{x} - q \leq 0, \tag{1}$$

with equality at each support point of the design $\boldsymbol{\psi}^*$. The function to the left of the above inequality is sometimes called the sensitivity function.

Often the worth of a design is measured by its efficiency relative to the optimal design, $\boldsymbol{\psi}^*$. If $\boldsymbol{\psi}$ is a design of interest and $\boldsymbol{\psi}^*$ is a locally D-optimal design for a GLM with q parameters in the linear predictor, the D-efficiency of $\boldsymbol{\psi}$ is

$$\left(\frac{\det(\boldsymbol{I}_{\psi})}{\det(\boldsymbol{I}_{\psi^*})}\right)^{1/q}. \tag{2}$$

If the ratio is one half, the design $\boldsymbol{\psi}$ requires twice as many replicates as the locally D-optimal design to obtain the same information. When the true optimum design is unknown, a lower bound on the D-efficiency of $\boldsymbol{\psi}$ is $\exp\{-\theta/q\}$, where θ is the maximum positive value of the sensitivity function across the design space (Pazman 1986). Clearly, $\theta = 0$ if and only if $\boldsymbol{\psi}$ is locally D-optimal, and the lower bound attains unity. We refer to the quantity in the numerator of (2), $\det(\boldsymbol{I}_{\psi})^{1/q}$, as the objective function value for the design and report its value for comparing different designs.

Atkinson and Woods (2015) provided an overview of design issues for generalized linear models. Some theoretical results exist for models with all discrete factors (Yang, Mandal, and Majumdar 2016) or all continuous factors (Yang, Zhang, and Huang 2011). When theoretical results are not available, computational methods are used to find optimal designs. Mandal, Wong, and Yu (2015) provided an overview of algorithms for generating optimal designs, including use of nature-inspired metaheuristic algorithms for finding a large class of optimal designs. Early techniques for generating optimal designs for experiments with outcomes modeled under GLMs include Fedorov–Wynn type algorithms (Fedorov 1972) and multiplicative algorithms (Titterington 1976). These approaches remain popular and often form the basis for more recent techniques such as the cocktail algorithm (Yu 2011). Specific applications of computational methods to solve real design problems for GLMs can be found in Woods et al. (2006), Dror and Steinberg (2006, 2008), Waterhouse et al. (2008), and Woods and van de Ven (2011).

There is little work on constructing efficient designs for experiments with mixed factors; a reason may be that the theory and algorithms for constructing D-optimal designs for GLMs when all factors are continuous or when all factors are discrete do not directly extend to the case when there are mixed factors. As far as we know, there is no efficient algorithm for finding D-optimal designs for such models. Some algorithms, such as quasi-Newton BFGS (Nocedal and Wright 1999), may be used to solve these mixed-factor design problems by optimizing the continuous factor levels and proportions for each fixed combination of discrete factor levels, but such approaches can be computationally inefficient (see Section 3.3). One common approach to these mixed factor problems is to discretize the continuous factors into a few levels and apply algorithms for studies with all discrete factors. However, the generated design is unlikely to be locally D-optimal for the original problem if the discretization is too coarse, as demonstrated in Section 3.2. Haphazard discretization of the continuous factors could also cause separation issues during analysis. This means that valid maximum likelihood estimates of the parameters do not exist because there is a hyperplane in the linear predictors that can perfectly separate the responses into two categories. More complicated and specialized estimation techniques, such as penalized maximum likelihood (Firth 1993; Heinze and Schemper 2002; Woods and van de Ven 2011; Woods, McGree, and Lewis 2017), or a modified logistic regression, such as a hidden logistic regression (Rousseeuw and Christmann 2003), will be required to produce meaningful parameter estimates.

The primary aim of this article is to propose a new method for finding D-optimal designs for GLMs with mixed factors and a binary outcome using quantum particle swarm optimization (QPSO). This QPSO is a nature-inspired metaheuristic algorithm based on particle swarm optimization (PSO), which is already widely used in engineering and computer science to tackle complicated optimization problems. A key advantage of working with PSO-type algorithms is that they require only an objective function which can be explicitly written down, and the design space does not have to be discretized. The latter

property is particularly useful when we design a study with multiple continuous factors.

This article is organized as follows. Section 2 first provides a brief review of PSO and QPSO before we describe our proposed d-QPSO algorithm. In Section 3, we apply the d-QPSO algorithm to find locally D-optimal designs for several real world problems. Section 4 demonstrates the flexibility of the d-QPSO algorithm to find pseudo-Bayesian D-optimal designs for situations in which the nominal values might be unknown. Section 5 summarizes our work with remarks on other possible applications of the d-QPSO algorithm. In the supplementary materials, we provide our d-QPSO algorithm code for the odor removal example and show how it may be used to investigate robustness properties of the D-optimal designs to violation of the model assumptions for the motivating example. We report computational time and accuracy of the d-QPSO-generated designs via simulations and also demonstrate that the proposed d-QPSO algorithm can be used to find exact optimal designs.

2. Swarm Optimization

We begin by briefly reviewing PSO and QPSO. We then describe how we modify QPSO to d-QPSO for finding D-optimal designs for models with mixed factors and a binary response.

2.1 Particle Swarm Type Algorithms

PSO is a metaheuristic optimization algorithm introduced by Kennedy and Eberhart (1995). It is a nature-inspired algorithm that mimics the behavior of a flock of birds as they search an area for food. Each member of the flock or swarm, known as a particle, represents a candidate solution to the problem of interest with a corresponding fitness, and the location of the food represents the optimum solution. Each particle has its own perception of where the food is located, based on its own experience. This position is known as the personal best position (pbest). Each particle is also aware of the overall best location that the flock has found, a position called the global best position (gbest). At each iteration, every particle moves in the direction of both its pbest position and the gbest position.

Since its inception, many variants of PSO have been developed, often to adapt PSO to a specific class of problems. For example, in public health research, Fu et al. (2009) used PSO to identify optimal screening nodes for spread of the SARS disease in Singapore; other applications are voluminously documented in the engineering literature, such as in the *IEEE Transactions*. Given its success in other application areas, PSO has also been modified to find optimal designs. Qiu et al. (2014) appeared to be the first to use the standard PSO to find a variety of optimal designs for biomedical problems, including optimal designs for estimating parameters in compartmental models and tumor growth models. It has also been used to find optimal designs under a nondifferentiable optimality criterion (Chen et al. 2015a), optimal designs for a variety of mixture models (Wong et al. 2015), optimal latin hypercube designs (Chen et al. 2013), and most recently, minimax projection designs (Mak and Joseph 2017). While the standard PSO is fast and effective for finding optimal designs for a few factors, it may not work very efficiently for complicated design problems, such as the case when we have moderate to large number of mixed factors in the regression model. This leads us to explore QPSO as a general optimization algorithm before modifying it to d-QPSO to specifically solve difficult design problems.

QPSO was developed after the trajectory analysis by Clerc and Kennedy (2002) where they showed that the swarm converges if each particle converges to the local attractor, which is defined as a point between the pbest and gbest positions in the standard PSO algorithm. QPSO was first introduced by Sun, Feng, and Xu (2004a), with the central idea that each particle can appear anywhere in the search space at any time, but has a higher probability to appear near its current position. This probabilistic scheme is unique to QPSO and justifies the use of the term "Quantum" in its name. Unlike the standard PSO, QPSO has no velocity term in its defining equations. Each particle's stochastic movement is accomplished by drawing positions from an exponential distribution with parameters determined by the distance between the particle and the best known positions. This probabilistic draw incorporates the local attractor for each particle and a position known as the "mainstream thought" or mbest, which is the average of all pbest positions at the current iteration (Sun, Xu, and Feng 2004b). By updating particle positions using both the local attractor and mbest (see Figure 1), QPSO is able to draw particles toward optimal positions without throwing away information from the particles with poor fitness values.

To our knowledge, this is the first work to apply QPSO to design experiments. The algorithm cannot simply be used off-the-shelf to find optimal designs because it assumes an unbounded search space and generally does not optimize discrete and continuous factors simultaneously. Thus, we develop a variant of QPSO for design of experiments and call it d-QPSO for short, where d stands for design. In particular, we modify the algorithm's behavior and use an elitist breeding mutation technique to maximize its performance for finding our sought-after designs. The overall idea of the d-QPSO algorithm is to generate multiple quantum-behaved particle swarms, where each swarm works to find the global best position. Information within swarms is shared just like in a typical QPSO algorithm, but with special attention paid to the nature of the covariates, the model structure, and the type of design space. Information is also pooled across swarms via an elitist breeding mutator without sacrificing the possibly distinct solutions obtained from different swarms.

2.2 d-QPSO: Algorithm Overview

Our proposed algorithm is a multi-swarm QPSO with elitist breeding that proceeds as follows. Suppose the design problem for our binary response experiment has k factors, and, to fix ideas, assume that all discrete factors have two levels. We first randomly generate s swarms each with w particles, where each particle is a design with L support points, and s, w, and L are user-selected positive integers. We refer to the collection of all swarms as a habitat. The components of each particle are the L support point settings and the proportion of observations at each support point. Thus each particle, $\boldsymbol{\psi}_i$, $i = 1, \ldots w$ has $L(k+1)$ elements and we search for an optimal design among them.

Within each swarm, $\boldsymbol{\psi}_i$, $i = 1, \ldots, w$ is generated randomly at the start of the search. We denote the best set of factor settings and proportion allocations found by particle i at iteration t by $\boldsymbol{\psi}_i^{(\text{pbest},t)} = (\psi_{i,1}^{(\text{pbest},t)}, \ldots, \psi_{i,L(k+1)}^{(\text{pbest},t)})^T$, and the best set of factor settings and proportion allocations found by the entire swarm by $\boldsymbol{\psi}^{(\text{gbest},t)} = (\psi_1^{(\text{gbest},t)}, \ldots, \psi_{L(k+1)}^{(\text{gbest},t)})^T$. At $t = 0$, the pbest positions are the initial positions of each particle and the gbest position is the best of all pbest positions, where "best" refers to the position corresponding to the design with the largest value of the objective function. Each iteration has two update steps, one at the swarm-level and the other at the habitat-level (Figure 1).

Below we enumerate the steps in the d-QPSO algorithm for updating each swarm and suppress the swarm indicator for notational simplicity. The updates must be performed differently for continuous factor settings, discrete factor settings, and the proportion allocated to each support point. We collect the components of each $\boldsymbol{\psi}_i$ into three sets, A_c, A_d, and A_p which correspond to continuous factors, discrete factors, and proportions, respectively. Clearly, $A_c \cup A_d \cup A_p = \{1, 2, \ldots, L(k+1)\}$, and updating each set updates the entire vector.

1. *Update the local attractors.* For each particle i, $i = 1, \ldots, w$, the local attractor $\mathbf{a}_i^{(t)} = (a_{i,1}^{(t)}, \ldots, a_{i,L(k+1)}^{(t)})^T$ is the central point around which the particle will appear at iteration t. For a given swarm, the local attractor for particle i at component m, $m = 1, \ldots, L(k+1)$ is calculated as

$$a_{i,m}^{(t)} = \begin{cases} \phi_{i,m}^{(t)} \times \psi_{i,m}^{(\text{pbest},t-1)} & \\ \quad +(1-\phi_{i,m}^{(t)}) \times \psi_m^{(\text{gbest},t-1)} & \text{if } m \in A_c \cup A_p, \\ \psi_m^{(\text{pbest},t-1)} & \text{if } m \in A_d \text{ and } r_{i,3} \le 0.5, \\ \psi_m^{(\text{gbest},t-1)} & \text{if } m \in A_d \text{ and } r_{i,3} > 0.5, \end{cases}$$

where $\phi_{i,m} = \frac{r_{i,1}}{r_{i,1}+r_{i,2}}$ and $r_{i,1}, r_{i,2}, r_{i,3}$ are independent draws from $U(0,1)$, the uniform distribution over the interval $(0,1)$.

2. *Update particle positions.* Each particle is drawn to both its local attractor, $\mathbf{a}_i^{(t)}$ and to $\boldsymbol{\psi}^{(\text{mbest},t)} = (\frac{1}{w}\sum_{i=1}^{w} \psi_{i,1}^{(\text{pbest},t-1)}, \ldots, \frac{1}{w}\sum_{i=1}^{w} \psi_{i,L(k+1)}^{(\text{pbest},t-1)})^T$, a position referred to as the "mbest." For discrete factors, that is, $m \in A_d$, $\psi_m^{(\text{mbest},t)}$ is rounded to 1 or -1. The position update step is

$$\psi_{i,m}^{(t)} = \begin{cases} a_{i,m}^{(t)} + \alpha \left| \psi_{i,m}^{(t-1)} \right. & \\ \quad \left. - \psi_m^{(\text{mbest},t)} \right| \log(r_{i,m,1}) & \text{if } m \in A_c \cup A_p \text{ and } r_{i,m,2} \ge 0.5\,, \\ a_{i,m}^{(t)} - \alpha \left| \psi_{i,m}^{(t-1)} \right. & \\ \quad \left. - \psi_m^{(\text{mbest},t)} \right| \log(r_{i,m,1}) & \text{if } m \in A_c \cup A_p \text{ and } r_{i,m,2} < 0.5, \\ -\psi_{i,m}^{(t-1)} & \text{if } m \in A_d \text{ with probability } \tau_l, \\ \psi_{i,m}^{(t-1)} & \text{if } m \in A_d \text{ with probability } 1-\tau_l, \end{cases}$$

where $r_{i,m,1}$, $r_{i,m,2}$ are independent $U(0,1)$ draws. Here α is known as the contraction-expansion coefficient and is decreased linearly from 1.4 to 0.4 as the algorithm runs (Sun et al. 2012). For the discrete factor update, τ_l is the probability of changing factor setting for factors belonging to $\boldsymbol{x}_l$ and is calculated using Hamming distance as described in Xi et al. (2016). Under this updating scheme, a vector of discrete factor settings is created by combining $\boldsymbol{\psi}^{(\text{mbest},t)}$ and $\boldsymbol{\psi}^{(\text{gbest},t-1)}$ using a crossover. In a crossover, two position vectors are split at a random location and the left partition of one is combined with the right partition of the other to form a new position vector. We use the vector with left elements taken from $\boldsymbol{\psi}^{(\text{mbest},t)}$ and right elements taken from $\boldsymbol{\psi}^{(\text{gbest},t-1)}$ and refer to this vector as $\boldsymbol{\psi}^{(\text{cbest},t)}$. The larger the Hamming distance between $\boldsymbol{\psi}^{(\text{cbest},t)}$ and the particle's current position, the more likely each element of $\boldsymbol{\psi}_i^{(t-1)}$ is to "flip" factor settings. For discrete factors, $m \in A_d$, τ_l is calculated based on the $\boldsymbol{x}_l$ to which m belongs. That is, for support point $\boldsymbol{x}_l$, the probability of flipping, τ_l, will be the same for all discrete factor settings of that support point. The value of τ_l is

$$\tau_l = \min\left\{ \frac{\alpha \times b_l(-\log(r))}{k_d}, 1 \right\},$$

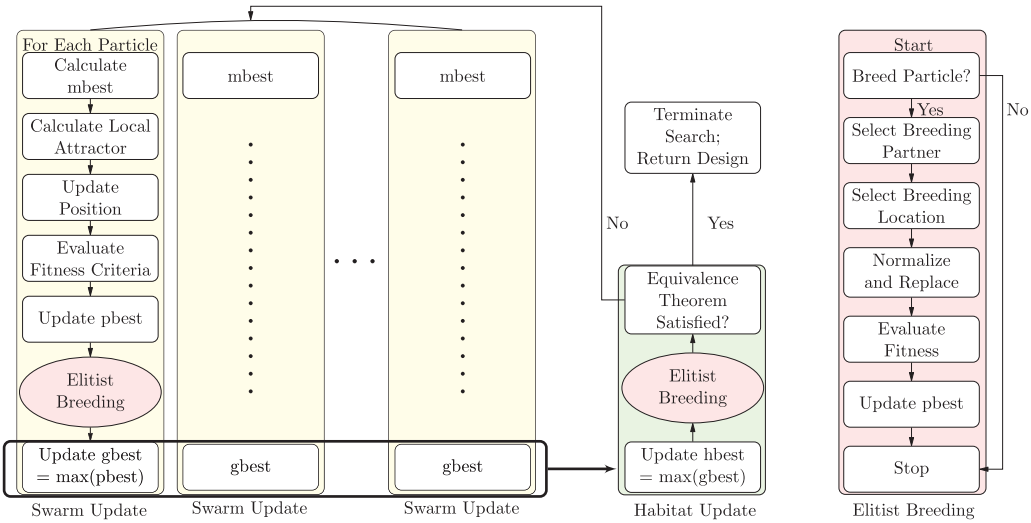

Figure 1. Steps in the QPSO update for generating locally D-optimal approximate designs. The swarm update is applied to each swarm individually, and the habitat update is performed on all swarms.

where k_d is the number of discrete factors in the model, r is a random draw from $U(0, 1)$, and b_l counts the number of elements in the current candidate $\boldsymbol{\psi}_i^{(t)}$ which are different from the corresponding elements of $\boldsymbol{\psi}^{(\text{cbest},t)}$, restricted to the discrete factors only, for $\boldsymbol{x}_l$. The probability of changing a factor setting, τ_l, increases as b_l increases. After the position update, any proportions that fall below 0 are set to 0. The proportions are then normalized to sum to one. If any value of a continuous factor falls outside its range, we assign this value to the nearest boundary on the design space.

3. *Update fitness.* For each particle i, $i = 1, \ldots, w$ in a given swarm, we calculate the value of the objective function. If a locally D-optimal design is sought, the fitness function is $\log\det(\boldsymbol{I}_{\psi_i})$, where the logarithm is used for numerical stability and to ensure the design criterion is concave for the application of the equivalence theorem. The fitness function used to find pseudo-Bayesian designs is the approximation to the expected log-determinant of the Fisher information matrix developed in Gotwalt, Jones, and Steinberg (2009).
4. *Update pbests and gbest.* For each particle i, $i = 1, \ldots, w$ in a given swarm, if the fitness of position $\boldsymbol{\psi}_i^{(t)}$ is greater than the fitness of $\boldsymbol{\psi}_i^{(\text{pbest},t-1)}$, we set $\boldsymbol{\psi}_i^{(\text{pbest},t)} = \boldsymbol{\psi}_i^{(t)}$, otherwise $\boldsymbol{\psi}_i^{(\text{pbest},t)} = \boldsymbol{\psi}_i^{(\text{pbest},t-1)}$. The $\boldsymbol{\psi}_i^{(\text{pbest},t)}$ with the largest fitness value is chosen as $\boldsymbol{\psi}^{(\text{gbest},t)}$ for the swarm.
5. *Elitist breeding.* To encourage exploration of the search space, we include an elitist breeding mutator similar to the one described in Yang, Wu, and Min (2015). At the end of each iteration, each particle undergoes elitist breeding with a small probability (we use probability 0.1). A new position vector is created $\boldsymbol{\psi}_i^{(\text{pbest}^*)} = \boldsymbol{\psi}_i^{(\text{pbest},t)}$ and updated by replacing a randomly selected element, m with a value taken from another randomly selected element, m^* of a random particle's pbest. Here both m and m^* belong to $\{1, 2, \ldots, L(k+1)\}$ and $\psi_{i,m}^{(\text{pbest}^*)}$ is updated as

$$\psi_{i,m}^{(\text{pbest}^*)} = \psi_{i^*,m^*}^{(\text{pbest},t)},$$

where $i^* \in \{1, \ldots, w\}$, and the position $\psi_{i^*,m^*}^{(\text{pbest},t)}$ is normalized to be in the range of component m. Figure 1 includes a schematic diagram of the elitist breeding mutator.

Following the swarm update, the habitat update keeps track of the best position from each swarm and the overall best position, which we denote by $\boldsymbol{\psi}^{(\text{hbest},t)}$, where hbest is short for "habitat best." We share information between swarms using another elitist breeding mutator. This acts to prevent the swarms from becoming stuck in local extrema. This elitist breeding update is similar to the swarm update, except that instead of selecting particles to breed from within the same swarm, particles are allowed to breed across swarms. To our knowledge, this is the first QPSO algorithm to share information across several swarms using such a technique. At each iteration, we determine whether the hbest position has improved over all previous iterations. If we are searching for a locally D-optimal approximate design, we check the local D-optimality of this generated design by examining if its sensitivity function satisfies the equivalence theorem. This can be done either by a grid search or another QPSO algorithm to maximize the sensitivity function and determine its minimum D-efficiency lower bound. The algorithm terminates if the desired D-efficiency lower bound is obtained or the maximum number of iterations has been reached. At termination, the design constructed using $\boldsymbol{\psi}^{(\text{hbest},t)}$ is returned as the d-QPSO algorithm-generated design.

Tuning the d-QPSO algorithm involves selecting the number of particles per swarm, the total number of swarms, the maximum number of iterations, and the maximum number of support points allowable in the design. For locally D-optimal approximate designs, the user must also supply a lower bound for the D-efficiency of the generated design; the algorithm terminates if the lower bound is met. Our general experience is that a relatively small number of particles (around $w = 30$) works well, allowing each particle to have a strong influence on the mean best position. As a rule of thumb, we suggest using k, swarms, where k is the number of factors in the model. We also suggest increasing the number of swarms when there are more factor levels in the study. The maximum number of iterations is decided by the user and this number should increase with the number of factors. In general, we find that a few thousand iterations work quite well. Finally, for approximate designs the maximum number of support points should be chosen based on the number of factors in the model. For small number of factors, it is appropriate to allow $L = 2^k$ support points; the d-QPSO algorithm will generally be able to find a design supported on fewer points. Our experience is that for many experiments with q parameters in the model, having particles each with $2q$ to $3q$ points in the algorithm seems to work well. Of course, if these numbers fail to provide an adequate design, the number of support points can always be increased.

We have run extensive simulations to assess the effectiveness of the d-QPSO algorithm. These results, along with several other sets of results obtained from the d-QPSO algorithm, are provided in the supplementary materials. In Section S1, we provide CPU time and accuracy simulations for designing mixed factor experiments using the d-QPSO algorithm. Section S2 uses the d-QPSO algorithm to find minimally supported designs. All computations in this article were carried out using a 2012 Macbook Pro 2.6GHz Intel Core i7 with 16G RAM on 64bit OSX El Capitan. Our code is written in C++ and called from R via the RCPP package (Eddelbuettel and François 2011).

3. Applications: Locally *D*-Optimal Designs

We now demonstrate that the d-QPSO algorithm can find locally D-optimal designs for GLMs with mixed factors and a binary response of increasing complexity. We start by revisiting the motivating odor removal example and find a more realistic locally D-optimal approximate design for the study. We then apply d-QPSO and obtain locally D-optimal approximate designs for an electrostatic discharge experiment and a 10-factor car refueling experiment. In all examples, we use the logit link function in the GLM, but other link functions can be used as well (see the supplementary materials Section S3.1).

Table 2. The d-QPSO algorithm-generated locally D-optimal approximate design for the odor removal experiment with nominal values $\boldsymbol{\beta} = (-1, 2, 0.5, -1, -0.25, 0.13)^T$.

Support point	Algae	Scav.	Resin	Comp.	Temp.	p_i(%)	Support point	Algae	Scav.	Resin	Comp.	Temp.	p_i(%)
1	−1	−1	−1	−1	9.040	3.70	8	−1	1	1	−1	16.894	2.20
2	−1	−1	−1	−1	25.788	4.30	9	−1	1	1	−1	33.366	8.80
3	−1	−1	−1	1	29.710	10.17	10	−1	1	1	1	35.000	6.10
4	−1	−1	1	−1	35.000	4.73	11	1	−1	−1	1	5.000	5.11
5	−1	−1	1	1	29.579	11.59	12	1	−1	1	−1	5.000	10.75
6	−1	1	−1	−1	5.000	9.80	13	1	−1	1	1	5.000	5.23
7	−1	1	−1	1	5.206	7.86	14	1	1	1	1	5.000	9.71

3.1 Odor Removal Experiment

In the motivating odor removal study conducted by Wang et al. (2016), the experimenters fixed the temperature variable and used a 2^{4-1} regular fractional factorial design with equal number of replicates. The d-QPSO algorithm allows us to construct a design that includes storage temperature as a continuous factor for the model $Y \sim \text{Bern}(\mu)$ with $\text{logit}(\mu) = \beta_0 + \beta_1\text{Algae} + \beta_2\text{Scavenger} + \beta_3\text{Resin} + \beta_4\text{Compatibilizer} + \beta_5\text{Temperature}$. Using information from their study, we take nominal values $\boldsymbol{\beta} = (-1, 2, 0.5, -1, -0.25, 0.13)^T$ and apply the d-QPSO algorithm to construct a locally D-optimal approximate design. The tuning parameters we used were 4 swarms, 30 particles in each swarm, and we initialized our search among designs with up to 20 support points. The termination rule was either a maximum of 6000 iterations or when the generated design attained a D-efficiency lower bound of 99%.

Table 2 displays the d-QPSO algorithm-generated locally D-optimal approximate design with 14 support points, where only some of the points have values at the extreme ends of the temperature range. The objective function value for this design is $0.0019^{1/6} = 0.3519$. The sensitivity plot of the design in Figure 2 confirms its D-optimality. It is instructive to compare this design to three locally D-optimal approximate designs obtained by breaking up the continuous temperature variable into 2, 3, and 4 uniformly spaced points across the design space. Designs using the discretized temperature variable were obtained using the lift-one algorithm (Yang, Mandal, and Majumdar 2016). The D-efficiencies of these three designs relative to the d-QPSO algorithm-generated design were 0.9737, 0.9907, and 0.9965, but the minimum allocations required at a support point were 0.46%, 1.54%, and 0.35% for the 2, 3, and 4 uniformly spaced point designs, respectively. This means that although theoretically these designs can be highly efficient, unless the total number of observations in the study is large, these lift-one algorithm-generated designs cannot be implemented in practice. In particular, the four-point lift-one algorithm-generated design requires $0.0035N$ observations at a support point versus $0.022N$ for the d-QPSO algorithm generated design.

3.2 Electrostatic Discharge Experiment

Whitman et al. (2006) described a similar experiment with continuous and discrete factors. The experimenters were interested in finding factors that influence the failure of semiconductors when exposed to electrostatic discharge (ESD). The response was whether or not a certain part of the semiconductor failed, and the model was logistic regression with five factors. The first two factors, Lot A and Lot B, describe the type of wafer used. The third factor was ESD handling: whether or not proper ESD precautions were taken. ESD testing requires a part to be "zapped" with a pulse at either a positive or a negative polarity and then to be zapped again by a pulse with the opposite polarity. A lack of standardization of which pulse order should be used resulted in the experimenters using pulse order as the fourth factor. The final factor, voltage applied to the chip, was continuous and ranged from 25 to 45. The experimental units were single chip TDMA power amplifiers, chosen for their lack of ESD protection circuitry. Without protection circuitry, the chips were expected to be sensitive to changes in experimental factors. Taking Y to

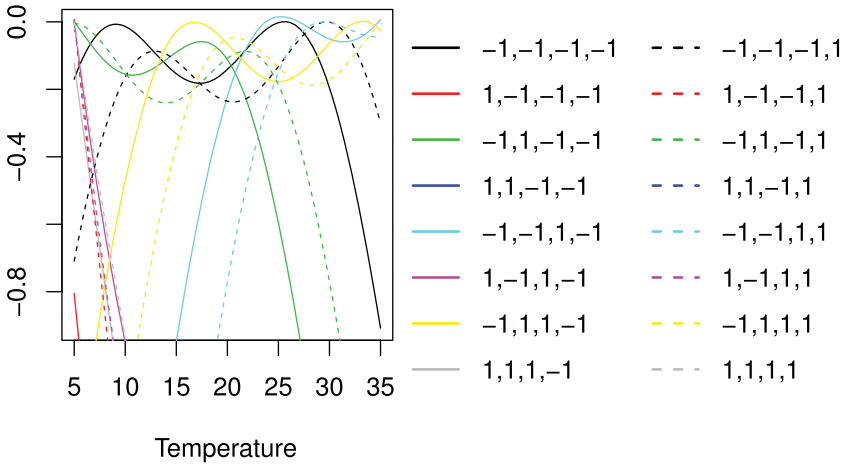

Figure 2. Sensitivity plot of the d-QPSO algorithm-generated locally D-optimal approximate design for each fixed combination of discrete factors in the odor removal experiment.

Table 3. The *d*-QPSO algorithm-generated locally *D*-optimal approximate design for the ESD experiment with nominal values $\boldsymbol{\beta} = (-7.5, 1.50, -0.2, -0.15, 0.25, 0.35, 0.4)^T$.

Support point	A	B	ESD	Pulse	Volt.	p_i (%)	Support point	A	B	ESD	Pulse	Volt.	p_i (%)
1	−1	−1	−1	−1	28.04	1.80	8	−1	1	−1	1	25.00	10.00
2	−1	−1	−1	−1	25.00	7.46	9	−1	1	1	−1	25.00	3.80
3	−1	−1	−1	1	25.00	2.49	10	−1	1	1	−1	32.93	13.43
4	−1	−1	−1	1	27.85	7.74	11	−1	1	1	1	25.00	9.20
5	−1	−1	1	−1	25.00	11.65	12	1	−1	1	−1	25.00	1.23
6	−1	−1	1	1	25.00	8.58	13	1	1	1	−1	25.00	13.40
7	−1	1	−1	−1	25.00	9.20							

be 1 if the chip fails and 0 otherwise, we have $Y \sim \text{Bern}(\mu)$, and the model of interest is $\text{logit}(\mu) = \beta_0 + \beta_1 \text{Lot A} + \beta_2 \text{Lot B} + \beta_3 \text{ESD} + \beta_4 \text{Pulse} + \beta_5 \text{Voltage} + \beta_{34}(\text{ESD} \times \text{Pulse})$.

The experimenters discretized the voltage variable into five levels: 25, 30, 35, 40, and 45 V. The full factorial design with 80 separate settings was implemented to test all possible combinations of factor settings. They did not provide a rationale for treating voltage as a discrete variable, but we note that such techniques are common when constructing exact response surface designs with continuous factors.

We redesign this experiment by treating voltage as a continuous variable that ranges from 25 to 45 instead of fixing it to 5 levels. We use the set of nominal values $\boldsymbol{\beta} = (-7.5, 1.50, -0.2, -0.15, 0.25, 0.35, 0.4)^T$, motivated by the parameter estimates reported in the article. The tuning parameters we used were 5 swarms, 30 particles per swarm, and we initialized our search among designs with up to 18 support points. The termination rule was either a maximum of 4000 iterations or a generated design that attains a *D*-efficiency lower bound of 99%. Table 3 lists the *d*-QPSO algorithm-generated design with 13 points and its sensitivity function in Figure 3 confirms its local *D*-optimality.

The *d*-QPSO algorithm-generated locally *D*-optimal approximate design has four unique voltage settings. The original experimental design has five unique voltage settings, and these were very different than the ones identified by the *d*-QPSO algorithm, which has an objective function value of $(1.2639 \times 10^{-5})^{1/7} = 0.1997$. The *D*-efficiency of the original 80 point design relative to the design identified by the *d*-QPSO algorithm is 32.85%, indicating that the latter design is approximately three times as *D*-efficient as the implemented design.

3.3 Car Refueling Experiment

We now apply the *d*-QPSO algorithm to solve a high-dimensional design problem. Grimshaw et al. (2001) described an experiment to test a vision-based car refueling system. Here, the investigators were interested in finding whether a computer-controlled nozzle was able to insert itself into the gas pipe correctly or not. Table 4 lists the four discrete factors and the six factors that are naturally continuous: reflective ring thickness, lighting angle, gas-cap angle (*Z*-axis), gas-cap angle (*Y*-axis skew), car distance, and threshold step value. For illustrative purposes, we take the set of nominal values for the model parameters to be: $\boldsymbol{\beta} = (3, 0.5, 0.75, 1.25, 0.8, 0.5, 0.8, -0.4, -1.00, 2.65, 0.65)^T$, where the order of the factors is the order given in Table 4. Other sets of nominal values could also be used.

We use the *d*-QPSO algorithm to search for a locally *D*-optimal approximate design for the main effects logistic model. The tuning parameters we used were 10 swarms, 30 particles per swarm, and we initialized our search among designs with up to 12 support points. The termination rule was a maximum of 7000 iterations or a generated design attaining a *D*-efficiency lower bound of 99%, which was determined using a second QPSO algorithm to find the maximum of the sensitivity function. This second search was used because the design has six continuous factors, and thus performing a grid search to find the maximum of the sensitivity function is difficult. While this search did not find any values to indicate that our design was not locally *D*-optimal via the equivalence theorem, we note that there is a possibility that we did not find the maximum of the sensitivity function that determines the minimum *D*-efficiency

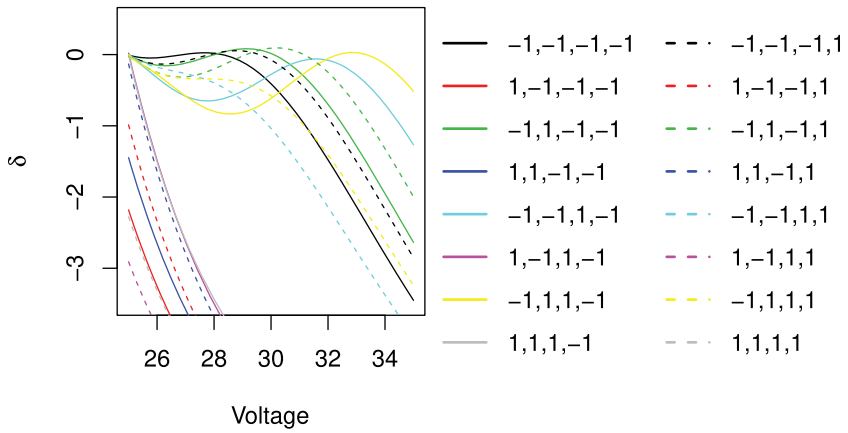

Figure 3. Sensitivity plot of the *d*-QPSO algorithm-generated locally *D*-optimal approximate design for each fixed combination of discrete factors in the ESD experiment.

Table 4. Factor types and levels for the car refueling experiment.

Type	Factor	Levels	
		Low	High
Discrete	Ring type	White paper	Reflective
	Lighting	Room lighting	2 flood lights and room lights
	Sharpen	No	Yes
	Smooth	No	Yes
Continuous	Lighting angle	from 50 degrees to 90 degrees	
	Gas-cap angle (Z-axis)	from 30 degrees to 55 degrees	
	Gas-cap angle (Y-axis skew)	from 0 degrees to 10 degrees	
	Car distance	from 18 in. to 48 in.	
	Reflective ring thickness	from 0.125 in. to 0.425 in.	
	Threshold step value	from 5 to 15	

Table 5. The d-QPSO algorithm-generated approximate design for the car refueling experiment with nominal values $\boldsymbol{\beta} = (3, 0.5, 0.75, 1.25, 0.8, 0.5, 0.8, -0.4, -1.00, 2.65, 0.65)^T$.

Ring type	Lighting	Sharpen	Smooth	Lighting angle	Z-axis	Y-axis skew	Car dist.	Ring thick.	Threshold stepsize	p_i (%)
−1	−1	−1	−1	50.00	30.00	4.20	48.00	0.125	5.00	9.1
−1	−1	−1	−1	50.00	30.00	10.00	48.00	0.125	8.57	9.1
−1	−1	−1	−1	50.00	30.00	10.00	45.68	0.125	5.00	9.1
−1	−1	−1	−1	54.64	30.00	10.00	48.00	0.125	5.00	9.1
−1	−1	−1	−1	50.00	32.90	10.00	48.00	0.125	5.00	9.1
−1	−1	−1	−1	50.00	30.00	10.00	48.00	0.125	5.00	8.1
−1	−1	−1	−1	50.00	30.00	10.00	48.00	0.425	5.00	7.7
−1	−1	−1	1	50.00	30.00	10.00	48.00	12.5	5.00	9.1
−1	−1	1	−1	50.00	30.00	10.00	48.00	12.5	5.00	9.1
−1	1	−1	−1	50.00	30.00	10.00	48.00	12.5	5.00	9.1
1	−1	−1	−1	50.00	30.00	10.00	48.00	12.5	5.00	7.5
1	−1	−1	−1	50.00	30.00	10.00	48.00	0.425	5.00	4.0

of the design generated. Table 5 displays the d-QPSO algorithm-generated design with an objective function value of $(2.5181 \times 10^{-16})^{1/11} = 0.0382$.

To further support our claim that this is design highly D-efficient, and is indeed locally D-optimal, we run the d-QPSO algorithm several more times, holding the number of swarms fixed at 10 and changing the number of particles, the limit on the number of possible support points, and the maximum number of iterations. Table 6 provides a summary of the tuning parameters used and the results obtained. In all cases, we obtain a design similar to the one presented in Table 5. Even the worst design, obtained using only 10 particles per swarm, still had a D-efficiency of 98% relative to the d-QPSO algorithm-generated locally D-optimal approximate design in Table 5.

We also attempted to solve this 10-factor problem by discretizing the design space and using current algorithms, such as the lift-one and Fedorov–Wynn type of algorithms; however, the number of candidate points became too large due to the number of factors, and the algorithms were unable to run successfully. We also applied quasi-Newton BFGS to tackle this problem via the optim package in R. This algorithm cannot search over the discrete factor settings, so we resorted to fixing the discrete factor combinations and applying quasi-Newton BFGS to find the continuous factor settings along with the proportions allocated to each support point. The algorithm took several hours to run, significantly longer than the d-QPSO algorithm. The resulting design had 16 support points and its D-efficiency was only 70% relative to the d-QPSO algorithm-generated locally D-optimal approximate design in Table 5.

Table 6. The d-QPSO algorithm tuning parameters and results for constructing locally D-optimal approximate designs for the car refueling experiment.

Max support points	Particles per swarm	Max iterations	log det	Support points	CPU time (Seconds)
12	35	5000	−35.91	12	87.58
12	35	7000	−36.08	11	49.38
14	35	5000	−36.04	11	47.51
14	35	7000	−36.02	11	244.70
14	20	7000	−36.01	12	140.80
16	35	5000	−35.93	12	70.44
16	10	7000	−36.15	12	67.12
16	20	7000	−35.98	12	85.60
16	35	7000	−35.92	12	211.34
16	75	7000	−35.93	12	424.52

4. Applications: Pseudo-Bayesian Designs

The previous section assumes that we have a given set of nominal values for the model parameters to find an optimal design. When no such reliable nominal values are available, Bayesian designs provide an attractive, robust solution to the design problem. These designs require priors for the parameter values and proceed by maximizing the expectation of the log-determinant of the Fisher information matrix by choice of a design. In general, Bayesian designs come at a high computational cost, as numerical integration must be performed every time the design changes. To circumvent these numerical difficulties, several pseudo-Bayesian methods have been proposed. Woods et al. (2006) identified exact compromise designs that are robust to misspecification of both the parameter values and link function.

Table 7. The *d*-QPSO algorithm-generated pseudo-Bayesian approximate design for the odor removal experiment using the independent uniform priors for all parameters specified in Section 4.1.

Support point	Algae	Scav.	Resin	Comp.	Temp.	p_i(%)	Support point	Algae	Scav.	Resin	Comp.	Temp.	p_i(%)
1	−1	−1	−1	−1	5.000	8.50	8	−1	1	1	1	14.674	5.00
2	−1	−1	−1	−1	34.911	3.87	9	−1	1	1	1	34.980	9.49
3	−1	−1	−1	1	28.818	6.60	10	1	−1	−1	1	5.000	9.80
4	−1	−1	1	−1	31.261	10.10	11	1	−1	1	−1	5.000	4.70
5	−1	1	−1	−1	5.000	7.60	12	1	−1	1	1	5.000	9.92
6	−1	1	−1	−1	21.550	3.15	13	1	1	1	−1	5.000	10.79
7	−1	1	−1	1	5.000	8.18	14	1	1	1	1	5.000	2.30

Table 8. The *d*-QPSO algorithm-generated pseudo-Bayesian exact design with 16 support points for the crystallography experiment with independent prior specification $\beta_0 \sim U(-3, 3)$, $\beta_1 \sim U(4, 10)$, $\beta_2 \sim U(5, 11)$, $\beta_3 \sim U(-6, 0)$, and $\beta_4 \sim U(-2.5, 3.5)$.

Support point	Agitation rate	Composition volume	Temperature	Evaporation rate	Support point	Agitation rate	Composition volume	Temperature	Evaporation rate
1	−1.000	0.378	−1.000	−1.000	9	0.594	−1.000	−1.000	−1.000
2	−1.000	0.791	−1.000	−1.000	10	1.000	−0.990	−1.000	−1.000
3	−1.000	0.212	−1.000	1.000	11	0.446	−1.000	−1.000	1.000
4	−1.000	0.708	−1.000	1.000	12	0.940	−1.000	−1.000	1.000
5	−0.969	1.000	1.000	−1.000	13	1.000	−0.652	1.000	−1.000
6	−0.487	1.000	1.000	−1.000	14	1.000	−0.223	1.000	−1.000
7	−1.000	1.000	1.000	1.000	15	1.000	−0.817	1.000	1.000
8	−0.522	1.000	1.000	1.000	16	1.000	−0.397	1.000	1.000

Gotwalt, Jones, and Steinberg (2009) used a quadrature scheme to approximate the value of the expected Fisher information matrix. Overstall and Woods (2017) developed an approximate coordinate exchange algorithm for identifying Bayesian designs through the use of a Gaussian process emulator. We next demonstrate the *d*-QPSO algorithm's ability to find pseudo-Bayesian designs using the odor removal experiment and the 16 run crystallography experiment described by Woods et al. (2006).

4.1 Odor Removal Experiment

We first return to the odor removal experiment. We use the *d*-QPSO algorithm to obtain a robust pseudo-Bayesian design by assuming independent uniform priors for each parameter with each prior centered at the nominal value and having a width twice the magnitude of the supposed nominal value used in Section 3.1. For example, if the nominal value of a parameter was 1, we use the uniform prior over $(0, 2)$. These independent priors for $\beta_0, \ldots, \beta_5$ are: $U(-2, 0)$ for β_0 and β_3, $U(0, 4)$ for β_1, $U(0, 1)$ for β_2, $U(-\frac{1}{2}, 0)$ for β_4, and $U(0, 0.26)$ for β_5. The tuning parameters we used were 2 swarms, 30 particles per swarm, and we initialized our search among designs with up to 16 support points. The termination rule was 10,000 iterations. Table 7 displays the *d*-QPSO algorithm-generated design. We compare robustness properties of this design with the *d*-QPSO algorithm-generated locally *D*-optimal approximate design in Section S3.2 of the supplementary materials.

4.2 Crystallography Experiment

Next we consider obtaining a pseudo-Bayesian design for the crystallography experiment described in Woods et al. (2006). Following Gotwalt, Jones, and Steinberg (2009), we use the prior labeled β_3 in Table 1 of Woods et al. (2006). This prior specification is $\beta_0 \sim U(-3, 3)$, $\beta_1 \sim U(4, 10)$, $\beta_2 \sim U(5, 11)$, $\beta_3 \sim U(-6, 0)$, and $\beta_4 \sim U(-2.5, 3.5)$. The experiment has four continuous factors corresponding to the agitation rate, composition volume, temperature, and evaporation rate. The *d*-QPSO algorithm tuning parameters we used were 5 swarms, 30 particles per swarm, and we initialized our search for the optimal exact design among all optimal exact designs with up to 16 support points. The termination rule was 7000 iterations. Table 8 displays the *d*-QPSO algorithm-generated design for this optimal exact design problem. For comparison, we generate one million randomly sampled parameter vectors from the prior and calculate the objective function value of the *d*-QPSO algorithm-generated design, the design in Woods et al. (2006), the design presented by Gotwalt, Jones, and Steinberg (2009), the design obtained using the method of Gotwalt, Jones, and Steinberg (2009) reported in Overstall and Woods (2017) (referred to as OW Gotwalt), and the design obtained using the approximate coordinate exchange (ACE) algorithm in the acebayes R package from Overstall and Woods (2017). Table 9 compares these methods using the average objective function values and the median *D*-efficiencies of the designs relative to the *d*-QPSO algorithm-generated design. The table shows that the *d*-QPSO algorithm is competitive with the other methods.

Table 9. Mean objective function value and median *D*-efficiency relative to the *d*-QPSO algorithm-generated pseudo-Bayesian exact design (RE) for each of the robust designs considered for the crystallography experiment over one million randomly sampled parameter vectors.

Design	Mean $\lvert X^T W X\rvert^{1/5}$	Median RE
d-QPSO	0.5734	1.0000
Gotwalt, Jones, and Steinberg (2009)	0.4643	0.8194
OW Gotwalt	0.5687	0.9896
Woods et al. (2006)	0.5394	0.9373
ACE	0.5685	0.9933

5. Summary and Conclusions

In this article, we proposed a novel and flexible *d*-QPSO algorithm to find several types of *D*-optimal designs for GLMs with mixed factors and a binary response. We demonstrated that the *d*-QPSO algorithm could find more *D*-efficient designs than those obtained by treating continuous factors as discrete. We applied the *d*-QPSO algorithm to find a locally *D*-optimal design for an experiment with 10 factors, and we also showed it can be used to find robust pseudo-Bayesian designs when there is uncertainty in the parameter values.

In conclusion, we view the use of metaheuristic optimization algorithms as an effective option for finding solutions to complicated design problems. We believe that there is definite potential for further use of nature-inspired metaheuristic algorithms to find different kinds of optimal designs and help us better understand properties of optimal designs. For example, Section S3.1 of the supplementary materials provides a brief study on sensitivity of the locally *D*-optimal designs to misspecification of the link function. Applying the *d*-QPSO algorithm to find optimal exact designs for correlated responses or minimum bias designs is potentially interesting. For such problems, there are no equivalence theorems to resort to for confirming optimality of the generated design because the design criteria are no longer concave. The only way to assess the optimality of the generated design by the *d*-QPSO algorithm or other algorithms is by developing theoretical results, which are usually only feasible for relatively simple models. A plausible strategy in this situation is to show the algorithm generates the same optimal designs that are already worked out analytically for simple cases, and then use the algorithm to find optimal designs for more complicated cases where theoretical designs are no longer available. For example, Chen et al. (2015b) used PSO to generate locally *D*-optimal exact designs for the Michaelis–Menten model with correlated errors and confirm their numerical results with the theoretical optimal designs available from Dette and Kunert (2014) when only a couple of time points are allowed for taking measurements. Chen et al. (2015b) then used PSO to generate optimal designs for a longitudinal study with more time points than those considered in Dette and Kunert (2014), where theoretical results are not available.

Supplementary Materials

- **S1. *d*-QPSO Computational Timing and Accuracy:** Simulations tracking the speed and accuracy of the *d*-QPSO algorithm for finding locally *D*-optimal approximate designs for models with up to six mixed factors and a binary response.
- **S2. Minimally Supported Designs:** Simulations using the *d*-QPSO algorithm to construct locally *D*-optimal approximate designs for models with one discrete and one continuous factor and a binary response. We investigate conditions under which the *d*-QPSO algorithm can find minimally supported designs.
- **S3. Sensitivity Study:** (i) An illustrative study of the robustness of two factor experiments with binary response to mis-specification of the link function and (ii) a study of the robustness of the optimal design for the odor removal experiment to parameter mis-specification.
- **S4. Locally *D*-optimal Exact Designs:** Selected *D*-optimal exact designs for the odor removal experiment for various total sample sizes *N*.
- **S5. Optimal Designs on an Irregular Design Space:** An illustrative example using *d*-QPSO to generate a locally *D*-optimal approximate design on an irregular design space.
- **S6. The *d*-QPSO Algorithm for Finding the Optimal Designs for the Odor Removal Experiment:** Provides the C++ code allowing the reader to reproduce the results for the odor removal study.

Acknowledgments

The authors would like to thank the editor, an associate editor, and the reviewers for comments and suggestions that substantially improved the quality of the article. The second author would like to thank Dr. Snehanshu Saha for his suggestions on the QPSO algorithm.

Funding

The research of Dr. Mandal was in part supported by NSA Grant H98230-13-1-025. The research of Dr. Wong reported in this article was partially supported by the National Institute of General Medical Sciences of the National Institutes of Health under Award Number R01GM107639. The contents in this article are solely the responsibility of the authors and do not necessarily represent the official views of the National Institutes of Health.

References

Atkinson, A. C., and Woods, D. C. (2015), "Designs for Generalized Linear Models," in *Handbook of Design and Analysis of Experiments*, eds. A. Dean, M. Morris, J. Stufken, and D. Bingham, Boca Raton, FL: Chapman & Hall/CRC, pp. 471–514. [78]

Chen, R.-B., Chang, S.-P., Wang, W., Tung, H.-C., and Wong, W. K. (2015a), "Minimax Optimal Designs via Particle Swarm Optimization Methods," *Statistics and Computing*, 25, 975–988. [79]

Chen, R.-B., Chen, P.-Y., Tung, H.-C., and Wong, W. K. (2015b), "Exact *D*-Optimal Designs for Michaelis-Menton Model with Correlated Observations by Particle Swarm Optimization," in *Festschrift in Honor of Hans Nyquist on the Occasion of His 65th Birthday*, ed. E. Fackle-Fornius, Stockholm, Sweden: Department of Statistics, Stockholm University, pp. 60–73. [86]

Chen, R.-B., Hsieh, D.-N., Hung, Y., and Wang, W. (2013), "Optimizing Latin Hypercube Designs by Particle Swarm," *Statistics and Computing*, 23, 663–676. [79]

Clerc, M., and Kennedy, J. (2002), "The Particle Swarm-Explosion, Stability, and Convergence in a Multidimensional Complex Space," *IEEE Transactions on Evolutionary Computation*, 6, 58–73. [79]

Dette, H., and Kunert, J. (2014), "Optimal Designs for the Michaelis-Menten Model With Correlated Observations," *Statistics*, 48, 1254–1267. [86]

Dror, H. A., and Steinberg, D. M. (2006), "Robust Experimental Design for Multivariate Generalized Linear Models," *Technometrics*, 48, 520–529. [78]

——— (2008), "Sequential Experimental Designs for Generalized Linear Models," *Journal of the American Statistical Association*, 103, 288–298. [78]

Eddelbuettel, D., and François, R. (2011), "Rcpp: Seamless R and C++ Integration," *Journal of Statistical Software*, 40, 1–18. [81]

Fedorov, V. V. (1972), *Theory of Optimal Experiments*, Moscow, Russia: Elsevier. [78]

Firth, D. (1993), "Bias Reduction of Maximum Likelihood Estimates," *Biometrika*, 80, 27–38. [78]

Ford, I., Torsney, B., and Wu, J. (1992), "The Use of a Canonical Form in the Construction of Locally Optimal Designs for Non-Linear Problems," *Journal of the Royal Statistical Society*, Series B, 54, 569–583. [78]

Fu, X., Lim, S., Wang, L., Lee, G., Ma, S., Wong, L., and Xiao, G. (2009), "Key Node Selection for Containing Infectious Disease Spread Using Particle Swarm Optimization," in *Swarm Intelligence Symposium, 2009. SIS'09*, Nashville, TN: IEEE, pp. 109–113. [79]

Gotwalt, C. M., Jones, B. A., and Steinberg, D. M. (2009), "Fast Computation of Designs Robust to Parameter Uncertainty for Nonlinear Settings," *Technometrics*, 51, 88–95. [85]

Grimshaw, S. D., Collings, B. J., Larsen, W. A., and Hurt, C. R. (2001), "Eliciting Factor Importance in a Designed Experiment," *Technometrics*, 43, 133–146. [83]

Heinze, G., and Schemper, M. (2002), "A Solution to the Problem of Separation in Logistic Regression," *Statistics in Medicine*, 21, 2409–2419. [78]

Kennedy, J., and Eberhart, R. (1995), "Particle Swarm Optimization," in *Proceedings of IEEE International Conference on Neural Networks*, Perth, Australia: IEEE, pp. 1942–1948. [79]

Kiefer, J. (1959), "Optimum Experimental Designs," *Journal of the Royal Statistical Society*, Series B, 21, 272–319. [77]

Kiefer, J., and Wolfowitz, J. (1959), "Optimum Designs in Regression Problems," *The Annals of Mathematical Statistics*, 30, 271–294. [78]

Mak, S., and Joseph, V. R. (2017), "Minimax and Minimax Projection Designs Using Clustering," *Journal of Computational and Graphical Statistics*, 28, 166–178. [79]

Mandal, A., Wong, W. K., and Yu, Y. (2015), "Algorithmic Searches for Optimal Designs," in *Handbook of Design and Analysis of Experiments*, eds. A. Dean, M. Morris, J. Stufken, and D. Bingham, Boca Raton, FL: Chapman & Hall/CRC, pp. 755–783. [78]

Nocedal, J., and Wright, S. J. (1999), *Numerical Optimization*, New York, NY: Springer Science. [78]

Overstall, A. M., and Woods, D. C. (2017), "Bayesian Design of Experiments Using Approximate Coordinate Exchange," *Technometrics*, 59, 458–470. [85]

Pazman, A. (1986), *Foundations of Optimum Experimental Design*, Reidel (Kluwer group), Dordrecht (co-pub. VEDA, Bratislava): Springer. [78]

Pukelsheim, F. (1993), *Optimal Design of Experiments*, Philadelphia, PA: SIAM. [78]

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. [79]

Rousseeuw, P. J., and Christmann, A. (2003), "Robustness Against Separation and Outliers in Logistic Regression," *Computational Statistics & Data Analysis*, 43, 315–332. [78]

Stufken, J., and Yang, M. (2012), "Optimal Designs for Generalized Linear Models," in *Design and Analysis of Experiments, Special Designs and Applications* (Vol. 3), ed. K. Hinkelmann, Hoboken, New Jersey: Wiley, pp. 137–164 [78]

Sun, J., Fang, W., Wu, X., Palade, V., and Xu, W. (2012), "Quantum-Behaved Particle Swarm Optimization: Analysis of Individual Particle Behavior and Parameter Selection," *Evolutionary Computation*, 20, 349–393. [80]

Sun, J., Feng, B., and Xu, W. (2004a), "Particle Swarm Optimization With Particles Having Quantum Behavior," in *Congress on Evolutionary Computation, 2004. CEC2004*, Portland, OR: IEEE, pp. 325–331. [79]

Sun, J., Xu, W., and Feng, B. (2004b), "A Global Search Strategy of Quantum-Behaved Particle Swarm Optimization," in *2004 IEEE Conference on Cybernetics and Intelligent Systems*, Singapore, Singapore: IEEE, pp. 111–116. [79]

Titterington, D. (1976), "Algorithms for Computing *D*-Optimal Designs on a Finite Design Space," in *Proceedings of the 1976 Conference on Information Science and Systems* (vol. 3), Baltimore, MD: John Hopkins University, pp. 213–216. [78]

Wang, K., Mandal, A., Ayton, E., Hunt, R., Zeller, A., and Sharma, S. (2016), "Modification of Protein Rich Algal-Biomass to Form Bio-Plastics and Odor Removal," in *Protein Byproducts: Transformation From Environmental Burden Into Value-Added Products*, ed. G. S. Dhillon, London, United Kingdom: Elsevier, pp. 107–117. [77,82]

Waterhouse, T. H., Woods, D. C., Eccleston, J. A., and Lewis, S. M. (2008), "Design Selection Criteria for Discrimination/Estimation for Nested Models and a Binomial Response," *Journal of Statistical Planning and Inference*, 138, 132–144. [78]

Whitman, C., Gilbert, T. M., Rahn, A. M., and Antonell, J. A. (2006), "Determining Factors Affecting ESD Failure Voltage Using DOE," *Microelectronics Reliability*, 46, 1228–1237. [82]

Wong, W. K., Chen, R.-B., Huang, C.-C., and Wang, W. (2015), "A Modified Particle Swarm Optimization Technique for Finding Optimal Designs for Mixture Models," *PLoS One*, 10, e0124720. [79]

Woods, D. C., Lewis, S. M., Eccleston, J. A., and Russell, K. G. (2006), "Designs for Generalized Linear Models With Several Variables and Model Uncertainty," *Technometrics*, 48, 284–292. [78,84,85]

Woods, D. C., McGree, J. M., and Lewis, S. M. (2017), "Model Selection via Bayesian Information Capacity Designs for Generalised Linear Models," *Computational Statistics & Data Analysis*, 113, 226–238. [78]

Woods, D. C., and van de Ven, P. (2011), "Blocked Designs for Experiments With Correlated Non-Normal Response," *Technometrics*, 53, 173–182. [78]

Xi, M., Sun, J., Liu, L., Fan, F., and Wu, X. (2016), "Cancer Feature Selection and Classification Using a Binary Quantum-Behaved Particle Swarm Optimization and Support Vector Machine," *Computational and Mathematical Methods in Medicine*, 2016, 1–9. [80]

Yang, J., Mandal, A., and Majumdar, D. (2016), "Optimal Designs For 2^k Factorial Experiments With Binary Response," *Statistica Sinica*, 26, 385–411. [78,82]

Yang, M., Zhang, B., and Huang, S. (2011), "Optimal Designs For Generalized Linear Models With Multiple Design Variables," *Statistica Sinica*, 21, 1415–1430. [78]

Yang, Z.-L., Wu, A., and Min, H.-Q. (2015), "An Improved Quantum-Behaved Particle Swarm Optimization Algorithm With Elitist Breeding for Unconstrained Optimization," *Computational Intelligence and Neuroscience*, 2015, 41–53. [81]

Yu, Y. (2011), "D-Optimal Designs via a Cocktail Algorithm," *Statistics and Computing*, 21, 475–481. [78]

d-QPSO: A Quantum-Behaved Particle Swarm Technique for Finding D-Optimal Designs for Models with Discrete and Continuous Factors and a Binary Response

Supplementary Materials

In the following sections, we further demonstrate the flexibility and utility of the d-QPSO algorithm for finding different types of optimal exact and approximate designs for GLMs with mixed factors and a binary response. We also show how it can be used to address uncertainty in the model assumptions and discuss the general performance of the algorithm.

S1 d-QPSO Computational Timing and Accuracy

We first report the average CPU run time and the average D-efficiency lower bound attained by the d-QPSO algorithm-generated design for some simple models. The model of interest is $logit(\mu) = \beta_0 + \sum_{i=1}^{k} \beta_i x_i$ and the number of factors, k, ranges from two to six, for all combinations of up to three discrete and three continuous factors. For each combination of factor types we apply the d-QPSO algorithm to construct 500 locally D-optimal approximate designs with $\beta_0, \beta_i \sim U(-3, 3)$. The design space is such that the discrete x_i's $\in \{-1, 1\}$ and the continuous x_i's $\in [-1, 1]$. The tuning parameters we used were 30 particles in each swarm, and the number of swarms was chosen to be equal to the number of continuous factors in the model. We initialized our search among designs with up to $\min\{16, 2^k\}$ support points, or in other words, each candidate design has at most 16 support points. The termination rule was either a maximum of $200 \times k$ iterations or when the generated design attained a D-efficiency lower bound of 98%. Grid searches were used to evaluate

Table S1: Average CPU times (seconds) and D-efficiency lower bounds (elb) for 500 simulated experimental designs generated with the d-QPSO algorithm.

Discrete Factors \ Continuous Factors	1		2		3	
	CPU time	elb	CPU time	elb	CPU time	elb
1	0.01	1.00	0.729	0.99	38.07	0.98
2	2.15	1.00	1.427	0.99	64.44	0.96
3	16.36	0.99	65.81	0.96	127.35	0.84

the sensitivity function of each generated design. These searches are included in the CPU time calculation, which is measured using the "user time" reported by R.

Table S1 displays the average CPU time required by the d-QPSO algorithm to obtain the locally D-optimal approximate design and the average D-efficiency lower bounds (elb) when there are different numbers of discrete and continuous factors in the experiment. Our results show that the d-QPSO algorithm is able to quickly identify a very highly D-efficient design or locally D-optimal approximate design.

S2 Minimally Supported Designs

In our second simulation, we delineate cases when and if a minimally supported locally D-optimal approximate design can be found by the d-QPSO algorithm. This is an interesting issue because some methods can only produce optimal designs with a fixed number of points (see for example, Yang et al. (2011)). Minimally supported optimal designs can be desirable because taking observations at a new point can be expensive.

Consider the model $Y \sim Bern(\mu)$, with $logit(\mu) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$ where $x_1 \in \{-1, 1\}$ and $x_2 \in [-1, 1]$, and the ranges for the nominal values are $\beta_0 \in \{1, 1.5, 2\}$, $\beta_1 \in [-1.5, 1.5]$ and $\beta_2 \in [-3, 3]$. We employ the d-QPSO algorithm to find locally D-optimal approximate designs. The tuning parameters we used were 2 swarms, 25 particles in each swarm, and we initialized our search among designs with up to 4 support points. The termination rule was either a maximum of 1000 iterations or when the generated design attained a D-efficiency lower bound of 99%. For the simulation, we discretize the parameter space for β_1 and β_2 using a grid with resolution 0.01, meaning that each parameter space is divided into a grid with points uniformly spaced 0.01 apart.

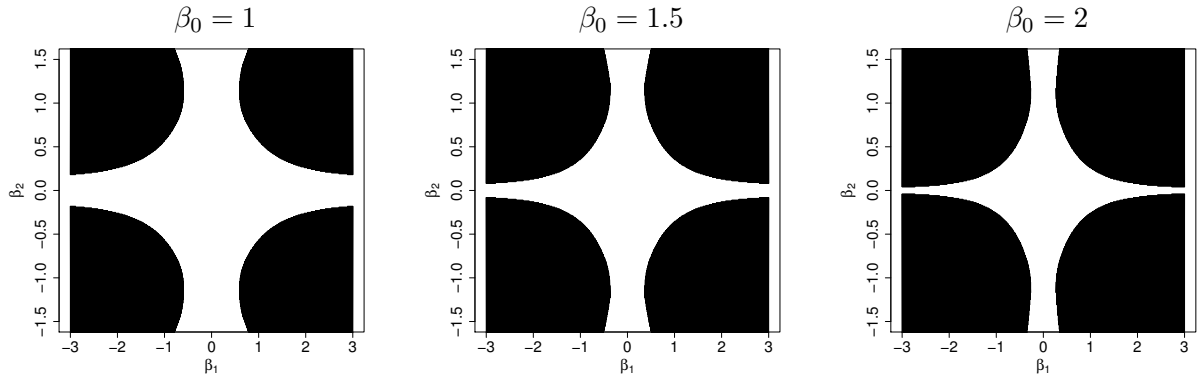

Figure S1: The black areas show the ranges of values for β_1 and β_2 for which a minimally supported locally D-optimal approximate design was found by the d-QPSO algorithm for the two-factor additive model when $\beta_0 = 1$, 1.5, and 2, respectively.

We use d-QPSO and generate designs for all combinations of β_1, β_2, with the intercept β_0 fixed. This results in a total of 180,901 (301 β_1 values $\times$ 601 β_2 values) d-QPSO algorithm-generated locally D-optimal approximate designs for each fixed β_0 setting.

The black curvilinear areas in Figure S1 show parameter values β_1 and β_2 for which the d-QPSO algorithm was able to construct minimally supported designs when $\beta_0 = \{1, 1.5, 2\}$. We observe that as the magnitude of β_0 increases, the region in which a minimally supported design can be constructed also increases. These pictures are similar to the ones obtained theoretically in Figure 2 on page 399 of Yang et al. (2016) and in Figures 1 and 3 of pages 11 and 19 of Yang et al. (2017).

S3 Sensitivity Study

S3.1 Robustness Under Mis-specification of the Link Function

Before a design is implemented, it is important to investigate its robustness properties to model mis-specification. For example, in GLMs with a binary response it is common to choose the logit link, but a prudent researcher should choose a design that reflects the actual goals of the study and has acceptable efficiency if there are violations in the model assumptions. There are several types of such violations. To fix ideas, suppose there is concern whether the link function is correctly specified and we want to know whether the locally D-optimal design found under the assumed link function remains efficient when the true link is another link function. In what follows, we use the

Table S2: Percentiles of the D-efficiencies of the logit link based d-QPSO algorithm-generated locally D-optimal designs relative to the d-QPSO algorithm-generated locally D-optimal approximate designs constructed under the probit, log-log, and complementary log-log link functions.

Percentile \ True Link	Probit	Log-log	C-log-log
0.99	1.0000	1.0000	1.0000
0.95	1.0000	1.0000	0.9900
0.90	1.0000	1.0000	0.9488
0.80	0.9900	0.9737	0.8692
0.70	0.9670	0.9106	0.7925

d-QPSO algorithm to investigate the robustness of locally D-optimal approximate designs found under the logit link when the true link function is probit, log-log, or complementary log-log.

We ran the d-QPSO algorithm using tuning parameters of 2 swarms, 25 particles in each swarm, and we initialized our search among designs with up to 4 support points. The termination rule was either a maximum of 1000 iterations or when the generated design attained a D-efficiency lower bound of 99%. We compare the D-efficiency of the logit link based d-QPSO algorithm-generated design relative to the d-QPSO algorithm-generated designs under the correct link function. In this study we considered the model $Y \sim Bern(\mu)$, with $logit(\mu) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$ with $x_1 \in \{-1, 1\}$ and $x_2 \in [-1, 1]$, but other models can be used. We take $\beta_0 = 1$ and explored $\beta_1 \in [-1.5, 1.5]$ and $\beta_2 \in [-3, 3]$ over a grid with resolution 0.1; this results in a total of $31 \times 61 = 1891$ individual locally D-optimal approximate designs generated for each link function. We then compare how the d-QPSO algorithm-generated locally D-optimal approximate design from the logit link function performs under various other link functions.

Table S2 provides results of the above simulation, and Figure S2 displays the D-efficiencies. We observe that many of the d-QPSO algorithm-generated designs are fairly robust against model misspecification in the link function. When the true link is the probit or log-log link, the logit-based designs tend to perform very well and less so when the true link function is the complementary log-log. Figure S2 suggests the problematic areas occur when β_1 is near 0 for the log-log link and when β_1 and β_2 are both near their extremes for the complementary log-log link. We note that these results assume $\beta_0 = 1$; for different values of β_0, the d-QPSO algorithm-obtained locally D-optimal approximate designs under an incorrect link function may behave differently.

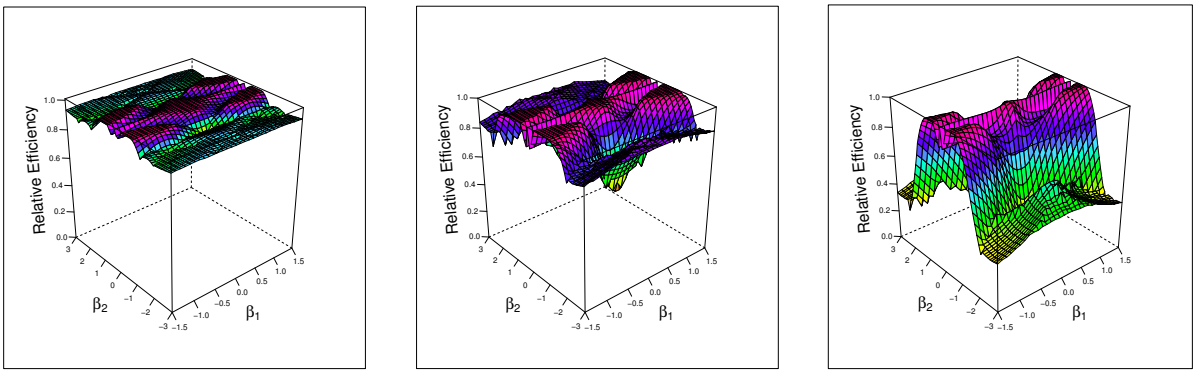

Figure S2: D-efficiencies of the logit link based d-QPSO algorithm-generated locally D-optimal approximate designs relative to the d-QPSO algorithm-generated locally D-optimal approximate designs constructed under the probit (left), log-log (middle), and complementary log-log (right) link functions.

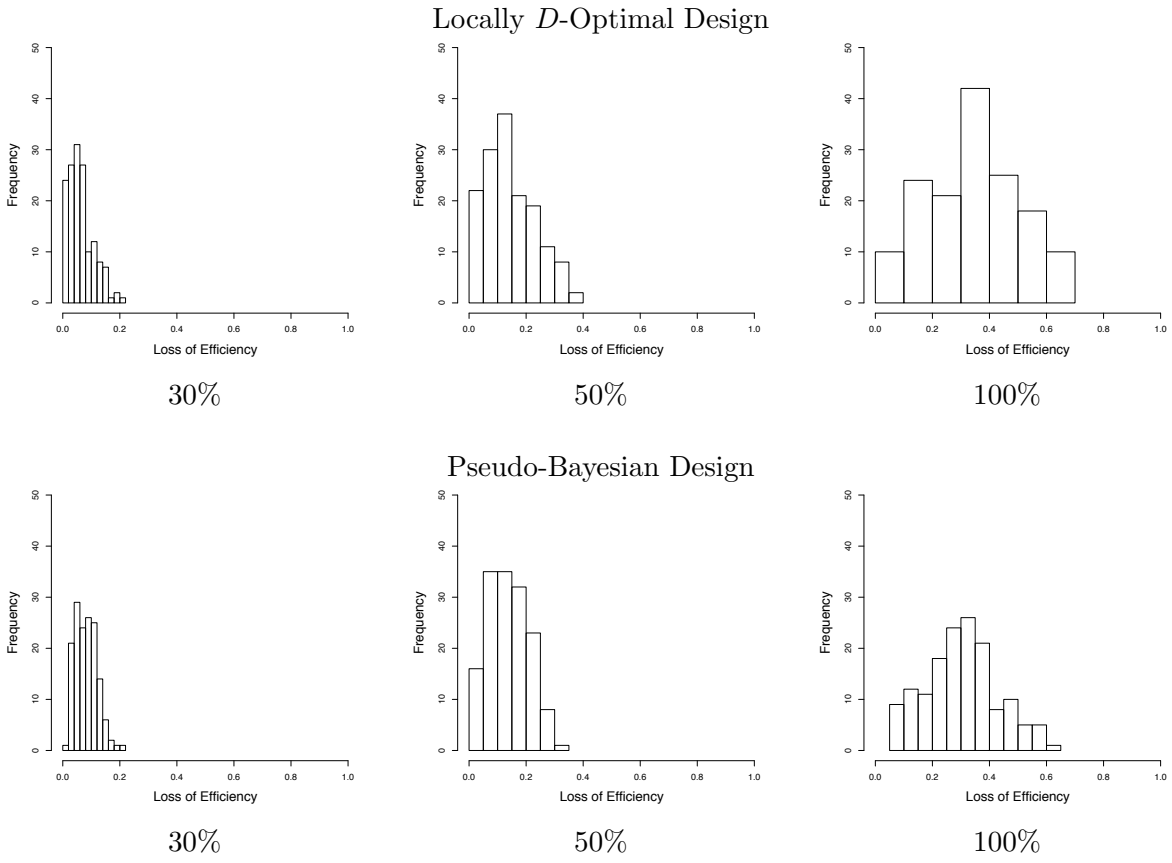

Figure S3: Loss of D-efficiency under 3 different levels of mis-specification: up to 30% of parameter magnitude, 50% of parameter magnitude, and 100% of parameter magnitude, corresponding to the columns in Table S5. Loss of D-efficiency was calculated by taking 1− (D-efficiency of each design to the d-QPSO algorithm-generated locally D-optimal approximate design).

S3.2 Sensitivities of Locally Optimal Designs to Mis-specified Nominal Values

Before a locally D-optimal design is implemented, it is important to investigate if it is robust to mis-specification of the nominal values. When there are multiple parameters the model, the problem becomes complicated since it may not be clear how to vary the nominal values systematically and draw meaningful conclusions. To fix ideas, let us return to the odor removal experiment, and conduct three robustness studies of the d-QPSO algorithm-generated locally D-optimal approximate design in Section 3.1. For the purpose of comparison, we also examine the d-QPSO algorithm-generated pseudo-Bayesian approximate design in Section 4.1. First, we investigate the drop in D-efficiency of the locally D-optimal approximate design when one of the nominal parameters is mis-specified by 10%, 20%, and 30% of its true value. Second, we examine the performance of the design when two parameters are mis-specified using the same setup as before, where all $\binom{6}{2} = 15$ combinations of parameters were considered for mis-specification.

As a third and probably more effective way to assess the effects of mis-specification of the nominal values on the optimal design, we consider cases where the entire nominal parameter vector is mis-specified to some extent. To this end, we perform similar robustness experiments to those carried out by Woods et al. (2006) and Gotwalt et al. (2009) and generate 150 random parameter vectors $\boldsymbol{\theta}_i$, $i = 1, \ldots, 150$ from the prior specification (uniform $\pm$ 100% the magnitude of the nominal values) and compute the locally D-optimal approximate designs $\boldsymbol{\psi}_{\boldsymbol{\theta}_1}, \boldsymbol{\psi}_{\boldsymbol{\theta}_2}, \ldots, \boldsymbol{\psi}_{\boldsymbol{\theta}_{150}}$ using d-QPSO. We evaluate the D-efficiencies of the d-QPSO algorithm-generated locally D-optimal approximate design $\boldsymbol{\psi}_{\boldsymbol{\theta}_0}$ in Section 3.1, and the pseudo-Bayesian design $\boldsymbol{\psi}_{\boldsymbol{B}}$ in Section 4.1, relative to the d-QPSO algorithm-generated locally D-optimal designs $\boldsymbol{\psi}_{\boldsymbol{\theta}_1}, \boldsymbol{\psi}_{\boldsymbol{\theta}_2}, \ldots, \boldsymbol{\psi}_{\boldsymbol{\theta}_{150}}$ as

$$RE_{L,i} = \left(\frac{\left| I_{\boldsymbol{\psi}_{\boldsymbol{\theta}_0}}(\boldsymbol{\theta}_i) \right|}{\left| I_{\boldsymbol{\psi}_{\boldsymbol{\theta}_i}}(\boldsymbol{\theta}_i) \right|} \right)^{1/6} \text{ and } \; RE_{\boldsymbol{B},i} = \left(\frac{\left| I_{\boldsymbol{\psi}_{\boldsymbol{B}}}(\boldsymbol{\theta}_i) \right|}{\left| I_{\boldsymbol{\psi}_{\boldsymbol{\theta}_i}}(\boldsymbol{\theta}_i) \right|} \right)^{1/6},$$

respectively, for $i = 1, \ldots, 150$. Here, $RE_{L,i}$ evaluates the objective function value of the design constructed under the nominal values, $\boldsymbol{\theta}_0$, at the true parameter vector, $\boldsymbol{\theta}_i$, and compares that value to the objective function value of the d-QPSO algorithm-generated locally D-optimal approximate design for $\boldsymbol{\theta}_i$. Similarly, for the pseudo-Bayesian design, $RE_{\boldsymbol{B},i}$ compares the value of the objective function of the design constructed under the prior vector at the true parameter vector $\boldsymbol{\theta}_i$ with that

Table S3: Mean and median D-efficiencies (RE) of the d-QPSO algorithm-generated locally D-optimal and pseudo-Bayesian approximate designs relative to the d-QPSO algorithm-generated locally D-optimal design when one parameter is mis-specified at a time by ±10%, ±20%, and ±30% of the magnitude of its nominal value.

Design	±10%		±20%		±30%	
	Mean RE	Median RE	Mean RE	Median RE	Mean RE	Median RE
Locally Optimal	0.9964	0.9989	0.9852	0.9945	0.9690	0.9872
Pseudo Bayesian	0.9618	0.9645	0.9538	0.9620	0.9424	0.9575

Table S4: Mean and median D-efficiencies (RE) of the d-QPSO algorithm-generated locally D-optimal and pseudo-Bayesian approximate designs relative to the the d-QPSO algorithm-generated locally D-optimal approximate designs when two parameters are mis-specified at a time by ±10%, ±20%, and ±30% of the magnitude of their nominal values (i.e. we randomly generate parameter vectors from uniform priors).

Design	±10%		±20%		±30%	
	Mean RE	Median RE	Mean RE	Median RE	Mean RE	Median RE
Locally Optimal	0.9934	0.9933	0.9745	0.9726	0.9475	0.9427
Pseudo Bayesian	0.9595	0.9617	0.9457	0.9544	0.9264	0.9427

of the d-QPSO algorithm-generated locally D-optimal approximate design for $\boldsymbol{\theta}_i$.

We also sample parameter vectors from two narrower priors (±30% and ±50%) and compare the loss in D-efficiency following the same procedure. We note that the uniform prior of nominal values ±100% was used to construct the pseudo-Bayesian design under consideration and the robustness was evaluated by constructing $\boldsymbol{\psi}_{\boldsymbol{\theta}_1}, \ldots, \boldsymbol{\psi}_{\boldsymbol{\theta}_{150}}$ for each of ±30%, ±50%, and ±100% mis-specification. Thus the two narrower mis-specifications correspond to situations where the experimenter took a very conservative approach even when the true parameter values were actually fairly close to the supposed nominal values.

Tables S3 and S4 list, respectively, results of one- and two-parameter mis-specification in the nominal values, and Table S5 shows the mean and median D-efficiencies based on the 150 simulated values for the full vector of model parameters. Figure S3 provides histograms of the corresponding loss in D-efficiency. For the one- and two-parameter mis-specification simulations we observe that the d-QPSO algorithm-generated locally D-optimal and pseudo-Bayesian designs are both very efficient, even with 30% mis-specification of the magnitude of the nominal values. For the full parameter vector mis-specification simulations, Table S5 and Figure S3 show that the d-QPSO algorithm-generated designs also appear to perform quite well.

Table S5: Mean and median D-efficiencies (RE) of the d-QPSO algorithm-generated locally D-optimal and pseudo-Bayesian approximate designs relative to the the d-QPSO algorithm-generated locally D-optimal approximate designs when all parameters are mis-specified and sampled from an independent uniform prior over $\pm 30\%$, $\pm 50\%$, $\pm 100\%$ of the parameter magnitudes, $\boldsymbol{\mu}$.

Design	$\boldsymbol{\beta} \sim U(0.7\boldsymbol{\mu}, 1.3\boldsymbol{\mu})$		$\boldsymbol{\beta} \sim U(0.5\boldsymbol{\mu}, 1.5\boldsymbol{\mu})$		$\boldsymbol{\beta} \sim U(0\boldsymbol{\mu}, 2\boldsymbol{\mu})$	
	Mean RE	Median RE	Mean RE	Median RE	Mean RE	Median RE
Locally Optimal	0.9359	0.9446	0.8559	0.8690	0.6538	0.6443
Pseudo Bayesian	0.9182	0.9204	0.8612	0.8624	0.6980	0.6995

S4 Locally D-optimal Exact Designs

In Section 4, we showed that the d-QPSO algorithm could be used to find a pseudo-Bayesian exact design. Here we further demonstrate that the algorithm can also find locally D-optimal exact designs. We apply the d-QPSO algorithm to generate locally D-optimal exact designs for the odor removal experiment with nominal values as $\boldsymbol{\beta} = (-1, 2, 0.5, -1, -0.25, 0.13)^T$. We find locally D-optimal exact designs when the total number of observations, N, is specified. We generate locally D-optimal exact designs for $N = 6, 10, 25, 50$ and 100 and note that (i) the case $N = 6$ corresponds to finding a minimally supported locally D-optimal exact design and (ii) when $N = 100$ (which is large), the d-QPSO algorithm-generated locally D-optimal exact design should be similar to the d-QPSO algorithm-generated locally D-optimal approximate design in Section 3.1.

The first three locally D-optimal exact designs were found by the d-QPSO algorithm using 10 swarms, 20 particles, and a termination rule of 5000 iterations. The last two designs were found using 15 swarms instead of 10. For each problem, we ran the d-QPSO algorithm four times to ensure the objective function value was about the same. Table S6 reports the number of support points in the d-QPSO algorithm-generated exact designs, along with their objective function values and the CPU times required to find them. Clearly, when $N > 25$ the objective function value of the exact designs becomes close to that of the d-QPSO algorithm-generated locally D-optimal approximate design, which has a value of 0.3519.

Table S7 displays the d-QPSO algorithm-generated exact designs for $N = 6, 10$, and 15 and shows how their support points are distributed. Table S8 compares the d-QPSO algorithm-generated locally D-optimal exact design for $N = 100$ (right) with the d-QPSO algorithm-generated locally D-optimal approximate design (left). The two designs are aligned by support points, such that each support point on the left is very similar to the one on the right. The three points listed at

Table S6: Properties of the d-QPSO algorithm-generated exact designs for the odor removal experiment with nominal values $\boldsymbol{\beta} = (-1, 2, 0.5, -1, -0.25, 0.13)^T$.

N	Number of Support Points	CPU Time	Objective Function Value
6	6	14.368	0.3368
10	10	15.616	0.3438
25	17	30.223	0.3504
50	18	89.696	0.3510
100	18	337.073	0.3513

Table S7: The d-QPSO algorithm-generated locally D-optimal exact designs for the odor removal experiment with nominal values $\boldsymbol{\beta} = (-1, 2, 0.5, -1, -0.25, 0.13)^T$ for $N = 6, 10$, and 25.

Algae	Scav.	Resin	Comp.	Temp.	N	Algae	Scav.	Resin	Comp.	Temp.	N	Algae	Scav.	Resin	Comp.	Temp.	N
−1	−1	−1	1	29.74	1	−1	−1	−1	1	13.25	1	−1	−1	−1	−1	7.60	1
−1	−1	1	1	27.95	1	−1	−1	−1	1	30.50	1	−1	−1	−1	−1	26.92	2
−1	1	−1	−1	5.00	1	−1	−1	1	−1	34.39	1	−1	−1	−1	1	28.48	2
−1	1	1	−1	33.59	1	−1	1	−1	−1	17.06	1	−1	−1	1	−1	24.95	1
1	−1	1	−1	5.00	1	−1	1	−1	−1	5.00	1	−1	−1	1	−1	35.00	1
1	1	1	1	5.00	1	−1	1	1	1	18.49	1	−1	−1	1	1	29.66	1
						−1	1	1	1	35.00	1	−1	−1	1	1	35.00	1
						1	−1	1	−1	5.00	1	−1	1	−1	−1	5.00	2
						1	−1	1	1	5.00	1	−1	1	−1	1	5.13	2
						1	1	1	1	5.00	1	−1	1	1	−1	19.24	1
												−1	1	1	−1	33.74	1
												−1	1	1	1	35.00	2
												1	−1	−1	1	5.00	1
												1	−1	1	−1	5.00	2
												1	−1	1	1	5.00	2
												1	1	1	−1	5.00	1
												1	1	1	1	5.00	2

Table S8: The d-QPSO algorithm-generated D-optimal exact design (right) for the odor removal experiment with nominal values $\boldsymbol{\beta} = (-1, 2, 0.5, -1, -0.25, 0.13)^T$ with $N = 100$ experimental units and the corresponding d-QPSO algrorithm-generated locally D-optimal approximate design (left).

Algae	Scav.	Resin	Comp.	Temp.	p_i (%)	Algae	Scav.	Resin	Comp.	Temp.	N
−1	−1	−1	−1	9.04	3.70	−1	−1	−1	−1	8.56	4
−1	−1	−1	−1	25.79	4.30	−1	−1	−1	−1	26.25	6
−1	−1	−1	1	29.71	10.17	−1	−1	−1	1	29.95	9
−1	−1	1	−1	35.00	4.73	−1	−1	1	−1	35.00	5
−1	−1	1	1	29.58	11.59	−1	−1	1	1	30.55	8
−1	1	−1	−1	5.00	9.75	−1	1	−1	−1	5.00	9
−1	1	−1	1	5.21	7.86	−1	1	−1	1	5.36	8
−1	1	1	−1	16.89	2.20	−1	1	1	−1	16.73	2
−1	1	1	−1	33.37	8.80	−1	1	1	−1	32.45	6
−1	1	1	1	35.00	6.10	−1	1	1	1	35.00	8
1	−1	−1	1	5.00	5.11	1	−1	−1	1	5.00	5
1	−1	1	−1	5.00	10.75	1	−1	1	−1	5.00	8
1	−1	1	1	5.00	5.23	1	−1	1	1	5.00	8
1	1	1	1	5.00	9.71	1	1	1	1	5.00	7
						−1	−1	−1	1	12.89	1
						−1	−1	1	−1	25.43	2
						1	1	1	−1	5.00	4

the bottom of the exact design have no similar points in the approximate design. The D-efficiency of the exact design relative to the approximate design is 99.8%. Clearly the exact design found by the d-QPSO algorithm is both highly efficient and very similar to the approximate design found in Section 3.1, which is what we expect when N is large.

S5 Optimal Designs on an Irregular Design Space

The bulk of the D-optimal designs reported in the literature are on prototype design spaces. For example, when factors are continuous the default design space is usually the unit cubiod, or, for mixture experiments, the design space is the regular simplex. In practice, some studies have irregularly shaped design spaces, and this is likely to pose additional difficulties for finding an analytical description of the D-optimal design. Such design problems seem to have not been well studied in the literature even though they appear in real problems. In this subsection we show that the d-QPSO algorithm is flexible and can be directly modified to find a locally D-optimal approximate design on an irregularly-shaped design space.

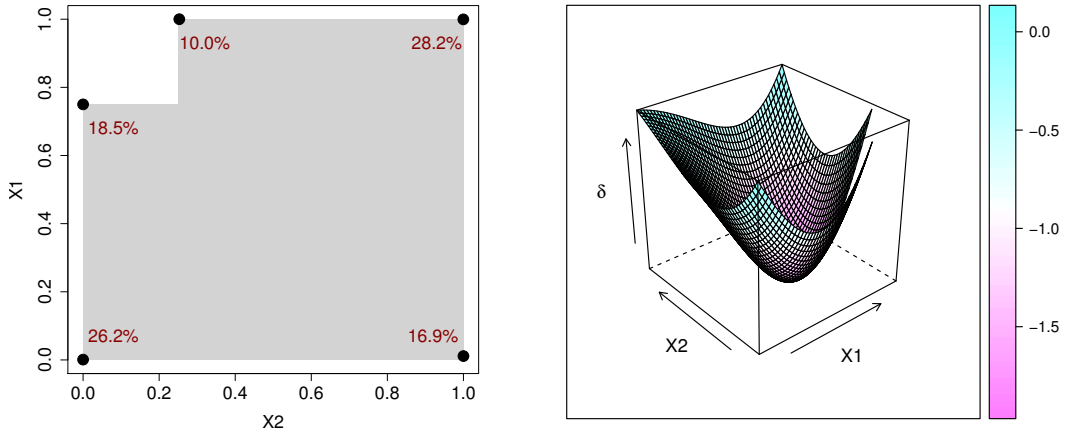

Figure S4: The d-QPSO algorithm-generated locally D-optimal approximate design on an irregular design space with a corner cut off from a box (left) and its sensitivity function (right) with nominal values $\boldsymbol{\beta} = (1.0, -1.7, 1.3)^T$.

To fix ideas, consider a design space which is box-shaped with a corner removed. We consider the model $Y \sim Bern(\mu)$ with $logit(\mu) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$, where Y takes values 1 or 0, and $x_1, x_2 \in [-1, 1]$. We remove the upper left hand corner of this design space by adding the constraint that we cannot have both X_1 above 0.75 and X_2 below 0.25. The nominal parameter vector is $\boldsymbol{\beta} = (1.0, -1.7, 1.3)^T$. We run the d-QPSO algorithm using tuning parameters of 2 swarms, 25 particles in each swarm, and we initialized our search among designs with up to 6 support points. The termination rule was either a maximum of 1000 iterations or when the generated design attained a D-efficiency lower bound of 99%. Figure S4 shows the d-QPSO algorithm-generated locally D-optimal approximate design and the sensitivity plot of the design in the same figure confirms its local D-optimality.

S6 The d-QPSO Algorithm for Finding the Optimal Designs for the Odor Removal Experiment

The C++ code that we provide is the d-QPSO algorithm for finding the D-optimal designs for the odor removal experiment. The code can generate locally D-optimal exact and approximate designs, and also pseudo-Bayesian designs.

References

Gotwalt, C. M., Jones, B. A., and Steinberg, D. M. (2009). Fast computation of designs robust to parameter uncertainty for nonlinear settings. *Technometrics*, 51(1):88–95.

Woods, D. C., Lewis, S. M., Eccleston, J. A., and Russell, K. G. (2006). Designs for generalized linear models with several variables and model uncertainty. *Technometrics*, 48(2):284–292.

Yang, J., Mandal, A., and Majumdar, D. (2016). Optimal designs for 2^k factorial experiments with binary response. *Statistica Sinica*, 26(1):385–411.

Yang, J., Tong, L., and Mandal, A. (2017). D-optimal designs with ordered categorical data. *Statistica Sinica*, 27(4):1879–1902.

Yang, M., Zhang, B., and Huang, S. (2011). Optimal designs for generalized linear models with multiple design variables. *Statistica Sinica*, 21(3):1415–1430.