Efficient inference for stochastic differential equation mixed-effects models using correlated particle pseudo-marginal algorithms

Citation for the original published paper (version of record):
Wiqvist, S., Golightly, A., McLean, A. et al (2021). Efficient inference for stochastic differential equation mixed-effects models using correlated
particle pseudo-marginal algorithms. Computational Statistics and Data Analysis, 157.
http://dx.doi.org/10.1016/j.csda.2020.107151

Computational Statistics and Data Analysis 157 (2021) 107151

Contents lists available at ScienceDirect

Computational Statistics and Data Analysis

journal homepage: www.elsevier.com/locate/csda

Efficient inference for stochastic differential equation mixed-effects models using correlated particle pseudo-marginal algorithms

Samuel Wiqvist [a,*], Andrew Golightly [b], Ashleigh T. McLean [b], Umberto Picchini [c]

[a] Centre for Mathematical Sciences, Lund University, Sweden
[b] School of Mathematics, Statistics and Physics, Newcastle University, UK
[c] Mathematical Sciences, Chalmers University of Technology and the University of Gothenburg, Sweden

ARTICLE INFO

Article history:
Received 1 March 2020
Received in revised form 22 November 2020
Accepted 25 November 2020
Available online 8 December 2020

Keywords:
Bayesian inference
Random effects
Sequential Monte Carlo
State-space model

ABSTRACT

Stochastic differential equation mixed-effects models (SDEMEMs) are flexible hierarchical models that are able to account for random variability inherent in the underlying time-dynamics, as well as the variability between experimental units and, optionally, account for measurement error. Fully Bayesian inference for state-space SDEMEMs is performed, using data at discrete times that may be incomplete and subject to measurement error. However, the inference problem is complicated by the typical intractability of the observed data likelihood which motivates the use of sampling-based approaches such as Markov chain Monte Carlo. A Gibbs sampler is proposed to target the marginal posterior of all parameter values of interest. The algorithm is made computationally efficient through careful use of blocking strategies and correlated pseudo-marginal Metropolis–Hastings steps within the Gibbs scheme. The resulting methodology is flexible and is able to deal with a large class of SDEMEMs. The methodology is demonstrated on three case studies, including tumor growth dynamics and neuronal data. The gains in terms of increased computational efficiency are model and data dependent, but unless bespoke sampling strategies requiring analytical derivations are possible for a given model, we generally observe an efficiency increase of one order of magnitude when using correlated particle methods together with our blocked-Gibbs strategy.

1. Introduction

Stochastic differential equations (SDEs) are arguably the most used and studied stochastic dynamic models. SDEs allow the representation of stochastic time-dynamics, and are ubiquitous in applied research, most notably in finance (Steele, 2012), systems biology (Wilkinson, 2018), pharmacokinetic/pharmacodynamic modeling (Lavielle, 2014) and neuronal modeling. SDEs extend the possibilities offered by ordinary differential equations (ODEs), by allowing random dynamics. As such, they can in principle replace ODEs in practical applications, to offer a richer mathematical representation for complex phenomena that are intrinsically non-deterministic. However, in practice switching from ODEs to SDEs

* Corresponding author.
E-mail address: samuel.wiqvist@matstat.lu.se (S. Wiqvist).

https://doi.org/10.1016/j.csda.2020.107151

is usually far from trivial, due to the absence of closed form solutions to SDEs (except for the simplest toy problems), implying the need for numerical approximation procedures (Kloeden and Platen, 1992). Numerical approximation schemes, while useful for simulation purposes, considerably complicate statistical inference for model parameters. For reviews of inference strategies for SDE models, see e.g. Fuchs (2013) (including Bayesian approaches) and Sørensen (2004) (classical approaches). Generally, in the non-Bayesian framework, the literature for parametric inference approaches for SDEs is vast, however there is no inference procedure that is applicable to general nonlinear SDEs and that is also easy to implement on a computer. This is due to the lack of explicit transition densities for most SDE models. The problem is particularly difficult for measurements that are observed without error, i.e. Markovian observations. On the other hand, the Bayesian literature offers powerful solutions to the inference problem, when observations arise from state-space models. In our case, this means that if we assume that observations are observed with error, and that the latent process is a Markov process, then the literature based on sequential Monte Carlo (particle filters) is readily available in the form of pseudo-marginal methods (Andrieu and Roberts, 2009), and closely related particle MCMC methods (Andrieu et al., 2010), which we introduce in Section 4.

Our goal is to produce novel Gibbs samplers embedding special types of pseudo-marginal algorithms allowing for exact Bayesian inference in a specific class of state-space SDE models. In this paper, we consider "repeated measurement experiments", modeled via mixed-effects, where the dynamics are Markov processes expressed via stochastic differential equations. These dynamics are assumed directly unobservable, i.e. are only observable up to measurement error. The practical goal is to fit observations pertaining to several "units" (i.e. independent experiments, such as measurements on different subjects) simultaneously, by formulating a state-space model having parameters randomly varying between the several individuals. The resulting model is typically referred to as a *stochastic differential equation mixed-effects model* (SDEMEM). SDEMEMs are interesting because, in addition to explaining intrinsic stochasticity in the time-dynamics, they also take into account random variation between experimental units. The latter variation permits the understanding of between-subjects variability within a population. When considered in a state-space model, these two types of variability (population variation and intrinsic stochasticity) are separated from the third source of variation, namely residual variation (measurement error). Thanks to their generality, and the ability to separate the three levels of variation, SDEMEMs have attracted attention, see e.g. Donnet and Samson (2013a) for a review and Whitaker (2016) for a more recent account. See also Section 2 for a discussion on previous literature.

In the present work, we mainly focus on a general, *plug-and-play* approach for exact Bayesian inference in SDEMEMs, meaning that analytic calculations are not necessary thanks to the flexibility of the underlying sequential Monte Carlo (SMC) algorithms. We also describe a non plug-and-play approach to handle specific situations. As in Picchini and Forman (2019), our random effects and measurement error can have arbitrary distributions, provided that the measurement error density can be evaluated point-wise. Unlike (Picchini and Forman, 2019), we use a Gibbs sampler to target the marginal parameter posterior. Subject specific, common and random effect population parameters are updated in separate blocks, with pseudo-marginal Metropolis–Hastings (PMMH) steps used to update the subject specific and common parameters, and Metropolis–Hastings (MH) steps used to update the random effect population parameters. We believe that, to date, our work results in the most general plug-and-play approach to inference for state-space SDEMEMs (a similar method has been concurrently and independently introduced (July 25 2019 on arXiv), in Botha et al. (2020); see the discussion in Section 6). However, the price to pay for such generality is that the use of pseudo-marginal methods guided by SMC algorithms is computationally consuming. In order to make pseudo-marginal methods scale better as the number of observations is increased, we exploit recent advances based on correlated PMMH (CPMMH). We combine CPMMH with a novel blocking strategy and show that it is possible to reduce considerably the number of required particles, and hence reduce the computational requirements for exact Bayesian inference. In our experiments, unless specific models admit bespoke efficient sampling strategies (e.g. Section 5.3 where it was possible to implement an advanced particle filter), CPMMH based algorithms with our novel blocking strategy are one order of magnitude more efficient than standard PMMH. Occasionally we even observed a 40-fold increase in efficiency, as in Section 5.1.

The remainder of this paper is organized as follows. Background literature is discussed in Section 2. Stochastic differential mixed-effects models and the inference task are introduced in Section 3. Our proposed approach to inference is described in Section 4. Applications are considered in Section 5, including a simulation study considering an Ornstein–Uhlenbeck SDEMEM, a tumor-growth model and finally a challenging neuronal data case-study. A discussion is in Section 6. Julia and R codes can be found at https://github.com/SamuelWiqvist/efficient_SDEMEM.

2. Background literature

Here we rapidly review key papers on inference for SDEMEMs, and refer the reader to https://umbertopicchini.github.io/sdemem/ for a comprehensive list of publications. Early attempts at inference for SDEMEMs use methodology borrowed from standard (deterministic) nonlinear mixed-effects literature such as FOCE (first order conditional estimation) combined with the extended Kalman filter, as in Overgaard et al. (2005). This approach can only deal with SDEMEMs having a constant diffusion coefficient, see instead (Leander et al., 2015) for an extension to state-dependent diffusion coefficients. The resulting inference in Overgaard et al. (2005) is approximate maximum likelihood estimation, and no uncertainty quantification is given. Moreover, only Gaussian random effects are allowed and measurement error is also assumed Gaussian. Other maximum likelihood approaches are in Picchini et al. (2010), Picchini and Ditlevsen (2011),

where a closed-form series expansion for the unknown transition density is found using the method in Ait-Sahalia (2008), however the methodology can only be applied to reducible multivariate diffusions without measurement error. Donnet et al. (2010) discuss inference for SDEMEMs in a Bayesian framework. They implement a Gibbs sampler when the SDE (for each subject) has an explicit solution, and consider Gaussian random effects and Gaussian measurement error. When no explicit solution exists, they approximate the diffusion process using the Euler–Maruyama approximation. The approach of Donnet and Samson (2013b) is of particular interest, since it is the first attempt to employ particle filters for inference in SDEMEMs: they construct an exact maximum likelihood strategy based on stochastic approximation EM (SAEM), where latent trajectories are "proposed" via particle Markov chain Monte Carlo. The major problem with using SAEM is the need for sufficient summary statistics for the "complete likelihood", which makes the methodology essentially impractical for arbitrarily complex models. Delattre and Lavielle (2013) also use SAEM, but they avoid the need for the (usually unavailable) summary statistics for the complete likelihood, and propose trajectories using the extended Kalman filter instead of particle MCMC. Unlike in Donnet and Samson (2013b), the inference in Delattre and Lavielle (2013) is approximate and measurement error and random effects are required to be Gaussian. Ruse et al. (2019) analyze multivariate diffusions under the conditions that the random effects are Gaussian distributed and that both fixed parameters and random effects enter linearly in the SDE. Whitaker et al. (2017) work with the Euler–Maruyama approximation and adopt a data augmentation approach to integrate over the uncertainty associated with the latent diffusion process, by employing carefully designed bridge constructs inside a Gibbs sampler. A linear noise approximation (LNA) is also considered. However, the limitations are that the observation equation has to be a linear combination of the latent states and measurement error has to be Gaussian. In addition, producing the bridge construct in the data augmentation approach or the LNA-based likelihood requires some careful analytic derivations. Consequently, neither approach can be regarded as a plug-and-play method (that is, a method that only requires forward simulation and evaluation of the measurement error density). In Picchini and Forman (2019), approximate and exact Bayesian approaches for a tumor growth study were considered: the approximate approach was based on synthetic likelihoods (Wood, 2010; Price et al., 2018), where summary statistics of the data are used for the inference, while exact inference used pseudo-marginal methodology via an auxiliary particle filter, which is suited to target measurements observed with a small error. It was found that using a particle approach to integrate out the random effects was very time consuming. Even though the data set was small (comprising 5–8 subjects to fit, depending on the experimental group, and around 10 observations per subject), the number of particles required to approximate each individual likelihood was in the order of thousands. This is very time consuming when the number of "subjects" (denoted M in the rest of this work) increases.

3. Stochastic differential mixed-effects models

Consider the case where we have M experimental units randomly chosen from a theoretical population. Our goal is to perform inference based on simultaneously fitting all data from the M units. Now assume that the experiment we are analyzing consists in observing a stochastically evolving dynamic process, and that associated with each unit i is a continuous-time d-dimensional Itô process $\{X_t^i, t \geq 0\}$ governed by the SDE

$$dX_t^i = \alpha(X_t^i, \kappa, \phi^i, D^i)\,dt + \sqrt{\beta(X_t^i, \kappa, \phi^i, D^i)}\,dW_t^i, \quad X_0^i = x_0^i, \quad i = 1, \ldots, M. \tag{1}$$

Here, α is a d-vector of drift functions, the diffusion coefficient β is a $d \times d$ positive definite matrix with a square root representation $\sqrt{\beta}$ such that $\sqrt{\beta}\sqrt{\beta}^T = \beta$, W_t^i is a d-vector of (uncorrelated) standard Brownian motion processes and D^i are unit-specific static or time-dependent deterministic input (e.g. covariates, forcing functions), see e.g. Leander et al. (2015). The p-vector parameter $\kappa = (\kappa_1, \ldots, \kappa_p)^T$ is common to all units whereas the q-vectors $\phi^i = (\phi_1^i, \ldots, \phi_q^i)^T$, $i = 1, \ldots, M$, are unit-specific random effects. In the most general random effects scenario we let $\pi(\phi^i|\eta)$ denote the joint distribution of ϕ^i, parameterized by the r-vector $\eta = (\eta_1, \ldots, \eta_r)^T$. The model defined by (1) allows for differences between experimental units through different realizations of the Brownian motion paths W_t^i and the random effects ϕ^i, accounting for inherent stochasticity within a unit, and variation between experimental units respectively.

We assume that each experimental unit $\{X_t^i, t \geq 0\}$ cannot be observed exactly, but observations $y^i = (y_1^i, \ldots, y_n^i)^T$ are available. Without loss of generality, we assume units are observed at the same integer time points $\{1, 2, \ldots, n\}$, that is in the following we write n instead of, say, n_i for all i. However this is only for convenience of notation, and we could easily accommodate the possibility of different units i having different values n_i and that, in turn, units are observed at different sets of times. The observations are assumed conditionally independent (given the latent process) and we link them to the latent process via

$$Y_t^i = h(X_t^i, S^i, \epsilon_t^i), \qquad \epsilon_t^i|\xi \overset{indep}{\sim} p_\epsilon(\xi), \qquad i = 1, \ldots, M \tag{2}$$

where Y_t^i is a d_o-vector, ϵ_t is a random d_o-vector, $d_o \leq d$, ϵ_t^i is the measurement noise, S^i is (as D^i) a unit-specific deterministic input, and $h(\cdot)$ is a possibly nonlinear function of its arguments. In the applications in Section 5 we have $D^i = S^i = \emptyset$, the empty set, for every i, and hence for simplicity of notation we disregard D^i and S^i in the rest of the paper. However having non-empty sets does not introduce any additional complication to our methodology. Notice, the possibility to have $d_0 < d$ implies that we may have some coordinate of the $\{X_t^i\}$ system that is unobserved at some (or all) t. We denote the density linking Y_t^i and X_t^i by $\pi(y_t^i|x_t^i, \xi)$. An important special case that arises from our flexible

observation model is when $h(X_t^i, \epsilon_t^i) = F^T X_t^i + \epsilon_t^i$ for a constant matrix F and $\epsilon_t^i | \Sigma \overset{indep}{\sim} N(0, \Sigma)$, allowing for observation of a linear combination of components of X_t^i, subject to additive Gaussian noise. Notice that our methodology in Sections 3.1–4.4 can be applied to an arbitrary $h(\cdot)$, provided this can be evaluated pointwise for any value of its arguments. For example, in Section 5.2 we have that $h(\cdot)$ is the logarithm of the sum of the components of a bivariate X_t^i.

We refer to the model constituted by the system (1)–(2) as a SDEMEM. This is a state-space model, due to the Markov property of the Itô processes $\{X_t^i, t \geq 0\}$, and the assumption of conditional independence of observations on latent processes. The model is flexible: equation (1) explains the intrinsic stochasticity in the dynamics (via β) and the variation between-units (via the random effects ϕ^i), while (2) explains residual variation (measurement error, via ξ).

3.1. Bayesian inference

Denote with $x = (x^1, \ldots, x^M)^T$ the set of unobserved states collected across all M diffusion processes $\{X_t^i\}$ at the same set of integer times $\{1, 2, \ldots, n\}$ as for data $y = (y^1, \ldots, y^M)^T$. Then given data $y = (y^1, \ldots, y^M)^T$, latent values x, the joint posterior for the common parameters κ, fixed/random effects $\phi = (\phi^1, \ldots, \phi^M)^T$, hyperparameters η and measurement error parameters ξ is

$$\pi(\kappa, \eta, \xi, \phi, x|y) \propto \pi(\kappa, \eta, \xi)\pi(\phi|\eta)\pi(x|\kappa, \phi)\pi(y|x, \xi) \tag{3}$$

where $\pi(\kappa, \eta, \xi)$ is the joint prior density ascribed to κ, η and ξ. These three parameters may be assumed *a priori* independent, and then we can write $\pi(\kappa, \eta, \xi) = \pi(\kappa)\pi(\eta)\pi(\xi)$, though this need not be the case and we can easily assume *a priori* correlated parameters. In addition we have that

$$\pi(\phi|\eta) = \prod_{i=1}^{M} \pi(\phi^i|\eta), \tag{4}$$

$$\pi(y|x, \xi) = \prod_{i=1}^{M}\prod_{j=1}^{n} \pi(y_j^i|x_j^i, \xi) \tag{5}$$

and

$$\pi(x|\kappa, \phi) = \prod_{i=1}^{M} \pi(x_1^i) \prod_{j=2}^{n} \pi(x_j^i|x_{j-1}^i, \kappa, \phi^i). \tag{6}$$

Note that $\pi(x_j^i|x_{j-1}^i, \kappa, \phi^i)$ will be typically intractable. In this case, we assume that it is possible to generate draws (up to arbitrary accuracy) from $\pi(x_j^i|x_{j-1}^i, \kappa, \phi^i)$ using a suitable numerical approximation. For example, the Euler–Maruyama approximation of (1) is

$$\Delta X_t^i \equiv X_{t+\Delta t}^i - X_t^i = \alpha(X_t^i, \kappa, \phi^i)\,\Delta t + \sqrt{\beta(X_t^i, \kappa, \phi^i)}\,\Delta W_t^i$$

and therefore

$$X_{t+\Delta t}^i = X_t^i + \alpha(X_t^i, \kappa, \phi^i)\,\Delta t + \sqrt{\beta(X_t^i, \kappa, \phi^i)}\,\Delta W_t^i \tag{7}$$

where $\Delta W_t^i \sim N(0, I_d \Delta t)$ and the time-step Δt, which need not be the inter-observation time, is chosen by the practitioner to balance accuracy and efficiency.

In what follows, we assume that interest lies in the marginal posterior for all parameters, given by $\pi(\kappa, \eta, \xi, \phi|y) = \int \pi(\kappa, \eta, \xi, \phi, x|y)dx$, where

$$\pi(\kappa, \eta, \xi, \phi|y) \propto \pi(\kappa)\pi(\eta)\pi(\xi)\pi(\phi|\eta)\pi(y|\kappa, \xi, \phi) \tag{8}$$

$$\propto \pi(\kappa)\pi(\eta)\pi(\xi) \prod_{i=1}^{M} \pi(\phi^i|\eta)\pi(y^i|\kappa, \xi, \phi^i). \tag{9}$$

This factorization suggests a Gibbs sampler with separate blocks for each parameter vector that sequentially takes draws from the full conditionals

1. $\pi(\phi|\kappa, \eta, \xi, y) \propto \prod_{i=1}^{M} \pi(\phi^i|\eta)\pi(y^i|\kappa, \xi, \phi^i)$,
2. $\pi(\kappa|\eta, \xi, \phi, y) = \pi(\kappa|\phi, \xi, y) \propto \pi(\kappa) \prod_{i=1}^{M} \pi(y^i|\kappa, \xi, \phi^i)$,
3. $\pi(\xi|\kappa, \eta, \phi, y) = \pi(\xi|\kappa, \phi, y) \propto \pi(\xi) \prod_{i=1}^{M} \pi(y^i|\kappa, \xi, \phi^i)$,
4. $\pi(\eta|\kappa, \xi, \phi, y) = \pi(\eta|\phi) \propto \pi(\eta) \prod_{i=1}^{M} \pi(\phi^i|\eta)$.

Of course, in practice, the observed individual data likelihood $\pi(y^i|\kappa, \xi, \phi^i) = \int p(y^i, x^i|\kappa, \xi, \phi^i)dx^i$ will be intractable. In what follows, therefore, we consider a Metropolis-within-Gibbs strategy, and in particular introduce auxiliary variables u to allow pseudo-marginal Metropolis–Hastings updates.

4. A pseudo-marginal approach

Consider again the intractable target in (8) and suppose that we can unbiasedly estimate the intractable observed data likelihood $\pi(y|\kappa,\xi,\phi)=\int p(y,x|\kappa,\xi,\phi)dx$. To this end let

$$\hat{\pi}_u(y|\kappa,\xi,\phi)=\prod_{i=1}^{M}\hat{\pi}_{u^i}(y^i|\kappa,\xi,\phi^i)$$

denote a (non-negative) unbiased estimator of $\pi(y|\kappa,\xi,\phi)$, where $u=(u^1,\ldots,u^M)^T$ is the collection of auxiliary (vector) variables used to produce the corresponding estimate, with density $\pi(u)=\prod_{i=1}^{M}g(u^i)$. In the context of inference for SDEs, the u may be the collection of pseudo-random standard Gaussian draws, these being necessary to simulate increments of the Brownian motion paths when implementing a numerical scheme such as Euler–Maruyama (Section 4.2), or produce draws from transition densities (in the rare instances when these are known). Notice in fact that the u need not have a specific distribution, though in stochastic simulation we need access to pseudo-random variates that are often uniform or Gaussian distributed (Devroye, 1986). When inference methods use particle filters, pseudo-random variates are also employed in the resampling step, and hence these variates can be included into u.

Now, the pseudo-marginal Metropolis–Hastings (PMMH) scheme targets

$$\pi(\kappa,\eta,\xi,\phi,u|y)\propto\pi(\kappa)\pi(\eta)\pi(\xi)\pi(\phi|\eta)\hat{\pi}_u(y|\kappa,\xi,\phi)\pi(u) \tag{10}$$

for which it is easily checked that

$$\begin{aligned}\int\pi(\kappa,\eta,\xi,\phi,u|y)du&\propto\pi(\kappa)\pi(\eta)\pi(\xi)\pi(\phi|\eta)\int\hat{\pi}_u(y|\kappa,\xi,\phi)\pi(u)du\\&\propto\pi(\kappa,\eta,\xi,\phi|y).\end{aligned}$$

Hence, marginalizing out u gives the marginal parameter posterior in (8). Directly targeting the high dimensional posterior $\pi(\kappa,\eta,\xi,\phi,u|y)$ with PMMH is likely to give very small acceptance rates. The structure of the SDMEM naturally admits a Gibbs sampling strategy. We outline our novel Gibbs samplers in the next section.

4.1. Gibbs sampling and blocking strategies

The form of (10) immediately suggests a Gibbs sampler that sequentially takes draws from the full conditionals. However, we can design two types of Gibbs samplers. Our first, novel strategy is denoted "naive Gibbs", where the u^i are updated with both the subject specific and common parameters.

Naive Gibbs:

1. $\pi(\phi^i,u^i|\kappa,\eta,\xi,y^i)\propto\pi(\phi^i|\eta)\hat{\pi}_{u^i}(y^i|\kappa,\xi,\phi^i)g(u^i)$, $i=1,\ldots,M$,
2. $\pi(\kappa,u|\eta,\xi,\phi,y,u)=\pi(\kappa,u|\phi,\xi,y)\propto\pi(\kappa)\prod_{i=1}^{M}\hat{\pi}_{u^i}(y^i|\kappa,\xi,\phi^i)g(u^i)$,
3. $\pi(\xi,u|\kappa,\eta,\phi,y,u)=\pi(\xi,u|\kappa,\phi,y)\propto\pi(\xi)\prod_{i=1}^{M}\hat{\pi}_{u^i}(y^i|\kappa,\xi,\phi^i)g(u^i)$,
4. $\pi(\eta|\kappa,\xi,\phi,y,u)=\pi(\eta|\phi)\propto\pi(\eta)\prod_{i=1}^{M}\pi(\phi^i|\eta)$.

Note that step 1 consists of a set of draws of M conditionally independent random variables since

$$\pi(\phi,u|\kappa,\eta,\xi,y)=\prod_{i=1}^{M}\pi(\phi^i,u^i|\kappa,\eta,\xi,y^i).$$

Hence, step 1 gives a sample from $\pi(\phi,u|\kappa,\eta,\xi,y)$. Draws from the full conditionals in 1–3 can be obtained by using Metropolis–Hastings within Gibbs. Taking the $[\phi^i,u^i]$ block as an example, we use a proposal density of the form $q(\phi^{i*}|\phi^i)g(u^{i*})$ and accept a move from $[\phi^i,u^i]$ to $[\phi^{i*},u^{i*}]$ with probability

$$\min\left\{1\,,\ \frac{\pi(\phi^{i*}|\cdot)}{\pi(\phi^i|\cdot)}\times\frac{\hat{\pi}_{u^{i*}}(y^i|\phi^{i*},\cdot)}{\hat{\pi}_{u^i}(y^i|\phi^i,\cdot)}\times\frac{q(\phi^i|\phi^{i*})}{q(\phi^{i*}|\phi^i)}\right\}.$$

Effectively, samples from the full conditionals in 1–3 are obtained via draws from pseudo-marginal MH kernels.

The above strategy is somewhat naive, since the auxiliary variables u need only be updated once per Gibbs iteration, instead in steps 1 to 3 of the naive Gibbs procedure vectors u^i are simulated anew in each of the three steps (notice $g(u^i)$ appears in each of the first three steps). We therefore propose to update the blocks $[\phi^i,u^i]$, $i=1,\ldots,M$ in step 1 only, and condition on the most recent value of u in the remaining steps. We call this second, novel strategy "blocked Gibbs".

Blocked Gibbs:

1. $\pi(\phi^i,u^i|\kappa,\eta,\xi,y^i)\propto\pi(\phi^i|\eta)\hat{\pi}_{u^i}(y^i|\kappa,\xi,\phi^i)g(u^i)$, $i=1,\ldots,M$,
2. $\pi(\kappa|\eta,\xi,\phi,y,u)=\pi(\kappa|\phi,\xi,y,u)\propto\pi(\kappa)\prod_{i=1}^{M}\hat{\pi}_{u^i}(y^i|\kappa,\xi,\phi^i)$,

3. $\pi(\xi|\kappa,\eta,\phi,y,u)=\pi(\xi|\kappa,\phi,y,u)\propto\pi(\xi)\prod_{i=1}^{M}\hat{\pi}_{u^i}(y^i|\kappa,\xi,\phi^i)$,
4. $\pi(\eta|\kappa,\xi,\phi,y,u)=\pi(\eta|\phi)\propto\pi(\eta)\prod_{i=1}^{M}\pi(\phi^i|\eta)$.

The aim of blocking in this way is to reduce the variance of the acceptance probability associated with steps 2 and 3, which involve the product of M estimates as opposed to a single estimate in each constituent part of step 1. Also, notice $g(u^i)$ appears only in the first step. The effect of blocking in this way is explored empirically in Section 5.

4.2. Estimating the likelihood

It remains that we can generate non-negative unbiased estimates $\hat{\pi}_u(y|\kappa,\xi,\phi)$. This can be achieved by running a sequential Monte Carlo procedure, also known as particle filter. The simplest approach is to use the bootstrap particle filter (Stewart and McCarty, 1992; Gordon et al., 1993) (see also Künsch, 2013) that, for a single experimental unit, recursively draws from the filtering distribution $\pi(x_t^i|y_{1:t}^i,\kappa,\xi,\phi^i)$ for each $t=1,\ldots,n$. Here, $y_{1:t}^i$ denotes the observations of experiment i for time-steps $1,\ldots,t$. Essentially, a sequence of importance sampling and resampling steps are used to propagate a weighted sample $\{(x_{t,k}^i, w(u_{t,k}^i)), k=1,\ldots,N_i\}$ from the filtering distribution, where N_i is the number of particles for unit i. Note that we let the weight depend explicitly on the tth component of the auxiliary variable $u^i=(u_1^i,\ldots,u_n^i)$, associated with experimental unit i. At time t, the particle filter uses the approximation

$$\hat{\pi}(x_t^i|y_{1:t}^i,\kappa,\xi,\phi^i)\propto\pi(y_t^i|x_t^i,\xi)\sum_{k=1}^{N_i}\pi(x_t^i|x_{t-1,k}^i,\kappa,\phi^i)w(u_{t-1,k}^i). \tag{11}$$

A simple importance sampling/resampling strategy follows, where particles are resampled (with replacement) in proportion to their weights, propagated via $x_{t,k}^i=f_t(u_{t,k}^i)\sim\pi(\cdot|x_{t-1,k}^i,\kappa,\phi^i)$ and reweighted by $p(y_t^i|x_{t,k}^i,\xi)$. Here, $f_t(\cdot)$ is a deterministic function of $u_{t,k}^i$ (as well as the parameters and previous latent state, suppressed for simplicity) that gives an explicit connection between the particles and auxiliary variables. An example of $f_t(\cdot)$ is to take the Euler–Maruyama approximation

$$f_t(u_{t,k}^i)=x_{t-1,k}^i+\alpha(x_{t-1,k}^i,\kappa,\phi^i)\,\Delta t+\sqrt{\beta(x_{t-1,k}^i,\kappa,\phi^i)\Delta t}\times u_{t,k}^i$$

where $u_{t,k}^i\sim N(0,I_d)$ and Δt is a suitably chosen time-step. In practice, unless Δt is sufficiently small to allow an accurate Euler–Maruyama approximation, $f_t(u_{t,k}^i)$ will describe recursive application of the numerical approximation.

Algorithm 1 Bootstrap particle filter for experimental unit i

Input: parameters κ, ϕ^i, ξ, auxiliary variables u^i, data y^i and the number of particles N_i.
Output: estimate $\hat{\pi}_{u^i}(y^i|\kappa,\xi,\phi^i)$ of the observed data likelihood.

1. Initialization ($t=1$).
   (a) **Sample** the prior. Put $x_{1,k}^i=f_1(u_{1,k}^i)\sim\pi(\cdot)$, $k=1,\ldots,N_i$.
   (b) **Compute** the weights. For $k=1,\ldots,N_i$ set
   $$\tilde{w}(u_{1,k}^i)=\pi(y_1^i|x_{1,k}^i,\xi),\qquad w(u_{1,k}^i)=\frac{\tilde{w}(u_{1,k}^i)}{\sum_{j=1}^{N_i}\tilde{w}(u_{1,j}^i)}.$$
   (c) **Update** observed data likelihood estimate. Compute $\hat{\pi}_{u_1^i}(y_1^i|\kappa,\xi,\phi^i)=\sum_{k=1}^{N_i}\tilde{w}(u_{1,k}^i)/N_i$.
2. For times $t=2,3,\ldots,n$:
   (b') **(optional) Sorting.** Use Euclidean sorting on particles $\{x_{t-1,1}^i,\ldots,x_{t-1,N_i}^i\}$ if using CPMMH.
   (b) **Resample.** Obtain ancestor indices a_{t-1}^k, $k=1,\ldots,N_i$ using systematic resampling on the collection of weights $\{w(u_{t-1,1}^i),\ldots,w(u_{t-1,N_i}^i)\}$.
   (c) **Propagate.** Put $x_{t,k}^i=f_t(u_{t,k}^i)\sim\pi\big(\cdot\,|x_{t-1,a_{t-1}^k}^i,\kappa,\xi,\phi^i\big)$, $k=1,\ldots,N_i$.
   (d) **Compute** the weights. For $k=1,\ldots,N_i$ set
   $$\tilde{w}(u_{t,k}^i)=\pi(y_t^i|x_{t,k}^i,\xi),\qquad w(u_{t,k}^i)=\frac{\tilde{w}(u_{t,k}^i)}{\sum_{j=1}^{N_i}\tilde{w}(u_{t,j}^i)}.$$
   (e) **Update** observed data likelihood estimate. Compute
   $$\hat{\pi}_{u_{1:t}^i}(y_{1:t}^i|\kappa,\xi,\phi^i)=\hat{\pi}_{u_{1:t-1}^i}(y_{1:t-1}^i|\kappa,\xi,\phi^i)\hat{\pi}_{u_t^i}(y_t^i|y_{1:t-1}^i,\kappa,\xi,\phi^i)$$
   where $\hat{\pi}_{u_t^i}(y_t^i|y_{1:t-1}^i,\kappa,\xi,\phi^i)=\sum_{k=1}^{N_i}\tilde{w}(u_{t,k}^i)/N_i$.

Algorithm 1 provides a complete description of the bootstrap particle filter when applied to a single experimental unit. However notice the addition of a non-standard and optional sorting step 2b', which turns useful when implementing a correlated pseudo-marginal approach, as described in Section 4.3. For the resampling step we follow (Deligiannidis et al., 2018) among others and use systematic resampling (see e.g. Murray et al., 2016), which only requires simulating a single uniform random variable at each time point. It is straightforward to augment the auxiliary variable u^i to include the random variables used in the resampling step. As a by-product of the particle filter, the observed data likelihood $\pi(y^i|\kappa, \xi, \phi^i)$ can be estimated via the quantity

$$\hat{\pi}_{u^i}(y^i|\kappa, \xi, \phi^i) = N_i^{-n} \prod_{t=1}^{n} \sum_{k=1}^{N_i} \tilde{w}(u_{t,k}^i). \tag{12}$$

Moreover, the corresponding estimator can be shown to be unbiased (Del Moral, 2004; Pitt et al., 2012).

The full Gibbs sampler for generating draws from the joint posterior (10) is given by Algorithm 2. For ease of exposition, we have blocked the updates for κ and ξ, but note that the use of separate updates for these parameters is straightforward. The precise implementation of step 4 of the Gibbs sampler is likely to be example specific, and we anticipate that a direct draw of $\eta^{(j)} \sim \pi(\cdot|\phi^{(j)})$ will often be possible. For example when the components of ϕ are assumed to be normally distributed and η consists of the corresponding means and precisions, for which a semi-conjugate prior specification is possible, see Section 5.1.

Algorithm 2 Blocked Gibbs sampler

Input: Data y, initial parameter values ϕ, κ, ξ, η and number of iterations n_{iters}.
Output: $\{\phi^{(j)}, \kappa^{(j)}, \xi^{(j)}, \eta^{(j)}\}_{j=1}^{n_{\text{iters}}}$.

1. Initialize $\phi^{(0)} = (\phi^{1,(0)}, \ldots, \phi^{M,(0)})$, $\kappa^{(0)}$, $\xi^{(0)}$. Draw $u^{i,(0)} \sim g(\cdot)$ and run Algorithm 1 for $i = 1, \ldots, M$ with $u^{i,(0)}$, $\phi^{i,(0)}$, $\kappa^{(0)}$, $\xi^{(0)}$ and y^i to obtain $\hat{\pi}_{u^{i,(0)}}(y^i|\kappa^{(0)}, \xi^{(0)}, \phi^{i,(0)})$. Set the iteration counter $j = 1$.
2. Update subject specific parameters. For $i = 1, \ldots, M$:
   (a) Propose $u^{i*} \sim g(\cdot)$ and $\phi^{i*} \sim q(\cdot|\phi^{i,(j-1)})$.
   (b) Compute $\hat{\pi}_{u^{i*}}(y^i|\kappa^{(j-1)}, \xi^{(j-1)}, \phi^{i*})$ by running Algorithm 1 with u^{i*}, ϕ^{i*}, $\kappa^{(j-1)}$, $\xi^{(j-1)}$ and y^i.
   (c) With probability
   $$\min\left\{1, \frac{\pi(\phi^{i*}|\eta)}{\pi(\phi^{i,(j-1)}|\eta)} \times \frac{\hat{\pi}_{u^{i*}}(y^i|\kappa^{(j-1)}, \xi^{(j-1)}, \phi^{i*})}{\hat{\pi}_{u^{i,(j-1)}}(y^i|\kappa^{(j-1)}, \xi^{(j-1)}, \phi^{i,(j-1)})} \times \frac{q(\phi^{i,(j-1)}|\phi^{i*})}{q(\phi^{i*}|\phi^{i,(j-1)})}\right\} \tag{13}$$
   put $\phi^{i,(j)} = \phi^{i*}$ and $u^{i,(j)} = u^{i*}$. Otherwise, store the current values $\phi^{i,(j)} = \phi^{i,(j-1)}$ and $u^{i,(j)} = u^{i,(j-1)}$.
3. Update common parameters.
   (a) Propose $(\kappa^*, \xi^*) \sim q(\cdot|\kappa^{(j-1)}, \xi^{(j-1)})$.
   (b) Compute $\hat{\pi}_{u^{(j)}}(y|\kappa^*, \xi^*, \phi^{(j)}) = \prod_{i=1}^{M} \hat{\pi}_{u^{i,(j)}}(y^i|\kappa^*, \xi^*, \phi^{i,(j)})$ by running Algorithm 1 for $i = 1, \ldots, M$ with $u^{i,(j)}$, $\phi^{i,(j)}$, κ^*, ξ^* and y^i.
   (c) With probability
   $$\min\left\{1, \frac{\pi(\kappa^*)\pi(\xi^*)}{\pi(\kappa^{(j-1)})\pi(\xi^{(j-1)})} \times \frac{\hat{\pi}_{u^{(j)}}(y|\kappa^*, \xi^*, \phi^{(j)})}{\hat{\pi}_{u^{(j)}}(y|\kappa^{(j-1)}, \xi^{(j-1)}, \phi^{(j)})} \times \frac{q(\kappa^{(j-1)}, \xi^{(j-1)}|\kappa^*, \xi^*)}{q(\kappa^*, \xi^*|\kappa^{(j-1)}, \xi^{(j-1)})}\right\} \tag{14}$$
   put $(\kappa^{(j)}, \xi^{(j)}) = (\kappa^*, \xi^*)$. Otherwise, store the current values $(\kappa^{(j)}, \xi^{(j)}) = (\kappa^{(j-1)}, \xi^{(j-1)})$.
4. Update random effect population parameters. Draw $\eta^{(j)} \sim \pi(\cdot|\phi^{(j)})$.
5. If $j = n_{\text{iters}}$, stop. Otherwise, set $j := j + 1$ and go to step 2.

Executing Algorithm 2 requires $n \sum_{i=1}^{M} N_i$ draws from the transition density governing the SDE in (1) *per iteration*. In scenarios where the transition density is intractable, draws of a suitable numerical approximation are required. For example, we may use the Euler–Maruyama discretization with time step $\Delta t = 1/m$, where $m \geq 1$ is chosen to limit the associated discretization bias (and typically $m \gg 1$). In this case, order $mn \sum_{i=1}^{M} N_i$ draws of (7) are required. As discussed by Andrieu et al. (2010), the number of particles per experimental unit, N_i, should be scaled in proportion to the number of data points n. Consequently, the use of PMMH kernels is likely to be computationally prohibitive in practice. We therefore consider the adaptation of a recently proposed correlated PMMH method for our problem.

4.3. A correlated pseudo-marginal approach

Consider again the task of sampling the full conditional $\pi(\phi^i, u^i|\kappa, \eta, \xi, y^i)$ associated with the ith experimental unit. In steps 2(a–c) of Algorithm 2, a (pseudo-marginal) Metropolis–Hastings step is used whereby the auxiliary variables u^i are proposed from the associated pdf $g(\cdot)$ (notice we could introduce a subject-specific $g_i(\cdot)$, but we refrain from doing so in the interest of a lighter notation). As discussed by Deligiannidis et al. (2018) (see also Dahlin et al., 2015), the proposal

kernel need not be restricted to the use of $g(u^i)$. The correlated PMMH (CPMMH) scheme generalizes the PMMH scheme by generating a new u^{i*} from $K(u^{i*}|u^i)$ where $K(\cdot|\cdot)$ satisfies the detailed balance equation

$$g(u^i)K(u^{i*}|u^i) = g(u^{i*})K(u^i|u^{i*}). \tag{15}$$

It is then straightforward to show that a MH scheme with proposal kernel $q(\phi^{i*}|\phi^i)K(u^{i*}|u^i)$ and acceptance probability Eq. (13) satisfies detailed balance with respect to the target $\pi(\phi^i, u^i|\kappa, \eta, \xi, y^i)$.

We take $g(u^i)$ as a standard Gaussian density and $K(u^{i*}|u^i)$ as the kernel associated with a Crank–Nicolson proposal (Deligiannidis et al., 2018). Hence

$$g(u^i) = \mathrm{N}\left(u^i;\ 0\ ,\ I_d\right) \qquad \text{and} \qquad K(u^{i*}|u^i) = \mathrm{N}\left(u^{i*};\ \rho u^i\ ,\ \left(1-\rho^2\right) I_d\right)$$

where I_d is the identity matrix whose dimension d is determined by the number of elements in u^i. The parameter ρ is chosen to be close to 1, to induce strong positive correlation between $\hat{\pi}_{u^i}(y^i|\kappa, \Sigma, \phi^i)$ and $\hat{\pi}_{u^{i*}}(y^i|\kappa, \Sigma, \phi^{i*})$, thus reducing the variance of the acceptance probability in Eq. (13), which is beneficial because it reduces the chance of accepting an overestimation of the likelihood function. Taking $\rho = 0$ gives the special case that $K(u^{i*}|u^i) = g(u^{i*})$, which corresponds to the standard PMMH. Iteration j of step 2 of Algorithm 2 then becomes

2. For $i = 1, \ldots, M$:
   (a) Propose $\phi^{i*} \sim q(\cdot|\phi^{i,(j-1)})$. Draw $\omega \sim \mathrm{N}(0, I_d)$ and put $u^{i*} = \rho u^{i,(j-1)} + \sqrt{1-\rho^2}\omega$.
   (b) Compute $\hat{\pi}_{u^{i*}}(y^i|\kappa^{(j-1)}, \xi^{(j-1)}, \phi^{i*})$ by running Algorithm 1 with u^{i*}, ϕ^{i*}, $\kappa^{(j-1)}$, $\xi^{(j-1)}$ and y^i.
   (c) With probability given by Eq. (13) put $\phi^{i,(j)} = \phi^{i*}$ and $u^{i,(j)} = u^{i*}$. Otherwise, store the current values $\phi^{i,(j)} = \phi^{i,(j-1)}$ and $u^{i,(j)} = u^{i,(j-1)}$.

Care must be taken here when executing Algorithm 1 in Step 2(b). Upon changing ϕ^i and u^i, the effect of the resampling step is likely to prune out different particles, thus breaking the correlation between successive estimates of observed data likelihood. Sorting the particles before resampling can alleviate this problem (Deligiannidis et al., 2018). We follow Choppala et al. (2016) (see also Golightly et al., 2019) by using a simple Euclidean sorting procedure which, for the case of a 1-dimensional latent state (e.g. when $\dim(X_t^i) = 1$ for every t) implies, prior to resampling the particles, to sort the particles from the smallest to the largest. This is step 2b' in algorithm 1, denoted "optional" as it only applies to CPMMH, not PMMH.

4.4. Tuning the number of particles for likelihood approximation

It remains that we can choose the number of particles N_i to be used to obtain estimates of the observed data likelihood contributions $\hat{\pi}_{u^i}(y^i|\kappa, \xi, \phi^i)$. Note that we allow a different number of particles per experimental unit to accommodate differing lengths of the y^i and potential model misspecification at the level of an individual unit. In the case of PMMH, a simple strategy is to fix ϕ^i, κ and ξ at some central posterior value (obtained from a pilot run), and choose N_i so that the variance of the log-likelihood (denoted $\sigma^2_{N_i}$) is around 2 (Doucet et al., 2015; Sherlock et al., 2015). When using a CPMMH kernel, we follow (Tran et al., 2016a; Choppala et al., 2016) by choosing N_i so that $\sigma^2_{N_i} = 2.16^2/(1-\rho_l^2)$ where ρ_l is the estimated correlation between $\log \hat{\pi}_{u^i}(y^i|\kappa, \xi, \phi^i)$ and $\log \hat{\pi}_{u^{i*}}(y^i|\kappa, \xi, \phi^i)$. Hence, an initial pilot run (with the number of particles set at some conservative value) is required to determine plausible values of the parameters. This pilot run can also be used to give estimates of $var(\phi^i|y^i)$, $i = 1, \ldots, M$, each of which can subsequently be used as the innovation variance in a Gaussian random walk proposal for ϕ^i.

4.5. Tuning the proposal distributions

The block structure of the Gibbs sampler (Algorithm 2) requires two proposal densities: $\phi^{i*} \sim q(\cdot|\phi^{i,(j-1)})$ and $(\kappa^*, \xi^*) \sim q(\cdot|\kappa^{(j-1)}, \xi^{(j-1)})$ that have to be chosen to achieve an algorithm that efficiently explores the posterior parameter space.

In Sections 5.1 and 5.3 we employ the generalized Adaptive Metropolis (AM) algorithm (Andrieu and Thoms, 2008) to tune the two proposal distributions. Regarding the generation of proposals ϕ^{i*}, in the first step of the blocked Gibbs scheme we tune subject-specific proposal distributions, separately for each ϕ^{i*}. In addition to these M proposal distributions we also tune a proposal distribution for (κ^*, ξ^*). Thus, we automatically tune overall $M + 1$ proposal distributions via the generalized AM algorithm. Additionally, in Sections 5.1 and 5.3 we found that the use of different proposal distributions for each ϕ^{i*} was beneficial since random effects for the different subjects varied around very different values.

5. Applications

5.1. Ornstein–uhlenbeck SDEMEM

We consider the following Ornstein–Uhlenbeck (OU) SDEMEM

$$\begin{cases} Y_t^i &= X_t^i + \epsilon_t^i, \quad \epsilon_t^i \overset{indep}{\sim} \mathrm{N}(0, \sigma_\epsilon^2), \qquad i = 1, \ldots, M \\ dX_t^i &= \theta_1^i(\theta_2^i - X_t^i)dt + \theta_3^i dW_t^i. \end{cases} \tag{16}$$

Here $\theta_2^i \in \mathbb{R}$ is the stationary mean for the $\{X_t^i\}$ process, $\theta_1^i > 0$ is a growth rate (expressing how rapidly the system reacts to perturbations) and θ_3^i is the diffusion coefficient. The OU process is a standard toy-model in that it is completely tractable, that is the associated SDE has a known (Gaussian) transition density, e.g. Fuchs (2013). This fact, coupled with the assumption that the $Y_t^i|X_t^i$ are conditionally Gaussian and linear in the latent states, implies that we can apply the Kalman filter to evaluate the likelihood function exactly. Therefore, exact inference is possible for the OU SDEMEM (both maximum likelihood and Bayesian). For all units i we simulate $n = 200$ observations, with constant observational time-step Δt. In our setup, all random effects $(\theta_1^i, \theta_2^i, \theta_3^i)$ are assumed strictly positive, and therefore we work with their log-transformed version and set $\phi^i = (\log\theta_1^i, \log\theta_2^i, \log\theta_3^i)$, where

$$\phi_j^i|\eta \overset{indep}{\sim} N(\mu_j, \tau_j^{-1}), \qquad j = 1, 2, 3$$

and $\eta = (\mu_1, \mu_2, \mu_3, \tau_1, \tau_2, \tau_3)$, with τ_j the precision of ϕ_j^i. The SDEMEM (16) has no parameters κ that are shared among subjects, and the full set of parameters that we want to infer is $(\mu_1, \mu_2, \mu_3, \tau_1, \tau_2, \tau_3, \sigma_\epsilon)$.

As already mentioned, we can compute the likelihood $\pi(y|\phi, \sigma_\epsilon) = \prod_{i=1}^M \pi(y^i|\phi^i, \sigma_\epsilon)$ exactly, using a Kalman filter (see Tornøe et al., 2005 and Donnet and Samson, 2013a for a description pertaining SDEMEMs). The filter can then be used in Algorithm 2, that is we avoid using the particle filter (Algorithm 1) and replace it with the Kalman filter in Algorithm 2. Results from Algorithm 2 when using this approach are denoted with "Kalman". The transition density for the latent state is known and therefore we do not need to use an Euler–Maruyama discretization when propagating the states forward in the particle filter. Instead we propagate the particles using the simulation scheme induced by the exact transition density:

$$X_{t+\Delta t}^i = \theta_2^i + (X_t^i - \theta_2^i)e^{-\theta_1^i \Delta t} + \sqrt{\frac{\theta_3^{i2}}{2\theta_1^i}(1 - e^{-2\theta_1^i \Delta t})} \times u_t^i, \tag{17}$$

where $u_t^i \sim N(0, 1)$ independently for all t and all i. Clearly, the u_t^i appearing in (17) are among the variates that we will correlate, when implementing CPMMH, in addition to the variates produced in the resampling steps.

We compare "Kalman" to four further methods: "naive PMMH", where we employ Algorithm 2 with the naive Gibbs scheme (see Section 4.1), "PMMH", which is Algorithm 2, "CPMMH-099", which is Algorithm 2 with a Crank–Nicolson proposal for the u^i using a correlation of $\rho = 0.99$, and "CPMMH-0999" where we use a correlation of $\rho = 0.999$. The number of particle used for each method was selected using the methods described in Section 4.4. All five methods return exact Bayesian inference, and while this is obvious for "Kalman", we remind the reader that this holds also for the other four approaches as these are instances of the pseudo-marginal approach. Therefore, special interest is in efficiency comparisons between the last four algorithms, "Kalman" being the obvious gold-standard.

We simulated data from the model in (16) with the following settings (data are in Fig. 1): $M = 40$ experimental units, $n = 200$ observations for each unit using a time step $\Delta t = 0.05$, $\sigma_\epsilon = 0.3$, and $\eta = (\mu_1, \mu_2, \mu_3, \tau_1, \tau_2, \tau_3) = (-0.7, 2.3, -0.9, 4, 10, 4)$. The prior for the observational noise standard deviation σ_ϵ was set to a Gamma distribution $Ga(1, 0.4)$, and the priors for the η parameters were set to

$$\begin{cases} \mu_j|\tau_j \overset{indep}{\sim} \mathrm{N}(\mu_{0_j}, M_{0_j}\tau_j), \qquad j = 1, 2, 3, \\ \tau_j \overset{indep}{\sim} Ga(\alpha_j, \beta_j), \end{cases} \tag{18}$$

where,

$$\begin{aligned} &(\mu_{0_1}, M_{0_1}, \alpha_1, \beta_1) = (0, 1, 2, 1), \\ &(\mu_{0_2}, M_{0_2}, \alpha_2, \beta_2) = (1, 1, 2, 0.5), \\ &(\mu_{0_3}, M_{0_3}, \alpha_3, \beta_3) = (0, 1, 2, 1). \end{aligned}$$

The priors in are semi-conjugate and we can therefore use a tractable Gibbs step to sample η in step 4 of Algorithm 2. An extended introduction to the semi-conjugate prior, including the tractable posterior can be found in Murphy (2007).

We ran all four methods for 60k iterations, considering the first 10k iterations to be the burn-in period. We set the starting value for σ_ϵ at $\sigma_{\epsilon_0} = 0.2$, which is far from its ground truth value. The starting values for the random effects ϕ_j^i were set to their prior means. The proposal distributions were adaptively tuned using the generalized AM algorithm and the particle filters were implemented on a single-core computer, thus no parallelization was utilized. We used the

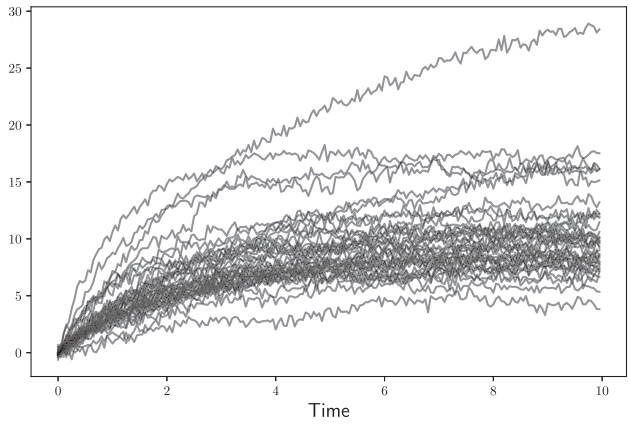

Fig. 1. Simulated data from the OU-SDEMEM model.

Table 1
OU SDEMEM. Correlation ρ, number of particles N, CPU time (in minutes m), minimum ESS (mESS), minimum ESS per minute (mESS/m) and relative minimum ESS per minute (Rel.) as compared to PMMH-naive. All results are based on 50k iterations of each scheme, and are medians over 5 independent runs of each algorithm on different data sets. We could only produce 5 runs due to the very high computational cost of PMMH.

Algorithm	ρ	N	CPU (m)	mESS	mESS/m	Rel.
Kalman	–	–	1.23	443.27	357.61	5140.18
PMMH-naive	0	3000	4601.87	229.01	0.05	1.00
PMMH	0	3000	4086.91	232.94	0.06	1.16
CPMMH-099	0.99	100	200.37	234.54	1.17	23.58
CPMMH-0999	0.999	50	110.88	235.63	2.13	41.48

same number of particles $N_i \equiv N$ for all units. Results are in Table 1 and Figs. 2–3. As a reference for the efficiency of the considered samplers, we take the minimum ESS per minute (mESS/m in Table 1) as measured on PMMH-naive as "base/default" value and set it to 1 in the rightmost column of Table 1. The minimum ESS per minute for the other samplers are relative to the PMMH-naive value. The mESS value is computed over all parameter chains (including individual random effects), i.e. the chains for ϕ, σ_ϵ and η. From Table 1 we conclude that CPMMH is about 20 to 40 times more efficient than PMMH in terms of mESS/m, depending on which correlation level we use. Furthermore, "Kalman" is about 5140 times more efficient than PMMH. However, the latter comparison is not very interesting since the Kalman filter can be applied only to a very restricted class of models. The marginal posteriors in Figs. 2–3 show that the several methods generate very similar posterior inference, which is reassuring. We left out the inference results from CPMMH-0999 for reasons of clarity. However we observed that with $N = 50$ CPMMH-0999 produces a slightly biased inference for σ_ϵ, due to failing to adequately mix over the auxiliary variable u, while inference for the remaining parameters is similar to the other considered methods. We verified (results not shown) that using $N = 100$ is enough to repair this problem. From Figs. 2–3 we can conclude that all parameters, with the possible exclusion of τ_2, are well inferred. Regarding τ_2, this is the precision for θ_2^i, the latter representing the stationary mean for a OU model. Clearly, by looking at Fig. 1, the occasional outlier in the upper part of the Figure may contribute to underestimating the true precision of the stationary mean. To check if CPMMH indeed is necessary, we tried to run PMMH with 100 particles (i.e., the same number of particles as for CPMMH-099). The inference results produced with PMMH with 100 particles gave considerable mismatch (in terms of posterior output) for both the η parameters and σ_ϵ relative to that obtained from CPMMH-099, resulting from the extremely poor mixing of the chain.

In summary, CPMMH is able to return reliable inference with a much smaller number of particles than PMMH, while resulting in a procedure that is about 20 to 40 times more efficient than PMMH (the 40-times figure is valid if we are ready to accept a small bias in σ_ϵ). Again, for most models exact inference based on a closed-form expression for the likelihood function is unavailable, therefore being able to obtain accurate inference using a computationally cheaper version of PMMH is very appealing.

Notice that while for this simple case study PMMH-naive has the same mESS than PMMH, this is not the case for the case study in Section 5.2, where using the blocked-Gibbs sampler produces a much larger mESS value compared to naive-Gibbs.

5.1.1. Investigating the choice of number of particles

A crucial problem when running methods based on particle filters is the selection of the number of particles N. In this section we investigate this problem by running CPMMH-099 and CPMMH-0999 with $N = [5, 10, 20, 50, 100]$ particles

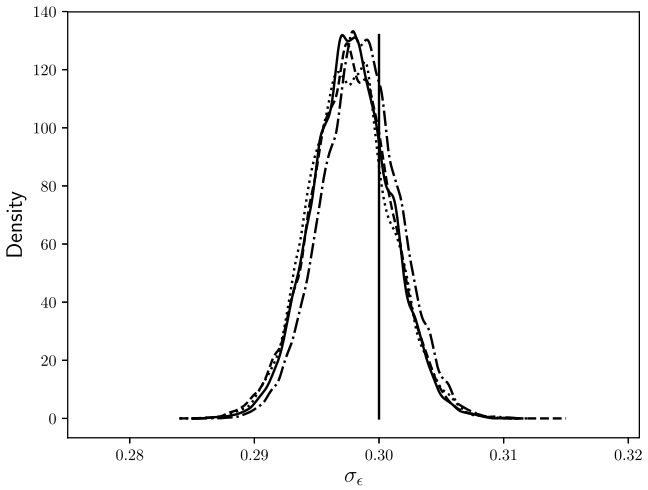

Fig. 2. OU SDEMEM: marginal posterior distributions for σ_ϵ. Solid line is Kalman, dashed line is PMMH-naive, dotted line is PMMH, dash-dotted line is CPMMH-099, vertical line is the ground truth.

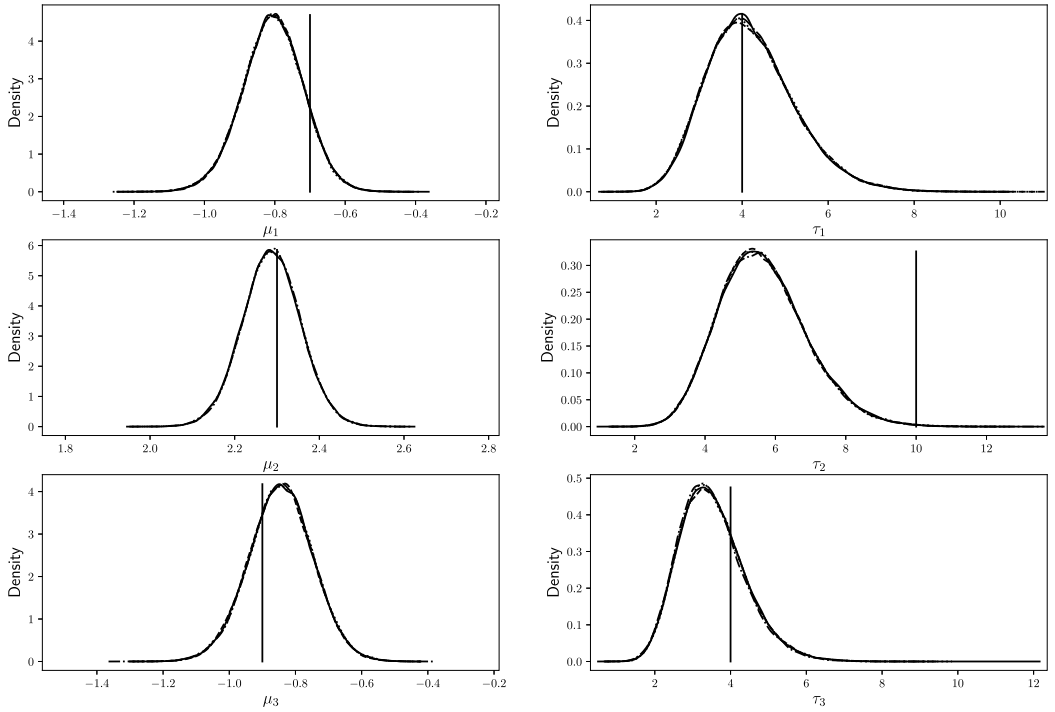

Fig. 3. OU SDEMEM: marginal posterior distributions for $\eta = (\mu_1, \mu_2, \mu_3, \tau_1, \tau_2, \tau_3)$. Solid line Kalman, dashed line PMMH-naive, dotted line PMMH, dash-dotted line CPMMH-099, vertical line ground truth.

using 25 different (simulated) data sets. We also ran the Kalman algorithm using the 25 different data sets for comparison purposes. In this analysis, we are only interested in investigating the quality and computational efficiency of the inference. Hence, we initialized all algorithms at the ground truth parameter values and ran each algorithm for 60k iterations, and discarding the first 10k iterations as burnin period. We first estimated the Wasserstein distance, between the marginal posteriors for σ_ϵ and η from the CPMMH algorithms and the corresponding Kalman-based marginal posteriors. This distance was computed via the POT package (Flamary and Courty, 2017) (we do not compute the Wasserstein distance for the marginal posterior of the random effects ϕ^i, since this is not of central interest for us). All Wasserstein distances are based on the last 5k samples of the corresponding chains. To obtain a performance measure that takes into account both the quality of the inference and the computational effort, we multiply the Wasserstein distances by the runtimes (in minutes) of the CPMMH algorithms, and obtain the performance measure *Wasserstein distance* $\times$ *runtime (m)*; see Figs. 4

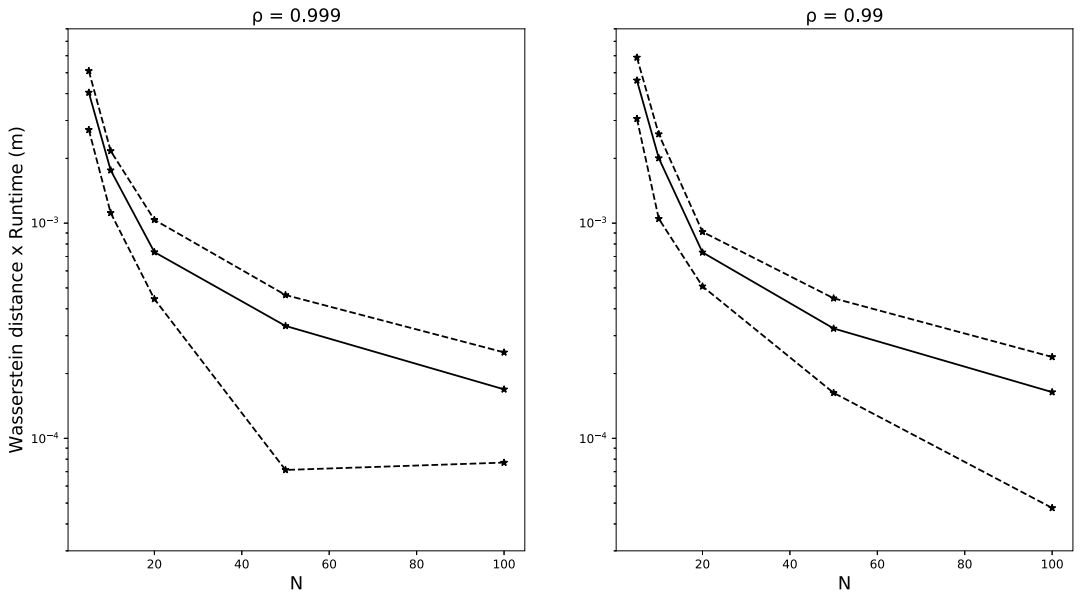

Fig. 4. OU SDEMEM: *Wasserstein distance* × *runtime (m)* performance measure for the marginal posterior of σ_ϵ, for several values of N and using $\rho = 0.999$ (left) and $\rho = 0.99$ (right). The solid line represents the mean value obtained from the 25 different data sets. The dashed confidence bands represent the 25th and 75th percentiles.

and 5. Smaller values of this measure are to be preferred as they indicate high computational efficiency and/or accurate inference. The reason for considering this performance measure is to take the quality of the inference into account, since for $N < 20$ we noticed that it is possible to obtain chains that do not indicate adequate convergence within a reasonable time-frame.

We can conclude that, on average, results for different correlation levels are similar. However, for σ_ϵ we obtain a better performance when using more particles (lower *Wasserstein distance* × *runtime (m)* value), this resulting from inaccurate inference for σ_ϵ when using too few ($N < 50$) particles, leading to a large Wasserstein distance. However, this is not the case for η since Fig. 5 shows that the performance is better with fewer particles, a result that we obtain since the inference for η is good even when using few particles (though not reported, in our analyses we observed that the Wasserstein distances for η are similar across all attempted values of N). Thus, if we want to infer the measurement noise parameter σ_ϵ accurately, in this case we will have to use $N \geq 50$ particles, while the inference for η is satisfactory, even with fewer particles.

Another issue that we analyze is the variability of mESS for the different data sets, based on 50k iterations of CPMMH. To investigate this we computed the 25th and 75th percentiles of mESS for CPMMH-099 with $N = 100$ and CPMMH-0999 with $N = 50$ based on the inference results on all unknown parameters from 25 simulated data sets. We obtain that the 25th and 75th percentiles of mESS for CPMMH-099 ($N = 100$) are $[227, 240]$, and for CPMMH-0999 ($N = 50$) are $[227, 252]$. Given that the several mESS are computed on different datasets, some degree of variation in the measure is expected and we conclude that the observed mESS variability is fairly small.

5.2. Tumor growth SDEMEM

We consider a stochastic differential mixed effects model that has been used to describe the tumor volume dynamics in mice receiving a treatment. Here we study a simplified version of the model in Picchini and Forman (2019), and is given by

$$\begin{aligned} dX^i_{1,t} &= \left(\beta^i + (\gamma^i)^2/2\right) X^i_{1,t} dt + \gamma^i X^i_{1,t} dW^i_{1,t} \\ dX^i_{2,t} &= \left(-\delta^i + (\psi^i)^2/2\right) X^i_{2,t} dt + \psi^i X^i_{2,t} dW^i_{2,t} \end{aligned} \tag{19}$$

for experimental units $i = 1, \ldots, M$. Here, $W_{1,t}$ and $W_{2,t}$ are uncorrelated Brownian motion processes, $X^i_{1,t}$ and $X^i_{2,t}$ are respectively the volume of surviving tumor cells and volume of cells killed by a treatment for mouse i. Let $V^i_t = X^i_{1,t} + X^i_{2,t}$ denote the total tumor volume at time t in mouse i. The observation model is given by

$$Y^i_t = \log V^i_t + \epsilon^i_t, \qquad \epsilon^i_t \overset{indep}{\sim} \mathrm{N}(0, \sigma_e^2). \tag{20}$$

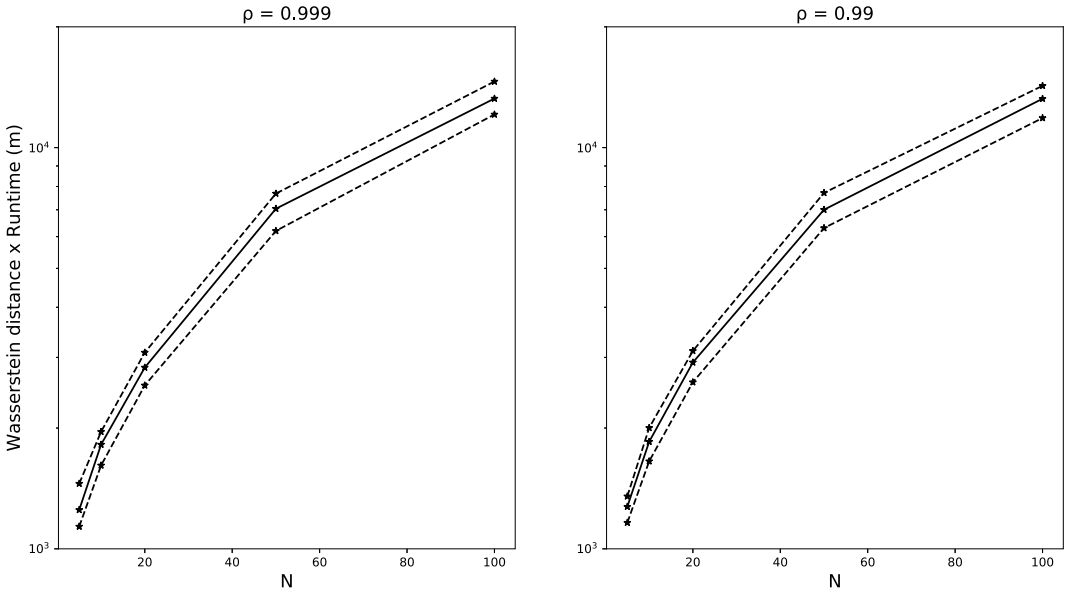

Fig. 5. OU SDEMEM: *Wasserstein distance* × *runtime (m)* performance measure for the marginal posterior of η, for several values of N and using $\rho = 0.999$ (left) and $\rho = 0.99$ (right). The solid line represents the mean value obtained from the 25 different data sets. The dashed confidence bands represent the 25th and 75th percentiles.

Let $\phi^i = (\log \beta^i, \log \gamma^i, \log \delta^i, \log \psi^i)$. We complete the SDEMEM specification via the assumption that

$$\phi^i_j|\eta \overset{indep}{\sim} \mathrm{N}(\mu_j, \tau_j^{-1}), \qquad j = 1, \ldots, 4 \tag{21}$$

so that $\eta = (\mu_1, \ldots, \mu_4, \tau_1, \ldots, \tau_4)$.

We recognize that $X^i_{1,t}$ and $X^i_{2,t}$ are geometric Brownian motion processes and (19) can be solved analytically to give

$$\begin{aligned} X^i_{1,t}|X^i_{1,0} = x^i_{1,0} &\sim logN\left(\log(x^i_{1,0}) + \beta^i t\,,\ (\gamma^i)^2 t\right) \\ X^i_{2,t}|X^i_{2,0} = x^i_{2,0} &\sim logN\left(\log(x^i_{2,0}) - \delta^i t\,,\ (\psi^i)^2 t\right) \end{aligned} \tag{22}$$

where $logN(\cdot, \cdot)$ denotes the log-Normal distribution. Despite the availability of a closed form solution to the underlying SDE model, the observed data likelihood is intractable, due to the nonlinear form of (20) as a function of $\log(X^i_{1,t} + X^i_{2,t})$. Nevertheless, a tractable approximation can be found, by linearizing $\log V^i_t$. The resulting linear noise approximation (LNA) is derived in B, and in what follows, we compare inference under the gold standard PMMH to that obtained under the LNA.

We mimicked the real data application in Picchini and Forman (2019) by generating 21 observations at integer times for $M = 10$ replicates. We took

$$\eta = (\log 0.29, \log 0.25, \log 0.09, \log 0.34, 10, 10, 10, 10)$$

and sampled $\phi^i_j|\eta$ using (21). The latent SDE process was then generated using (22) with an initial condition of $x^i_0 = (75, 75)^T$ (assumed known for all units), and each observation was corrupted according to (20) with $\sigma^2_e = 0.2$. The resulting data traces are consistent with the observations on total tumor volume of those subjects receiving chemo therapy in Picchini and Forman (2019) and can be seen in Fig. 6. We adopted semi conjugate, independent $\mathrm{N}(-2, 1)$ and $Ga(2, 0.2)$ priors for the μ_j and τ_j respectively. We took $\log \sigma_e \sim \mathrm{N}(0, 1)$ to complete the prior specification. Given the use of synthetic data of equal length for each experimental unit, we pragmatically took the number of particles as $N_i = N$, $i = 1, \ldots, 10$. Our choice of N was guided by the tuning advice of Section 4.4. For example, with CPMMH we obtain typical ρ_L values of around 0.75, when parameter values are fixed at an estimate of the posterior mean. This gives $\sigma^2_N = 10.6$ which is achieved with $N = 7$ particles. To avoid potentially sticky behavior of the chain in the posterior tails, we choose the conservative value $N = 10$. We compare four approaches: naive PMMH (where the u^i are updated with both the subject specific and common parameters), PMMH (where the u^i are only updated with the subject specific parameters – Algorithm 2), CPMMH (Algorithm 2 with a Crank–Nicolson proposal on the u^i) and the LNA based approach. We ran each scheme for $500k$ iterations. The results are summarized in Table 2 and Fig. 7.

Fig. 7 shows marginal posterior densities of the components of η. We see that inferences for these parameters are consistent with the true values that generated the data (with similar results obtained for the other parameters) and that inference via CPMMH is consistent with that from the gold-standard PMMH. Similar results are obtained for σ_ϵ (not shown

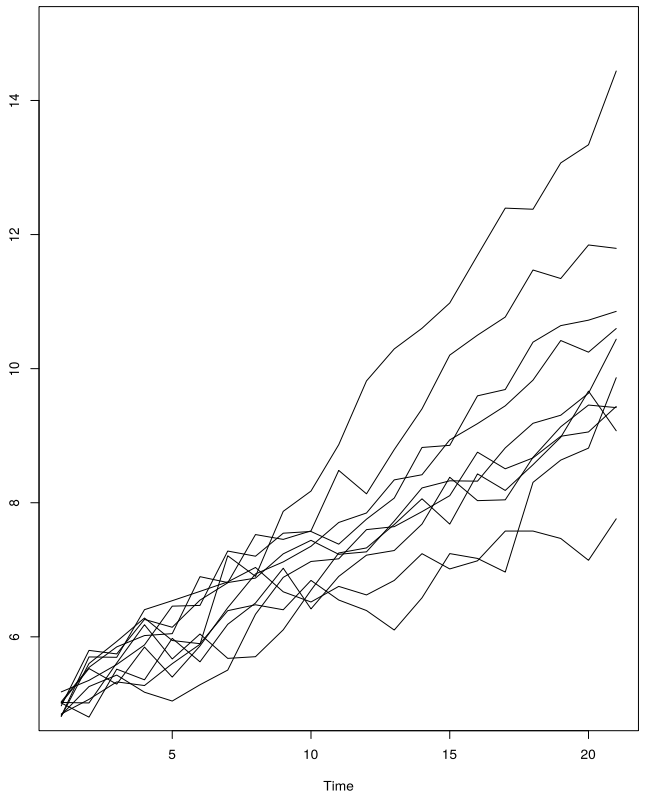

Fig. 6. Simulated data from the tumor growth model.

Table 2
Tumor model. Correlation ρ, number of particles N, CPU time (in minutes m), minimum ESS (mESS), minimum ESS per minute (mESS/m) and relative minimum ESS per minute (Rel.) as compared to PMMH-naive. All results are based on 500k iterations of each scheme.

Algorithm	ρ	N	CPU (m)	mESS	mESS/m	Rel.
LNA	–	–	1286	3676	2.858	13
PMMH - naive	0	30	3098	665	0.215	1
PMMH	0	30	2963	2559	0.864	4
CPMMH	0.999	10	957	2311	2.415	11

for brevity). At the same time, from Table 2 we note that CPMMH with $\rho = 0.999$ is about 11 times more efficient than the naive PMMH and almost 3 times more efficient than PMMH with additional blocking. Finally, the LNA-based approach provides an accurate alternative to PMMH, except for τ_4. However, everything considered, CPMMH is to be preferred here as its computational efficiency is comparable to LNA, but unlike the latter, CPMMH provides accurate inference for all parameters, and unlike LNA the CPMMH approach is plug-and-play.

5.2.1. Use of the Euler–maruyama approximation

We anticipate that for many applications of interest, an analytic solution of the underlying SDE will not be available. It is common place to use a numerical approximation in place of an intractable analytic solution. The simplest such approximation is the Euler–Maruyama (E–M) approximation. In this section, we investigate the effect of the E–M on the performance of PMMH and CPMMH for the tumor growth model.

The Euler–Maruyama approximation of (19) is

$$\Delta X^i_{1,t} = \left(\beta^i + (\gamma^i)^2/2\right) X^i_{1,t}\Delta t + \gamma^i X^i_{1,t}\Delta W^i_{1,t}$$
$$\Delta X^i_{2,t} = \left(-\delta^i + (\psi^i)^2/2\right) X^i_{2,t}\Delta t + \psi^i X^i_{2,t}\Delta W^i_{2,t}$$

where, for example, $\Delta X^i_{1,t} = X^i_{1,t+\Delta t} - X^i_{1,t}$ and $\Delta W^i_{1,t} \sim N(0, \Delta t)$, with other terms defined similarly. To allow arbitrary accuracy of E–M, the inter-observation time length Δt is replaced by a stepsize $\Delta t = 1/L$ for the numerical integration, for integer $L \geq 1$. We find that using $L = 5$ (giving 4 intermediate times between observation instants) allows sufficient

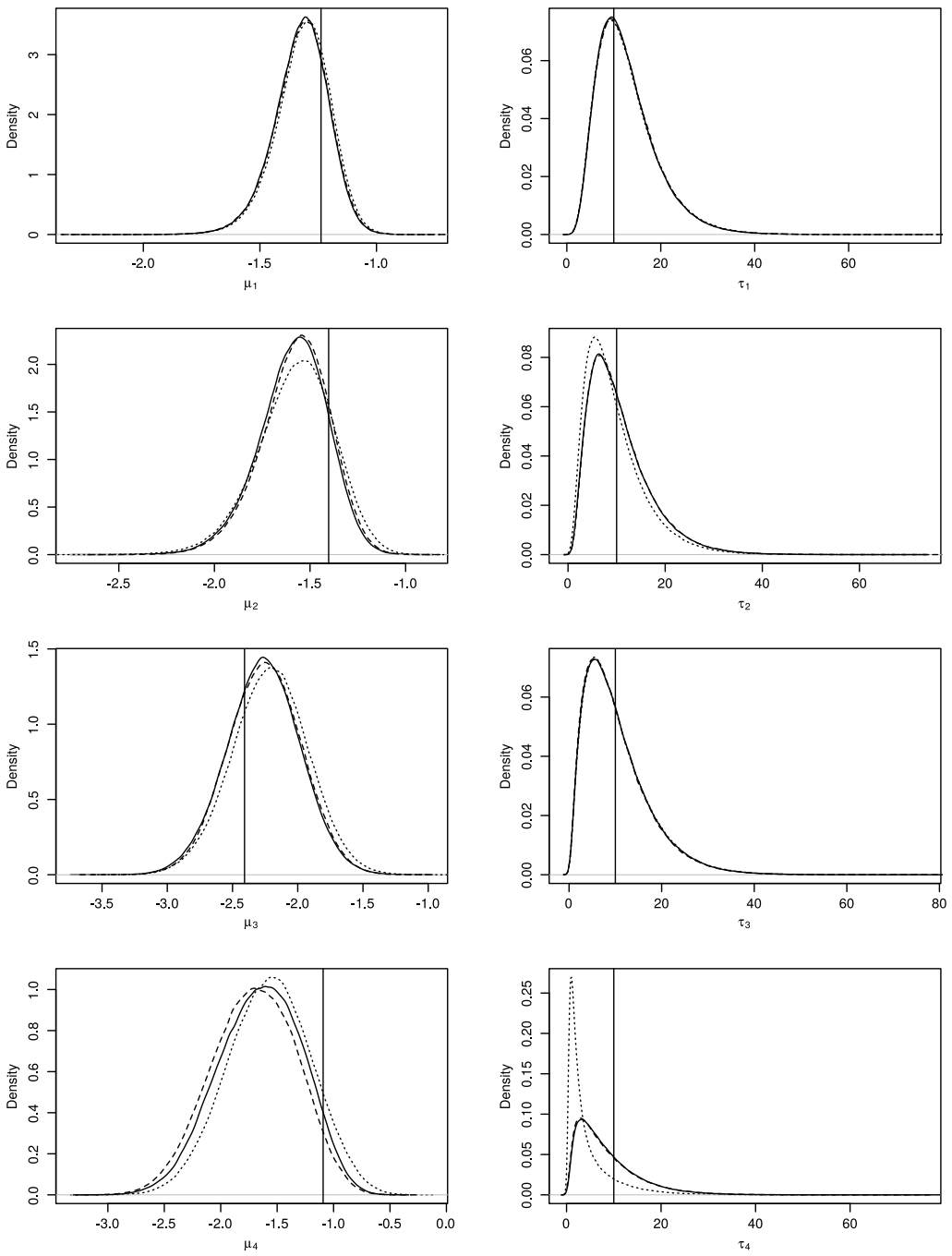

Fig. 7. Marginal posterior distributions for μ_i and τ_i, $i = 1, \ldots, 4$. Dotted line shows results from LNA scheme, solid line is from the CPMMH scheme and dashed line is the PMMH Scheme.

accuracy (compared to the analytic solution) to permit use of the same tuning choices when re-running PMMH (including the naive scheme) and CPMMH. Our findings are summarized by Table 3.

Unsurprisingly, inspection of Table 3 reveals that relative performance between the three computing pseudo-marginal schemes is similar to that obtained when using the analytic solution; CPMMH provides almost an order of magnitude increase in terms of mESS/m over a naive PMMH approach. We note that use of the Euler–Maruyama approximation requires computation and storage of an additional $1/\Delta t$ innovations per SDE component, inter-observation interval, particle and subject, thus accounting for the increase in CPU time compared to when using the analytic solution.

Table 3
Tumor model (Euler–Maruyama). Correlation ρ, number of particles N, CPU time (in minutes m), minimum ESS (mESS), minimum ESS per minute (mESS/m) and relative minimum ESS per minute (Rel.) as compared to PMMH-naive. All results are based on 500k iterations of each scheme.

Algorithm	ρ	N	CPU (m)	mESS	mESS/m	Rel.
PMMH - naive	0	30	7947	990	0.123	1
PMMH	0	30	7651	2240	0.293	2.4
CPMMH	0.999	10	1893	2172	1.15	9.2

Nevertheless, we find that our proposed approach is able to accommodate an intractable SDE scenario and provides a worthwhile increase in performance over competing approaches.

5.2.2. Comparison with ODEMEM

To highlight the potential issues that arise by ignoring inherent stochasticity, we consider inference for an ordinary differential equation mixed effects model (ODEMEM) of tumor growth. We take the SDEMEM in (19) and set $\gamma^i = \psi^i = 0$ to give

$$\begin{aligned} dx^i_{1,t} &= \beta^i x^i_{1,t} dt, \\ dx^i_{2,t} &= -\delta^i x^i_{2,t} dt \end{aligned} \tag{23}$$

for $i = 1, \ldots, M$. The observation model and random effects distributions remain unchanged from (20) and (21) upon omitting $\log \gamma^i$ and $\log \psi^i$ from ϕ^i. The ODE system in (23) can be solved to give

$$x^i_{1,t} = x^i_{1,0} \exp\{\beta^i t\}, \qquad x^i_{1,t} = x^i_{1,0} \exp\{\delta^i t\}.$$

The likelihood associated with each experimental unit is then obtained simply as

$$\pi(y^i|\phi^i, \sigma_e) = \prod_{t=1}^{21} \mathrm{N}\left(y^i_t; \log(x^i_{1,t} + x^i_{2,t}), \sigma_e^2\right).$$

Fitting the ODEMEM to the synthetic data set from Section 5.2 is straightforward, via a Metropolis-within-Gibbs scheme. Figs. 8 and 9 summarize our findings. Unsurprisingly, since the ODEMEM is unable to account for intrinsic stochasticity, the observation standard deviation is massively over-estimated. Fig. 8 shows little agreement between the marginal posteriors under the ODEMEM and SDEMEM for this parameter. In terms of model fit, both the observation (Y^1_t) and latent process ($X^1_t = \log V^1_t$) predictive distributions for unit 1 are over concentrated for the ODEMEM. Similar results (not shown) are obtained for the other experimental units. Notably, from Fig. 9, around half of the actual simulated X_t values lie outside of the 95% credible interval under the ODEMEM.

5.3. Neuronal data

Here we consider a much more challenging problem: modeling a large number of observations pertaining neuronal data. In particular, we are interested in modeling the neuronal membrane potential across inter-spike intervals (ISIs). The problem of modeling the membrane potential from ISIs measurements using SDEs has already been considered numerous times, also using SDEMEMs, see Picchini et al. (2008). In fact here we analyze the same data considered in Lansky et al. (2006) and Picchini et al. (2008), or actually a subset thereof, due to computational constraints. The "leaky integrate-and-fire" appears to be one of the most common models, in both artificial neural network applications and descriptions of biological systems. Deterministic and stochastic implementations of the model are possible. In the stochastic version, under specific assumptions (Lanski, 1984), it coincides with the Ornstein–Uhlenbeck stochastic process and has been extensively investigated in the neuronal context, for instance in Ditlevsen and Lansky (2005). Consider Fig. 10 as an illustrative example, reporting values of neuronal membrane depolarization studied in Höpfner (2007). Inter-spike-intervals are the observations considered between "firing" times of the neuron, the latter being represented by the spikes appearing in Fig. 10 (notice these are not the data we analyzed. This figure is only used for illustration). Data corresponding to the near-deterministic spikes are removed, and what is left constitutes data from several ISIs. As in Picchini et al. (2008), we consider data from different ISIs as independent. Hence, M is the number of considered ISIs. These are 312 in total, however, because of computational limitations, we will only analyze a subset of 100 ISIs, hence our results are based on $M = 100$ and a total of 162,610 observations. A challenge is posed by the fact that some ISIs are much longer than others (in our case they vary between 600 and 2,500 observations), meaning that longer ISIs could typically require a larger N to avoid particle depletion, but using the same large N to approximate all M likelihood terms would be a waste of computational resources. This is why CPMMH comes particularly useful, as it allows to keep a small N across all units while still avoiding sticky behavior in the MCMC chains. Data from the 100 ISIs are plotted on a common time-scale in Fig. 11 (after some translation to let each ISI start approximately at zero value at time zero). These consist

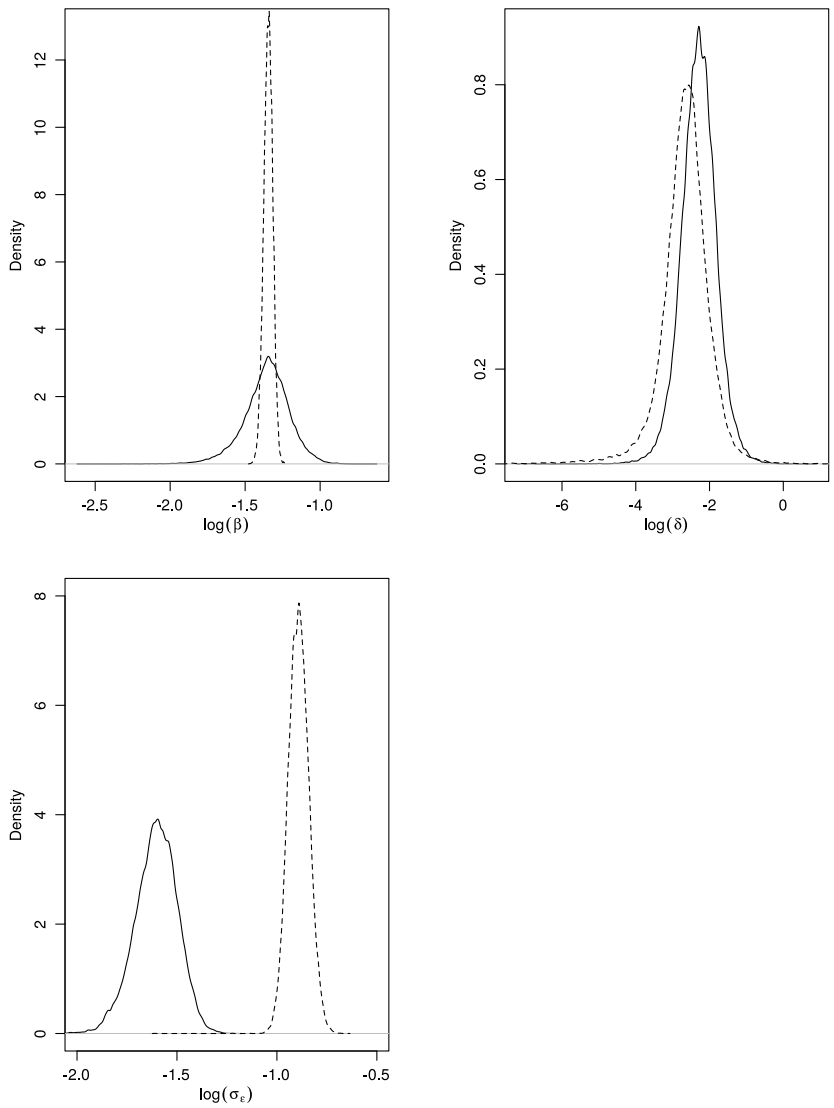

Fig. 8. Marginal posterior distributions for the (logged) subject specific parameters $\log \beta^1$, $\log \delta^1$, and the observation standard deviation $\log \sigma_\epsilon$. Dashed line shows results from ODEMEM, solid line is from SDEMEM.

of membrane potentials measured every 0.15 msec intracellularly from the auditory system of a guinea pig (for details on data acquisition and processing, see Yu et al., 2004).

Outside the mixed-effects context, if we denote the neuronal input with ν, and if the neuron is supposed to operate in a stationary state during some time of interest, then ν would be assumed constant during this period. Picchini et al. (2008) generalize by assuming that in addition to ν there is a random component changing from one ISI to the next, which could be caused by the naturally occurring variations of environment signaling, by experimental irregularities or by other sources of noise not included in the model. This fact can then be modeled by assuming that each ISI has its own input ν^i, and Picchini et al. (2008) specifically assume that the ν^i are iid Gaussian distributed with mean ν. An extension of the model in Picchini et al. (2008) is the following state-space type SDEMEM

$$\begin{cases} Y_t^i &= X_t^i + \epsilon_t^i, \quad \epsilon_t^i \overset{indep}{\sim} \mathrm{N}(0, \sigma_\epsilon^2), \qquad i = 1, \ldots, M, \\ dX_t^i &= (-\lambda^i X_t^i + \nu^i)dt + \sigma^i dW_t^i. \end{cases} \tag{24}$$

where the diffusion process $\{X_t^i; t \geq 0\}$ models the membrane potential [mV] in the ith ISI, with input ν^i[mV/msec]. The spontaneous voltage decay (in the absence of input) for the ith ISI is $(\lambda^i)^{-1}$[msec], which means that the stationary mean for $\{X_t^i\}$ is ν^i/λ^i, see e.g. Ditlevsen and Lansky (2005) for details. The diffusion coefficients σ^i have unit [mV/$\sqrt{\text{msec}}$]. Clearly, we assume that we are unable to observe $\{X_t^i\}$ directly, and instead can only observe a noisy realization from

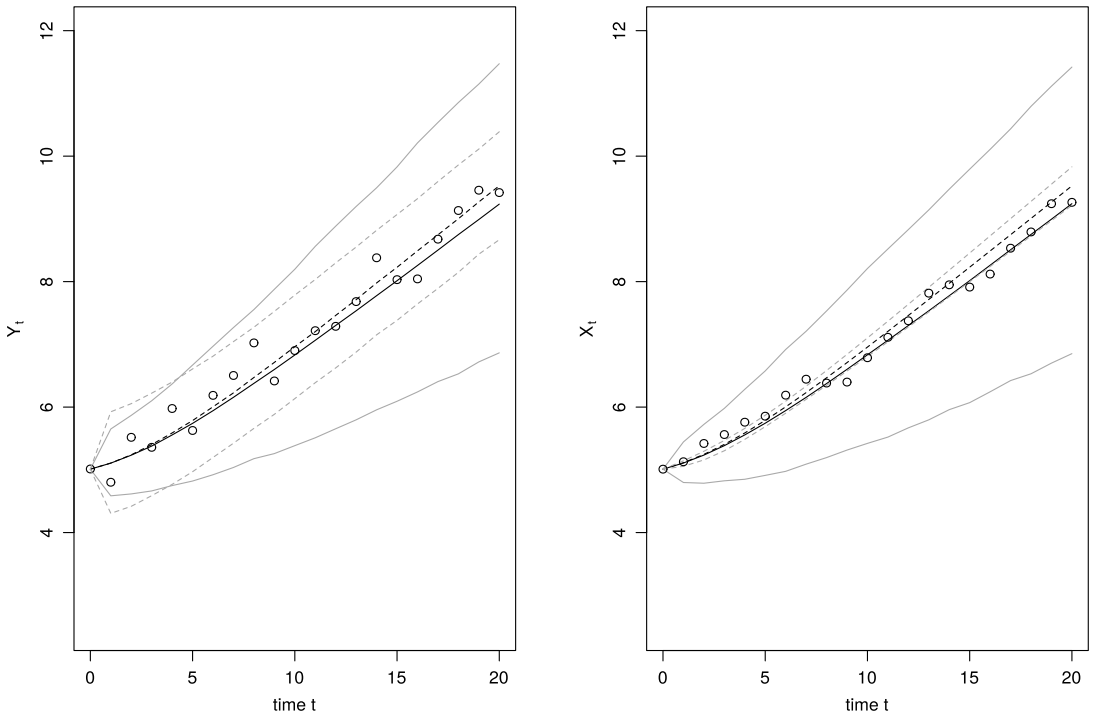

Fig. 9. Posterior predictive mean (black) and 95% credible intervals (gray) for the observed process Y_t^1 (circles, left panel) and the latent process $X_t^1 = \log Y_t^1$ (circles, right panel). Dashed line shows results from ODEMEM, solid line is from SDEMEM.

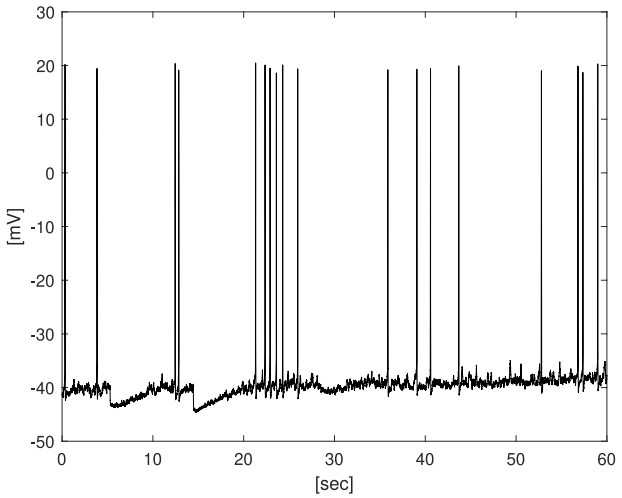

Fig. 10. An exemplificative plot of depolarization [mV] vs time [sec].
Source: Data from Höpfner (2007).

$\{Y_t; t \geq 0\}$. Differences with the SDEMEM in Picchini et al. (2008) are that: (i) their observations were assumed unaffected by measurement noise, i.e. observations were directly available from $\{X_t^i; t \geq 0\}$, $i = 1, \ldots, M$, which is a convenient assumption easing calculations towards obtaining exact maximum likelihood estimation, but that it is generally possible to argue against; (ii) in Picchini et al. (2008) the only random effect was ν^i, and remaining parameters were fixed-effects, while in the present case we have random effects λ^i and σ^i in addition to ν^i. Of course here we also need to estimate σ_ϵ, which was not done in Picchini et al. (2008) since no measurement error was assumed.

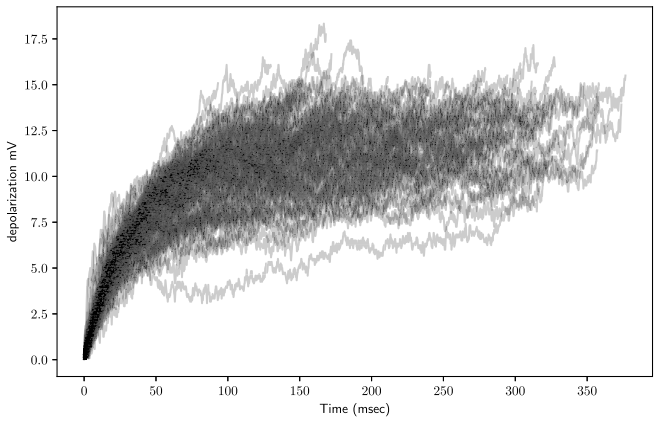

Fig. 11. Observations from 100 ISIs.

As in Section 5.1 the random effects are constrained to be positive and we therefore define $\phi^i = (\phi_1^i, \phi_2^i, \phi_3^i) = (\log \lambda^i, \log \nu^i, \log \sigma^i)$, where

$$\phi_j^i|\eta \overset{indep}{\sim} \mathrm{N}(\mu_j, \tau_j^{-1}), \qquad j = 1, 2, 3,$$

and $\eta = (\mu_1, \mu_2, \mu_3, \tau_1, \tau_2, \tau_3)$, with τ_j the precision of ϕ_j^i. Since we here have a similar setting as in Section 5.1, we employ the same semi-conjugate priors with hyperparameters

$$(\mu_{0_1}, M_{0_1}, \alpha_1, \beta_1) = (\log(0.1), 1, 2, 1),$$
$$(\mu_{0_2}, M_{0_2}, \alpha_2, \beta_2) = (\log(1.5), 1, 2, 1),$$
$$(\mu_{0_3}, M_{0_3}, \alpha_3, \beta_3) = (\log(0.5), 1, 2, 1).$$

The considered data are measured with techniques ensuring high precision, and we assume the following prior $\log \sigma_\epsilon \sim \mathrm{N}(-1, 1)$. Because of the small measurement noise, we expect that a bootstrap filter will perform poorly, leading to a very noisy approximation of the likelihood $\pi(y|\phi, \sigma_\epsilon) = \prod_{i=1}^{M} \pi(y^i|\phi^i, \sigma_\epsilon)$. To be able to obtain a good approximation of the likelihood, we instead use the bridge particle filter found in Golightly and Wilkinson (2011), since, as explained below, the bootstrap filter is statistically inadequate for this experiment (moreover, it is also computationally inadequate, since it would require a too large number of particles, which was impossible to handle with the limited memory of our computer). In A, we derive the bridge filter for the model in (24), and we also compare the forward propagation of the particles that we obtain using the bootstrap filter and the bridge filter. In A.2 we see that the likelihood approximation obtained from the bootstrap filter is very inaccurate, which is due to its inability to handle measurements with small observational noise. Consequently, the number of particles required to give likelihood estimates with low variance is computationally prohibitive. Therefore, for this example, we only report results based on the bridge filter (which is not a plug-and-play method).

We use the following four algorithms already defined in Section 5.1: Kalman, which obviously here is the gold-standard method; PMMH, using the bridge filter with $N = 1$ particle; CPMMH-0999 using the bridge filter also with 1 particle, and CPMMH-09 using the bridge filter with 1 particle. We find that, due to propagating particles conditional on the next observation, using a single particle was enough to give likelihood estimates with low variance. We ran all algorithms for 100k iterations, considering the first 20k iterations as burn-in. The starting value for σ_ϵ was set far away from the posterior mean that we obtained from a pilot run of the Kalman algorithm, and the starting values for the random effects ϕ_j^i were set to their prior means. For all algorithms, the proposal distributions were tuned adaptively using the generalized AM algorithm as described in Section 4.5. We ran the algorithms on a single-core computer so no parallelization was utilized. Posterior marginals in Figs. 12–13 show that inference results for all algorithms are very similar, except for CPMMH-0999, for which posterior samples of σ_ϵ are inconsistent with the output from the other competing schemes. We note that the case of $N = 1$ can be seen to correspond to a joint update of the parameters and latent process x. Inducing strong positive correlation between successive values of u therefore results in extremely slow mixing over the latent process and in turn, the parameters. This is particularly evident for σ_ϵ, whose update requires calculation of likelihood estimates over all experimental units. Reducing ρ to 0.9 appears to alleviate this problem. Runtimes and ESS values are in Table 4. As expected, Kalman is the most efficient algorithm, being 19 times more efficient than PMMH is terms of ESS/min. However, here PMMH and CPMMH have the same efficiency in terms of ESS/min. Thus, CPMMH does not seem to produce any efficiency improvement for this case study. This is due to the efficiency of the bridge filter in guiding state proposals towards the next observation, and therefore allowing us to run PMMH with very few particles, thus making the potential improvement brought by CPMMH essentially null.

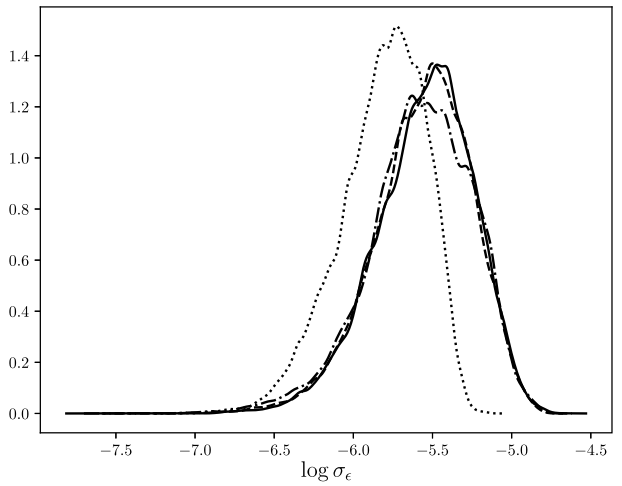

Fig. 12. Neuronal model: marginal posterior distributions for $\log \sigma_\epsilon$. Solid line is Kalman, dashed line is PMMH, dotted line is CPMMH-0999, dash-dotted line CPMMH-09.

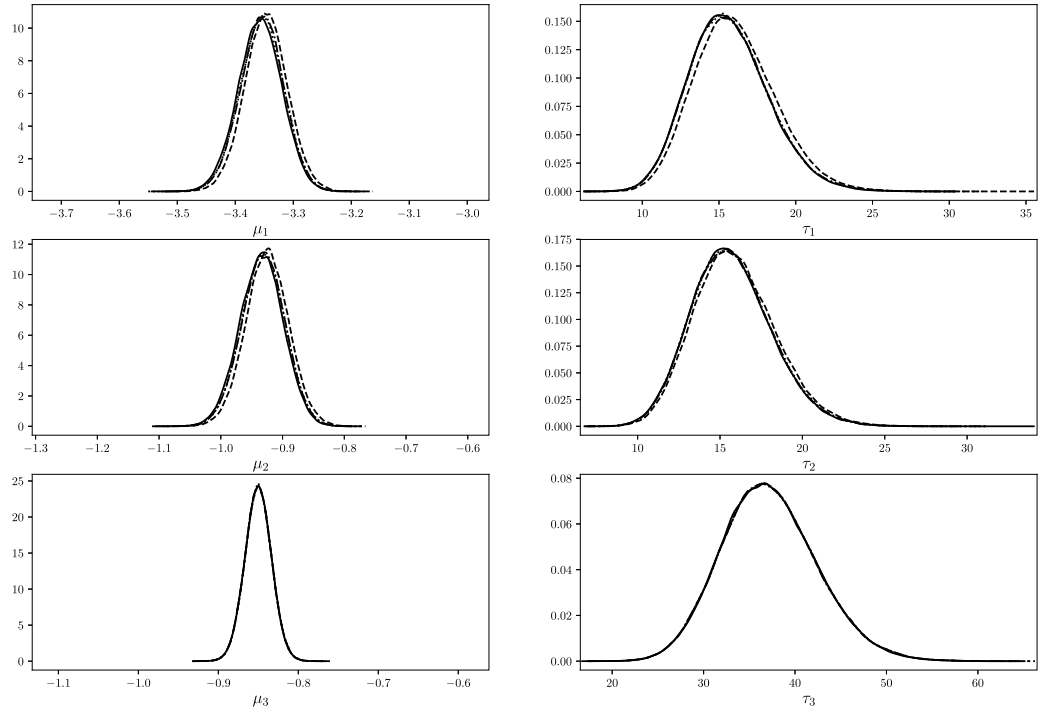

Fig. 13. Neuronal model: marginal posterior distributions for $\eta = (\mu_1, \mu_2, \mu_3, \tau_1, \tau_2, \tau_3)$. Solid line is Kalman, dashed line is PMMH, dotted line is CPMMH-0999, dash-dotted line CPMMH-09.

We compare our results with those in Picchini et al. (2008). Since we have assumed that the random effects $\phi^i = (\phi_1^i, \phi_2^i, \phi_3^i) = (\log \lambda^i, \log \nu^i, \log \sigma^i)$ are Gaussian, then the $(\lambda^i, \nu^i, \sigma^i)$ are log-Normal distributed with means (λ, ν, σ) and standard deviations $(\sigma_\lambda, \sigma_\nu, \sigma_\sigma)$ respectively. By plugging the posterior means for $(\log \lambda^i, \log \nu^i, \log \sigma^i)$ as returned by "Kalman" into the formulas for the mean and standard deviation of a lognormal distribution, we obtain that $\lambda = 0.036 (\sigma_\lambda = 0.009)$ [1/msec], $\nu = 0.406 (\sigma_\nu = 0.105)$ [mV/msec], and $\sigma = 0.433$, $(\sigma_\sigma = 0.072)$. In Picchini et al. (2008) we used a maximum likelihood approach, which is a fast enough procedure for Markovian data (there we did not assume a state-space model) that allowed us to obtain point estimates using all 312 ISIs (instead of 100 ISIs as in this case), but still slow enough to not permit bootstrapped confidence intervals to be obtained. Therefore, there we reported intervals based on asymptotic normality. There we had point estimates $\hat{\nu} = 0.494$ and $\hat{\sigma}_\nu = 0.072$, which are similar to our Bayesian estimation. It makes sense that the inferences are not very different, as in the end our estimation of σ_ϵ is very small,

Table 4
Neuronal model. Correlation ρ, number of particles N, CPU time (in minutes m), minimum ESS (mESS), minimum ESS per minute (mESS/m), and relative minimum ESS per minute (Rel.) as compared to PMMH. All results are based on 100k iterations of each scheme.

Algorithm	ρ	N	CPU (m)	mESS	mESS/m	Rel.
Kalman	–	–	56	630	11.30	18.9
PMMH	–	1	479	287	0.6	1.0
CPMMH-09	0.9	1	655	400	0.61	1.0
CPMMH-0999	0.999	1	653	372	0.57	1.0

meaning that we could assume nearly Markovian data. However here we have also inferences for random effects λ^i and σ^i, whereas in Picchini et al. (2008) these were assumed fixed (unknown) effects with maximum likelihood estimates $\hat{\lambda} = 0.047$ [1/msec] (it can be obtained from Table 1 in Picchini et al. (2008) via $1/0.021 = 47.62$ [1/sec]) and $\hat{\sigma} = 0.427$ [mV/$\sqrt{\text{msec}}$] (it can be obtained from Table 1 in Picchini et al. (2008) by converting 0.0135 [V/$\sqrt{\text{sec}}$] into [mV/$\sqrt{\text{msec}}$]). We appreciate how close our posterior means based on 100 ISIs are to the maximum likelihood estimates using 312 ISIs.

6. Discussion

We have constructed an efficient and general inference methodology for the parameters of stochastic differential equation mixed-effects models (SDEMEMs). While SDEMEMs are a flexible class of models for "population estimation", their use has been limited by technical difficulties that make the execution of inference algorithms (both classic and Bayesian) computationally intensive. Our work proposed strategies to both (i) produce Bayesian inference for very general SDEMEMs, without the limitations of previous methods; (ii) alleviate the computational requirements induced by the generality of our methods. The SDEMEMs we considered are general in the sense that the underlying SDEs can be nonlinear in the states and in the parameters; the random parameters can have any distribution (not restricted to the Gaussian family); the observations equation does not have to be a linear combination of the latent states. We produced a Metropolis-within-Gibbs algorithm (hereafter Gibbs sampler, Algorithm 2) with carefully constructed blocking strategies, where the technically difficult approximation to the unavailable likelihood function is efficiently handled via correlated particle filters. The use of correlated particle filters brings in the well-known benefit of requiring fewer particles compared to the particle marginal Metropolis–Hastings (PMMH) algorithm. In our experiments, the novel blocked-Gibbs sampler embedding a correlated PMMH (CPMMH) shows that it is possible to considerably reduce the number of required particles while still obtaining a value of the effective sample size (ESS) that is comparable to using standard PMMH in the Gibbs sampler. This means that the Gibbs sampler with embedded CPMMH is computationally efficient and on two out of three examples of increasing complexity we found that our algorithm is much more efficient than a similar algorithm using the standard PMMH, sometimes even 40 times more efficient. Some care must be taken when choosing ρ, which governs the level of correlation between successive likelihood estimates. Taking $\rho \approx 1$ can result in the sampler failing to adequately mix over the auxiliary variables. We found that this problem was exacerbated when using relatively few particles (such as $N = 1$), but can be overcome by reducing ρ. The fact that our approach is an instance of the pseudo-marginal methodology of Andrieu and Roberts (2009) implies that we produce exact (simulation-based) Bayesian inference for the parameters of our SDEMEMs, regardless the number of particles used. We mostly focus on producing "plug-and-play" methodology (but see below for exceptions), meaning that no preliminary analytic calculations should be required to run our methods, and forward simulation from the SDEs simulator should be enough. Instead, what is necessary to set is the number of particles N and, when correlated particles filters are used (CPMMH), the correlation parameter ρ (however this one is easily set within the interval $[0.90, 0.999]$). Finally, the usual settings for the MCMC proposal distribution should be decided (covariance matrix of the proposal function $q(\cdot)$). However, for the neuronal data example we had to employ a bridge filter, since the observational noise is very low for this case study, causing the bootstrap filter to perform poorly. The bridge filter is not plug-and-play (as discussed below), however in this paper we have decided to include a non-plug-and-play method to show how to analyze complex case studies with existing state-of-art sequential Monte Carlo filters. When considering a plug-and-play approach, our proposed methodology relies on the use of the bootstrap particle filter, within which particles are propagated according to the SDE solution or an approximation thereof. We note that in scenarios where the observations are particularly informative (e.g. the neuronal data case study in Section 5.3), it may be beneficial to propagate particles conditional on the observations, by using a carefully chosen bridge construct. We refer the reader to Golightly et al. (2019) for details on the use of such constructs within a CPMMH scheme for SDEs. However, notice that in order to use the constructs in Golightly et al. (2019) the conditional distribution of observations (i.e. (2) in our context) must be Gaussian. This is the underlying assumption that is exploited in Botha et al. (2020) to enable the use of bridge constructs in inference for SDEMEMs. In Botha et al. (2020) they also use methods based on correlated particle filters, in a work which has been proposed independently and concurrently to ours (July 25 2019 on arXiv). See for example their "component-wise pseudo-marginal" (CWPM) method, which is similar to the naive Gibbs strategy we also propose, and they found that CWPM was the best strategy among a battery of explored methods. In order to correlate the particles, Botha et al. (2020) advocate the use of the blockwise pseudo-marginal strategy of Tran et al. (2016b): this way, at each iteration of a CPMMH algorithm they randomly pick a unit in the set $\{1, \ldots, M\}$, and only for that unit

they update the corresponding auxiliary variates, whereas for the remaining $M-1$ units they reuse the same auxiliary variates u^i as employed in the last accepted likelihood approximation. This approach implies an estimated correlation between log-likelihoods of around $1-1/M$, which also implies that the correlation level is completely guided by the number of units. This means that for a small M (e.g. $M=5$ or 10, implying a correlation of 0.80 and 0.90 respectively) a blockwise pseudo-marginal strategy might not be as effective as it could be. On the other hand, assuming a very efficient and scalable implementation allowing measurements from $M=10{,}000$ units, the blockwise pseudo-marginal approach would produce highly correlated particles, which can sometimes be detrimental by not allowing enough variety in the auxiliary variates, and ultimately producing long-term correlations in the parameter chains, as we have documented in Section 5.3 when using a low number of particles N. We therefore think it is advantageous to use a method that allows the statistician to decide on the amount of injected correlation: even though this means having one more parameter to set (ρ in our treatment), we find this decision to be rather straightforward, as mentioned above.

We hope this work can push forward the use of SDEMEMs in applied research, as even though inference methods for SDEMEMs have been available from around 2005, the limitation of theoretical or computational possibilities has implied that only specific SDEMEMs could be efficiently handled, while other SDEMEMs needed ad-hoc solutions or computationally very intensive algorithms. We believe our work is promising as a showcase of the possibility to employ very general SDEMEMs for practical applications.

Acknowledgments

SW was supported by the Swedish Research Council (Vetenskapsrådet 2013-05167). UP was supported by the Swedish Research Council (Vetenskapsrådet 2019-03924). We thank the staff at the Center for Scientific and Technical Computing at Lund University (LUNARC) for help in setting up the computer environment used for the computations in Sections 5.1 and 5.3. We thank J. F. He for making the neuronal data available. We thank the editor and three anonymous reviewers for useful and insightful comments on this paper.

Appendix A. Bridge particle filter

A.1. Deriving the bridge filter

This section is not strictly pertaining mixed-effects modeling, hence we disregard the subject's index. We consider the bridge particle filter proposed in Golightly and Wilkinson (2011), with the exception that there an SDE was numerically solved using the Euler–Maruyama scheme. Here we provide the bridge particle filter for the special case where the exact (Gaussian) transition density is available, as considered for case studies in Sections 5.1 and 5.3 . Since we do not require numerical discretization, in terms of the notation established in Golightly and Wilkinson (2011) we have that $m=1$ and $j=0$. Furthermore, we let Δobs denote the step-length for the observational times grid. Thus we have that $\Delta t=\Delta$obs and $\Delta j=0=\Delta$obs.

Here the bridge filter is derived for the example in Section 5.3. The analytical transition density for the X_t process in 5.3 is

$$X_{t+\Delta t}|X_t=x_t\sim \mathrm{N}\left(x_t e^{-\lambda\Delta t}+\frac{\nu}{\lambda}(1-e^{-\lambda\Delta t}),\frac{\sigma^2}{2\lambda}(1-e^{-2\lambda\Delta t})\right).$$

The joint density for $X_{t+\Delta t}$ and $Y_{t+\Delta t}$, conditional on X_t, is

$$\begin{pmatrix}X_{t+\Delta t}\\ Y_{t+\Delta t}\end{pmatrix}|X_t=x_t\sim \mathrm{N}\left\{\begin{pmatrix}\alpha_0\\ \alpha_0\end{pmatrix},\begin{pmatrix}\beta_0 & \beta_0\\ \beta_0 & \beta_0+\sigma_\epsilon^2\end{pmatrix}\right\}$$

where $\alpha_0=x_t e^{-\lambda\Delta t}+\frac{\nu}{\lambda}(1-e^{-\lambda\Delta t})$, and $\beta_0=\frac{\sigma^2}{2\lambda}(1-e^{-2\lambda\Delta t})$. The conditional distribution used as proposal distribution in the bridge filter is

$$\hat{\pi}(x_{t+\Delta t}|x_t,y_{t+\Delta t})=\mathrm{N}(x_{t+\Delta t};\mu,\Sigma), \tag{A.1}$$

where $\mu=\alpha_0+\beta_0(\beta_0+\sigma_\epsilon^2)^{-1}(y_{t+\Delta t}-\alpha_0)$, $\Sigma=\beta_0(1-[\beta_0+\sigma_\epsilon^2]^{-1}\beta_0)$.

Eq. (A.1) can be used to propagate particles forward, which is a much more efficient approach than in the bootstrap filter case, where the sampler is myopic to the next observation, while (A.1) is able to look-ahead towards the next observation $y_{t+\Delta t}$. Thus, the bridge filter is similar in structure to Algorithm 1 with the difference that here the particles propagation step consists in sampling from (A.1), and the weights are given by

$$\tilde{w}_{t+\Delta t,k}=\frac{\pi(y_{t+\Delta t}|x_{t+\Delta t,k},\sigma_\epsilon^2)\pi(x_{t+\Delta t,k}|x_{t,k})}{\hat{\pi}(x_{t+\Delta t,k}|x_{t,k},y_{t+\Delta t})},\qquad w_{t+\Delta t,k}=\frac{\tilde{w}_{t+\Delta t,k}}{\sum_{j=1}^N\tilde{w}_{t+\Delta t,j}},\qquad k=1,\dots,N.$$

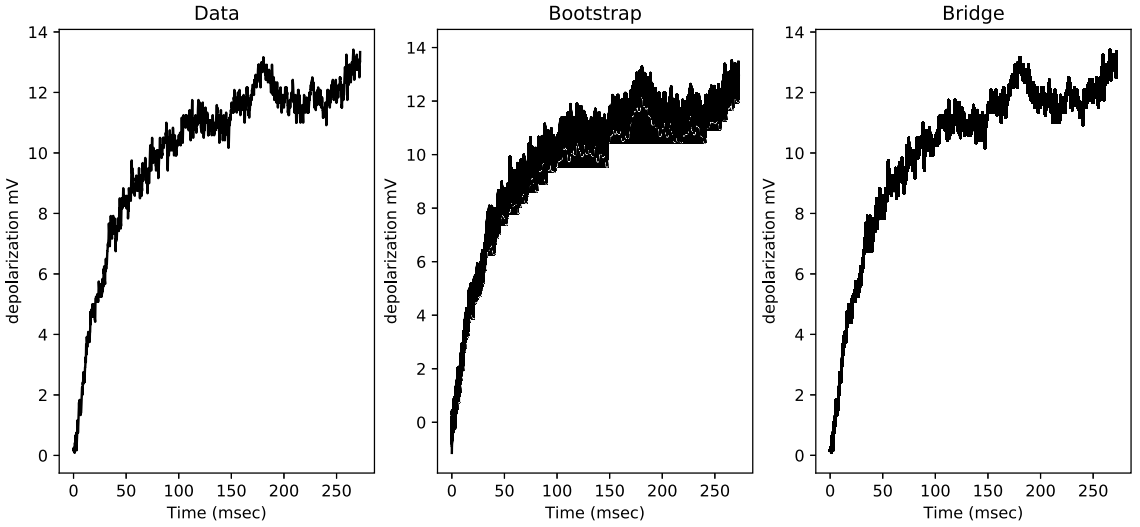

Fig. A.14. Neuronal model: forward propagation of the particles for bootstrap and bridge filter for one ISI (chosen at random; this ISI contained 1817 data points). Leftmost panel: observed data for that ISI. Central panel: forward propagation of the particles from the bootstrap filter. Rightmost panel: forward propagation of the particles from the bridge filter.

Table A.5
Comparing 100 log-likelihood estimations for the bootstrap and bridge filter.

	Log-likelihood	Std. Dev.	Runtime (s)
Kalman	62 091	–	0.012
Bootstrap	−2 594 152	119 905	21.51
Bridge	62 291	0.34	27.50

A.2. Comparing the bootstrap filter and the bridge particle filter

To compare the performance of the bootstrap and the bridge filter, we run both filters with the same number of particles (500 particles for each subject) using the 100 ISIs neuronal data from Section 5.3. Parameters are set at the posterior means obtained from the Kalman algorithm. The comparison is interesting since it illustrates the well known issue of running particle filters when the observational error is small (here we have that $\sigma_\epsilon \approx 0.001$), and hence it is expected that the bootstrap filter will produce sub-optimal results. This is due to its inability to "target" the next observation, thus producing very small weights due to the small σ_ϵ. In Fig. A.14, we compare the forward propagation of the particles for one ISI chosen at random. It is evident that the bridge filter follows the data more closely. Furthermore, we run each filter independently for 100 times and compare the averages of the log-likelihood values, the standard deviation of the 100 log-likelihood estimations, and the runtimes, see Table A.5. We can easily notice the superiority of the bridge filter returning an averaged log-likelihood value very close to the one provided by the Kalman filter. In particular, notice how the log-likelihood estimation is very unreliable (due to the small observation error).

We now compare the inference results for CPMMH when using the bridge filter and the bootstrap filter. We ran four algorithms: Kalman, PMMH with N = 1 particles using the bridge filter, CPMMH-09 with N = 1 particles using the bridge filter, CPMMH-099 with N = 100 particles using the bootstrap filter. We ran, Kalman, PMMH, and CPMMH-09 for 100k iterations, and ran CPMMH-099 for only 35k iterations, as this case is computationally more intensive. In Fig. A.15 we see that when using the bootstrap filter driven inference scheme, the σ_ϵ chain fails to adequately explore regions of high posterior density. We emphasize that this is due to using too few particles ($N = 100$). It is clear from Table A.5 that the number of particles required to match the efficiency of the bridge filter is computationally infeasible. Marginal posteriors for the remaining parameters (not shown) are however similar for all algorithms. The reason why the population parameters η appear to be unaffected by these issues, unlike σ_ϵ, is that step 4 of the Gibbs algorithms in Section 4.1 (both versions, naive and blocked one) does not depend on the approximated likelihood, whereas step 2 (which samples σ_ϵ) does depend on it.

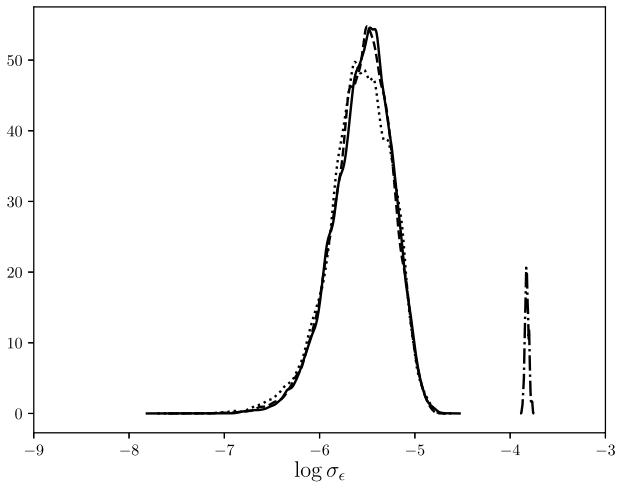

Fig. A.15. Neuronal model: marginal posterior distributions for $\log \sigma_\epsilon$. Solid line is Kalman, dashed line is PMMH using the bridge filter, dotted line is CPMMH-09 using the bridge filter, dash-dotted line is CPMMH-099 using the bootstrap filter. The marginal posteriors for Kalman, PMMH, and CPMMH-09 have been multiplied by a factor 40 for pictorial reasons.

Appendix B. Tumor growth — linear noise approximation

The linear noise approximation (LNA) can be derived in a number of more or less formal ways. We present a brief informal derivation here and refer the reader to Fearnhead et al. (2014) and the references therein for further details. We remark that the LNA is not a necessary feature of our general plug-and-play methodology outlined in Section 4 and Algorithm 2.

B.1. Setup

Consider the tumor growth model in (19), (20) and (21) and a single experimental unit so that the superscript i can be dropped from the notation. To obtain a tractable observed data likelihood, we construct the linear noise approximation of $\log V_t = \log(X_{1,t} + X_{2,t})$.

Let $Z_t = (Z_{1,t}, Z_{2,t}, Z_{3,t})^T = (\log V_t, \log X_{1,t}, \log X_{2,t})^T$. The SDE satisfied by Z_t can be found using the Itô formula, for which we obtain

$$dZ_t = \alpha(Z_t, \phi)dt + \sqrt{\beta(Z_t, \phi)}dW_t$$

where

$$\alpha(Z_t, \phi) = \begin{pmatrix} \left\{\beta + 0.5\gamma^2\right\} e^{Z_{2,t}-Z_{1,t}} + \left\{-\delta + 0.5\tau^2\right\} e^{Z_{3,t}-Z_{1,t}} - 0.5\left\{\gamma^2 e^{2(Z_{2,t}-Z_{1,t})} + \psi^2 e^{2(Z_{3,t}-Z_{1,t})}\right\} \\ \beta \\ -\delta \end{pmatrix}$$

$$\beta(Z_t, \phi) = \begin{pmatrix} \gamma^2 e^{2(Z_{2,t}-Z_{1,t})} + \tau^2 e^{2(Z_{3,t}-X_{1,t})} & \gamma^2 e^{2(Z_{2,t}-Z_{1,t})} & \psi^2 e^{2(Z_{3,t}-Z_{1,t})} \\ \gamma^2 e^{2(Z_{2,t}-Z_{1,t})} & \gamma^2 & 0 \\ \psi^2 e^{2(Z_{3,t}-Z_{1,t})} & 0 & \psi^2 \end{pmatrix}.$$

We apply the linear noise approximation (LNA) by partitioning Z_t as $Z_t = m_t + R_t$ where m_t is a deterministic process satisfying

$$\frac{dm_t}{dt} = \alpha(m_t, \phi) \tag{B.1}$$

and $\{R_t, t \geq 0\}$ is a residual stochastic process satisfying

$$dR_t = \{\alpha(Z_t, \phi) - \alpha(m_t, \phi)\}\, dt + \sqrt{\beta(Z_t, \phi)}dW_t.$$

By Taylor expanding α and β about the deterministic process m_t and retaining the first two terms in the expansion of α, and the first term in the expansion of β, we obtain an approximate residual stochastic process $\{\tilde{R}_t, t \geq 0\}$ satisfying

$$d\tilde{R}_t = J_t \tilde{R}_t dt + \sqrt{\beta(m_t, \phi)}dW_t$$

where J_t is the Jacobian matrix with (i,j)th element $(J_t)_{i,j} = \partial\alpha_i(m_t,\phi)/\partial m_{j,t}$. Assuming initial values $m_0 = z_0$ and $\tilde{R}_0 = 0$, the approximating distribution of Z_t is given by

$$Z_t|Z_0 = z_0 \approx \mathrm{N}(m_t, H_t) \tag{B.2}$$

where m_t satisfies (B.1) and, after several calculations which we omit for brevity, H_t is the solution to

$$\frac{dH_t}{dt} = H_t J_t^T + \beta(m_t,\phi) + J_t H_t. \tag{B.3}$$

B.2. Inference

Note that the observation model in (20) can be written as

$$Y_t = P^T Z_t + \epsilon_t, \qquad \epsilon_t \overset{indep}{\sim} \mathrm{N}(0,\sigma_e^2). \tag{B.4}$$

where P is a 3×1 'observation vector' with first entry 1 and zeros elsewhere. The linearity of (B.2) and (B.4) yields a tractable approximation to the marginal likelihood $\pi(y|\phi,\sigma_e)$, which we denote by $\pi_{\mathrm{LNA}}(y|\phi,\sigma_e)$. The approximate marginal likelihood $\pi_{\mathrm{LNA}}(y|\phi,\sigma_e)$ can be factorized as

$$\pi_{\mathrm{LNA}}(y|\phi,\sigma_e) = \pi_{\mathrm{LNA}}(y_1|\phi,\sigma_e)\prod_{i=2}^{n}\pi_{\mathrm{LNA}}(y_i|y_{1:i-1},\phi,\sigma_e) \tag{B.5}$$

where $y_{1:i-1} = (y_1,\ldots,y_{i-1})^T$. Suppose that $Z_1 \sim \mathrm{N}(a,C)$ *a priori*, for some constants a and C. The marginal likelihood under the LNA, $\pi_{\mathrm{LNA}}(y_{1:n}|\phi,\sigma_e) := \pi_{\mathrm{LNA}}(y|\phi,\sigma_e)$ can be obtained via a forward filter, which is given in Algorithm 3.

Algorithm 3 Forward filter

Input: Data y, parameter values ϕ and σ_e.
Output: Observed data likelihood $\pi_{\mathrm{LNA}}(y|\phi,\sigma_e)$.

1. Initialization. Compute

   $$\pi_{\mathrm{LNA}}(y_1|\phi,\sigma_e) = N\left(y_1\,;\, P^T a\,,\, P^T C P + \sigma_e^2\right)$$

   where $\mathrm{N}(\cdot\,;\,a\,,\,C)$ denotes the Gaussian density with mean vector a and variance matrix C. The posterior at time $t = 1$ is therefore $Z_1|y_1 \sim \mathrm{N}(a_1, C_1)$ where

   $$a_1 = a + CP\left(P^T C P + \sigma_e^2\right)^{-1}\left(y_1 - P^T a\right)$$
   $$C_1 = C - CP\left(P^T C P + \sigma_e^2\right)^{-1} P^T C\,.$$

2. For $i = 1, 2, \ldots, n-1$,

   (a) Prior at $i+1$. Initialize the LNA with $m_i = a_i$ and $H_i = C_i$. Integrate the ODEs (Eq. (B.1)) and (Eq. (B.3)) forward to $i+1$ to obtain m_{i+1} and H_{i+1}. Hence

   $$Z_{i+1}|y_{1:i} \sim \mathrm{N}(m_{i+1}, H_{i+1})\,.$$

   (b) One step forecast. Using the observation equation, we have that

   $$Y_{i+1}|y_{1:i} \sim N\left(P^T m_{i+1}, P^T H_{i+1} P + \sigma_e^2\right)\,.$$

   Compute

   $$\begin{aligned}\pi_{\mathrm{LNA}}(y_{1:i+1}|\phi,\sigma_e) &= \pi_{\mathrm{LNA}}(y_{1:i}|\phi,\sigma_e)\pi_{\mathrm{LNA}}(y_{i+1}|y_{1:i},\phi,\sigma_e)\\ &= \pi_{\mathrm{LNA}}(y_{1:i}|\phi,\sigma_e)\,N\left(y_{i+1}\,;\,P^T m_{i+1}\,,\,P^T H_{i+1} P + \sigma_e^2\right)\,.\end{aligned}$$

   (c) Posterior at $i+1$. Combining the distributions in (a) and (b) gives the joint distribution of Z_{i+1} and Y_{i+1} (conditional on $y_{1:i}$ and ϕ) as

   $$\begin{pmatrix} Z_{i+1} \\ Y_{i+1}\end{pmatrix} \sim N\left\{\begin{pmatrix} m_{i+1} \\ P^T m_{i+1}\end{pmatrix}, \begin{pmatrix} H_{i+1} & H_{i+1}P \\ P^T H_{i+1} & P^T H_{i+1} P + \sigma_e^2\end{pmatrix}\right\}$$

   and therefore $Z_{i+1}|y_{1:i+1} \sim \mathrm{N}(a_{i+1}, C_{i+1})$ where

   $$a_{i+1} = m_{i+1} + H_{i+1}P\left(P^T H_{i+1} P + \sigma_e^2\right)^{-1}\left(y_{i+1} - P^T m_{i+1}\right)$$
   $$C_{i+1} = H_{i+1} - H_{i+1}P\left(P^T H_{i+1} P + \sigma_e^2\right)^{-1} P^T H_{i+1}\,.$$

Inference for the SDEMEM defined by (19), (20) and (21) may be performed via a Gibbs sampler that draws from the following full conditionals

1. $\pi_{\mathrm{LNA}}(\phi|\eta,\sigma_e,y) \propto \prod_{i=1}^{M}\pi(\phi^i|\eta)\pi_{\mathrm{LNA}}(y^i|\sigma_e,\phi^i)$,
2. $\pi_{\mathrm{LNA}}(\sigma_e|\eta,\phi,y) \propto \pi(\sigma_e)\prod_{i=1}^{M}\pi_{\mathrm{LNA}}(y^i|\sigma_e,\phi^i)$,
3. $\pi(\eta|\sigma_e,\phi,y) \propto \pi(\eta)\prod_{i=1}^{M}\pi(\phi^i|\eta)$.

References

Ait-Sahalia, Y., 2008. Closed-form likelihood expansions for multivariate diffusions. Ann. Statist. 36 (2), 906–937.

Andrieu, C., Doucet, A., Holenstein, R., 2010. Particle Markov chain Monte Carlo methods (with discussion). J. R. Stat. Soc. Ser. B Stat. Methodol. 72 (3), 1–269.

Andrieu, C., Roberts, G.O., 2009. The pseudo-marginal approach for efficient computation. Ann. Statist. 37, 697–725.

Andrieu, C., Thoms, J., 2008. A tutorial on adaptive MCMC. Statist. Comput. 18 (4), 343–373.

Botha, I., Kohn, R., Drovandi, C., 2020. Particle methods for stochastic differential equation mixed effects models. Bayesian Anal. http://dx.doi.org/10.1214/20-BA1216.

Choppala, P., Gunawan, D., Chen, J., Tran, M.-N., Kohn, R., 2016. Bayesian inference for state space models using block and correlated pseudo marginal methods. Available from http://arxiv.org/abs/1311.3606.

Dahlin, J., Lindsten, F., Kronander, J., Schon, T.B., 2015. Accelerating pseudo-marginal Metropolis–Hastings by correlating auxiliary variables. Available from https://arxiv.1511.05483v1.

Del Moral, P., 2004. Feynman–Kac Formulae: Genealogical and Interacting Particle Systems with Applications. Springer, New York.

Delattre, M., Lavielle, M., 2013. Coupling the SAEM algorithm and the extended Kalman filter for maximum likelihood estimation in mixed-effects diffusion models. Stat. Interface 6 (4), 519–532.

Deligiannidis, G., Doucet, A., Pitt, M.K., 2018. The correlated pseudo-marginal method. J. R. Stat. Soc. Ser. B Stat. Methodol. 80, 839–870.

Devroye, L., 1986. Non-Uniform Random Variate Generation. Springer-Verlag, New York.

Ditlevsen, S., Lansky, P., 2005. Estimation of the input parameters in the Ornstein–Uhlenbeck neuronal model. Phys. Rev. E 71 (1), 011907.

Donnet, S., Foulley, J.-L., Samson, A., 2010. Bayesian analysis of growth curves using mixed models defined by stochastic differential equations. Biometrics 66 (3), 733–741.

Donnet, S., Samson, A., 2013a. A review on estimation of stochastic differential equations for pharmacokinetic/pharmacodynamic models. Adv. Drug Deliv. Rev. 65 (7), 929–939.

Donnet, S., Samson, A., 2013b. Using PMCMC in EM algorithm for stochastic mixed models: theoretical and practical issues. J. Soc. Fr. Stat. 155 (1), 49–72.

Doucet, A., Pitt, M.K., Kohn, R., 2015. Efficient implementation of Markov chain Monte Carlo when using an unbiased likelihood estimator. Biometrika 102, 295–313.

Fearnhead, P., Giagos, V., Sherlock, C., 2014. Inference for reaction networks using the linear noise approximation. Biometrics 70 (2), 457–466.

Flamary, R., Courty, N., 2017. POT Python Optimal Transport library. URL https://github.com/rflamary/POT.

Fuchs, C., 2013. Inference for Diffusion Processes with Applications in Life Sciences. Springer.

Golightly, A., Bradley, E., Lowe, T., Gillespie, C.S., 2019. Correlated pseudo-marginal schemes for time-discretised stochastic kinetic models. Computational Statistics & Data Analysis 136, 92–107.

Golightly, A., Wilkinson, D.J., 2011. Bayesian parameter inference for stochastic biochemical network models using particle Markov chain Monte Carlo. Interface Focus 1 (6), 807–820.

Gordon, N.J., Salmond, D.J., Smith, A.F.M., 1993. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. IEE Proc. F 140, 107–113.

Höpfner, R., 2007. On a set of data for the membrane potential in a neuron. Math. Biosci. 207 (2), 275–301.

Kloeden, P.E., Platen, E., 1992. Numerical Solution of Stochastic Differential Equations. Springer.

Künsch, H.R., 2013. Particle filters. Bernoulli 19, 1391–1403.

Lanski, P., 1984. On approximations of Stein's neuronal model. J. Theoret. Biol. 107 (4), 631–647.

Lansky, P., Sanda, P., He, J., 2006. The parameters of the stochastic leaky integrate-and-fire neuronal model. J. Comput. Neurosci. 21 (2), 211–223.

Lavielle, M., 2014. Mixed Effects Models for the Population Approach: Models, Tasks, Methods and Tools. Chapman and Hall/CRC.

Leander, J., Almquist, J., Ahlström, C., Gabrielsson, J., Jirstrand, M., 2015. Mixed effects modeling using stochastic differential equations: illustrated by pharmacokinetic data of nicotinic acid in obese Zucker rats. AAPS J. 17 (3), 586–596.

Murphy, K.P., 2007. Conjugate bayesian analysis of the gaussian distribution. https://www.cs.ubc.ca/~murphyk/Papers/bayesGauss.pdf.

Murray, L., Lee, A., Jacob, P.E., 2016. Parallel resampling in the particle filter. J. Comput. Graph. Statist. 25 (3), 789–805.

Overgaard, R.V., Jonsson, N., Tornøe, C.W., Madsen, H., 2005. Non-linear mixed-effects models with stochastic differential equations: implementation of an estimation algorithm. J. Pharmacokinet. Pharmacodyn. 32 (1), 85–107.

Picchini, U., De Gaetano, A., Ditlevsen, S., 2010. Stochastic differential mixed-effects models. Scand. J. Stat. 37 (1), 67–90.

Picchini, U., Ditlevsen, S., 2011. Practical estimation of high dimensional stochastic differential mixed-effects models. Comput. Statist. Data Anal. 55 (3), 1426–1444.

Picchini, U., Ditlevsen, S., De Gaetano, A., Lansky, P., 2008. Parameters of the diffusion leaky integrate-and-fire neuronal model for a slowly fluctuating signal. Neural Comput. 20 (11), 2696–2714.

Picchini, U., Forman, J.L., 2019. Bayesian inference for stochastic differential equation mixed effects models of a tumor xenography study. J. R. Stat. Soc. Ser. C. Appl. Stat. 68 (4), 887–913.

Pitt, M.K., dos Santos Silva, R., Giordani, P., Kohn, R., 2012. On some properties of Markov chain Monte Carlo simulation methods based on the particle filter. J. Econometrics 171 (2), 134–151.

Price, L.F., Drovandi, C.C., Lee, A., Nott, D.J., 2018. Bayesian synthetic likelihood. J. Comput. Graph. Statist. 27 (1), 1–11.

Ruse, M.G., Samson, A., Ditlevsen, S., 2019. Inference for biomedical data by using diffusion models with covariates and mixed effects. J. R. Stat. Soc. Ser. C. Appl. Stat. http://dx.doi.org/10.1111/rssc.12386.

Sherlock, C., Thiery, A., Roberts, G.O., Rosenthal, J.S., 2015. On the effciency of pseudo-marginal random walk Metropolis algorithms. Ann. Statist. 43 (1), 238–275.

Sørensen, H., 2004. Parametric inference for diffusion processes observed at discrete points in time. Internat. Statist. Rev. 72 (3), 337–354.

Steele, J.M., 2012. Stochastic Calculus and Financial Applications, Vol. 45. Springer Science & Business Media.

Stewart, L., McCarty, Jr., P., 1992. Use of Bayesian belief networks to fuse continuous and discrete information for target recognition, tracking, and situation assessment. In: Proc. SPIE Signal Processing, Sensor Fusion and Target Recognition, Vol. 1699, pp. 177–185.

Tornøe, C.W., Overgaard, R.V., Agersø, H., Nielsen, H.A., Madsen, H., Jonsson, E.N., 2005. Stochastic differential equations in NONMEM®: implementation, application, and comparison with ordinary differential equations. Pharm. Res. 22 (8), 1247–1258.

Tran, M.-N., Kohn, R., Quiroz, M., Villani, M., 2016a. The block pseudo-marginal sampler. arXiv:1603.02485.

Tran, M.-N., Kohn, R., Quiroz, M., Villani, M., 2016b. Block-wise pseudo-marginal Metropolis-Hastings. arXiv:1603.02485.

Whitaker, G.A., 2016. Bayesian Inference for Stochastic Differential Mixed-Effects Models (Ph.D. thesis). Newcastle University.

Whitaker, G.A., Golightly, A., Boys, R.J., Sherlock, C., 2017. Bayesian inference for diffusion driven mixed-effects models. Bayesian Anal. 12, 435–463.

Wilkinson, D.J., 2018. Stochastic Modelling for Systems Biology, third ed. Chapman & Hall/CRC Press, Boca Raton, Florida.

Wood, S.N., 2010. Statistical inference for noisy nonlinear ecological dynamic systems. Nature 466 (7310), 1102–1104.

Yu, Y.-Q., Xiong, Y., Chan, Y.-S., He, J., 2004. Corticofugal gating of auditory information in the thalamus: an in vivo intracellular recording study. J. Neurosci. 24 (12), 3060–3069.