Using Animal Instincts to Design Efficient Biomedical Studies

Jiaheng Qiu[a], Ray-Bing Chen[b], Weichung Wang[c], Weng Kee Wong[d]

[a]*Department of Biostatistics, University of California, Los Angeles, CA 90095, USA*
[b]*Department of Statistics, National Cheng-Kung University, Tainan 70101, Taiwan*
[c]*Department of Mathematics, National Taiwan University, Taipei, Taiwan*
[d]*Department of Biostatistics, University of California, Los Angeles, CA 90095, USA*

Abstract

Particle swarm optimization (PSO) is an increasingly popular metaheurisitc search algorithm for complex optimization problems. Its popularity is due to its repeated successes in finding an optimum or a near optimal solution for problems in many applied disciplines. The algorithm makes no assumption of the function to be optimized and for biomedical experiments like those presented here, PSO typically finds the optimal conditions in a few seconds of CPU time using a garden variety laptop. We apply PSO to find various optimal designs for popular models in real-life experiments and demonstrate its flexibility via a new website.

Keywords: Approximate design, D_s-optimal design, efficiency, metaheuristic algorithms, particle swarm optimization

1. Introduction

Optimal experimental designs are gaining in importance steadily over the last two decades [1]. A main interest is rising cost in conducting experiments and an increasing realization in more applied fields that optimal design ideas can save costs substantially without sacrifice in statistical efficiency. Some real-life examples are given in Dette and Beidermann [2], Dette et al. [3], Woods et al. [4], Lopez-Fidalgo et al. [5] and, Gilmour and Trinca [6], where the applications range from reaction kinetics to medical studies with a time-to-event outcome. Berger and Wong [7] also provide a catalogue of concrete applications of optimal designs that ranges from biomedical, social science applications to construction of groundwater wells in the Los Angeles basin.

In practice, the construction of optimal designs can be problematic. One reason is that many real-life experiments are studied using nonlinear models, which means that the optimal choice of the design depends on both the model and nominal values of the model parameters [8]. Except for the simplest nonlinear models, formulae for optimal designs are rarely possible and when they are available, they are oftentimes too complicated or limiting to be useful for the practitioners. For instance, the formula is invariably derived under a strict set of assumptions that may not apply to the problem at hand. In particular, either they are too simplistic or are made on technical grounds that are unrealistic for practical applications. As a specific example, the dose interval and the nonlinear mean function in a dose response study are always assumed in the construction of an optimal design and it may not be clear at all how the optimal design changes if the dose interval is changed or if the mean function is slightly changed.

There are many algorithms for finding optimal designs and most are based on heuristics or intuition and they do not have a theoretical basis. Only a couple of algorithms can be proven to converge to the optimal designs and prominent ones include Fedorov's and Wynn's algorithms for generating D and c-optimal designs [9, 10]. The former designs are useful for estimating all parameters in the mean function and the latter targets estimation of a specific function of the model parameters by minimizing, respectively, the volume of the confidence ellipsoid and the asymptotic variance of the estimated function of interest. For the few algorithms that can be shown to converge mathematically, problems may still exist and they include (i) they take too long to converge, (ii) they may fail to converge for more complicated setups that they are not designed, such as for nonlinear for mixed effects nonlinear models, and (iii) numerical issues due to rounding problems or the intrinsic nature of sequential process; for example, many algorithms produce clusters of support points as the algorithm proceeds and these clusters require periodic and judicious collapsing into the correct distinct but unknown support points.

In the next section, we propose an exciting, easy and effective algorithm to generate optimal designs for practitioners. This algorithm has been used for almost a dozen of years in the computer science and engineering circles, and exponentially more so in recent years due to its repeated successes in solving an increasing large class of applied problems. The main reasons for its popularity are its flexibility, ease of implementation and utility, and general applicability to solve complex optimization problems (or nearly so) without making specific assumptions on the objective function. Section 3 presents the methodology with a few specific applications to find a variety

of optimal designs for nonlinear models in the biomedical arena. Section 4 concludes with a discussion.

2. Particle Swarm Optimization (PSO)

Nature-inspired algorithms have been gaining popularity and dominance in the last decade both in academia and industrial applications after adjusting for different types of biases [11, 12].One of the most prominent examples of a nature-inspired algorithm is Particle Swarm Optimization (PSO) based on swarm intelligence. It is a metaheuristic algorithm and comes about from the research in fish and swarm movement behavior. PSO is intriguing in that they always seem to be able to quickly solve the optimization problem or provide good quality solutions for many types of complex optimization problems, even though the method itself lacks a firm theoretical justification to date. An attempt to explain its success from a biological viewpoint is given in Garnier, et al. [13] and an overview of PSO is available in Poli et al. [14]. Common characteristics of PSO includes its ability to find the optimal solution to a complex problem or gets close to the optimal solution quickly without requiring any assumption on the function to be optimized. The PSO generic code is easily available on many websites such as `http://www.swarmintelligence.org` and or in books on metaheuristic methods such as Yang [15]. MATLAB also has a toolbox for running the PSO code (`http://www.mathworks.com/matlabcentral/fileexchange/7506`).

PSO begins its search for the optimum with a population of randomly generated candidate particles that cover the search space. In our context, the particles are candidate designs or starting designs. The number of such designs to begin with is the flock size and is user-controlled. Each design or particle has the same number of support points which is again pre-selected by the user; this number is typically chosen to be the number of parameters in the model. At any time point, each particle sequentially adapts its movement toward where it believes the optimum is and does so with an velocity that depends on its current and perceived optimum locations (i.e. the self-perceived best position p_i) as well the location that other particles collectively believe is the optimum (i.e. the global position p_g). These movements are elegantly governed by two key equations with a few parameters that includes random vectors that are responsible for the stochastic movement of the designs, a cognitive learning parameter and a social learning parameter. These latter two parameters are time invariant and not specific to a specific particle. In terms of the animal interpretation as birds flocking

in the sky to seek for food, these parameters measure the self-learning ability and the sharing information ability of the flock to locate the optimum (where the food is on the ground). There is also another inertia parameter that controls the direction of the path it takes next based on knowledge at that particular time point.

Two basic equations that drive movement for the each particle in the PSO algorithm in its search to optimize an objective function $h()$ is as follows. At times t ant $t+1$, the movement of particle i is governed by

$$\mathbf{v}_i^{t+1} = \omega_t \mathbf{v}_i^t + \gamma_1 \boldsymbol{\beta}_1 \odot (\mathbf{p}_i - \mathbf{x}_i^t) + \gamma_2 \boldsymbol{\beta}_2 \odot (\mathbf{p}_g - \mathbf{x}_i^t), \tag{1}$$

and

$$\mathbf{x}_i^{t+1} = \mathbf{x}_i^t + \mathbf{v}_i^{t+1}. \tag{2}$$

Here, $\mathbf{v}_i^t$ is the particle velocity at time t and $\mathbf{x}_i^t$ is the current particle position at time t. The inertia weight ω_t adjusts the influence of the former velocity and can be a constant or a decreasing function with values between 0 and 1. For example, a linearly decreasing function over the specified time range with initial value 0.9 and end value 0.4 [16]. Further, the vector $\mathbf{p}_i$ is the personal best (optimal) position as perceived by the ith particle and the vector $\mathbf{p}_g$ is the global best (optimal) position as perceived by all particles, up to time t. This means that up to time t, the personal best for particle i is $pbest_i = h(\mathbf{p}_i)$ and $gbest = h(\mathbf{p}_g)$. The two random vectors in the PSO algorithm are $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ and their components are usually taken to be independent random variables from $U(0,1)$. The constant γ_1 is the cognitive learning factor and γ_2 is the social learning factor. These two constants determine how each particle moves toward its own personal best position or overall global best position. The default values for these two constants in the PSO codes are $\gamma_1 = \gamma_2 = 2$ and they really seem to work well in practice for nearly all problems that we have investigated so far. Note that in equation (1), the product in the middle two terms is Hadamard product. The pseudo code for the PSO procedure for a flock of size n is as follows.

(1) Initialize particles
  (1.1) Initiate positions $\mathbf{x}_i$ and velocities $\mathbf{v}_i$ for $i = 1, \ldots, n$.
  (1.2) Calculate the fitness values $h(\mathbf{x}_i)$ for $i = 1, \ldots, n$.
  (1.3) Determine the personal best positions $\mathbf{p}_i = \mathbf{x}_i$ and the global position $\mathbf{p}_g$.
(2) Repeat until stopping criteria are satisfied.
  (2.1) Calculate particle velocity according Eq. (1).
  (2.2) Update particle position according Eq. (2).
  (2.3) Calculate the fitness values $h(\mathbf{x}_i)$.
  (2.4) Update personal and global best positions $\mathbf{p}_i$ and $\mathbf{p}_g$.
(3) Output $\mathbf{p}_g = \arg\min h(\mathbf{x})$ with $gbest = h(\mathbf{p}_g)$.

In every iteration, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far. (The fitness value is also stored.) This personal best value is called *pbest*. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population. This best value is a global best and called *gbest*. When a particle takes part of the population as its topological neighbors, the best value is a personal best.

In practice, particles' velocities along each dimension are restricted to a user-specified maximum value of V_{max}. This is to prevent particles searching beyond the design space which sometimes happens because of accelerations due to the stochastic elements in the process. The initial velocity for each particle may be set equal to 0 or be randomly assigned from $U(0, 1)$.

3. Generating Optimal Designs for Biomedical Studies using PSO

In this section, we apply PSO to find various types of optimal designs for common models in the biomedical studies. These models may appear small in terms of number of parameters they have but as noted in Konstantinou el at. [8], among many others, they can still be difficult to find by traditional numerical methods or analytically. Here and throughout, our focus is approximate designs, which are probability measures defined on the given design space [17]. The worth of a design is measured by its Fisher information matrix defined as minus of the expectation of the second derivatives of its log likelihood function with respect to the parameters. Our goal is to minimize an appropriate convex function of the information matrix. Given a fixed sample size, a model and a design criterion, an optimal approximate design minimizes the criterion by choice of (i) the number (k) of points, (ii) where these design points $x_1, \ldots, x_k$ are and (iii) the mass distribution $p_1, \ldots, p_k$ at the design points subject to the constraint that the nonnegative weights p_i's sum to unity. Because each design criterion is convex, we can use results from convex analysis and verify the optimality of an approximate design ξ by examining the plot of its directional derivative over the design space. If the design is not optimal, using results from convex analysis, one can also ascertain how close the design is to the optimum without knowing the latter by means of an efficiency lower bound. Details can be found in standard design monographs such as Berger and Wong [7], Fedorov [9] and Silvey [18].

Our experience is that PSO can also find optimal designs for more complicated models just as efficiently. To date we had used PSO and successfully

found optimal designs for experiments up to 8 factors for a mixture model, nonlinear models up to 6 parameters and also for more involved design criteria, such as a minimax type of optimality criterion. We discuss these "larger" models further in the last section.

The PSO code is initiated as follows. The user first selects the flock size, the number of iterations desired and parameters for the design problem, such as the limits of the design space and the nominal values of the model parameters for the particular problem at hand. The rest of the PSO parameters like the inertia, cognitive and social learning parameters are all set to their default values. Consequently, we only have to fuss with the flock size and the number of iterations. This simplifies the process and is an appealing feature of PSO. The search for the optimal design by PSO begins with a randomly generated flock of size specified by the user. These designs all have the same fixed number of points, k equal to the number of parameters in the mean response function. When the algorithm is run, the k-point optimal design is usually found very quickly. When the design optimality criterion is convex, which is true for all our examples, the generated design can be verified using an equivalence theorem derived from the directional derivative.

If the above strategy fails to produce the optimal design, meaning that repeated searches by PSO with different number of iterations and different flock sizes produced a design that did not meet the equivalence theorem conditions, we continue the search to all designs with $k + 1$ points. Our experience is that such a strategy always produce a locally optimal design and we only need to search among designs with $k + j$ points where j is usually 1 or 2. For Bayesian optimal designs, j can be much larger than 2 especially if the prior distribution is diffuse. On the other hand, if we have over-specified the number of support points required by the optimal design, then PSO will report an optimal design with some identical points or some points with extremely small weights. The masses at these identical points are then summed to obtain the optimal design. Such situations arise when the optimal design has a singular information matrix and an example of such a case is provided below when we want to find the locally optimal design for estimating the area under curve in the 3-parameter compartmental model.

We have set up mirror websites at 3 places with examples discussed in this paper so that the reader can download the P-codes, verify our results and appreciate how PSO works in practice. One site is housed at UCLA at `http://optimal-design.biostat.ucla.edu/podpack/` and the other two sites are housed in Taiwan at `http://www.math.ntu.edu.tw/~optdesign` and `http://www.stat.ncku.edu.tw/optdesign`. Many of the codes can be

readily changed to find another type of optimal design for he same model or a different model. Typically, the only changes that are required are in the information matrix and the design criterion while leaving much of the rest of the code intact. For example, it took the first author about 15 minutes to re-code the PSO code for finding the c-optimal design for estimating the area under the curve in the compartmental model to one for estimating the two model parameters in the survival model, excluding of time to read the paper by Konstantinou el at. [8]. Our website has a swarm movement plot that shows whether the particles visually converge to a single location. However this plot sometimes can be misleading because of the default values we simply employed for the other parameters in the PSO algorithm. To obtain a more accurate picture of the swarm movement and its convergence, other PSO parameters will have to be properly fine-tuned. However, we do not think it is worth the trouble as the plot is only a visual guide whether the PSO-generated design is truly optimal or not. A more definitive way to confirm optimality is via an equivalence theorem by examining properties of the directional derivative of the convex functional evaluated at the PSO-generated design. For this purpose, we also provide an 'equivalence' plot to verify optimality of the PSO-generated design.

The instructions for downloading the MATLAB P-codes are available on our website. After the specific code for a particular problem is downloaded into the reader's computer and activated, the software provides a window for the reader to input the necessary design parameters for the problem. These typically include the design interval and the nominal values for the model parameters. For the PSO operation, at least two inputs are needed: the flock size and the number of iterations. Sometimes, for more complicated problems such as minimax design problems, additional PSO parameters will be required to be specified but for the most part, we minimize the need to change the default values that come with the PSO codes. Upon hitting the button 'run', PSO finds the optimal design iteratively and stops when the maximum number of iterations allowed is reached. Typically, one will notice PSO gets to the vicinity of the optimum quickly after a few iterations and spends the rest of the time converging to it and making sure the support points and weights all agree to 5 or 6 decimal places. By hitting the button 'Swarm plot' on the interface window we observe the swarm in action and whether it converges or not. Note that this plot is usually a 2-dimensional plot and sometimes a 3-dimensional plot, so not all support points of the PSO-generated design are displayed in the swarm plot. There is an additional button that says 'Equivalence Plot', and that shows the plot of the directional derivative of the PSO-generated design. If PSO finds the opti-

mum, the shape of the graph should exhibit properties of an optimal design discussed in the context of the equivalence theorem in Section 1. In each case, the CPU time required to generate the design is also reported in the interface window. We invite the reader to download the P-code into their own computer and use their MATLAB software to view demonstrations on how PSO works for some of these applications and verify the results claimed in the next section. Figures generated from the interface can be saved as eps files. Because of the stochastic nature of the algorithm, the numerical results from the MATLAB P codes and the reported CPU times in this paper will not be identical. All CPU times reported in the paper were obtained using a laptop equipped with an Intel i5 2520M, 2.5GHz. 8GB RAM, Windows 7 professional operating system and Matlab version 2010b.

In what follows, we present different types of optimal designs found by PSO for the following model: (i) a 3-parameter compartmental models used in pharmacokinetics, (ii) 2 and 3-parameter logistic models for studying binary responses, (iii) a double exponential model for studying tumor regrowth rate and (iv) a 2-parameter survival model. These models are selected to facilitate comparisons with known reported results from the literature. The design criteria are D or D_s or c-optimality but as we discuss in the last section, we have also applied PSO successfully to generate optimal designs under more complicated design criteria and optimization problems that involve a hundred dimensions or more. We provide a bit more technical details and discuss impact of choices of flock size and number of iterations for the first example to show some characteristics of PSO. Similar pattern of observations also apply to the other examples.

3.1. Locally optimal designs for compartmental models

Compartmental models are commonly used to model drug movement through the body, and is represented as

$$\eta(x, \theta) = \theta_3\{\exp(-\theta_2 x) - \exp(-\theta_1 x)\},\ x > 0.$$

They are not limited to pharmaceutical applications; for instance, the 3-parameter model described below was also used by Fresen [19] in veterinary science to model the effect of theophylline on horses and the 2-parameter model is used to study homogeneous chemical reactions [20, 21]. For space consideration, we focus on the 3-parameter model in this paper.

Let $\theta = (\theta_1, \theta_2, \theta_3)^T$ be the vector of model parameters for the 3-parameter compartmental model and let $\theta^0 = (\theta_1^0, \theta_2^0, \theta_3^0)^T$ be the nominal values for θ

for the compartmental model given by

$$\eta(x,\theta) \approx \eta(x,\theta_0) + [\frac{\partial \eta(x,\theta)}{\partial \theta}|_{\theta_0}]^T(\theta - \theta_0)$$

with $\theta_2 > \theta_1 > 0$ and $\theta_3 > 0$. A direct calculation shows the gradient of the approximated mean function at the point x is

$$f^T(x,\theta^0) = (\frac{\partial \eta(x,\theta)}{\partial \theta_1}, \frac{\partial \eta(x,\theta)}{\partial \theta_2}, \frac{\partial \eta(x,\theta)}{\partial \theta_3})|_{\theta^0},$$

with

$$\begin{aligned}
\frac{\partial \eta(x,\theta)}{\partial \theta_1} &= x\theta_3 exp(-\theta_1 x) \\
\frac{\partial \eta(x,\theta)}{\partial \theta_2} &= -x\theta_3 exp(-\theta_2 x) \\
\frac{\partial \eta(x,\theta)}{\partial \theta_3} &= exp(-\theta_2 x) - exp(-\theta_1 x).
\end{aligned}$$

When the sample size is fixed and the design ξ takes independent observations at $x_1, \ldots, x_n$, the total information matrix is proportional to $M(\xi) = \sum_i f(x_i,\theta^0) f^T(x_i,\theta^0)$. If there are replications, we use weights $p_1, \ldots, p_k$ to represent the proportions of observations at these points and the information matrix becomes $\sum_i p_i f(x_i,\theta^0) f^T(x_i,\theta^0)$, apart from an unimportant multiplicative constant. For D-optimality, the objective function is to maximize $\log(det(M(\xi))$, or equivalently, minimize the convex functional -log($det(M(\xi))$) by choice of a design. Here and elsewhere, we have for simplicity, suppressed the dependence of the information matrix on the vector of model parameters.

To implement PSO to find locally optimal design for the 3-parameter compartment model, we assumed for comparison purposes, we have the same dose interval and same set of nominal parameters used in Atkinson, Donev and Tobias's book [22] with $\theta_1^0 = 0.05884$, $\theta_2^0 = 4.298$ and $\theta_3^0 = 21.8$. These values were obtained from the least square estimates of the parameters from Fresen's dataset [19].

As a first attempt, we used 100 iterations and a flock size of 100 particles each with 3 support points in the PSO algorithm to find the locally D-optimal design for this compartmental model. This is a 5 dimension optimization problem to find the best choices of x_1, x_2, x_3, p_1, p_2, where $0 < x_1 < x_2 < x_3 < 30$, $0 < p_1 < 1, 0 < p_2 < 1$,$0 < p_3 = 1 - p_1 - p_2$.

The PSO-generated design was equally supported at 0.2288, 1.3886 and 18.4168 and it coincides with the locally D-optimal design in Atkinson, Donev and Tobias's book [22] on page 264. The equivalence plot confirms the optimality of the design over all designs on the designated dose interval and we do not need to search further. The CPU time was 0.408 seconds and if we had used just 50 particles and 50 iterations, we would also have obtained the same design in 0.381 seconds. In both cases, the swarm plot almost converged and the 'equivalence' plot confirms optimality. Interestingly, if had used 8 particles and 50 iterations, the PSO took 0.308 seconds to generate a design supported at 0.2305, 1.4197 and 18.8277 with mass distribution at these points equal to 0.3687, 0.3109 and 0.3204 respectively. The swarm converged but the plot shows the PSO-generated design is not optimal, suggesting that more particles are required and the swarm plot can be misleading. Even with 1000 iterations and 8 particles, the optimal design is still not found. A reason for this is that there was simply not enough starting designs (particles) to cover the design space adequately to bring about an effective search.

Three common characteristics of a drug are time to maximum concentration in the target compartment, maximum concentration and the average time it spends inside the compartment. We discussed only the latter two objective for space consideration because finding optimal design for the first objective can be carried out in a similar way. The locally c-optimal design for estimating the time to max concentration of a drug can be found by integration directly from the model. This time as a function of the model parameters is

$$g(\theta) = \frac{\log\theta_1 - \log\theta_2}{\theta_1 - \theta_2}.$$

Our goal then is to choose a design to minimize the asymptotic variance of the estimated time given by

$$[\frac{\partial g(x,\theta)}{\partial \theta^T}]M(\xi)^{-}\frac{\partial g(x,\theta)}{\partial \theta}.$$

Here $\frac{\partial g(x,\theta)}{\partial \theta^T} = (a/\theta_1 - b)/a^2, (b - a/\theta_2)/a^2, 0)$ where $a = \theta_1 - \theta_2$, $b = log\ \theta_1 - log\ \theta_2$ and $M(\xi)^{-}$ is a generalized inverse of the information matrix. Using the same set of nominal values of the parameters, we use PSO to minimize the variance of $g(\hat{\theta})$ in a similar way as before. In our case here, we started with two points implying that we sought to minimize by choice of (x_1, x_2, p_1), $0 < x_1 < x_2 < 30$, $0 < p_1 < 1$, $p_2 = 1 - p_1$.

The locally optimal design for estimating the time to maximum concentration is a c-optimal design with only 2 support points, which means that

its information matrix is singular because the matrix is now a sum of two rank-one matrices. To get around having to find the inverse of singular information matrix $M(\xi)$, we followed convention and added a small multiple to the identity matrix and worked with the invertible matrix

$$M_\epsilon(\xi) = M(\xi) + \epsilon I_3.$$

We implemented PSO to find the optimal values of x_1, x_2, p_1 subject to $0 < x_1 < x_2 < 10$ and $0 < p_1 < 1$ with $p_2 = 1 - p_1$. The PSO parameters we used were $\epsilon = 10^{-6}$ and 200 particles all with $k = 2$ points. Expecting a singular information matrix for the optimal design, we allowed for a larger number of iterations and after 1000 iterations, PSO generated a two-point design supported at 0.1793 and 3.5658 with weight 0.3938 at the latter point. This optimal design also agrees with the c-optimal design reported in Atkinson, Donev and Tobias's book [22] on page 264.

A similar procedure is used to find the optimal design for estimating the time the drug spends inside the compartment. A direct calculation shows this function is simply $1/\theta_1 - 1/\theta_2$. Proceeding as before, we applied PSO to minimize the asymptotic variance of the estimated AUC. Setting $k = 2$ and using 100 particles and 1000 iterations, PSO took 6.185 seconds to generate a two-point design supported at 0.2326 and 17.6339 with mass 0.0135 at the smaller point. This design also agreed with the optimal design in Atkinson, Donev and Tobias's book [22] on page 264.

It is interesting to observe what happens if we had used starting designs all with $k = 3$ points. the PSO-generated design obtained using 200 particles and 500 iterations was supported at 0.2337, 17.6269 and 17.7176 with mass distribution at these points equal to 0.0135, 0.8983 and 0.0882. Increasing the number of iterations to 1000 results in a design support at 0.2332, 17.6336 and 17.6626 with mass distribution at these points equal to 0.0135, 0.9535 and 0.03296. These results are consistent with the expectation that a longer iteration and/or more particles usually produces a higher quality solution. It also shows a very nice feature of PSO in that when we over-specify the number of support points the optimal design has, PSO can also automatically find the optimal design directly; in the above case, the 3 points found above get increasingly closer to 2 points as more iterations are used, leaving the mass at the smaller point unchanged.

3.2. Locally Optimal Designs for the simple and quadratic logistic models

For modeling binary responses, logistic models are among the most popular because of their simplicity and ease of interpretation. Frequently, we

have simple logistic models with two parameters and sometimes quadratic logistic models with three parameters. We consider locally D and D_s-optimal designs for estimating all or a user-selected subset of the model parameters. Probably the first description of the locally D-optimal design for the simple logistic model was given in a doctoral thesis [23] for the prototype interval $[-1, 1]$ and reported in Silvey [18]. The formula is complicated for a relatively simple model. When we ran PSO to compare results, we were unable to verify Ford's results. A corrected formula for the locally D-optimal design on an arbitrary interval was given in Sebastiani and Settimi [24] and we were able to verify the results using the MATLAB P-code available on the website.

Quadratic logistic models are sometimes employed to explore possible curvature in the model or for estimating an interesting characteristic of an agent in a dose response study. For example, in radiology and radiotherapy, there is often interest in estimating the ratio of the coefficients associated with the linear and quadratic terms in the quadratic logistic model [25] Selected locally c-optimal designs for the quadratic logistic model were found theoretically in Fornius and Nyquist [26]after re-parameterizing the model in the following way using their notation:

$$log\ \frac{Ey}{1-Ey} = \alpha + \beta(x-\mu)^2.$$

Here y is the binary response taking on values 0 or 1 with certain probabilities at dose x. We provide on our website PSO codes for generating locally D-optimal design on an arbitrary interval and as usual, we began our search for the locally D-optimal design among all 3- point designs first. As an example, suppose the design interval is $[-3, 1]$ and the nominal values for the 3 parameters are $\alpha = 2$, $\beta = 3$ and $\mu = 0$. With a flock size of 128 and the number of iterations set at 150, the PSO-generated design took 6.623 seconds to find a design equally supported at $-0.7270, 0$ and 0.7270. The directional derivative plot confirms the D-optimality of this design. If fewer number of iterations were used, say 50 iterations, the pattern of the optimal design will also emerge quickly and clearly, except that the weights are only approximately equal and the extreme support points are less symmetric about 0. PSO took 3.104 seconds to produce the design and also reports the design has an efficiency of 99.98%. Interestingly, when the maximum probability of response is high, there are 4-point locally D-optimal designs. For instance suppose the design interval is $[-1, 1]$ and the nominal values for the 3 parameters are $\alpha = 3$, $\beta = -5$ and $\mu = 0$. With a flock size of 256 and the number of iterations set at 200, the PSO-generated design took 18.317

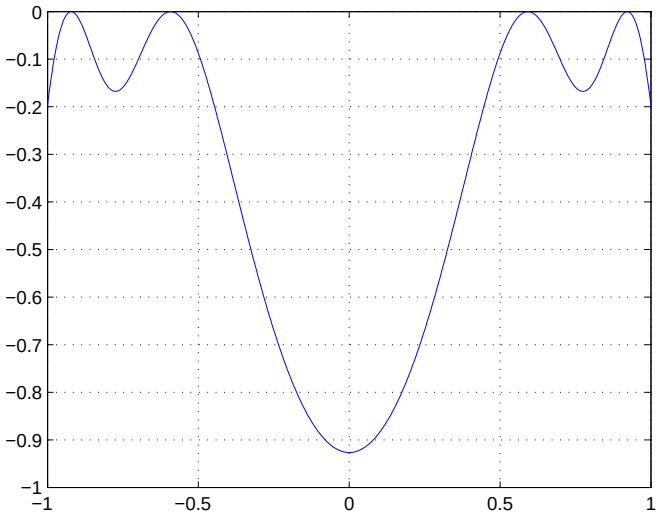

Figure 1: The plot of the directional derivative of the D-optimal criterion for the PSO-generated 4-point design for the quadratic logistic model when $(\alpha^0, \beta^0, \mu^0) = (3, -5, 0)$.

seconds to find a symmetric design supported at $-0.9217, -0.5921, 0.5921$ and 0.92170. The weights at -0.9217 and at -0.5921 are 0.2966 and 0.2034, respectively. As always here and elsewhere, in order to ensure a higher chance that PSO will generate the optimal design, we report flock size and number of iterations larger than are usually necessary. Frequently, smaller flock size and smaller number of iterations will suffice which mean shorter CPU time can usually also produce the optimal design. Figure 1 is obtained from the P-code and it shows the plot of the directional derivative of the D-optimal criterion for this PSO-generated 4-point design for the quadratic logistic model. The plot is bounded above by 0 throughout the design interval with equality at the support points of the PSO-generated design and so the figure confirms its D-optimality.

Oftentimes in practice, certain parameters are more important or more biologically meaningful than others. For example, in the Michaelis-Menten model, the Michaelis-Menten constant is clearly more interesting than the other parameter and so more resources should be used for estimating the more interesting parameter. Optimal designs for estimating selected model parameters are called D_s-optimal designs and they are usually more in-

volved than D-optimality [20, 27] One partitions the information matrix appropriately and minimizes only the determinant of the covariance matrix corresponding to the selected parameters of interest. For example, suppose we are interested to make inference on the last two parameters in the above compartmental model, i.e. θ_2 and θ_3. The optimal design that provides the best inference for this purpose is the design ξ^* that satisfies

$$\xi^* = \arg\min_{\xi} |M_{22}(\xi) - M_{21}(\xi)M_{11}^{+}(\xi)M_{12}(\xi)| = \arg\max_{\xi} |M(\xi)/|M_{11}(\xi)|.$$

Here $M_{11}(\xi)$ is the top left part or the $(1,1)$ element of the appropriately block partitioned 2×2 matrix M, $M_{11}^{+}(\xi)$ is the Moore-Penrose inverse of the matrix $M_{11}(\xi)$ with similar interpretations for the submatrices $M_{21}(\xi)$ and $M_{12}(xi)$. The resulting design ξ^* is called a D_s-optimal design with the subscript s standing for subset (of the model parameters).

We implemented PSO codes in a few lines of codes for finding locally D_s-optimal design on the website. For example, the PSO-generated D_s-optimal design for estimating both β and μ only was found to be unequally supported at $1, 1.2126$ and 1.5725 with mass distribution at these points equal to $0.1995, 0.3166$ and 0.4839. The dose interval was arbitrarily set to $[1, 4]$, the nominal values for the 3 parameters were $\alpha = -2$, $\beta = 4$ and $\mu = 0$ and PSO took 3.1 seconds to locate the above design among all 3-point designs using a flock size of 64 and 100 iterations. This design has at least 99.99% efficiency even though the directional derivative plot suggests optimality. If we had used only 25 iterations with everything else the same as before, the PSO-generated design found in 1.310 seconds still had an efficiency at least equal to 87.41%. In this case, of course the directional derivative plot clearly shows that the generated design is not optimal. Additional locally D-optimal designs and locally and D_s-optimal designs for estimating the two parameters β and μ in the quadratic logistic regression model are given in Tables 1 and 2.

There are additional interesting design questions for the quadratic model that a c-optimal can be useful. Quadratic logistic models are sometimes employed to explore possible curvature in the model or for estimating an interesting characteristic of an agent in a dose response study. In the former case estimating the coefficient associated with the quadratic term provides an indication of curvature presence in the the model. An example of an interesting quantity to estimate in radiology and radiotherapy is the ratio of the coefficients associated with the linear and quadratic terms in the quadratic logistic model [25]. PSO codes can also be implemented directly to find these c-optimal designs.

3.3. Locally optimal designs for a double exponential model

Double-exponential regrowth model was developed by Demidenko [**?**] to describe the dynamics of post-irradiated tumors based on the two-compartment model. One may categorize tumor cells as proliferating or quiescent and under appropriate assumptions, the natural logarithm of the tumor volume of the two kinds of cells may be expressed as

$$y_i = \alpha + \ln[\beta e^{\nu t_i} + (1-\beta)e^{-\phi t_i}] + \varepsilon_i,$$

where $\varepsilon_i \sim N(0, \sigma^2)$. After linearizing the model by a Taylor's first order expansion of the mean function, the gradient vector is

$$f(t,\theta) = (1, \frac{e^{\nu t} - e^{\phi t}}{\beta e^{\nu t} + (1-\beta)e^{-\phi t}}, \frac{\beta t e^{\nu t}}{\beta e^{\nu t} + (1-\beta)e^{-\phi t}}, \frac{-(1-\beta)te^{-\phi t}}{\beta e^{\nu t} + (1-\beta)e^{-\phi t}})^T,$$

where $\theta = (\alpha, \beta, \nu, \phi)^T$ is the vector of model parameters. To find the locally D-optimal design, we assume θ^0 is the vector of nominal values for θ and maximize $\det(M(\xi))$.

Li and Balakrishnan [28] had shown that $\det(M(\xi)$ depends only on β^0 and $\nu^0 + \phi^0$, which implies that only two nominal values are required to generate the locally D-optimal design, i.e. a nominal value for β and a nominal value for the sum of the two parameters ν and ϕ. Proceeding as before, PSO readily generated locally D-optimal designs that matched those in Li and Balakrishnan [28]. Some of these locally D-optimal designs for selected nominal parameter settings are given below. PSO codes for generating the locally D-optimal designs can be downloaded from the cited website and the reader can run and verify directly the above claimed results. For instance, suppose we used the default values in the code, namely set $\beta = \nu + \phi = 0.2$ and the interval is $[0, 10]$. Then the PSO code with 100 particles and 100 iterations produced in 1.041 seconds the locally D-optimal design equally supported at $0, 2.660, 6.707$ and 10 reported on the first line in Table 3 in Li and Balakrishnan [28]. Likewise, the same flock size and same number of iterations will also produce in 1.085 seconds the optimal design in the last row of their Table where $\beta = 0.8$ and $\nu + \phi = 1$. Other cases including the case for verifying the c-optimal designs reported in their paper can be similarly verified.

3.4. Locally optimal designs for a survival model

Konstantinou et al. [8] investigated a two-parameter exponential model with type I right censored data, where all individuals entered the study at the same time and stayed until a user-specified time c or until failure, whichever

was earlier. Right-censoring occurs when survival times are greater than c. Let $t_1, ..., t_n$ be the observed values for n subjects and $x_i \in \chi$ be the experimental condition at which the ith observation was taken. The design space could be a dose interval or the set $\{0, 1\}$ representing two treatment conditions, say treated or not. The maximum time for observing outcomes in the study is $c = 30$ so that all observations are right censored if the outcome is not observed by that time. They used an exponential regression model with probability density function $f(t_i)$ and a survival function $S(t_i)$ given respectively by

$$f(t_i) = e^{\alpha+\beta x_i} \exp(-t_i e^{\alpha+\beta x_i})$$

and

$$S(t_i) = \exp(-t_i e^{\alpha+\beta x_i}).$$

Without loss of generality, we assumed that the first k observations were failure times and rest $n-k$ observations were right censored. A direct calculation shows that when all observations are independent, the log-likelihood is

$$l(\alpha, \beta, x_1, ..., x_n) = \log\{\prod_{i=1}^{k} f(t_i) \prod_{i=k+1}^{n} S(t_i)\} = \sum_{i=1}^{k}(\alpha+\beta x_i) - \sum_{i=k+1}^{n} t_i \exp(\alpha+\beta x_i).$$

It follows that the information matrix using design ξ is

$$M(\xi, \alpha, \beta) = \sum_{1}^{n} p_i(1 - \exp(-ce^{\alpha+\beta x_i})) \begin{bmatrix} 1 & x_i \\ x_i & x_i^2 \end{bmatrix}.$$

The information matrix $M(\xi)$ depends on unknown parameters α and β because the model is nonlinear. Four sets of nominal value were used to find the locally D-optimal design, which is always equally supported at two points. Using 20 particles and 50 iterations, PSO was able to find the locally D-optimal designs as claimed in their paper.

In practice, the parameter β in the model always has a clear biological interpretations and so it is often of interest. If we have a dose response study, β measures the effect of increasing dose on the response and if we have two treatment conditions, β represents the effect on the hazard of death when say the new treatment is compared with the placebo condition. An appropriate design to use here for estimating β is the locally c-optimal design that gives the smallest asymptotic variance of the estimate. This design is the same as the D_s-optimal design for minimizing the criterion

$$\begin{bmatrix} 0 & 1 \end{bmatrix} M^{-}(\xi) \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

For this two-parameter model, it can be shown that the c-optimal design is always supported at 0 and 1 but with unequal weights that depend on the nominal values. Using 20 particles and 50 iterations, PSO was able to find and verify all the c-optimal designs for the four sets of nominal values considered in Konstantinou et al. [8] and these weights are shown in Table 3.

4. Discussion

We discussed using PSO to find locally D, D_s and c-optimal designs for compartmental models, logistic models, a double exponential model useful for monitoring tumor regrowth and estimating parameters in a survival model. The computational experience we had with these and many other problems we had looked at were similar to what is reported in the literature. First, many parameters in the PSO did not seem to matter much; following convention, we used default parameters in the PSO algorithm in all our examples. Interestingly, our experience also supports what is reported in the literature that setting $\gamma_1 = \gamma_2 = 2$ seemed to be the most efficient choice; other values tended to take a longer time for the swarm to converge, if it did at all. The only two parameters that we changed from problem to problem were number of iterations and the flock size. For optimal designs with a singular information matrix or with Bayesian optimal designs (not reported here), a larger number of iterations and a larger flock size are usually required for convergence.

In general, PSO generates the optimal designs very quickly for all our examples in this paper and elsewhere compared with our experiences with other algorithms. The CPU time for generating each optimal design each time for the same setting varies slightly because of the stochastic nature of the algorithm. Convergence for the PSO was defined as when the design criterion value does not change by more than 10^{-7} deviation from the known optimum value. Table 4 below reports the CPU times for PSO to generate the various optimal designs averaged over 30 replications, along with their standard deviations.

From Table 4, we observe that on average, less than 0.3 second in CPU time was required to find the locally D-optimal designs and a longer time is required to generate c-optimal designs with a singular information matrix. In the latter case, CPU time required is generally still short and only required less than 10 seconds to find the locally c-optimal designs for estimating time to maximum concentration or the area under the curve in the 3-parameter compartmental model. Not surprisingly, a larger number of particles to

cover a larger area of search space to begin with will require a longer time to find the optimal designs, but the increase in time is quite negligible for all our examples. We note that the standard deviations of the CPU times for PSO to generate the design with 20 particles are usually larger than those when a large number of particles is used. This is because the smaller number of starting designs (particles) were not enough to cover the design space adequately and in a consistent manner, and so more variability in the search capability.

PSO differs from current algorithms for finding optimal designs in a number of ways. First, unlike the setup for the Fedorov-Wynn's types of algorithms for finding optimal designs [9, 10], the design criterion does not need to be convex and differentiable for PSO to work. PSO requires no assumption on the function to be optimized. Second, PSO uses many random particles from the start to search for the optimum and these particles communicate with each other; traditional algorithms typically start with a single design and improves upon it sequentially to locate the optimum. PSO is therefore appealing because it uses many starting designs (particles) at the initial stage to cover the search space and so one can expect such an approach is preferable to the traditional approach that uses only a single starting design to find the optimal design. Third, the traditional algorithms usually produce several clusters of support points because at each iteration one point is added to the current design sequentially and over time one has to collapse these clusters of points judiciously to a few points that supposedly are the support points of the optimal design. PSO finds the optimal design neatly; very often it identifies a candidate optimal design after a few iterations and subsequent iterations only seek to ensure that all the points and weights of the candidate optimal design agree up to 4 or 5 decimal places.

Our experience is that in terms of time, PSO typically identifies the optimum usually in a few seconds of CPU time for most of the problems we had worked with. They include different types of optimal designs for nonlinear models with up to 5 parameters or models with rational polynomials as mean functions. Additionally, we have applied PSO to solve minimax design problems which are notoriously difficult to solve because they involve two layers of optimization over two spaces. If one of these spaces is discrete, as in E-optimal design problems or minimax single-parameter optimal design problems, time required to determine the optimal design is significantly shorter than when the two spaces are non-discrete, as in finding a design to minimize the maximum variance of the predicted response over a compact design interval. For such minimax problems, where optimization is sought

over two non-discrete compact spaces, we are not aware of any traditional algorithm that converge to the minimax optimal design even for linear models. Our boldest attempt to date was to test PSO search ability for finding the D-optimal design for the Scheffe's quadratic mixture model with 10 factors. This optimization problem involved optimizing hundreds of variables and we were pleased that PSO was able to get very close to the optimal design and found a design with a D-efficiency of 99.98%.

In conclusion, particle swarm optimization techniques seem like a very powerful, interesting but under-utilized tool for solving optimization problems in the pharmaceutical industry and more so in general statistical research work. We have shown here that PSO is an efficient and flexible method for finding optimal experimental designs for several biomedical studies, but clearly the applications are not restricted to biomedicine. A further strong point for PSO is that, it being a metaheuristic algorithm, it does not respect the technical requirements imposed on the problem to obtain the optimal designs. For example, Konstantinou et al. [8] and Li and Balakrishnan [28] assumed technical conditions to arrive at the theoretical descriptions of the optimal designs. PSO does not incorporate the technical conditions in its search, suggesting that PSO can generate optimal designs for a wider class of problems, including optimal design problems in this journal [29, 30, 31, 32].

We close with a note that the only role the convexity assumption in our design criterion plays is that it enables us to definitively confirm the quality of the generated design. Our repeated successes with PSO this far have now encouraged us to further apply PSO to find optimal designs under non-convex criteria, where there is no definitive and general way to check whether the generated design is optimal. Some examples are the design criteria for finding exact optimal designs, minimum bias designs and designs that minimize the mean squared error. We hope to report further results in the near future.

Acknowledgements

The research of Chen was partially supported by the National Science Council under grant NSC 101-2118-M-006-002-MY2 and the Mathematics Division of the National Center for Theoretical Sciences (South) in Taiwan. The research of Wang was partially supported by the National Science Council, the Taida Institute of Mathematical Sciences, and the National Center for Theoretical Sciences (Taipei Office). The idea for this manuscript originated at the Isaac Newton Institute for Mathematical Sciences at Cambridge, England when Wong was a visiting fellow there and a scientific adviser for a six month workshop in the design and analysis of experiment at

the Institute. He would like to thank the Institute for the support during his repeated visits there in the second half of 2011.

References

[1] Atkinson AC. The usefulness of optimum experimental designs. *Journal of Royal Statistical Society B* 1996;**58**:59-76.

[2] Dette H, Beidermann S. Robust and Efficient Designs for the Michaelis-Menten Model. *Journal of the American Statistical Association* 2003;**98**:679-686.

[3] Dette H, Melas VB, Wong WK. Optimal designs for goodness of fit of the Michaelis-Menten enzyme kinetic function. *Journal of the American Statistical Association* 2005;**100**:1370-1381.

[4] Woods DC, Lewis SM, Eccleston JA, Russell KG. Designs for generalized linear models with several variables and model uncertainty. *Technometrics* 2006;**48**:284-292.

[5] Lopez-Fidalgo Rivas-Lopez J, Del Campo R. Optimal designs for Cox regression. *Statistica Neerlandica* 2009;**63**:135-148.

[6] Gilmour SG, Trinca LA. Bayesian L-optimal exact design of experiments for biological kinetic models. *Applied Statistics* 2011;**61**:237-251.

[7] Berger MPF, Wong WK. *An Introduction to Optimal Designs for Social and Biomedical Research*. John Wiley & Sons: Chichester, 2009.

[8] Konstantinou M, Biedermann S, Kimber A. Optimal designs for two-parameter nonlinear models with application to survival models. *Southampton Statistical Research Institute, University of Southampton*, 2011.

[9] Fedorov VV. *Theory of Optimal Experiments*. Academic Press: New York, 1972.

[10] Wynn HP. Results in the theory and construction of D-optimum experimental designs. *Journal of Royal Statistical Society B* 1972;**34**:133-147.

[11] Whitacre JM. Recent trends indicate rapid growth of nature-inspired optimization in academia and industry. *Computing* 2011a; **93**:121-133.

[12] Whitacre JM. Survival of the flexible: explaining the recent dominance of nature-inspired optimization within a rapidly evolving world. *Computing* 2011b; **93**:135-146.

[13] Garnier S, Gautrais J, Thraulaz G. The biological principles of swarm intelligence. *Swarm Intell* 2007;**1**:3-31.

[14] Poli R, Kennedy J, Blackwell T. Particle swarm optimization: an overview. *Swarm Intell* 2007;**1**:33-57.

[15] Yang XS *Nature-Inspired Metaheuristic Algorithms. 2nd edition*. Frome: Luniver Press, 2010.

[16] Eberhart RC, Shi Y. Comparing inertia weights and constriction factors in particle swarm optimization. *Proceedings of the IEEE Congress Evolutionary Computation, 2000* 2000;**1**:84-88.

[17] Kiefer J. *Jack Carl Kiefer Collected Papers III: Design of Experiments*. Brown LD, Olkin I, Sacks J, Wynn HP. (eds). Springer-Verlag, 1980.

[18] Silvey SD. *Optimal Design*. Chapman and Hall: London, 1980.

[19] Fresen J. Aspects of bioavailiablity studies. M. Sc. Thesis. Department of Mathematical Statistics. University of Cape Town, 1984.

[20] Hill WJ, Hunter WG. Design of experiments for subsets of parameters. *Technometrics* 1974;**16**:425-434.

[21] Hamilton DC, Watts DG. A quadratic design criterion for precise estimation in nonlinear regression models. *Technometrics* 1985;**27**:241-250.

[22] Atkinson AC, Donev AN, Tobias RD. *Optimum Experimental Designs, with SAS*. Oxford University Press: USA, 2007.

[23] Ford I. *Ian Ford PhD Thesis*. University of Glasgow. Scotland, 1976.

[24] Sebastiani P, Settimi R. A note on D-optimal designs for a logistic regression model. *Journal of Statistical Planning and Inference* 1997;**59**:359-368.

[25] Taylor JMG. The design of in vivo multifraction experiments to estimate the $\alpha - \beta$ ratio. *Radiation Research* 1990;**121**:91-97.

[26] Fornius EF, Nyquist H. Using the canonical design space to obtain c-optimal designs for the quadratic logistic model. *Comm. in Statistics-Theory and Methods* 2010;**39**:144-157.

[27] O'Brien TE. Designing for parameter subsets in Gaussian nonlinear regression models. *Journal of Data Scince* 2005;**3**:179-197.

[28] Li G, Balakrishnan N. Optimal designs for tumor regrowth models. *Journal of Statistical Planning and Inference* 2011;**141**:644-654.

[29] Marschner IC. Optimal design of clinical trials comparing several treatments with a control. *Pharmaceutical Statistics* 2007;**6**: 33-33.

[30] Ogungbenro K, Dokoumetzidis A and Aarons L. Application of optimal design methodologies in clinical pharmacology experiments. *Pharmaceutical Statistics* 2009;**8**: 239-252.

[31] Vajjah R and Duffull SB. A generalisation of T-optimality for discriminating between competing models with an application to pharmacokinetic studies. *Pharmaceutical Statistics* 2012;**11**: 503-510.

[32] Biswas A and Lopez-Fidalgo J. Compound designs for dose-finding in the presence of nondesignable covariates. *Pharmaceutical Statistics* 2013;**12**: 92-101.

Table 1: Locally D-optimal designs for estimating the the three parameters in the quadratic logistic model for different nominal values and different design intervals.

$(\alpha^0, \beta^0, \mu^0)$	Design space	Locally D-optimal designs
$(0, -1, 0)$	$[-1, 1]$	$\begin{pmatrix} -1.0000 & 0.0000 & 1.0000 \\ 0.3333 & 0.3333 & 0.3333 \end{pmatrix}$
$(0, -1, 0)$	$[-2, 2]$	$\begin{pmatrix} -1.4073 & 0.0000 & 1.4073 \\ 0.3333 & 0.3333 & 0.3333 \end{pmatrix}$
$(3, -1, 0)$	$[-1, 1]$	$\begin{pmatrix} -1.0000 & 0.0000 & 1.0000 \\ 0.3333 & 0.3333 & 0.3333 \end{pmatrix}$
$(3, -1, 0)$	$[-2, 2]$	$\begin{pmatrix} -2.0000 & -1.2506 & 1.2506 & 2.0000 \\ 0.3061 & 0.1939 & 0.1939 & 0.3061 \end{pmatrix}$
$(3, -1, 0)$	$[-4, 4]$	$\begin{pmatrix} -2.0609 & -1.3239 & 1.3239 & 2.0609 \\ 0.2966 & 0.2034 & 0.2034 & 0.2966 \end{pmatrix}$

Table 2: Locally D_s-optimal designs for estimating the two parameters β and μ in the quadratic logistic model for different nominal values and different design intervals.

$(\alpha^0, \beta^0, \mu^0)$	Design space	Locally D_s-optimal designs for estimating β and μ
$(0, -1, 0)$	$[-1, 1]$	$\begin{pmatrix} -1.0000 & 0.0000 & 1.0000 \\ 0.3423 & 0.3153 & 0.3423 \end{pmatrix}$
$(0, -1, 0)$	$[-2, 2]$	$\begin{pmatrix} -1.5449 & 0.0000 & 1.5449 \\ 0.3779 & 0.2442 & 0.3779 \end{pmatrix}$
$(3, -1, 0)$	$[-1, 1]$	$\begin{pmatrix} -1.0000 & 0.0000 & 1.0000 \\ 0.3042 & 0.3916 & 0.3042 \end{pmatrix}$
$(3, -1, 0)$	$[-2, 2]$	$\begin{pmatrix} -2.0000 & -1.0516 & 1.0516 & 2.0000 \\ 0.2963 & 0.2037 & 0.2037 & 0.2963 \end{pmatrix}$
$(3, -1, 0)$	$[-4, 4]$	$\begin{pmatrix} -2.1428 & -1.1867 & 1.867 & 2.1428 \\ 0.3063 & 0.1937 & 0.1937 & 0.3063 \end{pmatrix}$

Table 3: Weights of selected locally c-optimal designs for the Survival Model.

Nominal values	p_1	p_2
$\alpha^0 = -2.163$, $\beta^0 = -0.1$	0.498	0.502
$\alpha^0 = -2.163$, $\beta^0 = -0.405$	0.491	0.509
$\alpha^0 = -2.163$, $\beta^0 = -1.526$	0.425	0.575
$\alpha^0 = -2.163$, $\beta^0 = -2.623$	0.324	0.676

Table 4: CPU times in seconds (standard deviations) required by PSO to generate the various locally optimal designs averaged over 30 replications.

Particle number	D-optimal design for compartmental model	AUC-optimal design for compartmental model	Time-To-Max Conc-optimal design for compartmental model
20	1.477 (4.005)	4.514 (2.455)	6.511 (2.871)
40	0.131 (0.027)	4.796 (3.822)	6.012 (1.808)
60	0.146 (0.032)	6.026 (1.346)	6.674 (2.064)
80	0.160 (0.022)	6.137 (1.867)	7.427 (2.089)
100	0.179 (0.018)	6.312 (1.548)	8.565 (2.195)
120	0.193 (0.020)	6.295 (1.924)	9.044 (2.711)
140	0.225 (0.026)	7.513 (1.367)	9.302 (2.236)
160	0.240 (0.031)	6.608 (2.362)	9.273 (2.530)
180	0.251 (0.029)	7.609 (2.730)	10.233 (3.325)
200	0.275 (0.020)	8.741 (2.519)	9.708 (3.082)

Particle number	D-optimal design for double exp. model	D-optimal design for survival model	c-optimal design for survival model
20	0.773 (2.117)	0.042 (0.021)	0.010 (0.004)
40	0.483 (0.258)	0.040 (0.004)	0.011 (0.003)
60	1.172 (2.641)	0.047 (0.005)	0.013 (0.002)
80	0.990 (1.138)	0.056 (0.007)	0.014 (0.001)
100	1.074 (1.043)	0.065 (0.007)	0.016 (0.002)
120	1.562 (1.226)	0.073 (0.008)	0.018 (0.003)
140	1.293 (0.615)	0.084 (0.008)	0.020 (0.002)
160	1.802 (1.780)	0.092 (0.009)	0.023 (0.003)
180	2.443 (2.428)	0.098 (0.009)	0.025 (0.002)
200	1.617 (0.773)	0.104 (0.017)	0.026 (0.002)