Shi *et al. Journal of Statistical Distributions and Applications* (2019) 6:3
https://doi.org/10.1186/s40488-019-0092-4

Journal of Statistical Distributions and Applications

RESEARCH Open Access

Particle swarm based algorithms for finding locally and Bayesian *D*-optimal designs

Yu Shi* , Zizhao Zhang and Weng Kee Wong

*Correspondence:
nicoleshi@ucla.edu
Department of Biostatistics, University of California at Los Angeles, 10833 Le Conte Avenue, Los Angeles, USA

Abstract

When a model-based approach is appropriate, an optimal design can guide how to collect data judiciously for making reliable inference at minimal cost. However, finding optimal designs for a statistical model with several possibly interacting factors can be both theoretically and computationally challenging, and this issue is rarely discussed in the literature. We propose nature-inspired metaheuristic algorithms, like particle swarm optimization (PSO) and its variants, to solve such optimization problems. We demonstrate that such techniques, which are easy to implement, can find different types of optimal designs for models with several factors efficiently. To facilitate use of such algorithms, we provide computer codes to generate tailor made optimal designs and evaluate efficiencies of competing designs. As applications, we apply PSO and find Bayesian optimal designs for Exponential models useful in HIV studies and re-design a car-refuelling study for a Logistic model with ten factors and some interacting factors.

Keywords: Bayesian design, Design efficiency, Generalized linear model, Metaheuristic algorithms

Introduction

Statistical models are getting increasingly complex to capture the finer features of a problem. Models incorporate more factors as data becomes high-dimensional and heterogeneous. When the model assumptions are tenable, it is important to develop and implement efficient design strategies to realize the most reliable statistical inference at minimal cost.

In the optimal design literature, we typically assume that the statistical model is fully parametrized, known and defined on a user-selected design space, apart from the unknown parameters in the model. An optimal design is found under a given criterion and the optimization is usually over all designs in the design space. Frequently, the goal is to estimate one or more parameters, the response surface or a couple of meaningful functions of the model parameters. Unless the model is relatively simple, closed form formulae for the optimal designs rarely exist. Sometimes, additional assumptions are imposed to derive the optimal designs analytically and some of the assumptions may be unrealistic. Further, the bulk of the work in the optimal design literature assumes models have a couple of explanatory factors only and when there are several of them, they are usually assumed to be additive to simplify the derivation. A practical and useful way to handle

design problems with many interacting explanatory factors is to develop efficient computational tools that find various types of optimal designs for a broad class of models under realistic assumptions.

We propose a state-of-the-art class of algorithms called nature-inspired metaheuristic algorithms for solving high dimensional design problems. We call them high dimensional design problems because there are many variables to optimize. Our experience is that traditional design algorithms tend to have problems finding optimal designs when there are several variables to optimize. They are likely to stall at a local optimum or break down because of the huge computational burden when there are many variables to optimize. Several researchers had reported similar experiences with traditional algorithms for finding optimal designs. An early one is Chaloner and Larntz (1989) who found both the (Wynn 1972) and (Fedorov 1972) algorithms very slow when they tried to find A- and c-Bayesian optimal designs for the two-parameter logistic model. They then used the general optimization algorithm proposed by (Nelder and Mead 1965) and found it to be adequate. Similarly, (Broudiscou et al. 1996) claimed that traditional algorithms, such as Fedorov-Wynn types of algorithms or exchange algorithms for finding optimal designs cannot be used to find non-standard designs, such as asymmetrical D-optimal designs. They found the algorithms performed poorly and difficult to handle and so not effective; they used genetic algorithms instead. Similarly, (Royle 2002) reported that the traditional exchange algorithms are not practical for finding large spatial designs when the criterion is computationally expensive to evaluate or the discretized design space is large. These may be reasons why the bulk of the optimal experimental designs reported in the literature concern a small number of factors.

Nature-inspired metaheuristic algorithms, such as particle swarm optimization (PSO) or one of its enhanced versions, such as competitive swarm optimizer (CSO), are more likely to solve optimization problems with a large number of variables to optimize. These algorithms are general purpose optimization tools and by construction, do not require any assumptions on the optimization problem. For example, these algorithms can solve optimization problems when the objective function is non-differentiable or even when the criterion cannot be written down explicitly. This article describes PSO, its variant CSO briefly and demonstrates their capability for finding optimal designs for a broad class of models with multiple factors, including Bayesian optimal designs.

The next section reviews the optimal design setup and "Particle swarm optimization based algorithms for generating optimal designs" section presents the particle swarm optimization algorithm. In "Websites for finding optimal designs" section, we present websites where codes for finding optimal designs are available and demonstrate with a simple example. In "Optimal designs for high dimensional models" section, we apply CSO to find high-dimensional D-optimal designs for Logistic and Poisson models. In "Bayesian optimal designs for biomedical studies" section, we apply PSO to find Bayesian D-optimal designs for Exponential models useful in HIV studies. We then conclude with a discussion on future work and a cautionary remark on use of optimal designs in practice.

Background

The statistical model of interest has the form

$$Ey = f(x, \theta), \quad x \in X,$$

where $f(x,\theta)$ is the mean response of the univariate response y and assumed to be known, apart from the $m \times 1$ vector θ of model parameters. There are p possibly interacting factors in the model and so X is p-dimensional. Given a design criterion and a predetermined n of independent observations to take for the study, the design questions are the optimal number (k) of design points to take from X, the optimal locations $x_1, \ldots, x_k$'s in X to observe the responses, and the optimal proportion (w_i) of observations to take at $x_i, i = 1, \ldots, k$. This results in an approximate design and is implemented by first rounding each nw_i to the nearest positive integer $\left[nw_i^*\right]$ and subject to the constraint that $\left[nw_1^*\right] + \left[nw_2^*\right] + \cdots + \left[nw_k^*\right] = n$. There is a theoretical framework for finding optimal approximate designs, including established algorithms for finding many of them and evaluating the proximity of a design to the optimum design even when the latter is unknown; for details on calculating the efficiency lower bound, see (Kiefer et al. 1985).

Following convention, the worth of a design ξ is gauged by its Fisher information matrix. For nonlinear models, the information matrix depends on the unknown values of the parameters θ and we denote this matrix by $M(\xi,\theta)$. The design criterion is then expressed as a function of this matrix and as a first step, we typically replace the unknown θ in the matrix by its nominal value, θ_0. The resulting optimal design is called locally optimal design since it depends on θ_0, which may come from a pilot study or from previous similar studies (Chernoff 1972).

The D-optimal design for estimating all model parameters using θ_0 as nominal values is the design ξ_D that minimizes the negative of the log-determinant of information matrix

$$\xi_D = \underset{\xi \in \Xi}{\operatorname{argmin}} \{ -\log(\det[\, M(\xi,\theta_0)\,]\,) \}, \tag{1}$$

where Ξ is the set of all designs on X. The smaller the criterion value is, the better is the design. When prior information on the model parameter θ is available in the form of a density $\pi(\theta)$, a Bayesian D-optimal design ξ_{BayesD} minimizes the same criterion after averaging out the model parameters with respect to the prior density. It is defined by

$$\xi_{BayesD} = \underset{\xi \in \Xi}{\operatorname{argmin}} \int \left\{ -\log\left(\det[\, M(\xi,\theta)\,]\,\right) \right\} \pi(\theta)\, d\theta, \tag{2}$$

and as before, the smaller the Bayesian D-optimality criterion value is, the better is the design. Clearly, when the prior density is degenerate, the resulting design becomes locally D-optimal. Both criteria are appropriate for estimating model parameters.

When the design criterion is convex in Ξ, as in the above two cases, an equivalence theorem is available to verify optimality of a design over all designs on X. Such a theorem comes from directional derivative considerations of a convex functional and is discussed in design monographs like (Fedorov 1972; Berger and Wong 2005). For example, if m is the dimension of θ and δ_x is the design that takes all observations at x, it can be shown that the design ξ_D is locally D-optimal designs if and only if

$$tr\left\{ M(\xi_D,\theta)^{-1} M(\delta_x,\theta) \right\} - m \le 0, \;\; \textit{for all } x \in X, \tag{3}$$

with equality at the support points of ξ_D. The function on the left is sometimes called the sensitivity function. Different convex design criteria lead to different sensitivity functions but they all have a similar form. In practice, the optimality of a design is verified by plotting the sensitivity function against the design space (equivalence plot) and checking whether the equivalence theorem is satisfied. If it is not, the design is not D-optimal. Similarly, ξ_{BayesD} is Bayesian D-optimal with respect to the prior density $\pi(\theta)$ if and only if

$$\int_{\theta} tr\left\{M(\xi_{BayesD},\theta)^{-1}M(\delta_x,\theta)\right\}\pi(\theta)d\theta - m \leq 0, \ \ for\ all\ x \in X, \tag{4}$$

with equality at the support points of ξ_{BayesD}.

We compare designs using relative efficiency, which is commonly defined as the ratio of the two criteria values, or a function thereof. When one of the designs been compared is the optimal design, relative efficiency becomes the design efficiency. Specifically, for D-optimality, we compare two designs ξ_1 and ξ_2 with nominal values θ_0 via the m^{th} root of the ratio of the determinants of their information matrices:

$$\left\{\frac{|M(\xi_1,\theta_0)|}{|M(\xi_2,\theta_0)|}\right\}^{1/m}. \tag{5}$$

The relative efficiency ratio compares performance of the two designs for estimating the model parameters. If the above ratio 0.5 or 50% efficiency, this means that the design ξ_1 needs twice as many observations for it to do as well as the design ξ_2. When ξ_2 is the D-optimal design, the above ratio is simply the D-efficiency of the design ξ_1.

The next section describes a nature-inspired metaheuristic algorithm and one of its variants for finding D-optimal designs for the Poisson regression models, Bayesian D-optimal designs for Exponential models and D-optimal designs for high-dimensional Logistic and Poisson models.

Particle swarm optimization based algorithms for generating optimal designs

Metaheuristic algorithms are increasingly common, and a key appeal of such algorithms is that there is no or minimum assumptions required for them to work well. Metaheuristic algorithms usually involve some randomization and local searches. In particular, they go through slightly different processes and end up with frequently not too different results (Yang 2010).

We focus on particle swarm optimization (PSO), which is a member of the class of metaheuristic algorithms. It is now widely and routinely used in the engineering field. PSO was first developed in 1995 by Eberhart and Kennedy (1995). Motivated by swarm intelligence, they simulated candidate designs for the optimum using them as birds in a swarm looking for food on the ground. The swarm collectively acts and communicates to update where each bird believes where the food is (personal best) and flies toward it in the direction of the group best, which is where the flock believes the food lies after sharing information within the flock. The objective function gets updated at each iteration as the bird flies over time in search of a quality solution. Many have reported that PSO can significantly outperform genetic algorithms (GA) in terms of number of function evaluations required (Hassan et al. 2005).

To initiate the PSO algorithm, the user first selects a swarm size where each particle in the swarm represents a randomly generated candidate design. Below is the pseudo-code for PSO (Kennedy 2006):

Begin
Initialize particle position and velocity
While maximum iterations or minimum error criteria is not attained
Do
For each particle
  Evaluate objective function

Update particle personal best
End
Set particle with the best objective function value as the group best
For each particle
Update particle velocity: $v_{id}^{t+1} = wv_{id}^{t} + c_1\psi_1\left(p_{id}^{t} - x_{id}^{t}\right) + c_2\psi_2\left(p_{gd}^{t} - x_{id}^{t}\right)$
Update particle position: $x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1}$
End
End

In the pseudo-code, w is inertia weight, v_{id}^{t} and x_{id}^{t} are velocity and position of d^{th} dimension of i^{th} particle at iteration step t, c_1, c_2 are weight constants, ψ_1 and ψ_2 are random values from uniform $[0, 1]$ distribution, p_{id} is the personal best of d^{th} dimension of particle i (the best position particle i ever visited), p_{gd} is the group best of d^{th} dimension of the swarm (the best position the group ever visited). Each particle represents a candidate design with k support points, and k ($\geq m$) is user selected. The dimension of each particle is therefore $2k - 1$ because we need to optimize the locations of the k support points and their corresponding weights. The dimension is one smaller since the weights sum to unity.

For solving high-dimensional optimization problems, it is helpful to use a variant of PSO because research shows that PSO can be prone to premature convergence (Yang and Pedersen 1997). This means that particles can quickly converge to a local optimum without enough space exploration. This phenomenon decreases the quality of the solution provided by PSO. Many strategies have been proposed to alleviate such premature convergence, and one successful PSO variant that shows great potential for solving complex optimization problems is the competitive swarm optimizer (CSO) algorithm (Cheng and Jin 2015). The researchers found that PSO premature convergence has strong connection with particle personal best and swarm group best, which seem to have too much influence on the convergence of each particle. They proposed CSO by removing those two "black holes" (namely, personal best and group best) and recast the updating formulas. Further, CSO pairs up particles and let the "loser" (particle with inferior objective function value) to learn from "winner" (particle with superior objective function value). The main change compared to PSO is that the updating mechanism for the "loser" particle velocity becomes

$$v_{id}^{t+1} = \psi_1 v_{id}^{t} + \psi_2\left(x_{jd}^{t} - x_{id}^{t}\right) + \psi_3\phi\left(\bar{x}_{d}^{t} - x_{id}^{t}\right)$$

where v_{id}^{t} and x_{id}^{t} are velocity and position of d^{th} dimension of i^{th} particle at iteration step t, x_{jd}^{t} is the position of d^{th} dimension of the paired j^{th} particle at iteration step t, ϕ is a tuning parameter, ψ_1, ψ_2, ψ_3 are random values from uniform$[0, 1]$ distribution, $\bar{x}_{d}^{t}$ is the d^{th} dimension of swarm center at iteration t. CSO makes the swarm more diverse without increasing much computational cost and shows that it is less likely to be trapped in local optimum (Cheng and Jin 2015).

We now show how PSO can find different types of optimal designs effectively for different types of generalized linear models. The examples are meant to be illustrative with some details for those new to the area; others may use our codes directly from the following websites that contain codes for finding optimal designs for more complicated situations.

Websites for finding optimal designs

There are websites with MATLAB PSO codes that we have written for finding various optimal designs for commonly used models. They include http://wkwong.bol.ucla.edu/podpack/index.html, http://www.math.ntu.edu.tw/~optdesign/ and http://www.stat.ncku.edu.tw/optdesign/. Each code is for a specific design problem for a particular model. The user inputs the required information for their design problem and the PSO code searches iteratively for the optimum.

The available codes on our website find optimal designs for different models under different criteria. Models include commonly used linear, Michaelis-Menten, mixture polynomial, logistic, compartmental, Hill's, double-exponential, exponential, Poisson, etc. Criteria include *D*-, D_s-, *A*-, *G*-, *E*-, minimax, etc.

The aim in many toxicity studies is to ascertain the joint toxicity effects from several toxicants on the number of organisms or cells that survive when we apply different dose combinations of the toxicants. There is limited work to address design issues for such studies and when they are available, they usually have one or two explanatory variables in the model (Russell et al. 2009; Wang et al. 2006; Qiu 2014). To fix ideas, we use a Poisson model with two toxicants to illustrate how our sites facilitate search for a locally *D*-optimal design for a Poisson regression model. The website has codes that are able to find designs with more interacting toxicants.

Let y_i be the observed number of organisms or cells that survive when we apply the i^{th} dose combination of the two interacting toxicants $x_i = (x_{i1}, x_{i2})^T$. Let the mean response of the Poisson regression model be λ_i, which is the same as its variance $Var(y_i|x_i)$. Further, we assume the mean structure in our statistical model is

$$\lambda_i = exp(\theta_0 + \theta_1 x_{i1} + \theta_2 x_{i2} + \theta_{12} x_{i1} x_{i2}) = exp\left(\boldsymbol{\theta}^T f(x_i)\right), \tag{6}$$

where $\boldsymbol{\theta}^T = (\theta_0, \theta_1, \theta_2, \theta_{12})$ and $f^T(x_i) = (1, x_{i1}, x_{i2}, x_{i1}x_{i2})$. The Fisher information for a design with k support points $\xi = (x_1, \ldots, x_k; w_1, \ldots, w_k)$ with Poisson rate λ_i, $i = 1, 2, \ldots, k$ has the form

$$I(\xi, \boldsymbol{\theta}) = F^T W F, \tag{7}$$

where $F = (f(x_1), \ldots, f(x_k))^T$ and $W = diag(w_1\lambda_1, \ldots, w_k\lambda_k)$. We consider a synergism effect in this example, and set the nominal values for θ_0, θ_1, θ_2 and the interaction term θ_{12} to be -0.1, -0.1, -0.1, and -0.01. The values are non-positive because we expect the effect of each toxicant is such that fewer cells will survive when the dose of the toxicant is increased. We also set the nominal value of θ_{12} to be smaller than the additive effects, which is usually the case in practice (Wang et al. 2006). Another restriction on the feasible design region representing doses or concentrations of the toxicants is that their values are non-negative.

Our environment for this demonstration uses a x86_64-win64 (64 bit) 3.6GHz, 16GB RAM computer with Intel i7-4790 CPU on Windows 10 Enterprise OS. The version of MATLAB we used is R2015b. We first download the codes from "Part H1: Locally *D*-optimal design for Poisson regression model with M = 2" from the website. Upon typing "run" in the command window, a Graphic User Interface (GUI) pops up as is shown in Fig. 1, whereupon the user inputs the nominal values of parameters, the anticipated number of support points for the optimal design and parameters for the PSO algorithm. The user can change the tuning parameters in PSO or change the nominal values for

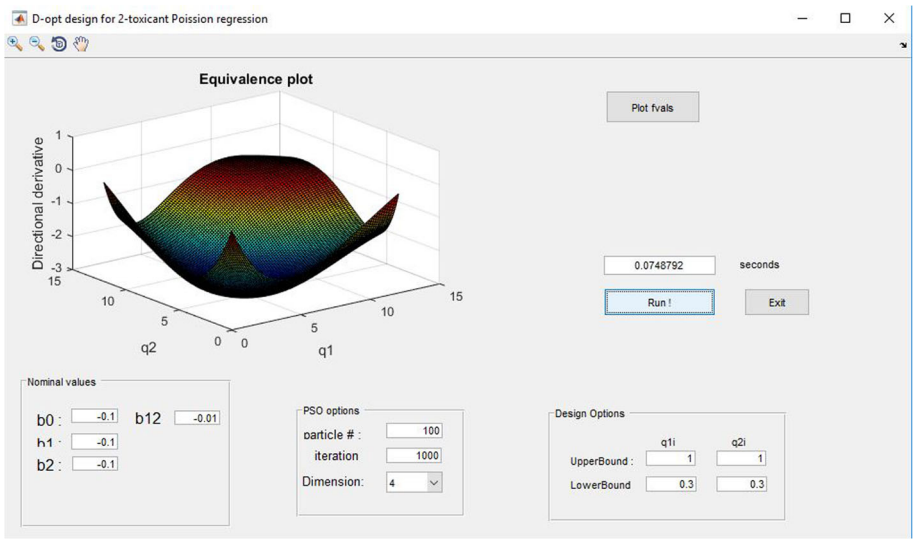

Fig. 1 Poisson regression model MATLAB Graphic User Interface (GUI)

the parameters. We assume there are 4 support points for the optimal design (minimally supported). For our demonstration, we set PSO tunning parameters as $c_1 = c_2 = 2$, w linearly decreasing from 0.9 to 0.4 (Shi and Eberhart 1999) and use 100 particles and 1000 iterations as tuning parameters for PSO. When the "Run!" button is clicked, the program runs and the search begins. The design found by the algorithm is displayed in the command window, as is shown in Fig. 2, with the dose levels of toxicant 1, toxicant 2 and their corresponding weights. We observe that the four support points are equally supported at (0.3, 1), (0.368, 0.368), (1, 0.3), and (1, 1). Additionally, the output displays the D-efficiency lower bound and the criterion value of the log-determinant of the Fisher information matrix. The generated design has a D-efficiency lower bound of 1, confirming that the design is locally D-optimal. The sub-window of Fig. 1 shows the sensitivity function plot of the PSO-generated design and visually also confirms optimality of the generated design among all designs. In this experiment we observe that the the criterion values become stable after about 100 iterations. This is typical of PSO where it tends to get to the proximity of the optimum quickly and afterwards exploits the locality to determine the optimum.

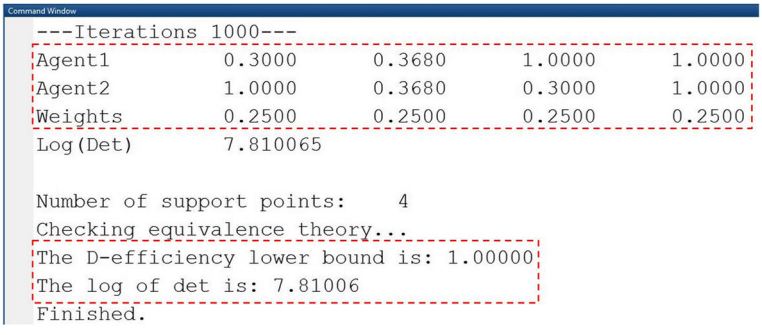

Fig. 2 The command window with the optimal designs, D-efficiency lower bound, and the criterion value

Optimal designs for high dimensional models

In practice, models are likely to have several explanatory factors. This is because a few explanatory factors may not capture the complex structure of the full data adequately. This section shows that CSO, a variant of PSO can tackle high-dimensional optimal design problems for the Logistic and Poisson models.

Locally *D*-optimal designs for 5-factor Logistic and Poisson regression models

Our models are generalized linear models and to fix ideas, consider the more popular Logistic model and a Poisson model, each with five explanatory factors and all pairwise interactions. The Logistic model is given by

$$logitE(y) = \theta_0 + \theta_1 x_1 + \cdots + \theta_5 x_5 + \theta_6 x_1 x_2 + \cdots + \theta_{15} x_4 x_5, \tag{8}$$

where the outcome y is Bernoulli-distributed, and each factor x_i resides in the design space $[-1, 1]$. For the Poisson model, its mean response is λ_i, which is the same as its variance $Var(y_i|x_i)$. In terms of the explanatory factors, the mean structure is

$$\lambda_i = exp(\theta_0 + \theta_1 x_1 + \cdots + \theta_5 x_5 + \theta_6 x_1 x_2 + \cdots + \theta_{15} x_4 x_5). \tag{9}$$

We expect the locally *D*-optimal design for each of the above models has at least $k \geq 16$ design points because there are 16 parameters in both models. This means that we have $k - 1$ weights to determine and at least $k \geq 16$ design points to determine, implying the total number of variables we need to optimize in this problem is at least 95. If $k = 25$, for instance, this number becomes 149 and so the problem becomes high-dimensional rapidly. In the event that the *D*-optimal design has $k = 16 = m$ support points, we have a minimally-supported design.

As always, the choice of the tuning parameters in an evolutionary algorithm deserves attention. For the hard high-dimensional problems in this paper, we used 200 particles. The values for the other parameters we used were suggested by (Cheng and Jin 2015); in particular, we set $\phi = 0.05$ in CSO. We stop the algorithm when the change in the values of objective function from successive iterations is smaller than 10^{-5}.

We implemented PSO and Genetic Algorithm (GA) and compared their performance with CSO for searching *D*-optimal designs for high dimensional models. The choices for the tuning parameters in PSO were $w = 0.9$ and $c_1 = c_2 = 2$ (Shi and Eberhart 1998). The tuning parameter EliteCount in Genetic Algorithm was 0.05, which is recommended by the Matlab official implementation of the code. The swarm size of PSO and GA was also 200. Since evolutionary algorithms are stochastic and produce slightly different result for each run, we ran the algorithm five times for each model and averaged the outputs.

For simulation purposes, parameters for Logistic models were generated randomly from uniform [-1, 1] and parameters for Poisson models were generated randomly from uniform [-3, 3]. These nominal values are listed in Table 1. The design space is $[-1, 1]^5$.

Table 2 displays the *D*-optimality criterion values, averaged over five runs, obtained by the three algorithms for the four models 8-1, 8-2, 9-1 and 9-2, along with their standard deviations in parentheses. Among the three algorithms, CSO consistently has better and more stable performance for searching *D*-optimal designs for the four simulated models. The equivalence theorem can be used to verify optimality of the CSO-generated designs but the high-dimensional sensitivity function plots are not necessarily easy to construct

Table 1 Parameter values for two Logistic and two Poisson simulated models

Model	$\theta_0, \theta_1, \cdots, \theta_{15}$
8-1 (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]
8-2 (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]
9-1 (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]
9-2 (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]

Parameters for Logistic models were generated randomly from uniform[-1, 1] and parameters for Poisson models were generated randomly from uniform [-3, 3]

and interpret visually. A more practical way is to determine the maximum of the sensitivity function of the generated design across the design space and compute its efficiency lower bound. For our examples, the average efficiency lower bounds of the generated designs for the four models are 93%, 96%, 99% and 99%, suggesting that the generated designs are highly *D*-efficient. We also observe that the average runtime for each algorithm shown at the bottom of the table confirms CSO not only produces the best quality solutions but also does so most efficiently.

As an example, we display at least 99% *D*-efficient design found by CSO for model 9-2 in Table 3. The first five columns show the support points and the last column shows the weight associated with each design point.

We verify the optimality of this and other designs by checking the values of sensitivity function over a user-selected discretized grid set in the design space. Clearly for models with several explanatory factors, the finer the grid set, the longer time it takes to check optimality. It is helpful to start with some initial testings with a rough grid to determine whether there are violations of the equivalence theorem; if there are, this suggests the design is not optimal and another should be generated. For this particular example, after initial testings, we discretized the search space with a total of $(2/0.2 + 1)^5 = 161051$ grid points for this 5-factor model, i.e. a step size of 0.2 for each of the factor spaces. We then plot the multi-dimensional sensitivity function over the grid set, which is now much harder to visualize and appreciate its properties than the case when there is only one explanatory factor. One option is to stretch the high-dimensional grid into a one-dimensional vector on the x-axis and plot the sensitivity function values along the x-axis.

Figure 3 is an example of such a plot where it shows the graph of the sensitivity function of the design in Table 3. The plot shows that there are many zero points in the graph. Such

Table 2 Average criterion values of the locally *D*-optimal designs found by genetic algorithm (GA), particle swarm optimization (PSO) and competitive swarm optimizer (CSO) for the 5-factor models, along with their standard deviations in parentheses

Model	GA	PSO	CSO
8-1	29.54(0.83)	31.05(1.51)	28.80(0.37)
8-2	29.76(1.12)	30.78(1.07)	28.91(0.54)
9-1	-167.30(1.31)	-163.11(0.92)	-169.04(1.24)
9-2	-100.14(1.71)	-93.23(1.40)	-100.35(0.64)
Average Runtime	95.2s	64.5s	42.3s

Table 3 A CSO generated 21 point design for model 9-2 with at least 99% D-efficiency

x_1	x_2	x_3	x_4	x_5	w
-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	0.296	0.046
-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	0.036
0.746	-1.000	-1.000	1.000	-1.000	0.047
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

discrepancy can be explained from several perspectives: (1) the design is very close to the true optimal design, but still not the true one; (2) when plotting, we systematically chose points from the high-dimensional space. One theoretical design point might be spatially close to more than one points in the grid we made; (3) although values at some points seem to achieve 0, if we amplify the graph, we may find out that they are not 0 points.

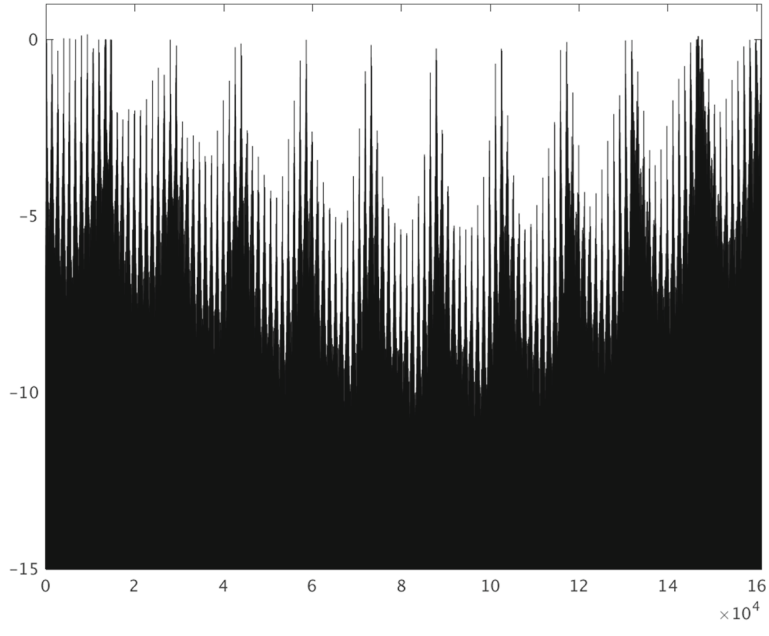

Fig. 3 Plot of the sensitivity function for the design in Table 3

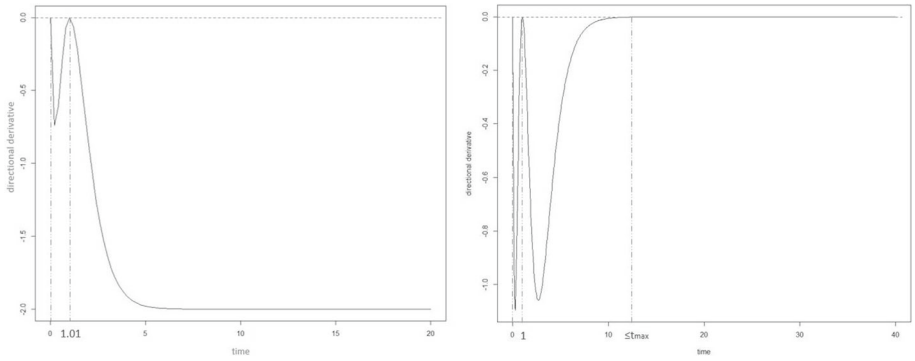

Fig. 4 Sensitivity function plots of the generated designs for Exponential regression model 10 (left) and model 11 (right)

A real-world application on car refueling experiment

In this section, we describe a real-world application which tries to find a high-dimensional optimal design for a 10-factor Logistic model. Grimshaw et al. (2001) described an experiment 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. The experiment has four binary explanatory factors taking values -1 or 1: ring type (x_1, white paper or reflective), lighting (x_2, room lighting or 2 flood lights and room lights), sharpening (x_3, without or with), smoothing (x_4, without or with); also included are six continuous factors: lightning angle (x_5, 50 to 90 degrees), gas-cap angle 1 (x_6, 30 to 55 degrees), gas-cap angle 2 (x_7, 0 to 10 degrees), can distance (x_8, 18 to 48 inches), reflective ring thickness (x_9, 0.125 to 0.425 inches) and threshold step vale (x_{10}, 5 to 15). Lukemire et al. (2019) employed a variant of PSO called quantum PSO to search for a locally D-optimal design for the 10-factor additive Logistic model and reported an average runtime of 140 seconds. Here we include some two and three-way interaction terms . All these terms are summarized in table 4. The model contains 10 factors and 16 parameters in total. Here, we started the search with a swarm, each with 20 support points. Finding optimal design for this problem is a high-dimensional problem: for each solution, there is (10+1)$\times$ 20=220 dimensions, as each design point has 10 factors and 1 corresponding weights, and there are 20 support points to be optimized.

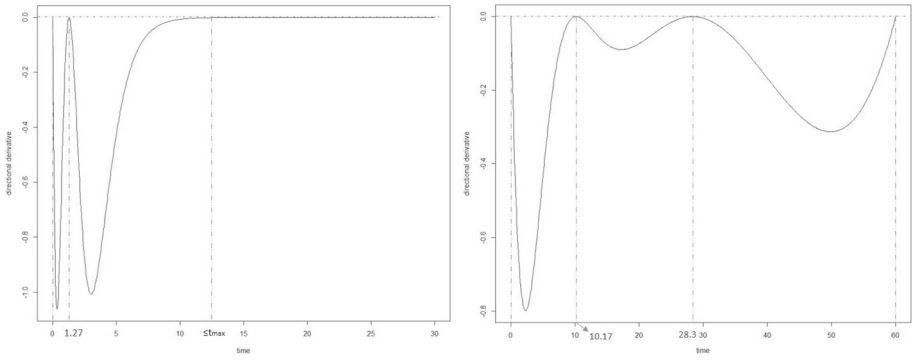

Fig. 5 Sensitivity function plots of the the generated design for the Exponential regression model 12 under specification 12-1 (left) and specification 12-2 (right)

Table 4 Variables in the car refueling experiment

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	$x_3x_4x_{10}$	-	-

To find the locally *D*-optimal design, a set of parameter values are proposed: $\boldsymbol{\theta}^T$ = (3.0, 0.5, 0.75, 1.25, 0.8, 0.5, 0.8, -0.4, -1.00, 2.65, 0.65, 1.1, -0.2, 0.9, -0.36, 1.07). We used 200 CSO particles to search the design space; we ran the simulation for 10 times in order to obtain the averaged runtime. We first tested the algorithm on the additive model used in (Lukemire et al. 2019), and CSO spent 24 seconds on average to find the optimal design, which is shown in Table 5 and the optimal criterion value is -35.95. This confirms that CSO has superior capability to find the optimal design efficiently. When testing CSO on the model with interaction terms, we found a design with an efficiency lower bound of 99% efficient using around 957 seconds. These efficiency was calculated on 1882384 sampled grid points uniformly drawn from the design space. Table 6 displays the 17-point locally *D*-optimal design. Its weight distribution on these points is in the last column and the criterion value for this design is 7.2562.

Table 5 The 12-point locally *D*-optimal design for the additive 10-factor car refueling experiment

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	w
-1	-1	-1	-1	50.000	30.000	4.200	48.000	0.125	5.000	0.091
-1	-1	-1	-1	50.000	30.000	10.000	48.000	0.125	8.570	0.091
-1	-1	-1	-1	50.000	30.000	10.000	45.680	0.125	5.000	0.091
-1	-1	-1	-1	54.640	30.000	10.000	48.000	0.125	5.000	0.091
-1	-1	-1	-1	50.000	32.900	10.000	48.000	0.125	5.000	0.091
-1	-1	-1	-1	50.000	30.000	10.000	48.000	0.125	5.000	0.081
-1	-1	-1	-1	50.000	30.000	10.000	48.000	0.425	5.000	0.077
-1	-1	-1	1	50.000	30.000	10.000	48.000	0.125	5.000	0.091
-1	-1	1	1	50.000	30.000	10.000	48.000	0.125	5.000	0.091
-1	1	1	-1	50.000	30.000	10.000	48.000	0.125	5.000	0.091
1	-1	-1	-1	50.000	30.000	10.000	48.000	0.125	5.000	0.075
1	-1	-1	-1	50.000	30.000	10.000	48.000	0.425	5.000	0.004

The vector of nominal values for the model parameters is $\boldsymbol{\theta}^T = (3.0, 0.5, 0.75, 1.25, 0.8, 0.5, 0.8, -0.4, -1.00, 2.65, 0.65)$

Table 6 The 17-point locally *D*-optimal design for the 10-factor car refueling experiment with two and three factor interactions

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	w
1	-1	-1	-1	50.000	30.000	0.026	31.494	0.125	5.000	0.062
1	-1	1	-1	90.000	30.000	0.285	18.000	0.425	5.000	0.063
1	-1.	1	-1	90.000	37.342	0.000	47.999	0.425	15.000	0.061
1	-1	1	1	68.511	55.000	0.209	29.239	0.425	15.000	0.062
1	1	-1	-1	90.000	30.000	0.085	28.026	0.125	15.000	0.062
1	1	-1	-1	90.000	31.591	0.000	34.269	0.425	5.000	0.062
1	1	1	-1	50.000	55.000	0.000	33.014	0.125	5.000	0.062
-1	-1	-1	-1	50.000	36.649	0.000	48.000	0.425	15.000	0.061
-1	-1	-1	-1	90.000	55.000	0.025	48.000	0.425	5.000	0.062
-1.	-1	-1	-1	90.000	55.000	0.091	36.073	0.125	15.000	0.061
-1	-1	-1	1	75.860	30.000	0.363	18.000	0.125	15.000	0.063
-1.	-1	1	-1	50.000	55.000	0.007	36.516	0.125	15.000	0.062
-1	-1	1	-1	90.000	30.000	0.029	38.137	0.425	15.000	0.020
-1	-1	1	-1	90.000	30.000	0.000	45.986	0.125	5.000	0.060
-1	1	-1	-1	50.000	30.000	0.000	34.471	0.125	15.000	0.057
-1	1	-1	1	67.477	30.000	0.070	48.000	0.125	15.000	0.063
-1	1	1	-1	50.000	30.000	0.011	18.361	0.425	15.000	0.056

The vector of nominal values for the model parameters is $\boldsymbol{\theta}^T = (3.0, 0.5, 0.75, 1.25, 0.8, 0.5, 0.8, -0.4, -1.00, 2.65, 0.65, 1.1, -0.2, 0.9, -0.36, 1.07)$

Bayesian optimal designs for biomedical studies

Bayesian optimal designs incorporate prior knowledge of the model parameters at the design stage. The prior knowledge usually comes in the form of a probability density function for the parameters and is averaged out by numerical integration before an optimization scheme is applied to find the optimal design. Because the integration and optimization spaces can be very different objects, each with varying magnitude, finding a Bayesian optimal design in a high dimensional problem can be very challenging. Here, we show that PSO is a promising tool for finding Bayesian *D*-optimal designs for Exponential models which are commonly used in pharmacokinetic/pharmacodynamic studies.

In HIV studies, Exponential models are frequently used to characterize viral load changes with time after administration of a potent inhibitor of HIV-1 protease in vivo (Perelson et al. 1996). Derived from a series of ordinary differential equations that describes the virus change in different compartments, the model is a good representation of longitudinal HIV dynamics. The important parameters in such a model include virus clearance rate and infected cell life span (Wu and Ding 1999). Some analytic locally optimal designs and Bayesian optimal designs are available for the Exponential models (Han and Chaloner 2003; Dette and Neugebauer 1997).

This section describes how to use prior information to study drug effect and understand the longitudinal viral dynamics. A well thought out design to draw plasma samples from patients to measure the HIV-1 RNA copies is essential for estimating the model parameters accurately. Han (2003)(Han and Chaloner 2003) provides some simple but useful models:

$$Y_j = P_1 e^{-\delta t_j} + \epsilon_j \tag{10}$$

$$Y_j = P_0 + P_1 e^{-\delta t_j} + \epsilon_j \tag{11}$$

and

$$Y_j = log\left(P_0 + P_1 e^{-\delta t_j}\right) + \epsilon_j. \tag{12}$$

Here Y_j is the viral load at time t_j and the sampling times are $t_j \in [t_{min}, t_{max}] \subseteq [0, 60]$. Both t_{min} and t_{max} are pre-selected and refer, respectively, to the minimum and maximum time where observations can be taken. We have $\epsilon_j \overset{iid}{\sim} N(0, \sigma^2)$ and the model parameters we wish to estimate are P_0, P_1 and δ (all > 0). Following (Han and Chaloner 2003), the prior densities for P_0, P_1 are both uniform [0.5, 1.5], and for δ is uniform [0.9, 1.1]. Additionally we use a different specification for model 12 with the usual P_0 and P_1 but with $\delta \sim$ uniform [0, 0.2]. We call the former specification model 12-1 and the latter model 12-2 and try to compare the properties of Bayesian optimal designs under different specifications. The priors can be quite flexible and they do not have to be independent. The above three models are simplified Exponential regression models with two to three parameters, and one can easily extend to the models with more parameters based on the disease stages. Specifically, model 12 describes the trajectory of plasma HIV RNA level under antiviral treatment (Wu and Ding 1999); model 10 and 11 are special cases when P_0 is treated as a nuisance parameter.

Similarly, we set PSO tunning parameters as $c_1 = c_2 = 2$, w =0.9 (Shi and Eberhart 1998) and use 40 particles and 1000 iterations. We use 1000 Monte Carlo samples to compute the numerical integral in the objective function (Eq. 2). Other numerical integration techniques such as Gaussian quadrature or sparse grid can be used to confirm the accuracy of the numerical integration. Table 7 provides the PSO found designs for different models.

The sensitivity function plots of the four PSO generated designs are shown in Figs. 4 and 5, below and they all confirm that the designs are Bayesian optimal. The plots suggest that the design remains optimal if $t = 14$ instead of t_{max}. We note that changing prior specifications considerably changes the optimal design. For example, when we compare model 12-2 with model 12-1, we observe that the locations of the support points and the number of support points also change. The optimal design for model 12-2 requires 4 support points compared to 3 support points for model 12-1. Moreover, the optimal design support points for model 12-2 are more dispersed, and include middle and end time points, whereas the optimal design support points for model 12-1 are concentrated in the earlier time. As is shown in Fig. 6, the differences in designs might be attributed to the fact that larger δ value in Eq. 12 flatten out responses quickly, and design points drawn at earlier time points are critical in providing insights on the exponential decay rate. On the contrary, the small δ value response curve decays smoothly, and later time points are still on the decreasing curve so that more information can be obtained by allocating design points sparsely.

Table 7 Designs found by PSO for the Exponential regression models

Model	# Model parameters	Design (time, weight)
10	2	((0, 1/2), (1.01, 1/2))
11	3	((0, 1/3), (1, 1/3), (t_{max}, 1/3))
12-1	3	((0, 1/3), (1.27, 1/3), (t_{max}, 1/3))
12-2	3	((0, 0.32), (10.17, 0.28), (28.30, 0.10), (60, 0.30))

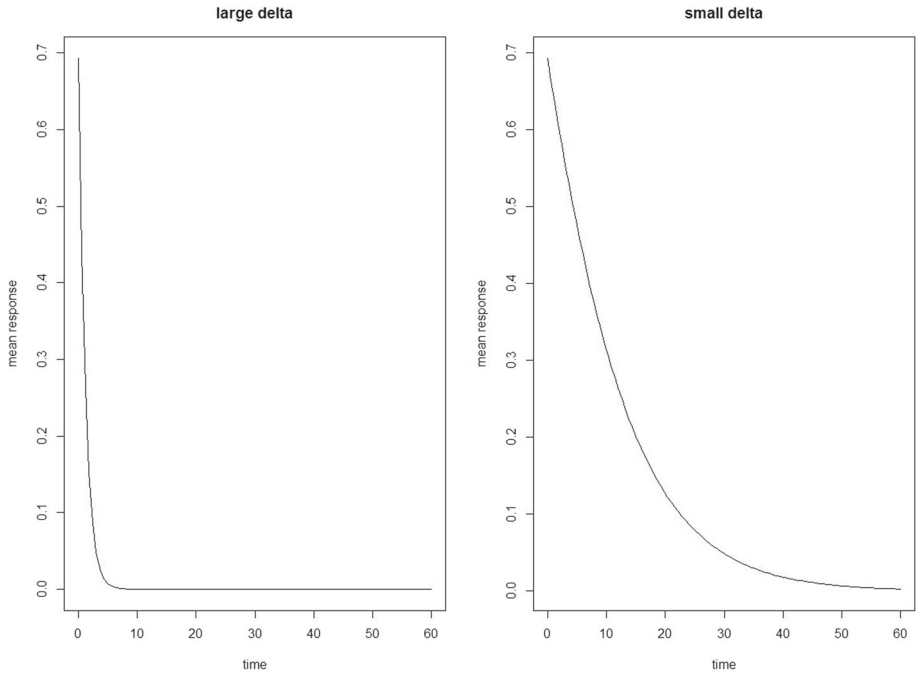

Fig. 6 Mean response curves of model 12 with δ fixed at 1 (left) and 0.1 (right), using mean values of the parameters

Discussion

Constructing efficient designs is critical to best use the data to reach reliable estimations. One potential issue in constructing high-dimensional or Bayesian designs is that the computational time will increase when the model becomes large and the number of the design points increases. Even for a simple linear additive model with 20 factors and 21 design points, the dimension of the optimal design problem is $20 \times 21 + 20 = 440$. To solve this constrained optimization problem, PSO has to optimize 440 variables. Consequently there is high computational costs. Fortunately, parallel computing techniques can be applied to accelerate the computations. Hung and Wang (2012) proposed a GPU-accelerated PSO (GPSO) algorithm by using a thread pool model and implemented GPSO on a Graphic Processing Unit (GPU). The authors demonstrated that the proposed GPSO can significantly reduce the computational burden with satisfactory parallel efficiency. Likewise, (Chen et al. 2013) proposed a discrete PSO approach, named LaPSO, to search for an optimal Latin hypercube design. The authors accelerated LaPSO by using GPU and showed that the GPU implementation can save computational time significantly for large optimization problems. We expect that the programs in this article can also be accelerated on parallel computers such as GPU. The parallelization will likely produce computing tools with faster response time and better user experience.

There are continuing challenging problems in our work. We do not claim we are able to find all types of optimal designs in a regression setup using the types of algorithms we proposed here. We point out a few of these problems:

- we have experiences with metaheuristic algorithms that work well for some nominal values for a model but not for other nominal values; similarly, the same algorithm may not work well when the design space is changed, and especially when it is

enlarged. These are likely scaling problems that our current work is trying to address and understand.

- confirming optimality of a generated design remains a challenge because it is difficult to appreciate interesting features in a high dimensional plot. An alternative is to find its efficiency lower bound, which amounts to solving another high-dimensional optimization problem to find the maximal value of the sensitivity function. This means that the metaheuristic algorithm has to be applied twice or find another more appropriate metaheuristic algorithm to solve this second optimization problem.
- as the number of explanatory factors in the model increases, so is the number of variables we need to optimize. For example, in the 10-factor car-refueling example, CSO fails to find a locally D-optimal design when the model includes all two-factor interaction terms and some three-factor interaction terms. Currently, the best design found by CSO seems to have a D-efficiency of about 82%. Clearly, some further enhancements of the metaheuristic algorithms may be needed. Hybridization to combine one or two more algorithms with CSO may also improve performance.

Summary

Our main contributions in this paper are 1) use PSO and its variants to find optimal designs for high-dimensional and Bayesian models; and 2) the creation of online tools for practitioners to generate different types of tailor made optimal designs for their problems. Web based tools can be very valuable to help practitioners make informed decisions on the study design. For instance, a successful website is the one housed in Houston at https://biostatistics.mdanderson.org/SoftwareDownload/, where an array of software is available for download to find many types of adaptive Bayesian designs for Phase I and II trials. It has more than 17,000 downloads to date indicating a high demand of such tools in practice. Some of our PSO codes for stand alone programs are in MATLAB, where the user can download to make changes, when necessary, for their problems. This website allows practitioners to compare designs and arrive at an informed decision on the choice of the design to implement.

We have applied PSO and its variants to search for Bayesian optimal designs and high-dimensional models. These are challenging tasks as they involve scaling problems and multiple integration over different types of parameter spaces. While such algorithms do not perform integration per se, they can be cleverly hybridized with effective tools for integration purposes to find these hard to find Bayesian optimal designs. Our current work includes hybridizing PSO or its variants with sparse grid algorithm and results are promising.

We close with a cautionary note that an optimal design should not be used religiously but should serve as a guide or benchmark. This is because the optimal design is found under a fixed set of assumptions that may not adequately reflect reality and so may not satisfy the needs of the practitioners. Different optimal designs under various settings should be compared carefully before the design is implemented. The guiding principle is that the implemented design should not stray too far from the optimum as measured by its efficiency relative to the optimum. PSO facilitates search for an efficient design, calculates an efficiency lower bound and compares competing designs. Our hope is that the practitioners are more informed of such algorithms and the availability of them on websites will help them implement more efficient designs.

Acknowledgement
The authors would like to thank Dr. Ray-Bing Chen and Dr. Weichung Wang for their support in maintaining the website.

Funding
The research in this publication were partially supported by the National Institute Of General Medical Sciences of the National Institutes of Health under Award Number R01GM107639. The funding did not influence the design of the study and collection, analysis, and interpretation of data and in writing the manuscript.

Availability of data and materials
Not applicable.

Authors' contributions
YS wrote the manuscript and provided PSO-generated designs, ZZ applied CSO and generated optimal designs for high-dimensional models and WKW supervised and edited the whole manuscript. All authors read and approved the final manuscript.

Competing interests
The authors declare that they have no competing interests.

Publisher's Note

Received: 1 November 2018 Accepted: 18 March 2019
Published online: 08 April 2019

References

Berger, M. P., Wong, W. K.: Applied Optimal Designs. John Wiley & Sons, England (2005)
Broudiscou, A., Leardi, R., Phan-Tan-Luu, R.: Genetic algorithm as a tool for selection of D-optimal design. Chemometr. Intell. Lab. Syst. **35**, 105–116 (1996)
Chaloner, K., Larntz, K.: Optimal bayesian design applied to logistic regression experiments. J. Stat. Plan. Infer. **21**, 191–208 (1989)
Chen, R. B., Hsieh, D. N., Hung, Y., Wang, W.: Optimizing latin hypercube designs by particle swarm. Stat. Comput. **23**(5), 663–676 (2013)
Cheng, R., Jin, Y.: A competitive swarm optimizer for large scale optimization. IEEE Trans. Cybern. **45**(2), 191–204 (2015)
Chernoff, H.: Sequential Analysis and Optimal Design. SIAM, Philadelphia (1972)
Dette, H., Neugebauer, H.-M.: Bayesian d-optimal designs for exponential regression models. J. Stat. Plan. Infer. **60**(2), 331–349 (1997)
Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory, MHS'95. In: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, pp. 39–43. IEEE, Nagoya, (1995)
Fedorov, V.V.: Theory of Optimal Experiments. Elsevier, New York (1972)
Grimshaw, S. D., Collings, B. J., Larsen, W. A., Hurt, C. R.: Eliciting factor importance in a designed experiment. Technometrics. **43**(2), 133–146 (2001)
Han, C., Chaloner, K.: D-and c-optimal designs for exponential regression models used in viral dynamics and other applications. J. Stat. Plan. Infer. **115**(2), 585–601 (2003)
Hassan, R., Cohanim, B., De Weck, O., Venter, G.: A comparison of particle swarm optimization and the genetic algorithm. In: 46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, p. 1897, (2005)
Hung, Y., Wang, W.: Accelerating parallel particle swarm optimization via gpu. Optim. Methods Softw. **27**(1), 33–51 (2012)
Kennedy, J.: Handbook of nature-inspired and innovative computing. Swarm intelligence. Springer, Boston (2006)
Kiefer, J. C., Brown, L., Olkin, I., Sacks, J.: Jack Carl Kiefer Collected Papers: Design of Experiments. Springer, New York (1985)
Lukemire, J., Mandal, A., Wong, W. K.: 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 (2019)
Nelder, J. A., Mead, R.: A simplex method for function minimization. Comput. J. **7**, 308–313 (1965)
Perelson, A. S., Neumann, A. U., Markowitz, M., Leonard, J. M., Ho, D. D.: Hiv-1 dynamics in vivo: virion clearance rate, infected cell life-span, and viral generation time. Science. **271**(5255), 1582 (1996)
Qiu, J.: Finding optimal experimental designs for models in biomedical studies via particle swarm optimization. PhD thesis, UCLA (2014). https://escholarship.org/uc/item/1cj4b854
Royle, J. A.: Exchange algorithms for constructing large spatial designs. J. Stat. Plan. Infer. **100**, 121–134 (2002)
Russell, K. G., Woods, D. C., Lewis, S., Eccleston, J.: D-optimal designs for poisson regression models. Stat. Sin. **19**, 721–730 (2009)
Shi, Y., Eberhart, R. C.: Parameter selection in particle swarm optimization. In: International Conference on Evolutionary Programming, pp. 591–600. Springer, Berlin, (1998)
Shi, Y., Eberhart, R. C.: Empirical study of particle swarm optimization. In: Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406) Vol. 3, pp. 1945–1950. IEEE, Washington, DC, (1999)
Wang, Y., Myers, R. H., Smith, E. P., Ye, K.: D-optimal designs for poisson regression models. J. Stat. Plan. Infer. **136**(8), 2831–2845 (2006)
Wu, H., Ding, A. A.: Population hiv-1 dynamics in vivo: Applicable models and inferential tools for virological data from aids clinical trials. Biometrics. **55**(2), 410–418 (1999)
Wynn, H. P.: Results in the theory and construction of d-optimum experimental design. J. R. Stat. Soc. Ser. B. **34**, 133–147 (1972)
Yang, X. S.: Nature-inspired Metaheuristic Algorithms. Luniver press, United Kingdom (2010)
Yang, Y., Pedersen, J. O.: A comparative study on feature selection in text categorization. In: Proceedings of the Fourteenth International Conference on Machine Learning, pp. 412–420. Morgan Kaufmann Publishers Inc., San Francisco, (1997)