Real-Time Semiparametric Regression via Sequential Monte Carlo

Mathew W. McLean[*] Chris. J. Oates[*][♮]
Matt P. Wand[*][♮]
{mathew.mclean, christopher.oates, matt.wand}@uts.edu.au

June 9, 2017

Abstract

Sequential Monte Carlo algorithms are considered for computation with Bayesian semiparametric regression models in streaming data applications. We work with a very flexible class of models that includes generalized linear mixed models and generalized additive models. Compared to existing variational Bayesian methods, we do not require restrictive assumptions about the form of the posterior density of interest and our approach can be parallelized. Numerical studies compare our approach to batch Markov Chain Monte Carlo and online mean field variational Bayes methods. A topical application to data involving step counts from a wearable activity tracker is presented. Our algorithms are implemented in a package written in the `R` software environment and make use of the `Stan` probabilistic programming language.

Keywords: real-time data analysis, generalized linear mixed models, penalized splines, online machine learning

1 Introduction

Advances in technology have lead to a proliferation of applications where data arrives in a streaming fashion. Often, it is necessary to process the data immediately as it is collected, so an end-user may make decisions in real-time. The majority of computational methods for regression models are designed to work only in the batch setting, where analysis occurs only after the entire data set has been collected. The streaming setting introduces additional challenges due to the speed at which the data arrive and the sheer volume of data that can be collected. In many applications, such as when performing calculations on a smartwatch or smartphone, there is also a limited computational budget because of the lightweight hardware involved and a need to conserve battery power.

Semiparametric regression models are a popular tool in applied statistics due to their flexibility and simple interpretation. There is an extensive literature on fitting semiparametric regression models to batch

[*]School of Mathematical and Physical Sciences, University of Technology Sydney, P.O. Box 123, Ultimo, NSW 2007, Australia
[♮]Australian Research Council Centre of Excellence for Mathematical and Statistical Frontiers

data in both the Bayesian and frequentist framework; see e.g., Ruppert et al. (2003) and Wood (2006) for book-length treatments. Recent work, including Luts et al. (2014) and Luts (2015), has been concerned with fitting semiparametric models to streaming data in real-time. In those works, online mean field variational Bayesian (MFVB) methods were used to provide approximate posterior distributions.

MFVB (e.g., Wainwright and Jordan 2008) assumes that the posterior distribution of interest factorizes into a convenient form to allow for quick, approximate inference, thus avoiding more computationally-intensive techniques such as Markov Chain Monte Carlo (MCMC). Variational Bayes for real-time data analysis has received recent attention in the literature, though the only work to consider online semiparametric regression as we do in this manuscript is Luts et al. (2014). This reference and numerous others demonstrate that MFVB is useful for estimating posterior modes. However, the product restriction it assumes can under-estimate the variability of posterior quantities of interest.

Sequential Monte Carlo (SMC) methods (Del Moral et al. 2006) are a popular approach for sequential Bayesian analysis that target the exact Bayesian posterior distribution. In particular they do not rely on assumptions about the posterior density admitting a particular product form. SMC methods are a specialisation of particle filtering methods (Chopin 2002) and are thus typically easy to implement (Doucet et al. 2001) and parallelizable at certain stages (Lee et al. 2010). Recent advances, such as Gerber and Chopin (2015), render SMC methods theoretically attractive in a range of applications. Unlike MCMC, these methods do not require a separate chain to be run for each new observation and instead leverage an estimate of the posterior at the previous time step as an importance sampling distribution. SMC was demonstrated to be effective for semiparametric regression in Fan et al. (2008); however, that work only considered the analysis of batch data and did not address the computational challenges associated with streaming data.

In this work, we investigated SMC methods for real-time Bayesian inference in semiparametric regression models. Our algorithms do not suffer from the loss in accuracy of MFVB while still being feasible when computational resources are limited. Details are provided for a model-independent algorithm that can be applied to many different response types and covariance structures. Thus, unlike MFVB, we do not need to derive special algorithms for each model of interest. We demonstrate the good performance of our algorithms on both synthetic data and in a topical application to physical activity monitors.

The paper begins with an overview of semiparametric regression and SMC in Section 2 and Section 3, respectively. Section 4 is the main section of the paper and presents our algorithms. Section 5 presents numerical studies examining the performance of our algorithms and Section 6 concludes with a discussion.

2 Semiparametric Regression

In this section, we set up the class of statistical models. For this work, we considered a flexible class of models known as generalized additive models (GAMs). First introduced in Hastie and Tibshirani (1986), a GAM has the form

$$y_i \sim \mathrm{EF}(\mu_i, \phi), \quad g(\mu_i) = \mathbf{x}_{0i}^T \boldsymbol{\beta} + \sum_{j=1}^{J} f_j(x_{ij}), \ \ i = 1, \ldots, N \tag{1}$$

where $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip_0})^T$, $\mathrm{EF}(\mu_i, \phi)$ denotes a two-parameter location-scale exponential family distribution with mean μ_i and (possibly-unknown) scale parameter ϕ, and the link function, $g(\cdot)$, is a known, smooth, monotonic function. The q_0-vector $\boldsymbol{\beta}$ is an unknown vector of coefficients for the linear predictors in the model while the $f_j(\cdot)$ are unknown smooth functions to be estimated from the data.

We represent the unknown functional terms in (1) using low-rank spline bases for each f_j; that is, $f_j(x) = \sum_{k=1}^{K_j} B_{jk}(x) b_{jk}$, where $\{B_{j1}(x), \ldots, B_{jK_j}(x)\}$ are known basis functions and the b_{jk} are spline coefficients to be estimated from the data. For our numerical experiments, we used a specific flavour of penalized splines known as O'Sullivan penalized splines (O'Sullivan 1986). These were chosen for their numerical stability, ease of implementation, and good performance in Bayesian semiparametric regression problems; as demonstrated in Wand and Ormerod (2008), which obtained a canonical form that makes the bases suitable for the GAM set-up under consideration.

Speed (1991) showed that it was possible to represent GAMs as generalized linear mixed models (GLMMs), with the parameters that control the smoothness of the f_j's becoming variance components in a GLMM. This allows us to fit the model (1) using any of the tools available for mixed models, including Monte Carlo methods for Bayesian hierarchical models. See Ruppert et al. (2003) for an in-depth treatment of fitting GAMs in this framework. Using this approach, it is possible to represent (1) as the GLMM (e.g., Wand and Ormerod 2008, Section 4)

$$y_i \overset{\text{i.i.d.}}{\sim} \mathrm{EF}(\mu_i, \phi), \quad g(\mu_i) = \mathbf{x}_i^T \boldsymbol{\beta} + \sum_{j=1}^{J} z_{ij}^T \mathbf{u}_j, \quad \mathbf{u}_j \sim N(0, \sigma_j^2 \mathbf{I}_{q_j}), \tag{2}$$

where $\mathbf{I}_p$ denotes a $p \times p$ identity matrix and $\mathbf{Z}_j \equiv (\mathbf{z}_{1j}^T, \ldots, \mathbf{z}_{Nj}^T)^T$ is the basis for the random effects, $\mathbf{u}_j$. The $\mathbf{Z}_j$ are obtained through a linear transformation of the original B-spline basis function evaluations that identifies (2) with (1).

We can simplify notation by writing $\mathbf{u} \equiv (\mathbf{u}_1^T, \ldots, \mathbf{u}_J^T)^T$, $\mathbf{C} \equiv [\mathbf{X} : \mathbf{Z}_1 : \cdots : \mathbf{Z}_J]$, and $\boldsymbol{\theta} \equiv (\boldsymbol{\beta}^T, \mathbf{u}^T)^T$. We may then express the linear predictor of the GLMM in (2) as $\eta \equiv \mathbf{C}\boldsymbol{\theta}$.

To complete a Bayesian specification of (2), we first place a weakly-informative independent Gaussian prior on the vector of fixed effects $\boldsymbol{\beta} \sim N(0, \sigma_\beta^2 \mathbf{I}_{q_0})$, with σ_β^2 a fixed hyperparameter. For the variance components, σ_j^2, we considered two alternatives. We implemented both half-Cauchy priors, which was the recommended choice by Gelman (2006) for variance components in mixed models and also implemented uniform priors on the logarithm of the variance components, as was done for the SMC algorithms in Chopin (2002). We also used these priors for the scale parameter in the Gaussian response case.

3 Sequential Monte Carlo

In this section, we provide a brief overview of SMC methods. A comprehensive review can be found in; e.g., Del Moral et al. (2006). SMC refers to a class of algorithms that iteratively approximate a sequence of distributions (π_t) using a weighted collection of particles:

$$\widehat{\pi}_t = \sum_{p=1}^{P} w_p^t \delta(\boldsymbol{\Theta}_p^t),$$

where $\delta(\cdot)$ is Dirac measure. In our Bayesian setting, $\pi_t(\boldsymbol{\Theta}) = \pi_t\{\boldsymbol{\Theta} | (y_1, \mathbf{x}_1), \ldots, (y_t, \mathbf{x}_t)\}$ is the posterior distribution of the parameters in (2) after the first t data points have been observed. The number of particles P must be chosen large enough so that $\sum_{p=1}^{P} w_p^t \delta(\boldsymbol{\Theta}_p^t)$ provides a sufficiently accurate approximation to π_t. SMC is useful in applications where data are processed sequentially due to the natural sequence of slowly-varying targets, such that (almost) independent local updates performed on the collection of particles can be used to change the target of the algorithm from π_t to π_{t+1}.

SMC starts with an initial set of P particles and weights $\{\boldsymbol{\Theta}_p^0, w_p^0 \equiv P^{-1}; p = 1, \ldots, P\}$, drawn from an initial distribution π_0. The weights for the pth particle at time t are given by

$$w_p^t = w_p^{t-1} \frac{\pi_t(\boldsymbol{\Theta}_p^{t-1})}{\pi_{t-1}(\boldsymbol{\Theta}_p^{t-1})} \propto w_p^{t-1} p(y_t | \mathbf{x}_t, \boldsymbol{\Theta}_p^{t-1}),$$

where the proportionality follows from factorization of the likelihood over the data y_t and where we have defined π_0 to be the prior distribution for the model.

Each reweighting step adds more variability to the estimates as π_t moves away from π_0 and fewer and fewer particles carry significant weight. This is known as "degeneracy" and is computationally wasteful because even particles that do not contribute much to the representation of the target must still be processed. To remove the insignificant particles from the algorithm, a resampling step is usually added. Popular approaches include multinomial resampling (Gordon et al. 1993) and stratified resampling (Kitagawa 1996), and more recently, quasi Monte Carlo resampling (Gerber and Chopin 2015).

To further combat degeneracy, Gilks and Berzuini (2001) proposed the resample-move algorithm, which adds a move/rejuvenation step in which particles are moved according to a Markov transition kernel, K, that leaves the target distribution invariant. This algorithm is summarized in Algorithm 1.

The resampling in Algorithm 1 is only performed if the empirical variance of the weights is above a certain threshold. That is, the effective sample size (ESS) given by $\mathrm{ESS}(\mathbf{w}^t) = \{\sum_{p=1}^{P} (w_p^t)^2\}^{-1}$ (assuming the weights have been normalized to sum to one) is smaller than τP, where $\tau \in (0, 1)$, is a user-specified tuning parameter.

Choice of an efficient transition kernel K is important because this is the most computationally-demanding step. A standard choice that we found to work well in our application is the Metropolis-Hastings kernel, which we detail in the next section. Under mild and verifiable conditions, that include ergodicity of the Markov chain generated by K, the SMC estimate produced by Algorithm 1 has been proven to be consistent in the limit as $P \to \infty$ (Chapter 7, Del Moral et al. 2006).

Algorithm 1 Resample-move algorithm

```
Initialize: $\Theta_p^0 \overset{\text{i.i.d.}}{\sim} \pi_0$, $w_p^0 = P^{-1}$, $p = 1, \ldots, P$
for $t = 1, 2, \ldots$ do
    Reweight: $w_p^t \leftarrow w_p^{t-1} \frac{\pi_t(\Theta_t^{t-1})}{\pi_{t-1}(\Theta_p^{t-1})}$; $p = 1, \ldots, P$
    Normalize: $\mathbf{w}^t \leftarrow \mathbf{w}^t / \sum_{p=1}^{P} w_p^t$
    Resample: $\{\Theta_p^{t-1}, w_p^t\} \leftarrow \{\Theta_{i_p}^{t-1}, P^{-1}\}$; $p = 1, \ldots, P$; where the indices $i_p \in \{1, \ldots, P\}$ are
        chosen via some resampling scheme such as multinomial resampling
    Move: $\Theta_p^t \overset{\text{i.i.d.}}{\sim} K_t(\Theta_p^{t-1}, \cdot)$; $p = 1, \ldots, P$; where $K_t$ is a transition kernel with stationary distribution $\pi_t$
end for
```

4 SMC Algorithm for Real-Time Semiparametric Regression

In this section, we provide the full details of our resample-move algorithms generalized semiparametric regression models.

4.1 Choice of Transition Kernel

In this section, we first detail the Gaussian response case, where conjugacy allows us to use a Gibbs sampler for the move step, before considering the non-Gaussian exponential family case in the next subsection where we employ a hybrid (Metropolis-within-Gibbs) sampler.

4.1.1 Gaussian Response

For the Gaussian response case, we have a closed-form for every full conditional distribution in the model, so we may use Gibbs sampling for the move steps, as outlined below.

For $\boldsymbol{\theta}$, note that we may express the prior on the regression coefficients as

$$\boldsymbol{\theta} \sim N(\mathbf{0}, \boldsymbol{\Sigma}), \qquad \boldsymbol{\Sigma} \equiv \text{blockdiag}(\sigma_\beta^2 \mathbf{I}_{q_0}, \sigma_1^2 \mathbf{I}_{q_1}, \ldots, \sigma_J^2 \mathbf{I}_{q_J}).$$

With this prior, standard algebraic manipulations show that the full conditional for $\boldsymbol{\theta}$ is then

$$\boldsymbol{\theta}|\,\text{rest} \sim N\big\{(\mathbf{C}^T\mathbf{C} + \sigma_\epsilon^2 \boldsymbol{\Sigma}^{-1})^{-1}\mathbf{C}^T\mathbf{y}, \sigma_\epsilon^2(\mathbf{C}^T\mathbf{C} + \sigma_\epsilon^2\boldsymbol{\Sigma}^{-1})^{-1}\big\},$$

where σ_ϵ^2 denotes the scale parameter of the Gaussian response.

For the variance components, we use the result (see, e.g., Luts et al. 2014) that a random variable $\sigma \sim$ half-Cauchy(A) has an equivalent representation using inverse-gamma auxiliary variables as

$$\sigma^2|\,a \sim \text{IG}(1/2, 1/a), \quad a \sim \text{IG}(1/2, 1/A^2).$$

Here, $X \sim \text{IG}(A, B)$ refers to a random variable with density $f(x) = B^A \Gamma(A)^{-1} x^{-A-1} \exp(-B/x)$, $x > 0$, where $\Gamma(\cdot)$ denotes the gamma function. Under this representation, the full conditionals are given by

$$\sigma_j^2|\,\text{rest} \sim \text{IG}\big\{(q_j+1)/2, \tfrac{1}{2}\mathbf{u}_j^T\mathbf{u}_j + 1/a_j\big\}, \qquad a_j|\text{rest} \sim \text{IG}(1, 1/\sigma_j^2 + 1/A_j^2), \quad j = 1, \ldots, J;$$

and

$$\sigma_j^2| \,\text{rest} \sim \text{IG}\big\{(t+1)/2, \tfrac{1}{2}(\mathbf{y}-\mathbf{C}\boldsymbol{\theta})^T(\mathbf{y}-\mathbf{C}\boldsymbol{\theta}) + 1/a_\epsilon\big\}, \qquad a_\epsilon|\text{rest} \sim \text{IG}(1, 1/\sigma_j^2 + 1/A_\epsilon^2).$$

4.1.2 Generalized Case

When the response distribution is not Gaussian, the full conditional for the regression coefficients does not have a closed-form. Hence, we cannot use a Gibbs sampling step to update those parameters and instead resort to Metropolis-Hastings.

We chose to use an independent Metropolis-Hastings kernel with proposal density

$$N(\mu_\theta, \epsilon\boldsymbol{\Sigma}_\theta), \quad \mu_\theta \equiv \sum_{p=1}^{P} w_p \boldsymbol{\theta}_p, \quad \boldsymbol{\Sigma}_\theta \equiv \sum_{p=1}^{P} w_p(\boldsymbol{\theta}_p - \mu_\theta)(\boldsymbol{\theta}_p - \mu_\theta)^T,$$

where $\epsilon > 0$ is a user-specified tuning parameter. This propsoal was was also used in Chopin (2002). It is justified asymptotically because for regular models, π_n tends to a Gaussian distribution as $n \to \infty$. Thus, as more data arrives, our moves become more efficient, which is important because, of course, they become more computationally demanding. This proposal is also model-independent and can be easily extended to many response types or covariance structures, $\boldsymbol{\Sigma}$, that encorporate more complex nonparametric terms and different penalization.

Note that it would be inappropriate to use a random walk proposal because this does not allow us to monitor the Monte Carlo mixing through examination of the acceptance rate as is usual for MCMC. This is because the acceptance rate does not take into account the possible deterioration in particle weights; it does not matter if a particle with zero weight is mixing well.

For a specific example considered in our numerical studies, logistic regression, we have $p(\mathbf{y}|\,\boldsymbol{\theta}) = \exp[\mathbf{y}^T\mathbf{C}\boldsymbol{\theta} - \mathbf{1}^T\log\{1+\exp(\mathbf{C}\boldsymbol{\theta})\}]$, so that the posterior for $\boldsymbol{\theta}$ is given by

$$p(\boldsymbol{\theta}|\,\text{rest}) \propto p(\mathbf{y}|\,\boldsymbol{\theta})p(\boldsymbol{\theta}|\,\boldsymbol{\Sigma}) \propto \exp\big[\mathbf{y}^T\mathbf{C}\boldsymbol{\theta} - \mathbf{1}^T\log\{1+\exp(\mathbf{C}\boldsymbol{\theta})\}\big]\exp\big[-\tfrac{1}{2}\boldsymbol{\theta}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\theta}\big].$$

The acceptance probability for a proposal $\boldsymbol{\theta}'$ when the current value is $\boldsymbol{\theta}_p$ is

$$r \equiv \min\left\{\frac{p(\boldsymbol{\theta}'|\,\text{rest})\phi(\boldsymbol{\theta}_p;\boldsymbol{\mu}_\theta,\boldsymbol{\Sigma}_\theta)}{p(\boldsymbol{\theta}_p|\,\text{rest})\phi(\boldsymbol{\theta}';\boldsymbol{\mu}_\theta,\boldsymbol{\Sigma}_\theta)}, 1\right\} = \min\big\{\exp\big(\mathbf{y}^T\mathrm{C}[\boldsymbol{\theta}'-\boldsymbol{\theta}_p] - \mathbf{1}^T\log[1+\exp\{\mathrm{C}(\boldsymbol{\theta}'-\boldsymbol{\theta}_p)\}]\big)$$
$$\times\ \exp\big(-\tfrac{1}{2}[\boldsymbol{\theta}'-\boldsymbol{\theta}_p]^T[\boldsymbol{\Sigma}_p^{-1} - \boldsymbol{\Sigma}_\theta^{-1}][\boldsymbol{\theta}'-\boldsymbol{\theta}_p]\big), 1\big\}.$$

For the variance components, we can use half-Cauchy priors and Gibbs sampling as detailed in the previous section for the Gaussian response case. Another possibility, considered in Chopin (2002), is to update the log-variance components along with $\boldsymbol{\theta}$ using the independent Metropolis-Hastings kernel just described.

4.2 Initial Values and Validation Period

Owing to the adaptive importance distribution used, it is necessary to initialize the algorithm with starting values drawn from a partial posterior π_{t_w} for some $t_w > 0$ to avoid degeneracy in the early stages of the algorithm (Chopin 2002). To do this, we first fit the model in batch to an intial set of data of size n_w using the NO-U-Turn sampler of Hoffman and Gelman (2014). We refer to this period as the "warm-up".

We also use a short validation period of size t_v where we use visual diagnostics to compare the fit from the SMC algorithm to the batch fits at fixed intervals of the validation period. A similar approach was used in Luts et al. (2014). For example, the user may specify $t_w = 500$ and $t_v = 100$ with five checks during the validation period. In this example, the SMC algorithm would be initialized using values determined from MCMC on the data set of size 500, then SMC would be run over the validation period until $t = 600$, at which point the user would visually compare the SMC results with the results of batch MCMC for sample sizes of $520, 540, 560, 580,$ and 600; and if necessary, increase the warm-up period and/or the number of particles.

An example plot is shown in Figure 1 involving simulated data with a binary response, one linear predictor, z, and one nonlinear predictor, x. Point estimates and confidence bands are plotted for the SMC and batch MCMC estimates at each of the sample sizes $t = 520, 540, 560, 580, 600$. In this example, the length of the warm-up appears to be satisfactory as there is good agreement between the SMC and MCMC estimates.

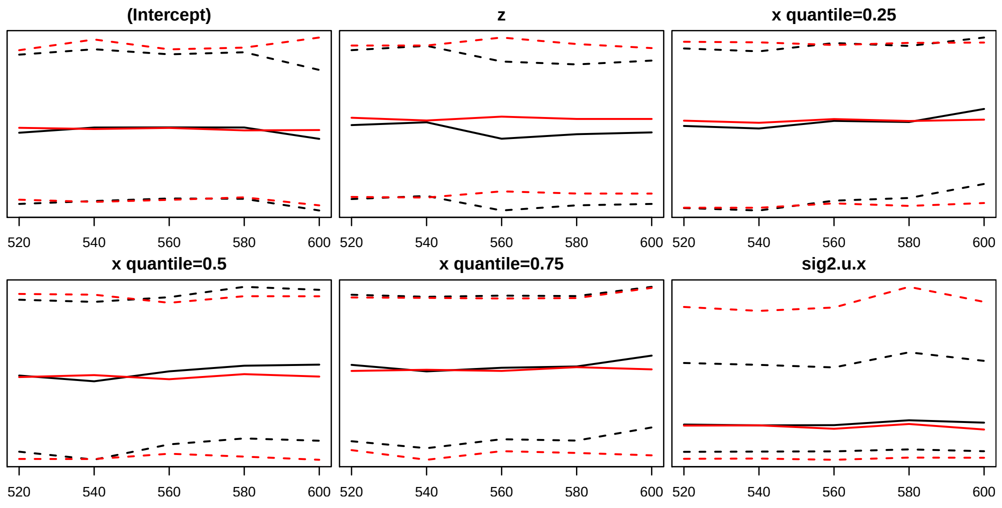

Figure 1: Example validation diagnostic plot for logistic additive model with one linear predictor, z, and one nonlinear predictor, x.

4.3 Resampling and Reweighting

We use stratified resampling for the resampling step in our algorithms. This has been demonstrated to have superior performance to multinomial resampling (Douc et al. 2005; Hol et al. 2006). The steps needed to do the resampling are provided in Algorithm 2.

In the Gaussian additive model case when we use Gibbs sampling for the move steps, reweighting proceeds exactly as outlined in Section 3. When using an independent Metropolis-Hastings kernel, as we do in the logistic regression case, the reweighting step must be modified to take into account the (in)effectiveness of the move step, as discussed in Chopin (2002).

To accomplish this, unmoved replicate particles are replaced by a single particle with weight proportional to the number of unmoved replicates. For example, if $\boldsymbol{\theta}_p$ has n_p replicates after resampling, but $n'_p \leq n_p$ of those particles have not changed after the move step, then those n'_p particles are replaced by a single particle with weight proportional to n'_p. This gives a more accurate representation of the level of degeneracy, higher values for the empirical variance of the weights, and more instances of resampling when needed. It also avoids unnecessary repitition of computations for identical particles.

4.4 Full Algorithms

The algorithm for the Gaussian response case is given in Algorithm 2. We write $\theta \leftarrow \pi(\cdot)$ to denote that the parameter θ is updated using a random draw from the density $\pi(\cdot)$. Note that each parameter gains a subscript p in the algorithm to denote the dependence on particle index. We use $\mathbf{c}_t^T$ to denote the tth row of $\mathbf{C}$.

The algorithm for the generalized case is provided in Algorithm 3.

5 Experimental Results

In this section, we evaluate the performance of our SMC algorithms on both real and synthetic data. We compare our algorithms with online MFVB and also with refitting the model at each time step using MCMC. Though the latter approach is computationally infeasible in most streaming data settings when estimates are needed in real-time, we include it here as a sort of oracle because we can use it to achieve arbitrarily accurate answers by running a long enough Markov chain.

All data analysis was conducted in `R` (R Core Team 2016). Our SMC algorithms are available in an `R` package, which can be obtained from the first author. Batch MCMC was performed using the `R` package `rstan` (Stan Development Team 2016). The online MFVB algorithms of Luts et al. (2014) where run using code provided by the authors.

5.1 Synthetic Data

5.1.1 Simple Logistic Model

We begin with a simple logistic regression example. Our aim here is to highlight a possible defiency in the MFVB solution presented in Luts et al. (2014). We generate data from the model

$$x \sim \text{Unif}(0,1), \quad Y \,|\, x \sim \text{Bernoulli}(\text{logit}^{-1}(\beta_0 + \beta_1 x)),$$

Algorithm 2 Gaussian additive model SMC algorithm

```
Initialize: Inputs $P, \tau, t_w, t_v, \sigma^2_\beta, A_\epsilon, A_u$

    for $p = 1, \ldots, P$ do
        $\boldsymbol{\theta}_p \leftarrow \pi_{t_w}(\boldsymbol{\theta}|\,\text{rest})$; $\sigma^2_{j,p} \leftarrow \pi_{t_w}(\sigma^2_j|\,\text{rest})$; $a_{j,p} \leftarrow \pi_{t_w}(a_j|, \text{rest})$; $j = 1, \ldots, J$;
        $\sigma^2_{\epsilon,p} \leftarrow \pi_{t_w}(\sigma^2_\epsilon|\,\text{rest})$; $a_{\epsilon,p} \leftarrow \pi_{t_w}(a_\epsilon|, \text{rest})$; $j = 1, \ldots, J$; $w_p \leftarrow P^{-1}$
    end for
for $t = t_w + 1, 2, \ldots$ do
    Reweight: $w_p \leftarrow w_p p(y_t|\,\boldsymbol{\theta}_p, \sigma^2_{\epsilon,p})$; $p = 1, \ldots, P$
    Normalize: $\mathbf{w} \leftarrow \mathbf{w}/\sum_{p=1}^P w_p$
    if $\text{ESS}(\mathbf{w}) < \tau P$ then
        Resample:
            $v'_p \leftarrow \text{Unif}(0, 1)$; $p = 1, \ldots P$
            $v_p \leftarrow (p - 1 + v'_p)/P$; $p = 1, \ldots P$
            $i_p \leftarrow i$ s.t. $v_p \in \Big[\sum_{p=1}^{i-1} w_p, \sum_{p=1}^{i} w_p\Big)$; $p = 1, \ldots P$
            $\{\boldsymbol{\theta}_p, w_p\} \leftarrow \{\boldsymbol{\theta}_{i_p}, P^{-1}\}$; $p = 1, \ldots P$
    end if
    Move:
        $\mathbf{C} \leftarrow [\mathbf{C}^T : \mathbf{c}_t]^T$;    $\mathbf{y} \leftarrow (\mathbf{y}^T, y_t)^T$
        for $p = 1, \ldots, P$ do
            $\boldsymbol{\theta}_p \leftarrow N\big\{(\mathbf{C}^T\mathbf{C} + \sigma^2_{\epsilon,p}\boldsymbol{\Sigma}_p^{-1})^{-1}\mathbf{C}^T\mathbf{y}, \sigma^2_{\epsilon,p}(\mathbf{C}^T\mathbf{C} + \sigma^2_{\epsilon,p}\boldsymbol{\Sigma}_p^{-1})^{-1}\big\}$
            for $j = 1, \ldots, J$ do
                $a_{j,p} \leftarrow \text{IG}(1, 1/\sigma^2_{j,p} + 1/A^2_j)$
                $\sigma^2_{j,p} \leftarrow \text{IG}\big\{(q_j + 1)/2, \frac{1}{2}\mathbf{u}^T_{j,p}\mathbf{u}_{j,p} + 1/a_{j,p}\big\}$
            end for
            $a_{\epsilon,p} \leftarrow \text{IG}(1, 1/\sigma^2_{\epsilon,p} + 1/A^2_\epsilon)$
            $\sigma^2_{\epsilon,p} \leftarrow \text{IG}\big\{(t + 1)/2, \frac{1}{2}(\mathbf{y} - \mathbf{C}\boldsymbol{\theta}_p)^T(\mathbf{y} - \mathbf{C}\boldsymbol{\theta}_p) + 1/a_{\epsilon,p}\big\}$
        end for
    Validation:
        if $t = t_w + t_v$ then
            Use visual diagnostics to compare current estimates from SMC with batch MCMC
            if SMC not converging then increase $P$ and/or $t_w$ and restart algorithm end if
        end if
end for
```

Algorithm 3 Generalized additive model SMC algorithm

```
**Initialize:** Inputs $P, \tau, t_w, t_v, \epsilon, \sigma^2_\beta, A_u$

    **for** $p = 1, \dots, P$ **do**
        $\boldsymbol{\theta}_p \leftarrow \pi_{t_w}(\boldsymbol{\theta} | \text{rest})$; $\sigma^2_{j,p} \leftarrow \pi_{t_w}(\sigma^2_j | \text{rest})$; $a_{j,p} \leftarrow \pi_{t_w}(a_j |, \text{rest})$; $j = 1, \dots, J$; $w_p \leftarrow P^{-1}$
    **end for**
**for** $t = t_w + 1, 2, \dots$ **do**
    **Reweight:** As described in Section 4.3
    **Normalize:** $\mathbf{w} \leftarrow \mathbf{w} / \sum_{p=1}^P w_p$
    **if** $\text{ESS}(\mathbf{w}) < \tau P$ **then**
        **Resample:** As described in Algorithm 2
    **end if**
    **Move:**
        $\mathbf{C} \leftarrow [\mathbf{C}^T : \mathbf{c}_t]^T$;   $\mathbf{y} \leftarrow (\mathbf{y}^T, y_t)^T$;   $\boldsymbol{\mu}_\theta \leftarrow \sum_{p=1}^P w_p \boldsymbol{\theta}_p$;   $\boldsymbol{\Sigma}_\mu \leftarrow \sum_{p=1}^P w_p (\boldsymbol{\theta}_p - \boldsymbol{\mu}_\theta)(\boldsymbol{\theta}_p - \boldsymbol{\mu}_\theta)^T$
        **for** $p = 1, \dots, P$ **do**
            $\boldsymbol{\theta}' \leftarrow N(\boldsymbol{\mu}_\theta, \epsilon \boldsymbol{\Sigma}_\theta)$;   $r \leftarrow p(\boldsymbol{\theta}' | \text{rest}) \phi(\boldsymbol{\theta}_p; \boldsymbol{\mu}_\theta, \boldsymbol{\Sigma}_\theta) / \{p(\boldsymbol{\theta}_p | \text{rest}) \phi(\boldsymbol{\theta}'; \boldsymbol{\mu}_\theta, \boldsymbol{\Sigma}_\theta)\}$;   $u \leftarrow \text{Unif}(0, 1)$
            **if** $u < \min(r, 1)$ **then**
                $\boldsymbol{\theta}_p \leftarrow \boldsymbol{\theta}'$
            **end if**
            **for** $j = 1, \dots, J$ **do**
                $a_{j,p} \leftarrow \text{IG}(1, 1/\sigma^2_{j,p} + 1/A^2_j)$;   $\sigma^2_{j,p} \leftarrow \text{IG}\big\{(q_j + 1)/2, \tfrac{1}{2}\mathbf{u}^T_{j,p}\mathbf{u}_{j,p} + 1/a_{j,p}\big\}$
            **end for**
        **end for**
    **Validation:** As described in Algorithm 2
**end for**
```

where we consider $\beta \equiv 16.1$ and $\beta_1 \equiv -19.05$. For these values of β_0 and β_1, MFVB as implemented in Luts et al. (2014) is inappropriate due to the high amount of posterior correlation between the parameters.

We use $t_w = 100$ for both MFVB and SMC warm-up. We apply Algorithm 3 with $P = 10000$ and $\tau = 0.5$ to obtain density estimates for β_0 and β_1 at each time step. We run batch MCMC for 2500 iterations with the first 1000 iterations being used for warm-up.

Results for a representative simulated data set are shown in Figure 2, where we plot density estimates from SMC, MFVB, and batch MCMC at sample sizes $N = 600, 1100, 1600, 2100$. As expected, we see that MFVB struggles in this scenario. We also observe close agreement between SMC and batch MCMC.

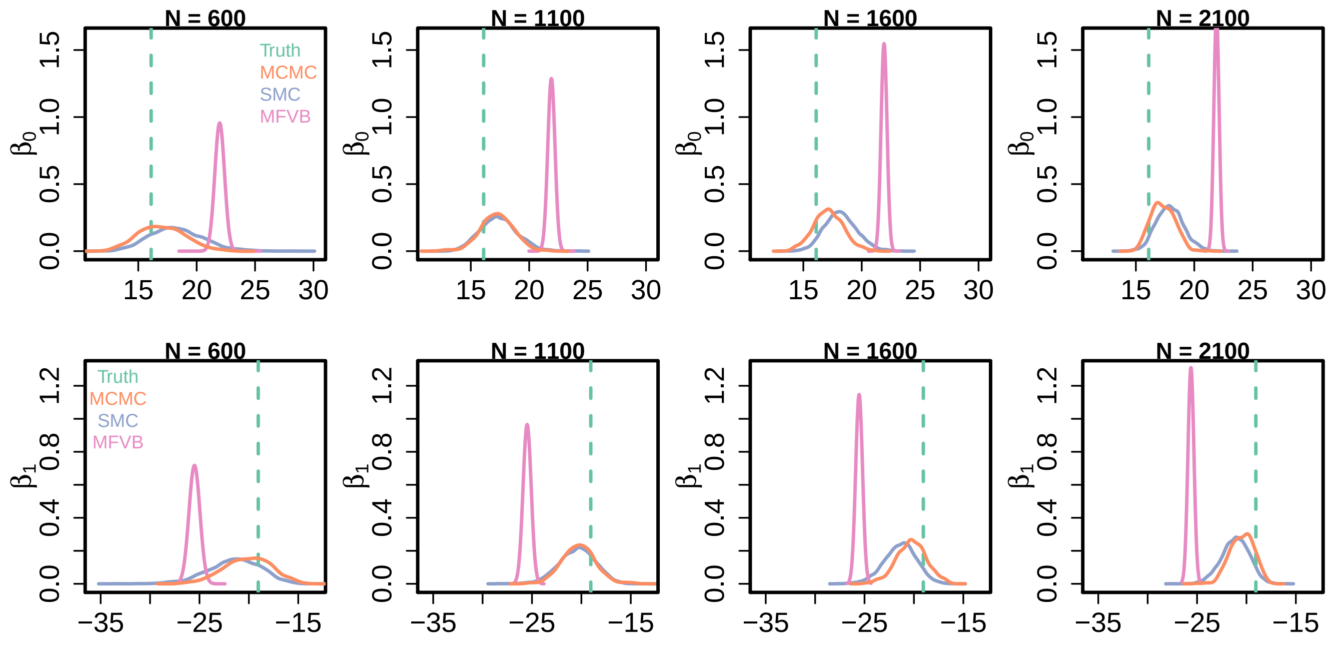

Figure 2: Density estimates for β_0 (top row) and β_1 (bottom row) for SMC, online MFVB, and batch MCMC at various sample sizes. True values are shown as dashed vertical lines.

5.1.2 Gaussian Penalized Spline

In this example, we generate synthetic data as in Luts et al. (2014) to assess the performance of Algorithm 2. The data are generated according to

$$x_1 \sim \text{Bernoulli}(1/2), \quad x_2, x_3 \sim \text{Unif}(0,1), \quad Y \,|\, x_1, x_2, x_3 \sim N(\beta_1 x_1 + f_2(x_2) + f_3(x_3), \sigma_\epsilon^2),$$

where $\beta_1 \equiv 0.2$, $f_1(x) = \cos(4\pi x) + 2x$, $f_2(x) = \sin(2\pi x^2)$, and $\sigma_\epsilon^2 = 1$.

Luts et al. (2014) already demonstrated that the mean field approximation was reasonable and that MFVB performs well in this scenario. In applying Algorithm 2, we used a warm-up of size 500 and set $P = 10000$ and $\tau = 0.5$.

Figure 3 compares the results of running our SMC Algorithm 2 with online MFVB and batch MCMC at sample sizes of $N = 1000, 2000, 3000,$ and 4000 for a representative simulation. Good agreement is shown between the estimates from batch MCMC and those from the MFVB and SMC methods.

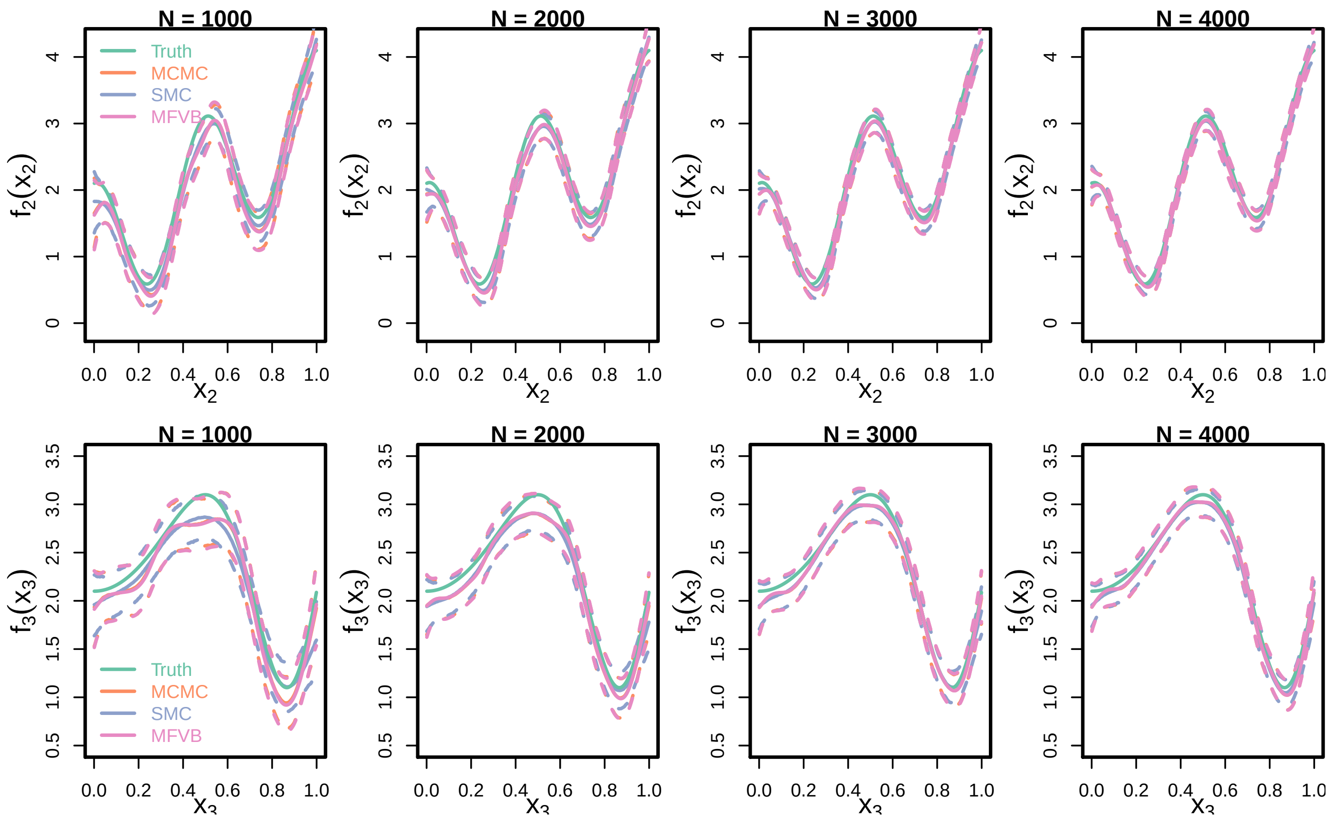

Figure 3: Comparison of SMC, online MFVB, and batch MCMC fits for simulated real-time Gaussian semiparametric regression.

5.1.3 Poisson Nonparametric Regression

Finally, we consider the simulation scenario used in Luts and Wand (2015) to validate the use of Algorithm 3 on count data. Data are generated according to

$$x \sim \text{Unif}(0, 1), \quad Y|\, x \sim \text{Pois}[\exp\{\cos(4\pi x) + 2x\}].$$

We use $t_w = 100$ for warm-up and apply Algorithm 3 with $P = 10000$ and $\tau = 0.5$. A comparison with batch MCMC and MFVB fits at several time steps is displayed in Figure 4. We see close agreement between SMC and MCMC with no signs of degeneracy in the SMC approximation at these sample sizes.

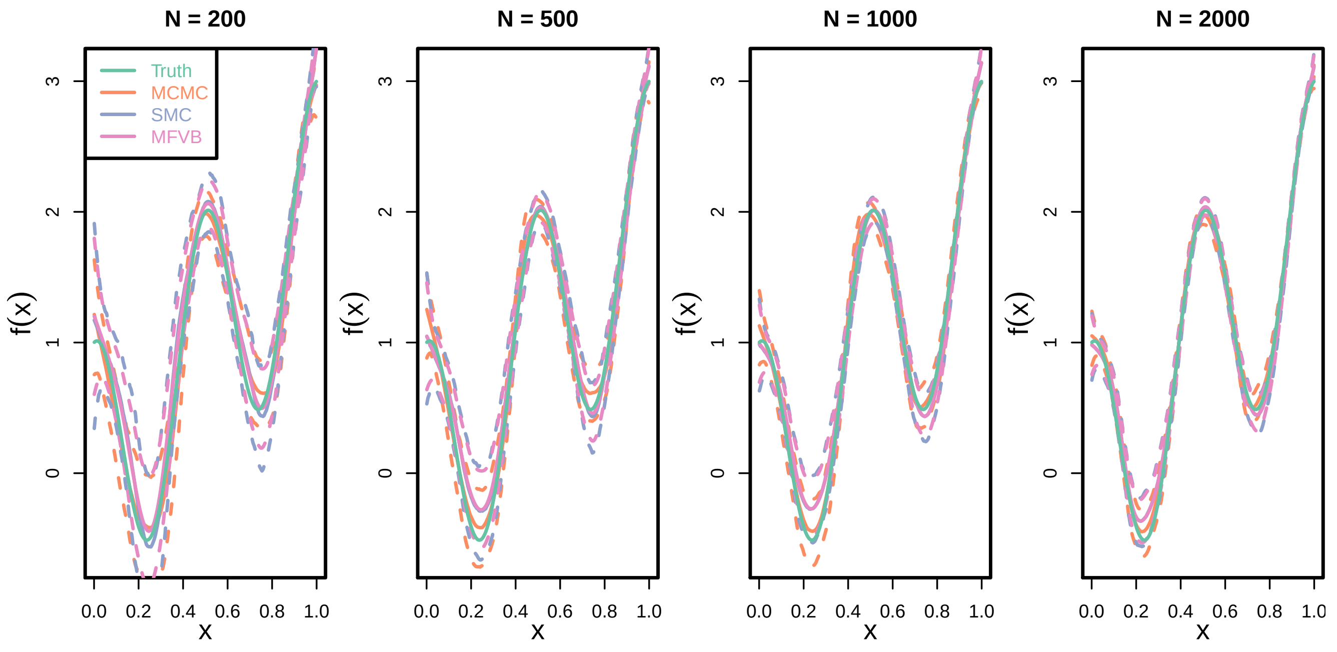

Figure 4: Comparison of SMC, online MFVB, and batch MCMC fits for simulated real-time Poisson nonparametric regression.

5.2 Fitbit® Data

Wearable activity monitors, such as the Fitbit®, have grown in popularity recently as their cost, size, and accuracy has increased. This has lead to a wealth of data and research, with the hope that wearing these monitors to assess physical activity will actually lead to improvements in the wearer's health and fitness. Whether this is actually the case remains an open area of research (Patel et al. 2015). The Fitbit® is one of the most popular consumer physical activity trackers available. It has been demonstrated to be an accurate and reliable measure of physical activity (Diaz et al. 2015).

In this section we apply Algorithm 3 to data from a Fitbit®. We envision a scenario where the wearer of the activity tracker wants real-time modelling of their physical activity, with Bayesian inferences being made in an online fashion on their wearable device. Thus, we have a limited computational budget and batch MCMC at each time interval is not a possibility.

The data come from a single Fitbit® reporting the number of steps the wearer has taken in five-minute intervals. The average number of steps for each day of the week is shown in Figure 5. The data set is included in the `R` package that implements our algorithms.

As a proof of principal, we consider a two-predictor model with a goal of future work being to refine the model after consultation with health sciences experts. We consider the Poisson log-linear model

$$\text{steps}_{ij} \sim \text{Pois}\Big[\exp\{\text{night}_{ij} + f(\text{time.of.day}_{ij})\}\Big], \tag{3}$$

where steps_{ij} is the step count reading on day i at five-minute interval j, night_{ij} is an indicator variable for time

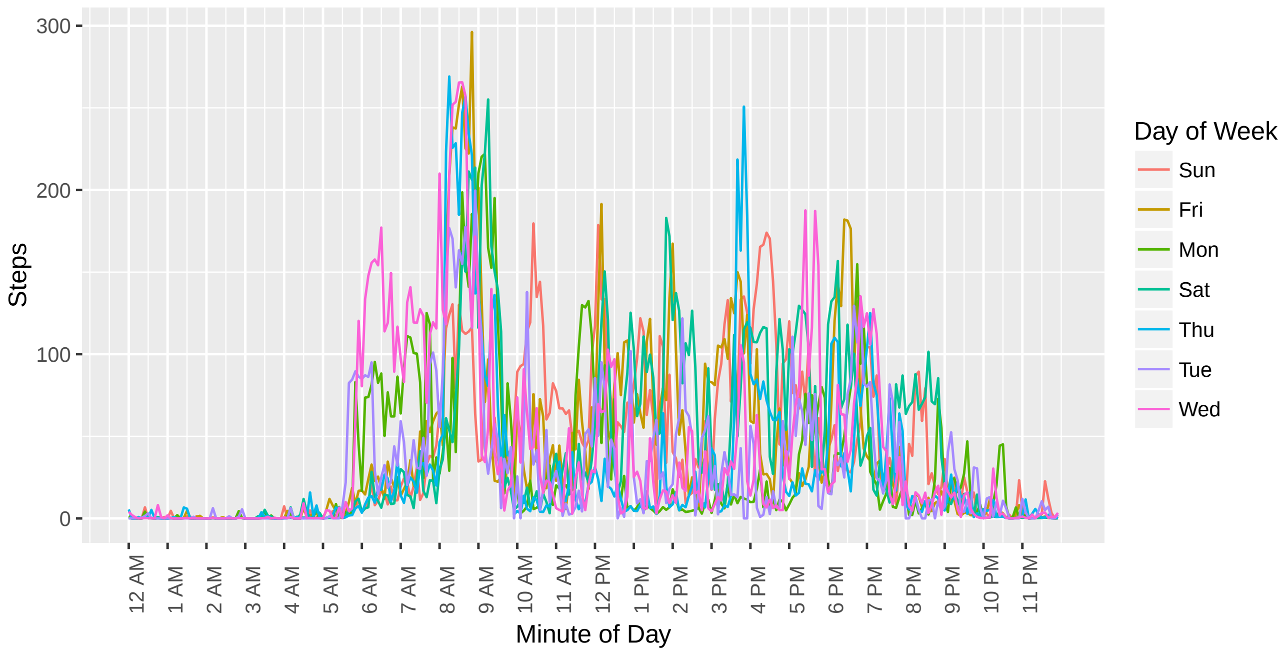

Figure 5: Average steps for each day of week for Fitbit® data.

of day being between 9PM and 6AM, and time.of.day$_{ij}$ is the minute of the day standardized to have mean zero and variance one.

For the batch MCMC, we run the No-U-Turn Sampler for 5000 iterations, with a warm-up of size 2000. As before, we use $P = 10000$ and $\tau = 0.5$ in Algorithm 3. We warm-up using one week of data and run the algorithm on the second week of data.

Figure 6 plots the estimate $\widehat{f}$ of f in (3), using both Algorithm 3, online MFVB, and batch MCMC at 12 hour intervals during the second week of using the activity tracker.

Agreement between SMC and batch MCMC is generally good. We see some minor evidence of degeneracy as their is an increase in disagreement between the fits on later days. The disagreement is mostly at the start and end of the time interval where penalized splines are known to have more bias. We also point out that given the cyclic nature of the covariate, it would be more appropriate to use a cyclic penalized spline (e.g., Wood 2006) as opposed to the O'Sullivan splines we implemented for this work. We see greater disagreement between online MFVB and batch MCMC, and note that disagreement also increases as the time window increases. It is especially significant in the early AM when the subject is most likely sleeping.

6 Conclusion

We have shown that SMC methods may be applied to fit semiparametric regression models in a streaming data setting. We demonstrated on synthetic data that SMC methods can be applied with a high degree of accuracy for semiparametric models with Gaussian, binary, and count responses. We see SMC methods as offering a

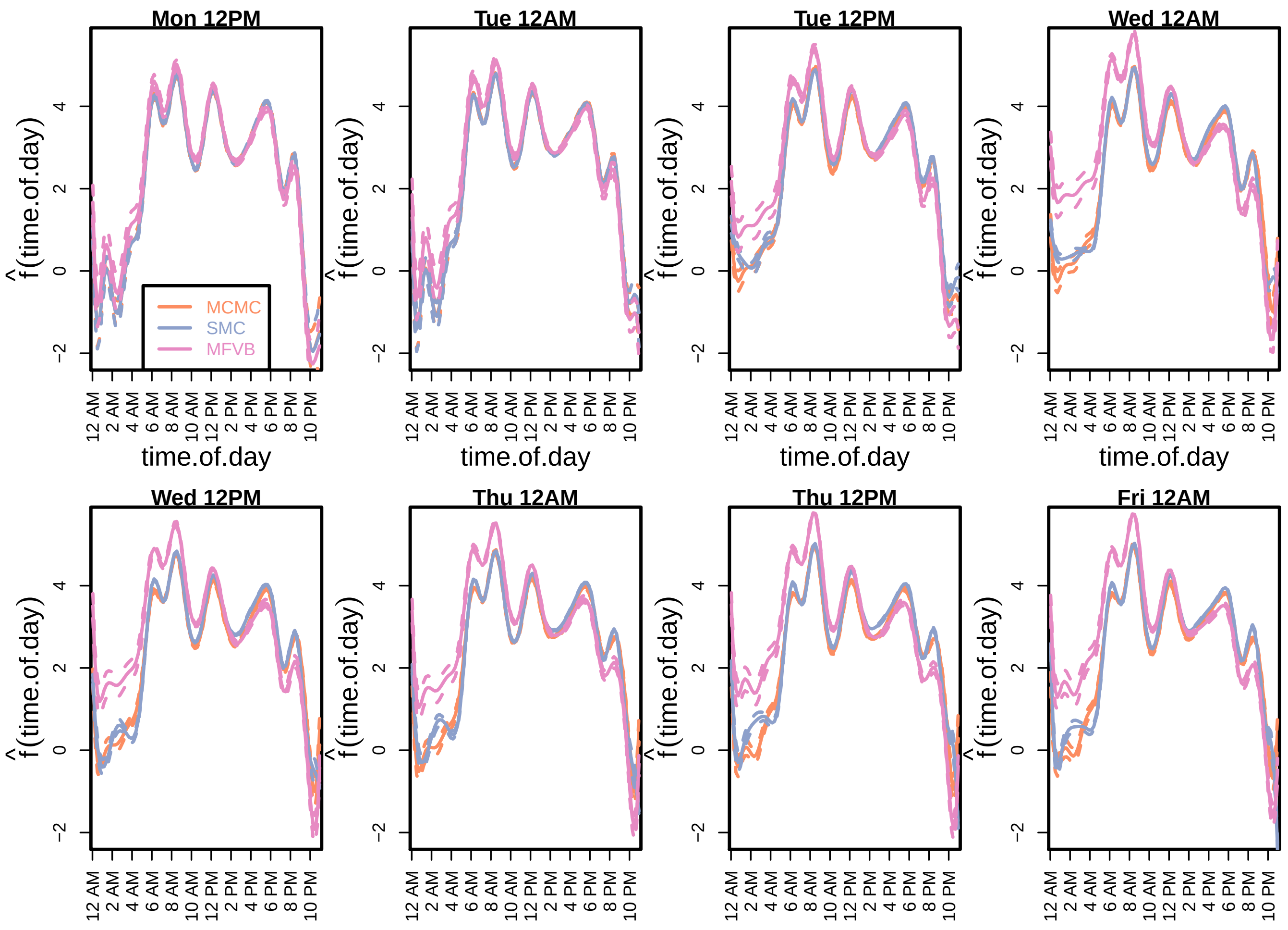

Figure 6: Comparison of SMC, online MFVB, and batch MCMC fits for Fitbit® data.

middle ground in the trade-off between speed/accuracy of MFVB and batch MCMC.

Future work will explore developing a more realistic model for the step count data to include additional covariates, account for overdispersion in the counts, and experiment with other types of splines that would be more appropriate for the spiky, cyclic nature of the data. Including additional covariates and larger datasets motivate further research effort to reduce the computational budget. This research provides an important proof of concept toward this goal.

Acknowledgements

This research was partially supported by Australian Research Council Discovery Project DP140100441 and the Australian Research Council (ARC) Centre of Excellence for Mathematical and Statistical Frontiers.

References

Chopin, N. (2002). “A sequential particle filter method for static models”. *Biometrika* 89 (3), pp. 539–552.

Del Moral, P., A. Doucet, and A. Jasra (2006). “Sequential Monte Carlo samplers”. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 68 (3), pp. 411–436.

Diaz, K. M., D. J. Krupka, M. J. Chang, J. Peacock, Y. Ma, J. Goldsmith, J. E. Schwartz, and K. W. Davidson (2015). “Fitbit®: An accurate and reliable device for wireless physical activity tracking.” *International journal of cardiology* 185, pp. 138–140.

Douc, R., O. Cappé, and E. Moulines (2005). “Comparison of resampling schemes for particle filtering”. In: *ISPA 2005. Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis, 2005*. IEEE, pp. 64–69. DOI: 10.1109/ISPA.2005.195385.

Doucet, A., N. de Freitas, and N. Gordon, eds. (2001). *Sequential Monte Carlo Methods in Practice*. New York: Springer-Verlag. ISBN: 0387951466.

Fan, Y., D. S. Leslie, and M. P. Wand (2008). “Generalised linear mixed model analysis via sequential Monte Carlo sampling”. *Electronic Journal of Statistics* 2, pp. 916–938.

Gelman, A. (2006). “Prior distributions for variance parameters in hierarchical models (comment on article by Browne and Draper)”. *Bayesian analysis* 1 (3), pp. 515–534. DOI: 10.1214/06-BA117A.

Gerber, M. and N. Chopin (2015). “Sequential quasi Monte Carlo”. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 77 (3), pp. 509–579.

Gilks, W. R. and C. Berzuini (2001). “Following a moving target—Monte Carlo inference for dynamic Bayesian models”. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 63 (1), pp. 127–146.

Gordon, N. J., D. J. Salmond, and A. F. Smith (1993). “Novel approach to nonlinear/non-Gaussian Bayesian state estimation”. *IEE Proceedings F - Radar and Signal Processing* 140 (2), pp. 107–113.

Hastie, T. and R. Tibshirani (1986). “Generalized additive models”. *Statistical Science* 1 (3), pp. 297–318.

Hoffman, M. D. and A. Gelman (2014). “The No-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo.” *Journal of Machine Learning Research* 15 (1), pp. 1593–1623.

Hol, J. D., T. B. Schon, and F. Gustafsson (2006). “On resampling algorithms for particle filters”. In: *Nonlinear Statistical Signal Processing Workshop, 2006 IEEE*. IEEE, pp. 79–82. DOI: 10.1109/NSSPW.2006.4378824.

Kitagawa, G. (1996). “Monte Carlo filter and smoother for non-Gaussian nonlinear state space models”. *Journal of Computational and Graphical Statistics* 5 (1), pp. 1–25.

Lee, A., C. Yau, M. B. Giles, A. Doucet, and C. C. Holmes (2010). “On the utility of graphics cards to perform massively parallel simulation of advanced Monte Carlo methods”. *Journal of computational and graphical statistics* 19 (4), pp. 769–789.

Luts, J. and M. P. Wand (2015). “Variational inference for count response semiparametric regression”. *Bayesian Analysis* 10 (4), pp. 991–1023. DOI: 10.1214/14-BA932.

Luts, J. (2015). “Real-time semiparametric regression for distributed data sets”. *IEEE Transactions on Knowledge and Data Engineering* 27 (2), pp. 545–557.

Luts, J., T. Broderick, and M. P. Wand (2014). “Real-time semiparametric regression”. *Journal of Computational and Graphical Statistics* 23 (3), pp. 589–615.

O’Sullivan, F. (1986). “A statistical perspective on ill-posed inverse problems”. *Statistical Science* 1 (4), pp. 502–527.

Patel, M. S., D. A. Asch, and K. G. Volpp (2015). “Wearable devices as facilitators, not drivers, of health behavior change”. *Jama* 313 (5), pp. 459–460.

R Core Team (2016). *R: A Language and Environment for Statistical Computing*. Version 3.4.0. Vienna, Austria: R Foundation for Statistical Computing. URL: `https://www.R-project.org/`.

Ruppert, D., M. P. Wand, and R. J. Carroll (2003). *Semiparametric Regression*. Cambridge University Press. ISBN: 0521785162.

Speed, T. (1991). "[That BLUP is a good thing: the estimation of random effects]: comment". *Statistical Science* 6 (1), pp. 42–44. DOI: `10.1214/ss/1177011930`.

Stan Development Team (2016). *Stan Modeling Language User's Guide and Reference Manual*. Version 2.12.1. URL: `http://mc-stan.org/` (visited on 2016-09-22).

Wainwright, M. J. and M. I. Jordan (2008). "Graphical models, exponential families, and variational inference". *Foundations and Trends® in Machine Learning* 1 (1-2), pp. 1–305.

Wand, M. P. and J. T. Ormerod (2008). "On semiparametric regression with O'Sullivan penalized splines". *Australian & New Zealand Journal of Statistics* 50 (2), pp. 179–198.

Wood, S. N. (2006). *Generalized Additive Models: An Introduction with R*. CRC Press. ISBN: 9781584884743.