Stat Comput (2016) 26:1121–1136
DOI 10.1007/s11222-015-9596-z

A pseudo-marginal sequential Monte Carlo algorithm for random effects models in Bayesian sequential design

J. M. McGree[1] · C. C. Drovandi[1] · G. White[1] · A. N. Pettitt[2]

Received: 15 October 2014 / Accepted: 13 July 2015 / Published online: 5 August 2015

Abstract Motivated by the need to sequentially design experiments for the collection of data in batches or blocks, a new pseudo-marginal sequential Monte Carlo algorithm is proposed for random effects models where the likelihood is not analytic, and has to be approximated. This new algorithm is an extension of the idealised sequential Monte Carlo algorithm where we propose to unbiasedly approximate the likelihood to yield an efficient exact-approximate algorithm to perform inference and make decisions within Bayesian sequential design. We propose four approaches to unbiasedly approximate the likelihood: standard Monte Carlo integration; randomised quasi-Monte Carlo integration, Laplace importance sampling and a combination of Laplace importance sampling and randomised quasi-Monte Carlo. These four methods are compared in terms of the estimates of likelihood weights and in the selection of the optimal sequential designs in an important pharmacological study related to the treatment of critically ill patients. As the approaches considered to approximate the likelihood can be computationally expensive, we exploit parallel computational architectures to ensure designs are derived in a timely manner.

Keywords Graphics processing unit · Importance Sampling · Intractable likelihood · Laplace approximation · Nonlinear regression · Optimal design · Parallel computing · Particle filter · Randomised quasi Monte Carlo

1 Introduction

Experiments for the collection of data in batches or blocks are prevalent in applied science and technology in areas such as pharmacology (Mentré et al. 1997), agriculture (Patterson and Hunter 1983) and aeronautics (Woods and van de ven 2011). When modelling data from such experiments, it is important to account for the correlation or dependence of observations collected in a given batch or a given block. The same is of course true when constructing an optimal experimental design. Unfortunately, this task is generally computationally prohibitive, and therefore has received limited attention from researchers. In a sequential design setting, the computational challenge is efficiently updating prior information as new data are observed and the need to find an optimal design for the collection of data from the next batch or block each time this prior information has been updated. The latter requires locating the design that maximises the expected utility where this expected utility is generally a functional of the posterior distribution averaged over uncertainty in the model, parameter values and supposed observed data. Hence, it is necessary to efficiently sample from or accurately approximate a larger number of posterior distributions. This poses a significant computational challenge and renders many algorithms such as standard Markov chain

✉ J. M. McGree
james.mcgree@qut.edu.au

C. C. Drovandi
c.drovandi@qut.edu.au

G. White
gentry.white@qut.edu.au

A. N. Pettitt
a.pettitt@qut.edu.au

[1] School of Mathematical Sciences Faculty, Queensland University of Technology, GPO Box 2434, Brisbane, QLD 4001, Australia

[2] Australian Research Council Centre of Excellence for Mathematical & Statistical Frontiers (ACEMS), Parkville, Australia

Monte Carlo (MCMC) computationally infeasible (Ryan et al. 2015). Hence, the requirement for an efficient computational algorithm for Bayesian inference motivates the consideration of the sequential Monte Carlo (SMC) algorithm. Computational efficiency is achieved when using the SMC algorithm as prior information can be updated as more data are observed avoiding the need to re-run posterior sampling approaches based on the full data set. Further, in using the SMC algorithm, one can handle model uncertainty by running SMC algorithms for each rival model (Drovandi et al. 2014) avoiding any between-model moves. One also avoids computationally expensive approximations of the evidence or marginal likelihood of a given model through the availability of a convenient approximation which is a by-product of the SMC algorithm (Del Moral et al. 2006).

The particular design problem considered in this work is a sequential design problem in pharmacology, and involves pharmacokinetic (PK) models and extra corporeal membrane oxygenation (ECMO). In 2009, during the worldwide H1N1 pandemic, ECMO was a vital treatment for H1N1 patients requiring advanced ventilator support in Australia and internationally (Davies et al. 2009). ECMO is a modification of cardiopulmonary bypass (CPB). However, unlike conventional CPB, ECMO is utilised in already critically ill patients, and lasts for days rather than hours increasing the likelihood of complications. Nonlinear random effect models for data from such trials have been considered recently (Ryan et al. 2014). As the likelihood based on a value of the random effects is available analytically, these models are, in principle, straightforward to estimate, for example, via MCMC. However, the need to draw from many posterior distributions renders this sampling method computationally infeasible in the context of experimental design, and thus we propose a new SMC algorithm for sequential design and inference.

To apply the SMC algorithm in the PK context (and in general), the likelihood needs to be evaluated a large number of times. Unfortunately, for nonlinear random effect models, this likelihood is typically unavailable analytically (Kuk 1999). We therefore propose to unbiasedly estimate the likelihood within SMC forming an exact-approximate algorithm to facilitate efficient Bayesian inference and design for random effects models. In our research, we consider four approaches for the approximation. Firstly, we consider standard Monte Carlo (MC) integration. Here, Q random effect values are randomly drawn from the (current) prior distribution, and the average conditional likelihood (over Q) is taken as the estimate of the likelihood. Secondly, we extend standard MC integration by choosing randomised low discrepancy sequences of random numbers for the integration. This is known as randomised quasi-Monte Carlo (RQMC), and can yield more efficient estimates when compared to standard MC (Niederreiter 1978). Thirdly, we consider Laplace importance sampling (LIS) where a Laplace approximation is used to form the importance distribution in importance sampling (Kuk 1999). Lastly, we consider the combination of LIS and RQMC where draws from the importance distribution are chosen as (transformed) randomised low discrepancy random numbers. These approaches form new pseudo-marginal algorithms for random effect models, and are explained in Sect. 3. We also compared these methods within a sequential Bayesian design context in Sect. 5.

To facilitate the construction of designs in a reasonable amount of time, we propose the exploitation of parallel computational architectures. In particular, we explore the use of a graphics processing unit (GPU). We are not aware of the use of such hardware in the derivation of optimal designs. However, they have been used recently within SMC to reduce run times (Durham and Geweke 2011; Vergé et al. 2015). Actually, the SMC algorithm is often labelled an “embarrassingly parallel” algorithm (see for example Gramacy and Polson (2011)), and hence naturally lends itself to such endeavors. We note that our use of the GPU is not within the standard SMC algorithm but rather for the approximation of the likelihood.

A recent review of modern computational algorithms for Bayesian design has been given by Ryan et al. (2015), and discusses some work in a sequential design context. Approaches based on MCMC techniques have been explored for fixed effects models by Weir et al. (2007), McGree et al. (2012). In each case, importance sampling was used in selecting the next design point to avoid running many MCMC posterior simulations. Approaches based on SMC have also been considered for fixed effects models by Drovandi et al. (2013, 2014). In the 2013 paper, a variety of utility functions were considered to construct a design to precisely estimate model parameters. In the following 2014 paper, a utility function for model discrimination was developed and applied within a number of nonlinear settings. The generic algorithms used in both of these papers provides a basis for the work proposed in this paper. However, this generic algorithm was limited to independent data settings, and therefore our proposed methods for random effect models present a significant extension of this previous research.

Müller (1999) has considered an MCMC approach to derive static (non sequential) Bayesian experimental designs by considering the design variable as random and exploring a target distribution comprised of the design variable, the model parameter and the data. Extensions to this approach have been given by Amzal et al. (2006) who used a particle method to explore the utility surface and employed simulated annealing (Corana et al. 1987) to concentrate the samples near the modes. Such methodology has been applied in Bayesian experimental design, but is currently restricted to simply comparing fixed designs (Han and Chaloner 2004), optimising designs in low dimensions, for a single, fixed effects model (Müller 1999) and/or limited utility functions (Stroud et al. 2001).

Our paper proceeds with a description of the inference framework within which we develop our methodology. This is followed by Sect. 3 which decribes our proposed SMC algorithm for random effects models. In Sect. 4, we define a utility function called Bayesian A-optimality for parameter estimation and show how it is approximated within our algorithm. In Sect. 5, our proposed methods are applied to a PK study in sheep. We conclude with a discussion of our work, and suggestions for further research.

2 Inferential framework

Consider the sequential problem where a design is required for the collection of data in a batch or block to precisely estimate parameters across one or a finite number of K models defined by the random variable $M \in \{1,\ldots,K\}$. We follow the M-closed perspective of Bernardo and Smith (2000), and assume that one of the K models is appropriate to describe the observed data. Each model m contains parameters $\boldsymbol{\theta}_m = (\boldsymbol{\mu}_m, \boldsymbol{\Omega}_m, \sigma_m)$ defining the model parameters $\boldsymbol{\mu}_m$, the between batch or block variability of the model parameters $\boldsymbol{\Omega}_m$ and the residual variability parameter σ_m. Note that the subscript m will be dropped if only one model is under consideration. Define the likelihood function $f(\boldsymbol{y}_{1:j}|M = m, \boldsymbol{\theta}_m, \boldsymbol{d}_{1:j})$ for all data $\boldsymbol{y}_{1:j}$ observed up to batch or block j at design points $\boldsymbol{d}_{1:j}$. To construct this likelihood for each model, we assume that batch/block effects are independent random draws from a population of batches/blocks so that data from block j denoted as y_j are conditionally independent of $\boldsymbol{y}_{1:j-1}$ and $\boldsymbol{d}_{1:j-1}$ given d_j. Then, the likelihood is formed as follows:

$$
\begin{aligned}
& f\left(\boldsymbol{y}_{1:j}|M = m, \boldsymbol{\theta}_m, \boldsymbol{d}_{1:j}\right) \\
& \quad = \Pi_{k=1}^{j} f\left(y_k|M = m, \boldsymbol{\theta}_m, d_k\right), \quad \text{for} \quad j = 1,\ldots,J.
\end{aligned}
$$

Then, the likelihood for data from batch or block j can be expressed as

$$
\begin{aligned}
f\left(y_j|M = m, \boldsymbol{\theta}_m, d_j\right) = \int & f\left(y_j|\boldsymbol{b}_{mj}, M = m, \boldsymbol{\theta}_m, d_j\right) \\
& \times p\left(\boldsymbol{b}_{mj}|\boldsymbol{\mu}_m, \boldsymbol{\Omega}_m, M = m\right) d\boldsymbol{b}_{mj},
\end{aligned}
\tag{1}
$$

where $\boldsymbol{b}_{mj} \sim p(\boldsymbol{\mu}_m, \boldsymbol{\Omega}_m|M = m)$ denotes the random effect associated with block j for model m.

Prior distributions are placed on $\boldsymbol{\theta}_m$ for each model denoted as $p(\boldsymbol{\theta}_m|M = m)$, and we also define a probability distribution for the random effects $p(\boldsymbol{b}_{mj}|\boldsymbol{\mu}_m, \boldsymbol{\Omega}_m, M = m)$, where in our work this is a multivariate normal distribution with mean $\boldsymbol{\mu}_m$ and variance-covariance $\boldsymbol{\Omega}_m$ for model m. We also place prior model probabilities on each model denoted as $p(M = m)$. All of this prior information is sequentially updated as data are observed on each block, and then used in finding the optimal design for data collection in the next batch or block.

3 Sequential Monte Carlo algorithm

SMC is an algorithm for sampling from a smooth sequence of target distributions. Originally developed for dynamic systems and state space models (Gordon et al. 1994; Liu and Chen 1998), the algorithm has also been applied to static parameter models through the use of a sequence of artificial distributions (Chopin 2002; Del Moral et al. 2006). In our sequential design setting, the sequence of target distributions presents as a sequence of posterior distributions through data annealing (for example, see Gilks and Berzuini 2001). We first introduce the standard or idealised SMC algorithm, then present our new developments.

3.1 Idealised sequential Monte Carlo algorithm

As given in Chopin (2002), for a particular model m, the sequence of target distributions built up through data annealing is given by

$$
\begin{aligned}
& p\left(\boldsymbol{\theta}_m|M = m, \boldsymbol{y}_{1:j}, \boldsymbol{d}_{1:j}\right) \propto f\left(\boldsymbol{y}_{1:j}|M = m, \boldsymbol{\theta}_m, \boldsymbol{d}_{1:j}\right) \\
& \quad \times p\left(\boldsymbol{\theta}_m|M = m\right), \quad \text{for} \quad j = 1,\ldots,J.
\end{aligned}
$$

For a given model, there are essentially three steps in the SMC algorithm; re-weighting, resampling and mutation steps. As data are observed, the algorithm generates a set of N weighted particles for each model m to represent each target/posterior distribution in the sequence. This is achieved by initially drawing equally weighted particles for each model from the respective prior distributions. As data are observed, particles for each model are continually re-weighted via importance sampling (Hammersley and Handscomb 1964) until the effective sample size (ESS_m) of the importance approximation of the current target distribution for each model falls below a predefined threshold (E). For models where $ESS_m < E$, within model resampling is performed to replicate particles with relatively large weight while eliminating particles with relatively small weight. This is followed by the mutation step where an MCMC kernel (Metropolis et al. 1953; Hastings 1970) that maintains invariance of the current target is used to diversify each particle set. Alternative kernels are possible that lead to $\mathcal{O}(N^2)$ rather than $\mathcal{O}(N)$ SMC algorithms (Del Moral et al. 2006).

In using this SMC algorithm, there is also the availability of an efficient estimate of the evidence for a given model (leading to an efficient estimate of posterior model probabilities). As shown in Del Moral et al. (2006), the evidence of a given model can be approximated as a by-product of the SMC algorithm. To show this, we note that the ratio of normalising constants for a given model m ($Z_{m,j+1}/Z_{m,j}$) is equivalent to the predictive distribution of the next observation y_{j+1} given current data $\boldsymbol{y}_{1:j}$:

$$Z_{m,j+1}/Z_{m,j} = \int_{\boldsymbol{\theta}_m} f\left(y_{j+1}|M=m,\boldsymbol{\theta}_m,d_{j+1}\right) \times p\left(\boldsymbol{\theta}_m|M=m,\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j}\right) d\boldsymbol{\theta}_m.$$

Here, we form a particle approximation to the above integral as follows:

$$Z_{m,j+1}/Z_{m,j} \approx \sum_{i=1}^{N} W^i_{m,j} f\left(y_{j+1}|M=m,\boldsymbol{\theta}^i_{m,j},d_{j+1}\right).$$

This approximation of the evidence is therefore available at negligible additional computational cost, and allows for efficient design and analysis in sequential settings where there exists uncertainty about the model (Drovandi et al. 2014).

3.2 Pseudo-marginal sequential Monte Carlo algorithm for random effect models

Implementing the idealised SMC algorithm requires evaluating the likelihood many times (in the re-weight and move steps). Unfortunately for nonlinear random effect models, this is generally analytically intractable, see Eq. (1). Here, we propose to extend the idealised SMC algorithm to random effect models through unbiasedly approximating this likelihood. Four different approaches are considered for this approximation. Firstly, we consider standard MC integration for the approximation. For each particle of a given model $\boldsymbol{\theta}^i_m$, the likelihood can be approximated as follows:

$$\begin{aligned} f\left(y_j|M=m,\boldsymbol{\theta}^i_m,d_j\right) &= \int f\left(y_j|\boldsymbol{b}_{mj},M=m,\boldsymbol{\theta}^i_m,d_j\right) \\ &\quad\times p\left(\boldsymbol{b}_{mj}|\boldsymbol{\mu}^i_m,\boldsymbol{\Omega}^i_m,M=m\right) d\boldsymbol{b}_{mj}, \\ &\approx \frac{1}{Q}\sum_{q=1}^{Q} f\left(y_j|\boldsymbol{b}^q_{mj},M=m,\boldsymbol{\theta}^i_m,d_j\right) \\ &= \hat{f}\left(y_j|M=m,\boldsymbol{\theta}^i_m,d_j\right), \end{aligned} \qquad (2)$$

where

$$\boldsymbol{b}^q_{mj} \overset{iid}{\sim} p\left(\cdot|\boldsymbol{\mu}^i_m,\boldsymbol{\Omega}^i_m,M=m\right), \quad q=1,\ldots,Q.$$

Alternatively, QMC methods can be used to approximate the integral in Eq. (2). To implement QMC, the Q integration nodes are replaced with (transformed) deterministic nodes that are more evenly distributed over $(0,1]^{dim(\boldsymbol{\mu}_m)}$. Examples of such sequences include the Halton (Halton 1960), Sobol' (Sobol' 1967) and Faure (Faure 1982) sequences. In order to maintain an unbiased estimate, randomised versions of these deterministic sequences can be used. Here, we consider a rank-1 lattice shift which entails applying a random shift modulo 1 (Cranley and Patterson 1976; L'Ecuyer and Lemieux 2000) and the Baker's transformation to each coordinate in the sequence. Pseudo code for this is given in Algorithm 1, where v_k is the shift applied to the lattice. Such an approach has been used to estimate the likelihood function of a mixed logit model (Munger et al. 2012).

Algorithm 1 Pseudo code for rank-1 lattice shift of u in $(0,1]^{dim(\boldsymbol{\mu}_m)}$

```
1: Initialize $v_k$ and $u_{q,k} \in (0,1]^{dim(\boldsymbol{\mu}_m)}$ for $i = 1,\ldots,Q$ and $k = 1,\ldots,dim(\boldsymbol{\mu}_m)$
2: for $q = 1 : Q$ do
3:   for $k = 1 : dim(\boldsymbol{\mu}_m)$ do
4:     $u_{q,k} = 2(((q-1)v_k + k) \bmod 1)$
5:     if $u_{q,k} \geq 1$ then
6:       $u_{q,k} = 2 - u_{q,k}$
7:     end if
8:   end for
9: end for
10: Output: Rank-1 lattice shift of $u_{q,k}$ for $i = 1,\ldots,Q$ and $k = 1,\ldots,dim(\boldsymbol{\mu}_m)$
```

Implementing this approach to randomise the deterministic sequences in our framework means that the actual values of $\boldsymbol{b}^q_{mj}$ are based on $\boldsymbol{u}_q \in (0,1]^{dim(\boldsymbol{\mu}_m)}$ that are more evenly distributed over $(0,1]^{dim(\boldsymbol{\mu}_m)}$ than random draws. Such an approach is termed RQMC, and has been shown to be superior to standard Monte carlo integration in terms of the efficiency of an estimate (Morokoff and Caflisch 1995). In our work, the random shift is applied to an initial Halton sequence of length Q in $(0,1]^{dim(\boldsymbol{\mu}_m)}$ each 'new' time the likelihood is estimated. This ensures appropriate unbiasedness for our exact-approximate algorithm. Then, the Cholesky factorization of $\boldsymbol{\Omega}^i_m$ is found and applied to the $\Phi^{-1}(\boldsymbol{u}_q)$s to generate the $\boldsymbol{b}^q_{mj}$s from the appropriate distribution, where Φ denotes the normal cumulative distribution function.

The third approach we consider to approximate this likelihood is LIS. This approach was proposed by Kuk (1999) to estimate the likelihood for generalised linear mixed models. For each particle $\boldsymbol{\theta}^i_m$, a numerical optimiser was used to find the mode of the following with respect to $\boldsymbol{b}_{mj}$:

$$f\left(y_j|\boldsymbol{b}_{mj},M=m,\boldsymbol{\theta}^i_m,d_j\right) p\left(\boldsymbol{b}_{mj}|\boldsymbol{\mu}^i,\boldsymbol{\Omega}^i_m,M=m\right),$$

and a multivariate Normal distribution (denoted as $p_{LA}(\boldsymbol{\mu}^i_{LA},\boldsymbol{\Omega}^i_m,M=m)$) was formed with mean $\boldsymbol{\mu}^i_{LA}$ being the mode and the variance-covariance matrix $\boldsymbol{\Omega}^i_m$ being the random effect variability corresponding to the ith particle for model m. Then, by noting that

$$f\left(y_j|M=m,\boldsymbol{\theta}_m^i,d_j\right) = \int \frac{f\left(y_j|\boldsymbol{b}_{mj},M=m,\boldsymbol{\theta}_m^i,d_j\right) p\left(\boldsymbol{b}_{mj}|\boldsymbol{\mu}_m^i,\boldsymbol{\Omega}_m^i,M=m\right)}{p_{LA}\left(\boldsymbol{b}_{mj}|\boldsymbol{\mu}_{LA}^i,\boldsymbol{\Omega}_m^i,M=m\right)} \times p_{LA}\left(\boldsymbol{b}_{mj}|\boldsymbol{\mu}_{LA}^i,\boldsymbol{\Omega}_m^i,M=m\right) d\boldsymbol{b}_{mj}, \tag{3}$$

the likelihood can be approximated as follows:

$$\hat{f}\left(y_j|M=m,\boldsymbol{\theta}_m^i,d_j\right) = \frac{1}{Q}\sum_{q=1}^{Q}\frac{f\left(y_j|\boldsymbol{b}_{mj}^q,M=m,\boldsymbol{\theta}_m^i,d_j\right) p\left(\boldsymbol{b}_{mj}^q|\boldsymbol{\mu}_m^i,\boldsymbol{\Omega}_m^i,M=m\right)}{p_{LA}\left(\boldsymbol{b}_{mj}^q|\boldsymbol{\mu}_{LA}^i,\boldsymbol{\Omega}_m^i,M=m\right)},$$

where $\boldsymbol{b}_{mj}^q \sim p_{LA}(\boldsymbol{\mu}_{LA}^i,\boldsymbol{\Omega}_m^i,M=m)$ for each particle i.

The fourth approach combines LIS and RQMC. Instead of drawing randomly from the Laplace approximation, these draws are chosen based on randomised Halton sequences using $\boldsymbol{\mu}_{LA}^i$ and $\boldsymbol{\Omega}_m^i$.

In comparing these approximations, standard MC and RQMC will potentially perform poorly when the corresponding Laplace approximation does not overlap with the prior density $p(\boldsymbol{b}_{mj}|\boldsymbol{\mu}_m^i,\boldsymbol{\Omega}_m^i,M=m)$, with MC generally producing more variable weights. Such cases could occur, for example, when the random effect values are relatively far from $\boldsymbol{\mu}_m^i$. Hence, it is believed that MC and RQMC will produce more variable likelihood weights (and therefore more variable estimates) than LIS and LIS+RQMC. However, this comes at the computational cost of finding the mode for each particle. This potential trade-off between variability of weights and computational cost will be explored in Sect. 5.

Any of the above approaches can be used to unbiasedly approximate the likelihood to form a pseudo-marginal SMC algorithm for random effect models. Andrieu and Roberts (2009) consider a similar psuedo-marginal approach within an MCMC framework, and Tran et al. (2014) consider Bayesian inference via importance sampling for models with latent variables based on an unbiased estimate of the likelihood. Our algorithm is similar to the framework proposed by Chopin et al. (2013) for state space models. Further, there have been developments using QMC and RQMC in the SMC algorithm, see Gerber and Chopin (2014) who provide empirical evidence that using QMC methods may significantly outperform the standard implementation of SMC in terms of approximation error. As noted in Gerber and Chopin (2014), the error rate for standard MC is $\mathcal{O}(N^{-1/2})$ improving to $\mathcal{O}(N^{-1+\epsilon})$ using QMC, and improving again to $\mathcal{O}(N^{-3/2+\epsilon})$ using RQMC, under certain conditions (Owen 1997a, b, 1998b). Owen (1998a) also proposed a method based on RQMC for 'very high' dimensional problems, but notes that the advantages of using QMC and RQMC diminish as the dimension increases or for integrals that are not smooth (Morokoff and Caflisch 1995).

3.3 Implementation of the pseudo-marginal sequential Monte Carlo algorithm

Pseudo code for our SMC algorithm is given in Algorithm 2, and now explained. Let $\{W_{m,j}^i,\boldsymbol{\theta}_{m,j}^i\}_{i=1}^N$ denote the particle approximation for model m, for target $p_j(.)$, then the reweight step is given by

$$w_{m,j+1}^i = W_{m,j}^i \hat{f}\left(y_{j+1}|M=m,\boldsymbol{\theta}_{m,j}^i,d_{j+1}\right),$$

as the batch or block data are conditionally independent and an MCMC kernel is used in the mutation step. From Chopin (2002) and Del Moral et al. (2006), the $\hat{f}(y_{j+1}|M=m,\boldsymbol{\theta}_{m,j}^i,d_{j+1})$ are the approximate incremental weights given by target $j+1$ divided by target j, and are given by the approximate likelihood, Eq. (2). Once the new weights $w_{m,j+1}^i$ are normalized to give $W_{m,j+1}^i$, the particle approximation $\{W_{m,j+1}^i,\boldsymbol{\theta}_{m,j}^i\}_{i=1}^N$ approximates target $j+1$.

Algorithm 2 SMC algorithm for random effect models incorporating model uncertainty

```
1: Draw $\boldsymbol{\theta}_{m,0}^i \sim p(\boldsymbol{\theta}_m|M=m)$ and set $W_{m,0}^i = 1/N$, for $m = 1,\ldots,K$ and $i = 1,\ldots,N$
2: Set $\hat{Z}_{m,0} = 1$ for $m = 1,\ldots,K$
3: Let $\boldsymbol{d} \in \boldsymbol{D}$
4: for $j = 0 : J-1$ do
5:   Find design point $d_{j+1}$ and collect data point $y_{j+1}$
6:   for $m = 1 : K$ do
7:     Re-weight step: $w_{m,j+1}^i = W_{m,j}^i \hat{f}(y_{j+1}|M=m,\boldsymbol{\theta}_{m,j}^i,d_{j+1})$, for $i = 1,\ldots,N$
8:     Estimate marginal likelihood for each model via $Z_{m,j+1}/Z_{m,j} \approx \sum_{i=1}^N W_{m,j}^i \hat{f}(y_{j+1}|M=m,\boldsymbol{\theta}_{m,j}^i,d_{j+1})$
9:     Normalize weights $W_{m,j+1}^i = w_{m,j+1}^i / \sum_{k=1}^N w_{m,j+1}^k$, for $i = 1,\ldots,N$
10:    Calculate $\text{ESS}_m = 1/\sum_{i=1}^N (W_{m,j+1}^i)^2$
11:    if $\text{ESS}_m < E$ OR $j = J-1$ then
12:      Resample step: $\{\boldsymbol{\theta}_{m,j}^i, W_{m,j+1}^i\}_{i=1}^N \rightarrow \{\boldsymbol{\theta}_{m,j+1}^i, 1/N\}_{i=1}^N$
13:      Calculate the random walk variance terms for MCMC proposal $q_{m,j+1}(.|.)$ using particles $\{\boldsymbol{\theta}_{m,j}^i, W_{m,j+1}^i\}_{i=1}^N$
14:      for $i = 1 : N$ do
15:        Move step: Perform $R_m$ moves on particle $\boldsymbol{\theta}_{m,j+1}^i$ with an MCMC kernel of invariant distribution $p_{j+1}(\boldsymbol{\theta}_m|M=m,\boldsymbol{y}_{1:j+1},\boldsymbol{d}_{1:j+1})$ with acceptance probability $\alpha$ with $\hat{f}(\boldsymbol{y}_{1:j+1}|M=m,\boldsymbol{\theta}_{m,j+1}^i,\boldsymbol{d}_{1:j+1})$ being recorded and re-used.
16:      end for
17:    else
18:      Set $\boldsymbol{\theta}_{m,j+1}^i = \boldsymbol{\theta}_{m,j}^i$, for $i = 1,\ldots,N$
19:    end if
20:  end for
21: end for
22: Output: $\{\boldsymbol{\theta}_{m,J}^i, 1/N\}_{i=1}^N$ for $m = 1,\ldots,K$
```

We assess the adequacy of this approximation by estimating the ESS_m by $1/\sum_{i=1}^{N}(W^i_{m,j+1})^2$ (Liu and Chen 1995). If the ESS_m falls below E, the particle set is replenished by resampling the particles with probabilities proportional to the normalised weights. In our work, we used multinomial resampling. However, other resampling techniques such as systematic or residual resampling could be considered (Kitagawa 1996). Following this step, the particles are diversified by applying R_m MCMC steps to each particle to increase the probability of the particle moving. Assuming a symmetric proposal distribution, the acceptance probability α for a proposal $\boldsymbol{\theta}^*_{m,j+1}$ for a given model is given by the Metropolis probability

$$\alpha = \min\left\{\frac{p\left(\boldsymbol{\theta}^*_{m,j+1}\right)\hat{f}\left(\boldsymbol{y}_{1:j+1}|M=m,\boldsymbol{\theta}^*_{m,j+1},\boldsymbol{d}_{1:j+1}\right)}{p\left(\boldsymbol{\theta}^i_{m,j+1}\right)\hat{f}\left(\boldsymbol{y}_{1:j+1}|M=m,\boldsymbol{\theta}^i_{m,j+1},\boldsymbol{d}_{1:j+1}\right)},1\right\}.$$

The proposal distribution in the MCMC kernel $q(.|.)$ is efficiently constructed based the current set of particles (as they are already distributed according to the target $p_{j+1}(.)$). This avoids tuning the algorithm or having to implement other schemes such as adaptive MCMC. We also note that as each rival model has a particle set, any between model 'jumps' are avoided.

As shown in Sect. 3.1, there is also the availability of an efficient estimate of the evidence for a given model. For random effect models, we base this on the approximate likelihood. We note that an estimate of the evidence is generally difficult to obtain for nonlinear models, and, in particular, mixed effects models. The particle approximation for Eq. (3) is then based on the approximate likelihood as follows:

$$Z_{m,j+1}/Z_{m,j} \approx \sum_{i=1}^{N} W^i_{m,j}\hat{f}\left(y_{j+1}|M=m,\boldsymbol{\theta}^i_{m,j},d_{j+1}\right).$$

Tran et al. (2014) show the validity of this estimate. As $Z_0 = 1$, the evidence can be approximated sequentially in the algorithm as data are observed. Further, posterior model probabilities $p(m|\boldsymbol{y}_{1:j+1},\boldsymbol{d}_{1:j+1})$ are estimated based on the above estimates of evidence (for each given model).

4 Experimental design

SMC has been employed within a sequential design framework for estimation of a fixed effects model (Drovandi et al. 2013) and for discriminating between fixed effects models (Drovandi et al. 2014). In each paper, appropriate utility functions were defined for the experimental goals. Here, we consider an estimation utility termed Bayesian A-optimality, and extend to the case where one wishes to precisely estimate parameters over a finite number of models. This utility is based on A-optimality (Kiefer 1959), where the total or average variance of the parameter estimates is minimised (Atkinson et al. 2007).

Consider a design $\boldsymbol{d}$. In general, the expected utility for this design for a single model can be expressed as

$$u(\boldsymbol{d}) = \int_{\boldsymbol{\theta}}\int_{\boldsymbol{y}} u(\boldsymbol{d},\boldsymbol{\theta},\boldsymbol{y})p(\boldsymbol{\theta},\boldsymbol{y}|\boldsymbol{d})\mathrm{d}\boldsymbol{y}\mathrm{d}\boldsymbol{\theta}. \tag{4}$$

This can be extended as follows when uncertainty about the true model is considered

$$u(\boldsymbol{d}) = \sum_{m=1}^{K} p(M=m|\boldsymbol{y},\boldsymbol{d})\int_{\boldsymbol{\theta}_m}\int_{\boldsymbol{y}} u(\boldsymbol{d},\boldsymbol{y},m,\boldsymbol{\theta}) \times p(\boldsymbol{\theta}_m,\boldsymbol{y},M=m|\boldsymbol{d})\,\mathrm{d}\boldsymbol{y}\mathrm{d}\boldsymbol{\theta}_m. \tag{5}$$

Two utility functions will be considered in the examples that follow. For a single model, the utility is given by the inverse of the trace of the posterior variance; *Bayesian A-optimality* as follows:

$$u(\boldsymbol{d},\boldsymbol{y}) = 1/\mathrm{trace}\ \left(\mathrm{VAR}\left[\boldsymbol{\theta}|\boldsymbol{y},\boldsymbol{d}\right]\right).$$

This is extended to the case of K models by maximising the inverse of the product of the Bayesian A-optimality utility values over the K models (scaled appropriately such that each utility value is between 0 and 1). This can be simplified by taking the logarithm which leads to the consideration of maximising the inverse of the sum of the logarithm of traces of the posterior variances for all K models. That is,

$$u(\boldsymbol{d},\boldsymbol{y}) = 1/\sum_{l=1}^{K}\log\mathrm{trace}\ (\mathrm{VAR}[\boldsymbol{\theta}_l|\boldsymbol{y},\boldsymbol{d},M=l]).$$

We now show how to estimate utilities of the forms given in Eqs. (4) and (5) in a sequential design framework. Suppose we have collected data up until batch or block j denoted as $\boldsymbol{y}_{1:j}$ collected at design points $\boldsymbol{d}_{1:j}$. Define a general utility $u(\boldsymbol{d},z,m,\boldsymbol{\theta}_m|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j})$, where $\boldsymbol{d}$ is a proposed design for future observation z taken from model m with parameter $\boldsymbol{\theta}_m$. Then, the expected utility of a given design $\boldsymbol{d}$ conditional on data already observed $\boldsymbol{y}_{1:j}$ at design points $\boldsymbol{d}_{1:j}$, denoted as $u(\boldsymbol{d}|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j})$, is given by

$$\begin{aligned} u\left(\boldsymbol{d}|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j}\right) &= E_{z,m,\boldsymbol{\theta}_m|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j}}\left[u\left(\boldsymbol{d},z,m,\boldsymbol{\theta}_m|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j}\right)\right] \\ &= \sum_{m=1}^{K}\int_z\int_{\boldsymbol{\theta}_m} u\left(\boldsymbol{d},z,m,\boldsymbol{\theta}_m|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j}\right) \\ &\quad \times p\left(z,m,\boldsymbol{\theta}_m|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j},\boldsymbol{d}\right)d\boldsymbol{\theta}_m dz \\ &= \sum_{m=1}^{K}\int_z\int_{\boldsymbol{\theta}_m} u\left(\boldsymbol{d},z,m,\boldsymbol{\theta}_m|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j}\right) \end{aligned}$$

$$\times p\left(z|m,\boldsymbol{\theta}_m,\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j},\boldsymbol{d}\right)$$
$$\times p\left(\boldsymbol{\theta}_m|m,\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j}\right)$$
$$\times p\left(m|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j}\right)d\boldsymbol{\theta}_m dz$$
$$=\sum_{m=1}^{K}p\left(m|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j}\right)$$
$$\int_z\int_{\boldsymbol{\theta}_m}u\left(\boldsymbol{d},z,m,\boldsymbol{\theta}_m|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j}\right)$$
$$\times p\left(z|m,\boldsymbol{\theta}_m,\boldsymbol{d}\right)p\left(\boldsymbol{\theta}_m|m,\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j}\right)d\boldsymbol{\theta}_m dz,$$

where the summation over the posterior model probabilities is dropped if only a single model is under consideration.

As shown in Algorithm 2, suppose, up to batch or block j, we have a particle set for each model defined as $\{W_{m,j}^i,\boldsymbol{\theta}_{m,j}^i\}_{i=1}^N$, for $m=1,\ldots,K$. We approximate the above integrals via simulating $z_{m,j}^i$ from the posterior predictive distribution $p(z|m,\boldsymbol{\theta}_{m,j}^i,\boldsymbol{d})$. This gives a weighted sample $\{W_{m,j}^i,\boldsymbol{\theta}_{m,j}^i,z_{m,j}^i\}$ from $p(z,\boldsymbol{\theta}_m|m,\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j},\boldsymbol{d})$. Then, MC integration can be used to approximate the above integrals as follows:

$$u\left(\boldsymbol{d}|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j}\right)\approx\sum_{m=1}^{K}p\left(m|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j}\right)\sum_{i=1}^{N}W_{m,j}^i$$
$$\times u\left(\boldsymbol{d},z_{m,j}^i|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j}\right),$$

where $u(\boldsymbol{d},z_{m,j}^i|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j})$ is approximated by forming a particle approximation (via importance sampling) to the posterior distribution where $z_{m,j}^i$ (and $\boldsymbol{d}$) are supposed observed data. That is, in the case of model uncertainty,

$$u\left(\boldsymbol{d},z_{m,j}^i|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j}\right)$$
$$=1\bigg/\sum_{l=1}^{K}\log\text{trace}\left(\text{VAR}[\boldsymbol{\theta}_l|\boldsymbol{y}_{1:j},z_{m,j}^i,\boldsymbol{d}_{1:j},\boldsymbol{d},M=l]\right).$$

5 Examples

Data have been collected on sheep which have been treated with ECMO, and have been modelled previously by Ryan et al. (2014). Each sheep was subjected to ECMO for 24 hours and infused with various antibiotic drugs. Blood samples were then collected at various times. In the original studies, Davies et al. (2009) and Shekar et al. (2013) were interested in the PK profile of antibiotic drugs in healthy sheep receiving ECMO for 24 hours as it is known that drugs are absorbed in the circuitry of ECMO. We propose to re-design this study to minimise the uncertainty about population PK parameters such that differences between PK profiles for ECMO versus non-ECMO sheep can be investigated.

The general form of the models considered in Ryan et al. (2014) and in this paper are as follows. Let the time of blood

Algorithm 3 Estimation of utility function within the SMC algorithm for random effect models incorporating model uncertainty

```
1: We have particles $\{W_{m,j}^i,\boldsymbol{\theta}_{m,j}^i\}_{i=1}^N$, for $m=1,\ldots,K$
2: Initialise data $\boldsymbol{y}_{1:j}$ and designs $\boldsymbol{d}_{1:j}$
3: for $\boldsymbol{d}\in\boldsymbol{D}$ do
4:   for $m=1,\ldots,K$ do
5:     for $i=1,\ldots,N$ do
6:       Simulate $z_{m,j}^i\sim p(z|M=m,\boldsymbol{\theta}_{m,j}^i,\boldsymbol{d})$
7:       Form temporary weights $\tilde{w}_m^i$ via re-weight step $\tilde{w}_m^i = W_{m,j}^i\hat{f}(z_{m,j}^i|m,\boldsymbol{\theta}_{m,j}^i,\boldsymbol{d})$
8:       Evaluate utility $u(\boldsymbol{d},z_{m,j}^i|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j})$ based on $\tilde{w}_m^i$
9:     end for
10:  end for
11:  $u(\boldsymbol{d}|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j}) \approx \sum_{m=1}^K p(m|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j})\sum_{i=1}^N W_{m,j}^i u(\boldsymbol{d}, z_{m,j}^i|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j})$
12: end for
13: Output: Estimate of $u(\boldsymbol{d}|\boldsymbol{y}_{1:j},\boldsymbol{d}_{1:j})$ for $\boldsymbol{d}\in\boldsymbol{D}$.
```

samples in minutes for sheep j (where each sheep was considered as a block) be denoted as $d_j=(d_{1j},d_{2j})$, where the possible sampling times available for blood collection are:

$$\begin{bmatrix}6 & 15\\ 15 & 30\\ 30 & 45\\ 45 & 60\\ 60 & 120\\ 120 & 180\\ 180 & 240\\ 240 & 300\\ 300 & 360\end{bmatrix}.$$

That is, two repeated measures are taken on each sheep, and the design problem is to choose which two sampling times (that is, which row of the above matrix) to actually use to collect plasma samples. Then

$$y_j\sim MVN\left(g\left(\boldsymbol{\mu}_j,d_j\right),\sigma^2\text{diag }g\left(\boldsymbol{\mu}_j,d_j\right)^2\right),$$
$$\boldsymbol{b}_j\sim MVN\left(\boldsymbol{0},\boldsymbol{\Omega}\right),\quad \boldsymbol{\mu}_j=\boldsymbol{\mu}+\boldsymbol{b}_j,$$

with $\boldsymbol{\mu}_j$ defined later, and priors

$$\boldsymbol{\mu}\sim MVN\left(\boldsymbol{\delta},\boldsymbol{\Sigma}\right),\ \text{for }\boldsymbol{\delta}\text{ and }\boldsymbol{\Sigma}\text{ known}$$
$$\boldsymbol{\Omega}\sim InvWish\left(\boldsymbol{\Psi},\nu\right),\ \text{for }\boldsymbol{\Psi}\text{ and }\nu\text{ known}$$
$$\log\sigma\sim N(a,b),\ \text{for }a\text{ and }b\text{ known.}$$

In both of the examples that follow, we assumed the broad spectrum antibiotic meropenem was administered to 10 sheep which were on ECMO. Meropenem was delivered as a single, IV bolus dose of 500 mg ($D=500$), infused over 30 minutes ($Tinf=30$). The design problem was to sequentially determine the optimal times to collect blood samples in plasma from 10 sheep in order to precisely estimate population PK parameters. We considered Bayesian A-optimality for this purpose, and, as a comparison, we also considered a random design (selects sampling times at random). To investigate the properties of the utility functions, 500 trials were simulated with each utility used to select the sampling times. As this is a sequential design problem, data needed to be 'collected' once an optimal design for a given batch or block was found. This was facilitated by generating data from an assumed model; the parameter values of which were chosen as the modes of the prior distributions for a given model. Utility values under Bayesian A-optimality were recorded for all trials, and the chosen sampling times were also recorded. These results were then used to compare the utilities.

In the first example, we explore the proposed methods for approximating the likelihood in terms of the estimated likelihood weights and the optimal sampling times selected. The results from this investigation will be used to determine which approximation will be used in the subsequent simulation studies. In these studies, we use $N = 1000$ particles and $Q = 1000$ integration nodes for both examples. Further, R_m is set to 10 and 20 for the models introduced in the first and second examples, respectively.

Throughout these examples, the Q evaluations of the conditional likelihood for the approximation of Eq. (1) were performed on a GPU. For a given approximation, this requires evaluating the conditional likelihood Q times for each of the N particles yielding $N \times Q$ times in total, for each model considered. With the number of particles N being in the interval $10^2 - 10^5$ and Q the number of conditional likelihood evaluations per particle of the order of 10^3, it is obvious that there is substantial gain in computational speed by evaluating these likelihoods in parallel. This approach is implemented in CUDA (NVIDIA 2012) via the MATLAB CUDA API (MATLAB 2013). The complete SMC algorithm is implemented in MATLAB with a `mex` function written in CUDA to evaluate the approximation to the likelihood using the GPU. This implementation is compared to C code compiled and called in a similar manner. The resulting code for implementing these approaches is available upon request.

5.1 Example 1: Comparison of approximations to the likelihood

Initially, we assumed that a one-compartment infusion model was appropriate to describe the metabolism of meropenem. This model can be described as follows:

$$g(\boldsymbol{\mu}_j, d_j) = \begin{cases} \frac{D}{Tinf}\frac{1}{k_j v_j}\left(1 - \exp\left(-k_j d_j\right)\right), & \text{for } d_j \le Tinf \\ \frac{D}{Tinf}\frac{1}{k_j v_j}\left(1 - \exp\left(-k_j Tinf\right)\right) & \\ \quad \exp\left(-k_j\left(d_j - Tinf\right)\right), & \text{otherwise} \end{cases}$$

where $(k_j, v_j) = \exp(\boldsymbol{\mu}_j)$.

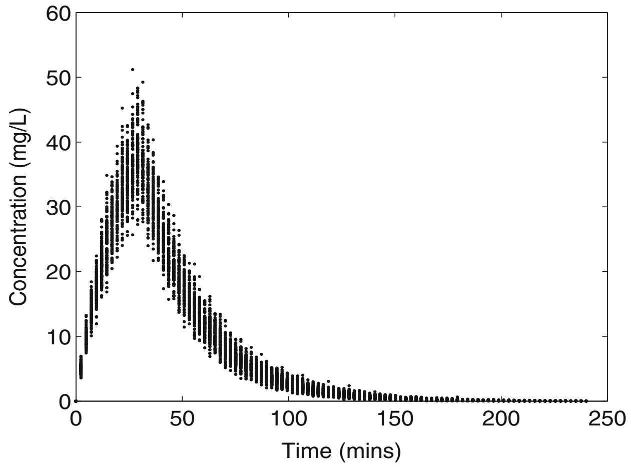


Fig. 1 Prior predictive plot for the one-compartment infusion model as described in Section 6.1 based on 500 simulations

Prior information about parameter values was based on the results from Ryan et al. (2014) with;

$$\boldsymbol{\delta} = (-3.26, 8.94),\ \boldsymbol{\Sigma} = 0.01\boldsymbol{I},\ \boldsymbol{\Omega} = \begin{bmatrix} 0.0071 & -0.0057 \\ -0.0057 & 0.0080 \end{bmatrix},$$

$$\boldsymbol{\Psi} = 0.02\boldsymbol{I},\ \nu = 5,\ a = -2.3,\ b = 0.5 \text{ and } \sigma = 0.1.$$

The prior predictive distribution under this model is shown in Fig. 1. One can observe that the maximum concentration is reached at 30 mins (the infusion time), and that the variability is largest here primarily because proportional residual variability was assumed. Further, it appears that the drug will have been eliminated from plasma 200 mins after the infusion has started.

We start by investigating the four approximations of the likelihood proposed in this paper. A trial of 10 sheep was simulated (with $N = 500$ and $Q = 1000$). Each time new data were observed, the likelihood weights from each of the four approximations were evaluated and recorded. These weights are shown in Fig. 2. The likelihood weights as given by all four approximations appear to agree well for the first 7 sheep, with some differences becoming apparent for the 8th, 9th and 10th sheep. There appears to be no differences between the likelihood weights for the two Laplace methods, in general, with some differences seen when considering MC and RQMC. We investigate these approximation methods further by comparing the estimated expected values of the Bayesian A-optimality utility for different designs.

To compare estimates of the expected Bayesian A-optimality utility, another trial of 10 sheep was simulated. Each time an optimal design needed to be determined, all four approximation methods were run and the expected Bayesian A-utility values for each proposed design were recorded. These expected utility values are shown in Fig. 3 for all four

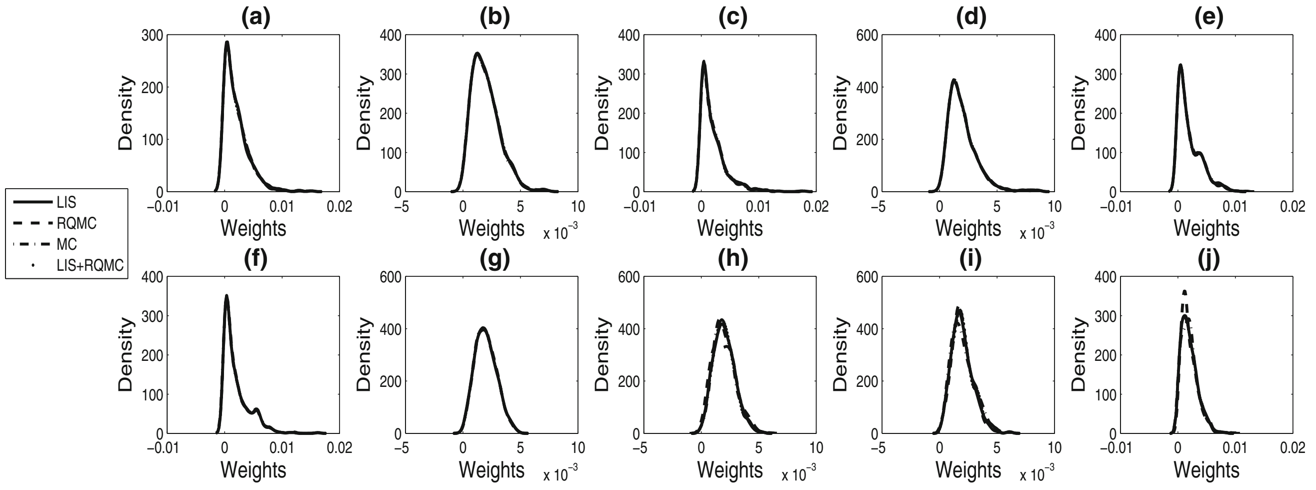


Fig. 2 Distribution of approximate likelihood weights as given by standard MC, RQMC, LIS and LIS+RQMC for a simulated trial on 10 sheep (plots (a) to (j) respectively) with $N = 500$ and $Q = 1000$

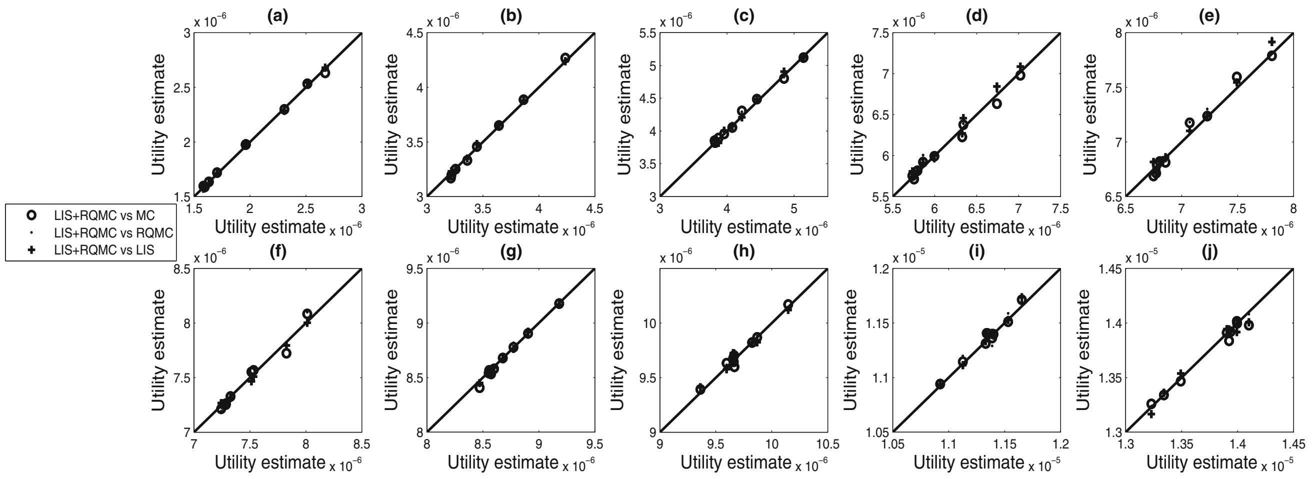


Fig. 3 Comparison of approximate expected utility values for Bayesian A-optimality as given by standard MC, RQMC, LIS and LIS+RQMC for a simulated trial on 10 sheep (plots (a) to (j) respectively) with $N = 500$ and $Q = 1000$

approximation methods. In general, there is a linear relationship between the expected utility values for all methods. Importantly, all four methods find the same design as being optimal for all 10 sheep. It seems for our purposes (that is, our design space, example, etc) that any of the considered methods could be used to select the optimal designs.

We now need to choose which approximation method to run in the examples that follow. In terms of choosing the optimal designs, all four methods appear similar. So then the choice between the methods will be based on the variability of the likelihood weights. To more comprehensively compare the variability of the likelihood weights between the methods, the ESS values of particle sets for the four approaches were compared. This involved running additional simulation studies of 500 trials each of 10 sheep with MC, RQMC, LIS and then LIS+RQMC being used to approximate the likelihood weights. However, in order for the comparison of ESS values to be deemed reasonable, the sampling times and subsequent data generated in the sequential design process were fixed. This means that the ESS values from each approximation method can be compared as they are based on the same target distributions. These comparisons are shown in Figs. 4a–d for MC compared to RQMC and 5a–d for MC compared to LIS, for $Q = 10$, 100, 500 and 1000. The plots for MC compared with LIS+RQMC are omitted as the results are similar to the comparison of MC and LIS.

From Fig. 4, we can see that overall there seems to be a one-to-one relationship between the ESS values from MC and RQMC, with the values become less variable as Q increases. Initially, there does not appear to be much of a difference between the ESS values. However, we investigated these values further by considering the number of times (out of 5000) RQMC gave an ESS value greater than MC. This yielded the following percentages 56.0, 55.9, 49.0 and 53.1 %

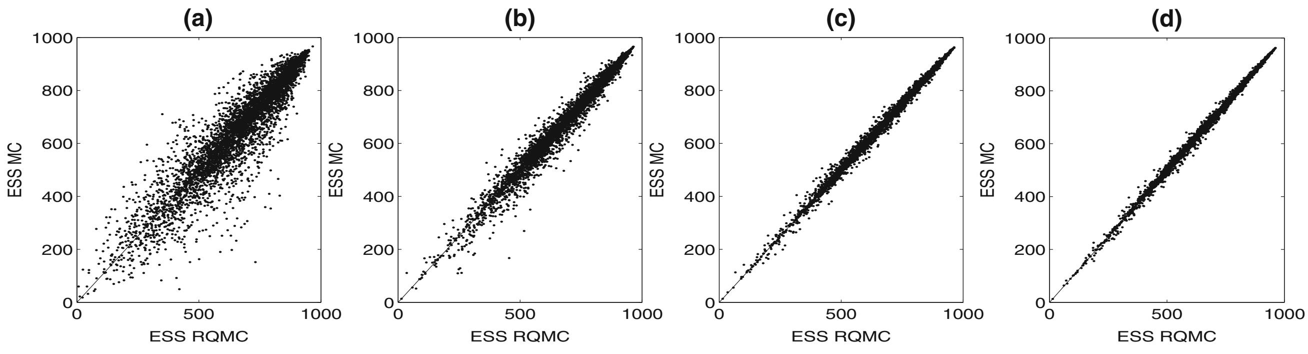


Fig. 4 Comparison of ESS as given by standard MC and RQMC for 500 simulated trials of 10 sheep with $N = 1000$ and $Q = 10, 100, 500$ and 1000 for *plots* (**a**)–(**d**), respectively

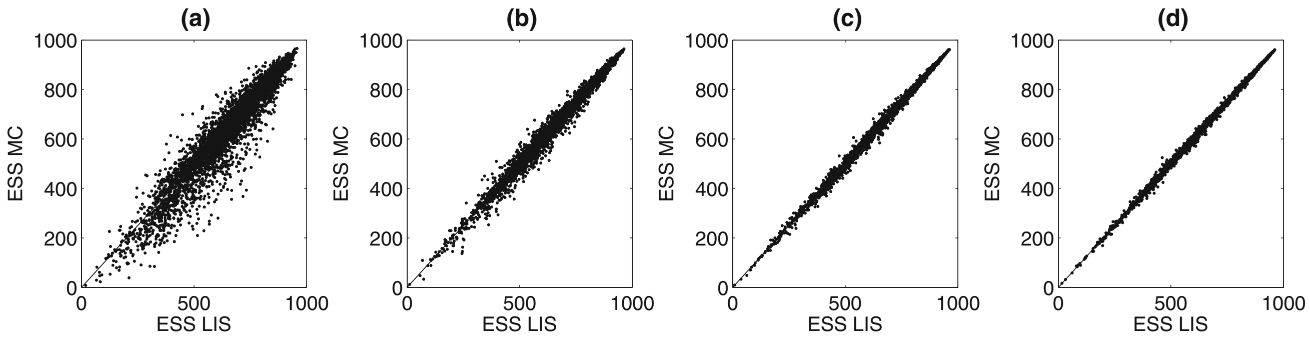


Fig. 5 Comparison of ESS as given by standard MC and LIS for 500 simulated trials of 10 sheep with $N = 1000$ and $Q = 10, 100, 500$ and 1000 for *plots* (**a**)–(**d**), respectively

for $Q = 10, 100, 500$ and 1000, respectively. The results are mixed but for $Q = 10, 100$ and 1000, RQMC gave better ESS values with the average (median) difference being 11.6 (6.0), 4.2 (1.86) and 0.8 (0.3), respectively. This suggests that there are potentially quite reasonable gains when using RQMC. Of course, when $Q = 500$, MC gave better ESS values more often. This suggests that the performance of RQMC may be implementation specific. In fact, performance also varies depending upon the value of v_k chosen in Algorithm 1. In our work, we arbitrarily selected $v_k = [0.1, 0.1]$, where v_k could in actual fact be chosen to minimize a measure of discrepancy (Dick et al. 2004; Munger et al. 2012). This could provide improved ESS values for RQMC.

From Fig. 5, again it appears that overall there is a one-to-one relationship between the ESS values. However, the percentage of times (out of 5000) LIS gave an ESS value greater than MC was 53.9, 50.7, 50.1 and 50.5 % for $Q = 10, 100, 500$ and 1000, respectively. Further, LIS gave better ESS values than MC with the average (median) difference being 13.0 (3.5), 1.4 (0.1), 0.5 (0.02) and 0.03 (0.04) for $Q = 10, 100, 500$ and 1000, respectively. When comparing RQMC and LIS, it appears that RQMC yields the smallest variability of the likelihood weights (when $Q = 10, 100$ and 1000). Therefore, this method will be used in the examples that follow. We note also that the computation time when using RQMC when compared to LIS is significantly reduced as the need to numerically find the mode a large number of times poses considerable computational burden. In fact, the computation time required for RQMC is only incrementally larger than that as given by standard MC.

5.2 Example 1 continued: Bayesian A-optimality for one compartment pharmacokinetic model

The Bayesian A-optimal sampling times for each of the 10 sheep over 500 simulated studies are shown in Fig. 6, where the figure shows the empirical probability distribution of the first of the two selected sampling times in each sheep. It is clear that early sampling times are prefered with the majority of sampling times being selected before 30 mins. Indeed, for the first 4 sheep, sampling times [6, 15] mins were selected 100 % of the time. Larger sampling times were selected for sheep near the end of the trial. Overall, it appeared that design selection was largely driven by the parameter which was most variable. In most cases, this was the parameter representing the volume of distribution.

Utility values for the 500 simulated trials for Bayesian A-optimality and the random design are shown in Fig. 7. It can be seen that there can potentially be over a 10 fold improvement in the (inverse of the) total or average variance

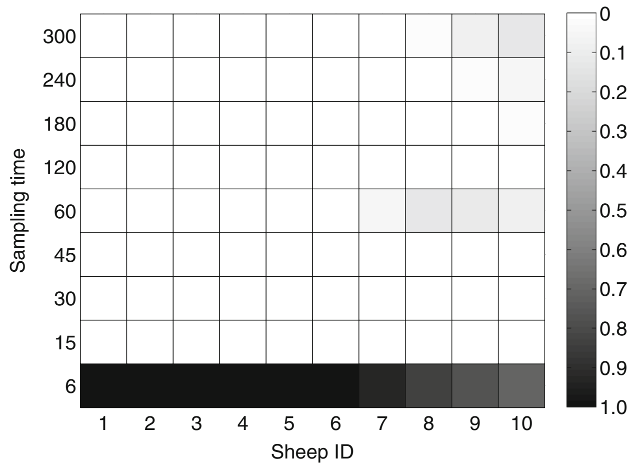


Fig. 6 Empirical probability distribution of sampling times selected in 500 simulated trials of 10 sheep for the one-compartment infusion model under Bayesian A-optimality. Note: Only the first sampling time has been plotted

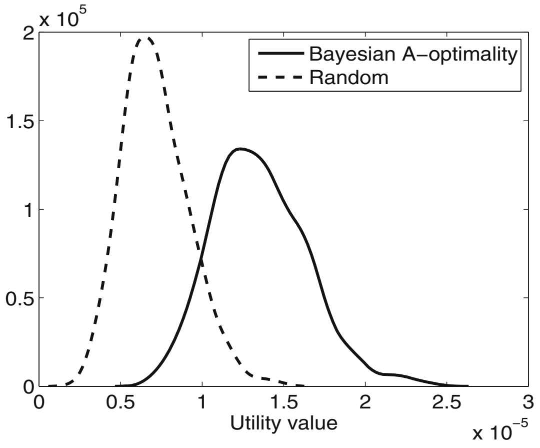


Fig. 7 Utility values for the 500 simulated trials for Bayesian A-optimality and the random design

of the parameters when using the Bayesian A-optimality as opposed to the random design. It can also be seen that there are occasions where the random design may have yielded a higher utility value than A-optimality. This is presumably because, given the small number of potential designs to use for data collection, by chance, the random design has selected sampling times that lead to precise parameter estimates. Further, varibility in the simulated data may also contribute to the occurrence of such instances. Notably, this overlap is only seen in about the lower 50th percentile of the utility values for the Bayesian A-optimality utility function suggesting that this utility is in general outperforming random selection.

5.3 Example 1 continued: Comparison of run times

Of interest are the run times for evaluating the likelihood under different implementations. These are shown for implementations in C and CUDA (GPU) for this example in Table 1 and Fig. 11a. The run times shown are for the evaluation of the likelihood, averaged over 25 evaluations. The results shown in Fig. 11 are just the log base 10 of those shown in Table 1. These runs were performed on a Windows desktop computer with an Intel(R) Xeon(R) CPU E5-1620 0 @ 3.60 GHz processor and a NVIDIA Quadro 2000 1GB GPU. Further, we note that the evaluation of the likelihood requires the generation of random numbers which can contribute significantly to run times. In all run times shown, the required random numbers were generated before the timed likelihood call. In regards to the implementations, it is believed that the comparisons between the C and CUDA implementations are true representations of what can be gained through using a GPU. This is because the CUDA implementation is essentially the same C code but compiled to run on a GPU. Hence, it is essentially the hardware that is being compared.

From Table 1 and Fig. 11:

– The CUDA code runs about 18 times faster than the C code;
– There is a roughly linear increase in computing time with respect to Q for the GPU implementation;
– Increases in time are not linear with respect to N for the GPU implementation. For example, there is not a huge increase in computing time between $N = 100$ and $N = 1000$. Moreover, in general, there is about a 10

Table 1 Average run times (s) for the evaluation of the likelihood under different implementations for data from a single block for the one compartment and two compartment infusion models

Implementation	$Q = 1000$			$Q = 2000$			$Q = 4000$		
	$N = 100$	$N = 1000$	$N = 10000$	$N = 100$	$N = 1000$	$N = 10000$	$N = 100$	$N = 1000$	$N = 10000$
Example 1									
C (serial)	0.0218	0.2221	2.2451	0.0457	0.4432	4.4381	0.0878	0.8858	8.8913
CUDA (parallel)	0.0107	0.0137	0.1263	0.0210	0.0273	0.2470	0.0430	0.0541	0.4964
Example 2									
C (serial)	0.0291	0.2873	2.9043	0.0584	0.5721	5.8276	0.1174	1.1702	11.6148
CUDA (parallel)	0.0213	0.0269	0.2386	0.0422	0.0537	0.4751	0.0813	0.1074	0.9705

fold increase in run times when $N = 100$ compared to $N = 10{,}000$;

– For the C implementations, increases in time are roughly linear with N and Q;

The reduction in computing times presented when implementing the C and CUDA code is quite reasonable, with the maximum benefits for using CUDA coming for the largest N and Q, which is the case where computational time is most expensive.

5.4 Example 2: Bayesian A-optimality for one and two compartment pharmacokinetic models

Consider an example where there is uncertainty around the form of $g(\boldsymbol{\mu}_j, d_j)$. As well as the model in the first example, the following two-compartment infusion model was also contemplated:

$$g(\boldsymbol{\mu}_j, d_j) = \begin{cases} \frac{D}{Tinf}\left[\frac{A_j}{\alpha_j}(1-\exp(-\alpha_j d_j)) + \frac{B_j}{\kappa_j}(1-\exp(-\kappa_j d_j))\right], & d_j \le Tinf \\ \frac{D}{Tinf}\left[\frac{A_j}{\alpha_j}(1-\exp(-\alpha_j Tinf))\exp(-\alpha_j(d_j - Tinf)) + \frac{B_j}{\kappa_j}(1-\exp(-\kappa_j Tinf))\exp(-\kappa_j(d_j - Tinf))\right], & \text{otherwise} \end{cases},$$

for $(k_j, k_{12j}, k_{21j}, v_j) = \exp(\boldsymbol{\mu}_j)$, where

$$A_j = \frac{1}{v_j}\frac{\alpha_j - k_{21j}}{\alpha_j - \kappa_j}, \quad B_j = \frac{1}{v_j}\frac{\kappa_j - k_{21j}}{\kappa_j - \alpha_j},$$
$$\alpha_j = \frac{k_{21j}k_j}{\kappa_j},$$
$$\kappa_j = \frac{1}{2}\left[k_{12j} + k_{21j} - k_j - \sqrt{\left(k_{12j} + k_{21j} + k_j\right)^2 - 4k_{21j}k_j}\right].$$

Results obtained from a two compartment analysis of data from Ryan et al. (2014) were used to give the prior distribution values:

$$\boldsymbol{\delta} = (-2.502, 0.8326, 0.6563, 8.225), \boldsymbol{\Sigma} = 0.01\boldsymbol{I},$$
$$\boldsymbol{\Omega} = \begin{bmatrix} 0.0120 & -0.0012 & 0.0012 & 0.0018 \\ -0.0012 & 0.0085 & 0.0002 & -0.0051 \\ 0.0012 & 0.0002 & 0.0104 & 0.0046 \\ 0.0018 & -0.0051 & 0.0046 & 0.0195 \end{bmatrix},$$
$$\boldsymbol{\Psi} = 0.02\boldsymbol{I}, \nu = 7, a = -2.3, b = 0.5 \text{ and } \sigma = 0.1.$$

The prior predictive distribution of this two-compartment model is shown in Fig. 8. This appears similar in shape to the one-compartment model with the peak concentration at 30 mins after the infusion has started. The additional compartment in this model yields the characteristic kink in the tail, and, in this case, there appears to be more variability around the typical response when compared to the one-compartment model.

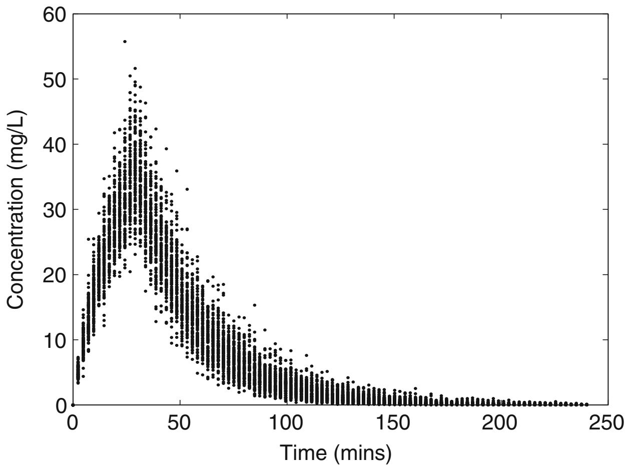


Fig. 8 Prior predictive *plot* for two-compartment infusion model as described in Section 6.4 based on 500 simulations

The Bayesian A-optimal sampling times for each of the 10 sheep over 500 simulated studies are shown in Fig. 9 for when the one-compartment model was supposed responsible for the sequential data generation. Again, the plot shows the empirical probability distribution of the first of the two

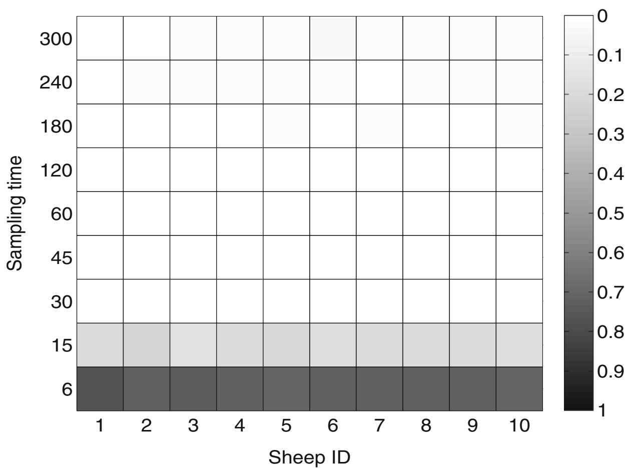


Fig. 9 Empirical probability distribution of sampling times selected in 500 simulated trials of 10 sheep for Example 2 under Bayesian A-optimality. Note: Only the first sampling time has been *plotted*

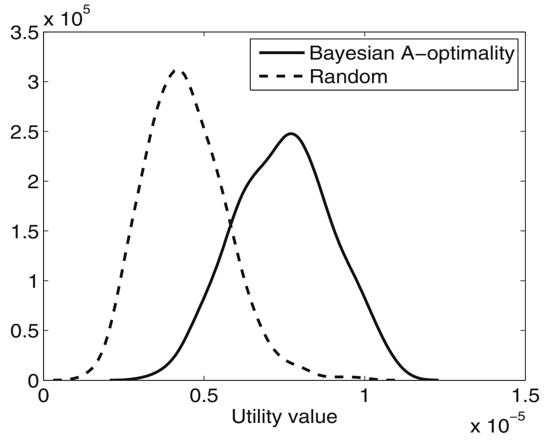


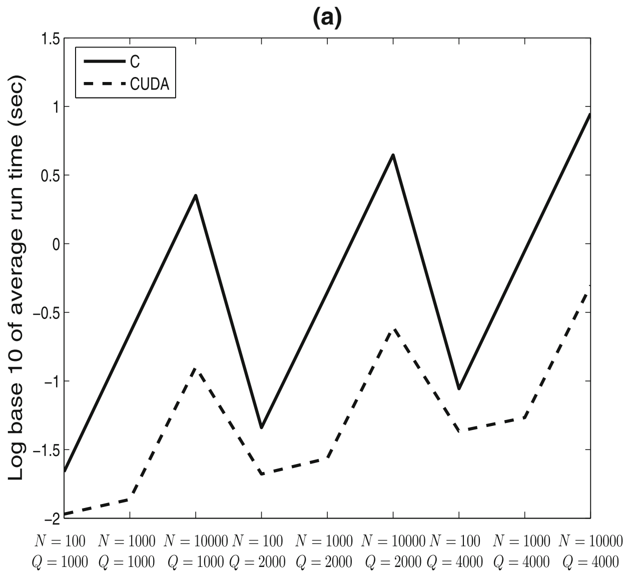
Fig. 10 Utility values for each simulated trial for Bayesian A-optimality and the random design for Example 2

selected sampling times in each sheep. There appears to be differences in the selected sampling times when one allows for the possibility of a two-compartment model being responsible for data generation. There is still a perference for early sampling times, however, there are more sampling times closer to the peak concentration and sampling times far beyond this point can be observed from the first sheep (rather than from the fourth sheep as seen in the first example). The different sampling times selected betwen the two examples could highlight potential sub-optimalities that may be observed if one were to simply design for a single model when model uncertainty exists.

Figure 10 displays density plots of utility values for each of the 500 completed trials for when the Bayesian A-optimality utility was used for design selection compared to the random design. Despite the introduction of more uncertainty, in particular around the true model, one can see that Bayesian A-optimality is generally performing better than the random design. There is potentially around a threefold improvement in the (inverse of the) total or average variance of the parameters when using the Bayesian A-optimality utility. Again, there are occassions where the random design may have performed better, and this is by chance. In comparison with Example 1, the distributions of utility values, as given by Bayesian A-optimality and the random design, appear to overlap more. Presumably, this is a consequence of having more variability to deal with when constructing the designs.

This example was also re-run with the two compartment model used to generate the sequential data. This yielded similar results to those presented here when the one compartment model was used for data generation, but are omitted. This suggests that the Bayesian A-optimality utility is selecting designs that are robust to model uncertainty. Indeed, this is how the utility was constructed to perform. Robustness to such uncertainty is obviously an important characteristic of an optimal design, and our methodology extends straightforwardly to the consideration of more than two models for data generation.

Table 1 and Fig. 11b show similar run time results for the two compartment model when compared with the one compartment model, with the benefits of CUDA when compared to C reduced slightly. In this example, the CUDA code is only roughly ten times faster than the C implementation. This reduced improvement may be due to the increased model complexity, or more precisely the larger size of the



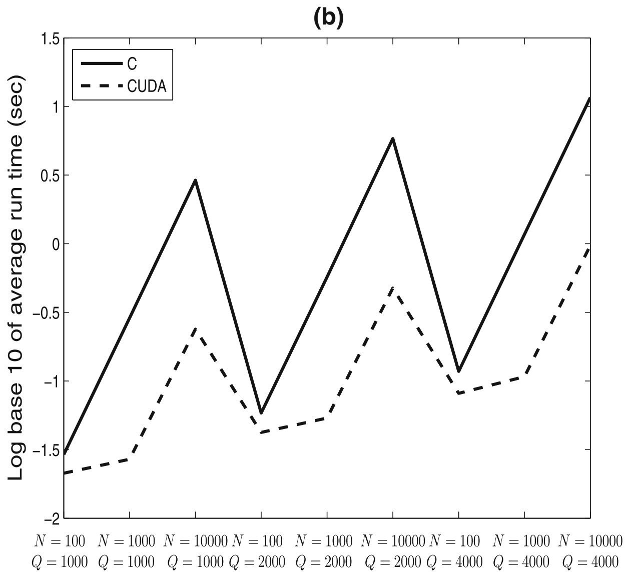


Fig. 11 Log base 10 of average computing times (s) for evaluating the likelihood under different implementations in: **a** Example 1 and **b** Example 2

instructions set required to compute the likelihood for the two compartment model. The GPU consists of a large number of light-weight processors with limited memory or register space to store instructions. When the size of the instruction set exceeds a certain limit, the GPU is no longer able to use all its processors at once, reducing the number of available processors for parallel operations; in turn reducing the computational advantages of the GPU.

6 Conclusion

In this paper, we have proposed a new pseudo-marginal SMC algorithm for sequentially designing experiments that yield batch or block data in the presence of model and parameter uncertainty when the likelihood is not analytic and has to be approximated. Our work was motivated by the need for an efficient Bayesian inference and design algorithm where data are observed sequentially, and the SMC algorithm served this purpose well. Our developments of a new pseudo-marginal algorithm have extended the use of SMC to random effects models, where we can achieve efficient estimates of important statistics such as the model evidence. With respect to implementation, a nice feature of the idealised and our new SMC algorithm is that it can be implemented in either a coarse-grained way or a fine-grained way. Thus with little effort, it can perform well on both multi-core CPUs or GPUs. In our work, computational efficiency was achieved via the use of a GPU when evaluating the approximation of the likelihood for a given model. This GPU implementation was up to 18 times faster than the C implementation, and made this research possible in a reasonable amount of time. We argue that the run time comparisons between C and CUDA are reasonably fair comparisons as the code and computing language for running both implementations is essentially the same.

We considered standard MC, RQMC, LIS and LIS+RQMC integration techniques for approximating the likelihood. All approaches produced comparable results for design selection, however, differences were observed in the estimated likelihood weights. These differences were explored further where it was found that, under certain implementations, RQMC provides the larger ESS values overall. As such, RQMC was proposed for use in the examples, and we note that this method is generally faster to implement when compared with LIS and LIS+RQMC. However, other approaches may prove more useful here. Indeed, in terms of optimal experimental design, it may not be necessary to consider an exact-approximate algorithm for design selection. For example, one could consider deterministic approaches for fast posterior approximations such as those given by an integrated-nested Laplace approximation (Rue et al. 2009) or variational methods (Beal 2003). We plan to investigate this in future research.

We considered an important PK study in sheep being treated with ECMO, and results suggested that prolonged use of ECMO may not be required for the estimation of PK parameters. Reduced trial lengths may provide more ethical studies while not compromising experimental results.

In our PK examples, the experimental aims reflected precise parameter estimation with the possibility of model uncertainty but other aims (and therefore utility functions) may be of interest. For example, experimenters may be more interested in determining the form of the model, that is, model discrimination, and a utility function based on mutual information has been considered in previous research for this purpose (Drovandi et al. 2014). However, implementation was shown to be computationally challenging, even for fixed effects models. We note that the methodology presented here can be applied to find designs for this purpose within a mixed effects setting, and is an avenue for future research. It may also be of interest to not only derive designs for model discrimination, but also for the precise estimation of parameters. Dual objective or compound utility functions could be considered for this purpose.

A limitation of our implementation is the discretisation of the design space. On-going research in this area is in the consideration of Gaussian Processes to model the expected utility surface/s (Overstall and Woods 2015). Such models are known to be powerful nonlinear interpolation tools, and therefore could prove useful here. The choice of design points to 'observe' the expected utility is a design problem within itself. Of further interest would be the choice of covariance function.

We would also like to mention that there are further opportunities to reduce run times. In particular, running the actual SMC (inference) algorithm in parallel would certainly significantly reduce computing times. Moreover, as proposed designs are independent, then the design selection phase of the algorithm could be run in parallel (for example, one thread per proposed design). This may also prove useful in overcoming the limitation of discretising the design space. One could also potentially improve run times by adopting a different kernel in the move step. For example, Liu and West (2001) propose a kernel which, by accepting all proposed parameters, preserves the first two moments of the target distribution. In our implementation, this would therefore reduce our move step iterations from R_m to one, and has been used in sequential design previously (Azadi et al. 2014). This approximation would work well if the posterior distributions were well approximated by mixtures of Gaussian distributions. The number of likelihood evaluations could also be reduced by considering a Markov kernel (Del Moral et al. 2006). For this kernel, the normalising constant needs to be estimated yielding an $O(N^2)$ algorithm (as opposed to the $O(N)$ algorithm proposed here). However, using this kernel would reduce the number of evaluations of the likelihood by a factor of R_m. The Liu and West (2001) kernel could be corrected for non-Gaussian posteriors by this approach or using a Metropolis-Hastings step.

Acknowledgments This work was supported by the Australian Research Council Centre of Excellence for Mathematical & Statistical Frontiers. The work of A.N. Pettitt was supported by an ARC Discovery Project (DP110100159), and the work of J.M. McGree was supported by an ARC Discovery Project (DP120100269). We would also like to thank the two referees who offered helpful comments to improve the article.

References

Amzal, B., Bois, F.Y., Parent, E., Robert, C.P.: Bayesian-optimal design via interacting particle systems. J. Am. Stat. Assoc. **101**, 773–785 (2006)

Andrieu, C., Roberts, G.O.: The pseudo-marginal approach for efficient Monte Carlo computations. Ann. Stat. **37**, 697–725 (2009)

Atkinson, A.C., Donev, A.N., Tobias, R.D.: Optimum experimental designs, with SAS. Oxford University Press Inc., New York (2007)

Azadi, N.A., Fearnhead, P., Ridall, G., Blok, J.H.: Bayesian sequential experimental design for binary response data with application to electromyographic experiments. Bayesian Anal. **9**, 287–306 (2014)

Beal, M.J.: *Variational algorithms for approximate inference*. Ph.D. thesis. University of London (2003)

Bernardo, J.M., Smith, A.: Bayesian Theory. Wiley, Chichester (2000)

Chopin, N.: A sequential particle filter method for static models. Biometrika **89**, 539–551 (2002)

Chopin, N., Jacob, P., Papaspiliopoulos, O.: SMC^2: an efficient algorithm for sequential analysis of state space models. J. R. Stat. Soc. **75**, 397–426 (2013)

Corana, A., Marchesi, M., Martini, C., Ridella, S.: Minimizing multimodal functions of continuous variables with the 'simulated annealing' algorithm. ACM Trans. Math. Softw. **13**, 262–280 (1987)

Cranley, R., Patterson, T.: Randomisation of number theoretic methods for multiple integration. SIAM J. Numer. Anal. **13**, 904–914 (1976)

Davies, A., Jones, D., Bailey, M., Beca, J., Bellomo, R., Blackwell, N., Forrest, P., Gattas, D., Granger, E., Herkes, R., Jackson, A., McGuinness, S., Nair, P., Pellegrino, V., Pettilä, V., Plunkett, B., Pye, R., Torzillo, P., Webb, S., Wilson, M., Ziegenfuss, M.: Extracorporeal membrane oxygenation for 2009 influenza A(H1N1) acute respiratory distress syndrome. J. Am. Med. Assoc. **302**, 1888–1895 (2009)

Del Moral, P., Doucet, A., Jasra, A.: Sequential Monte Carlo samplers. J. R. Stat. Soc. **68**, 411–436 (2006)

Dick, J., Sloan, I., Wang, X., Wozniakowski, H.: Liberating the weights. J. Complex. **5**, 593–623 (2004)

Drovandi, C.C., McGree, J.M., Pettitt, A.N.: Sequential Monte Carlo for Bayesian sequentially designed experiments for discrete data. Comput. Stat. Data Anal. **57**, 320–335 (2013)

Drovandi, C.C., McGree, J.M., Pettitt, A.N.: A sequential Monte Carlo algorithm to incorporate model uncertainty in Bayesian sequential design. J. Comput. Gr. Stat. **23**, 3–24 (2014)

Durham, G., Geweke, J.: Massively parallel sequential Monte Carlo for Bayesian inference. Manuscript, URL http://www.censoc.uts.edu.au/pdfs/gewekepapers/gpworking9.pdf (2011)

Faure, H.: Discrkpance de suites associkes b un systkme de numeration (en dimension s). Acta Arith. **XLI**, 337–351 (1982)

Gerber, M., Chopin, N.: Sequential-quasi Monte Carlo. Eprint arXiv:1402.4039 (2014)

Gilks, W.R., Berzuini, C.: Following a moving target—Monte Carlo inference for dynamic Bayesian models. J. R. Stat. Soc. **63**, 127–146 (2001)

Gordon, N.J., Salmond, D.J., Smith, A.: Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In: IEE Proceedings F Radar and Signal Processing, vol. **140**, pp. 107–113 (1994)

Gramacy, R.B., Polson, N.G.: Particle learning of Gaussian process models for sequential design and optimization. J. Comput. Gr. Stat. **20**, 102–118 (2011)

Halton, J.H.: On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. Numer. Math. **2**, 84–90 (1960)

Hammersley, J.M., Handscomb, D.C.: Monte Carlo Methods. Methuen & Co Ltd, London (1964)

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

Hastings, W.K.: Monte carlo sampling methods using markov chains and their applications. Biometrika **57**, 97–109 (1970)

Kiefer, J.: Optimum experimental designs (with discussion). J. R. Stat. Soc. **21**, 272–319 (1959)

Kitagawa, G.: Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. J. Comput. Gr. Stat. **5**, 1–25 (1996)

Kuk, A.: Laplace importance sampling for generalized linear mixed models. J. Stat. Comput. Simul. **63**, 143–158 (1999)

L'Ecuyer, P., Lemieux, C.: Variance reduction via lattice rules. Manag. Sci. **46**, 1214–1235 (2000)

Liu, J., West, M.: Combined parameter and state estimation in simulation based filtering. In: Doucet, A., de Freitas, J.F.G., Gordon, N.J. (eds.) Sequential Monte Carlo in Practice, pp. 197–223. Springer, New York (2001)

Liu, J.S., Chen, R.: Blind deconvolution via sequential imputations. J. Am. Stat. Assoc. **90**, 567–576 (1995)

Liu, J.S., Chen, R.: Sequential Monte Carlo for dynamic systems. J. Am. Stat. Assoc. **93**, 1032–1044 (1998)

MATLAB.: *version 8.2.0.701 R2013b*. The Mathworks Inc, Natick, MA (2013)

McGree, J.M., Drovandi, C.C., Thompson, M.H., Eccleston, J.A., Duffull, S.B., Mengersen, K., Pettitt, A.N., Goggin, T.: Adaptive Bayesian compound designs for dose finding studies. J. Stat. Plan. Inference **142**, 1480–1492 (2012)

Mentré, F., Mallet, A., Baccar, D.: Optimal design in random-effects regression models. Biometrika **84**, 429–442 (1997)

Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E.: Equation of state calculations by fast computing machines. J. Chem. Phys. **21**, 1087–1092 (1953)

Morokoff, W.J., Caflisch, R.E.: Quasi-Monte Carlo integration. J. Comput. Phys. **122**, 218–230 (1995)

Müller, P.: Simulation-based optimal design. Bayesian Stat. **6**, 459–474 (1999)

Munger, D., L'Ecuyer, P., Bastin, F., Cirillo, C., Tuffin, B.: Estimation of the mixed logit likelihood function by randomized quasi-Monte Carlo. Transp. Res. Part B **46**, 305–320 (2012)

Niederreiter, H.: Quasi-Monte Carlo methods and pseudo-random numbers. Bull. Am. Math. Soc. **84**, 957–1041 (1978)

NVIDIA.: *NVIDIA CUDA C Programming Guide 4.1*. NVIDIA (2012)

Overstall, A., Woods, D.: The approximate coordinate exchange algorithm for Bayesian optimal design of experiments arXiv:1501.00264v1 [stat.ME] (2015)

Owen, A.B.: Monte Carlo variance of scrambled net quadrature. SIAM J. Numer. Anal. **34**, 1884–1910 (1997a)

Owen, A.B.: Scramble net variance for integrals of smooth functions. Ann. Stat. **25**, 1541–1562 (1997b)

Owen, A.B.: Latin supercube sampling for very high-dimensional simulations. ACM Trans. Model. Comput. Simul. **8**, 71–102 (1998a)

Owen, A.B.: Scrambling Sobol' and Niederreiter-Xing points. J. Complex. **14**, 466–489 (1998b)

Patterson, H.D., Hunter, E.A.: The efficiency of incomplete block designs in National List and Recommended List cereal variety trials. J. Agric. Sci. **101**, 427–433 (1983)

Rue, H., Martino, S., Chopin, N.: Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations (with discussion). J. R. Stat. Soc. **71**, 319–392 (2009)

Ryan, E., Drovandi, C., McGree, J., Pettitt, A.: A review of modern computational algorithms for Bayesian optimal design. Int. Stat. Rev. Accepted for publication (2015)

Ryan, E., Drovandi, C., Pettitt, A.: Fully Bayesian experimental design for Pharmacokinetic studies. Entropy **17**(3), 1063–1089 (2014)

Shekar, K., Roberts, J., Smith, M., Fung, Y., Fraser, J.: The ECMO PK project: an incremental research approach to advance understanding of the pharmacokinetic alterations and improve patient outcomes during extracorporeal membrane oxygenation. BMC Anesthesiol. **13**, 7(2013)

Sobol', I.M.: On the distribution of points in a cube and the approximate evaluation of integrals. USSR Comput. Math. Math. Phys. **7**, 86–112 (1967)

Stroud, J.R., Müller, P., Rosner, G.L.: Optimal sampling times in population pharmacokinetic studies. J. R. Stat. Soc. Ser. C **50**, 345–359 (2001)

Tran, M.N., Strickland, C., Pitt, M.K., Kohn, R.: Annealed important sampling for models with latent variables. arXiv:1402.6035 [stat.ME] (2014)

Vergé, C., Dubarry, C., Del Moral, P., Moulines, E.: On parallel implementation of sequential Monte Carlo methods: the island particle model. Statistics and Computing. To appear (2015)

Weir, C.J., Spiegelhalter, D.J., Grieve, A.P.: Flexible design and efficient implementation of adaptive dose-finding studies. J. Biopharm. Stat. **17**, 1033–1050 (2007)

Woods, D.C., van de ven, P.: Blocked designs for experiments with correlated non-normal response. Technometrics **53**, 173–182 (2011)