Chapter 1

MCMC for State Space Models

Paul Fearnhead

1.1 Introduction: State-space models

In this chapter we look at MCMC methods for a class of time-series models, called state-space models. The idea of state-space models is that there is an unobserved state of interest the evolves through time, and that partial observations of the state are made at successive time-points. We will denote the state by X and observations by Y, and assume that our state space model has the following structure:

$$X_t|\{x_{1:t-1}, y_{1:t-1}\} \sim p(x_t|x_{t-1}, \theta), \tag{1.1.1}$$

$$Y_t|\{x_{1:t}, y_{1:t-1}\} \sim p(y_t|x_t, \theta). \tag{1.1.2}$$

Here, and throughout, we use the notation $x_{1:t} = (x_1, \dots, x_t)$, and write $p(\cdot|\cdot)$ for a generic conditional probability density or mass function (with the arguments making it clear which conditional distribution it relates to). To fully define the distribution of the hidden state we further specify an initial distribution $p(x_1|\theta)$. We have made explicit the dependence of the model on an unknown parameter θ, which may be multi-dimensional. The assumptions in this model are that, conditional on the parameter θ, the state model is Markov, and that we have a conditional independence property for the observations: observation Y_t only depends on the state at that time, X_t.

For concreteness we give three examples of state-space models:

Example 1: Stochastic Volatility

The following simple stochastic volatility model has been used for modelling the time-varying variance of log-returns on assets. For fuller details see Hull and White (1987) and Shephard (1996). The state-space model is

$$X_t|\{x_{1:t-1}, y_{1:t-1}\} \sim \mathrm{N}(\phi x_{t-1}, \sigma^2),$$

where $|\phi| < 1$, and with initial distribution $X_1 \sim \mathrm{N}(0, \sigma^2/(1-\phi^2))$, and

$$Y_t|\{x_{1:t}, y_{1:t-1}\} \sim \mathrm{N}(0, \beta^2 \exp\{x_t\}).$$

The parameters of the model are $\theta = (\beta, \phi, \sigma)$. The idea of the model is that the variance of the observations depends on the unobserved state, and the unobserved state is modelled by an AR(1) process.

Example 2: Discrete Hidden Markov Model

A general class of models occurs when the underlying state is a discrete-valued Markov model, with a finite state-space. Thus we can assume without loss of generality that $X_t \in \{1, 2, \dots, K\}$ and that the model for the dynamics of the state (1.1.1) is defined by a $K \times K$ transition matrix P. Thus for all $i, j \in \{1, \dots, K\}$:

$$\Pr(X_t = j|X_{t-1} = i, x_{1:t-2}, y_{1:t-1}) = P_{ij}. \tag{1.1.3}$$

Usually it is assumed that the distribution for X_1 is given by the stationary distribution of this Markov chain. The observation equation (1.1.2) will depend on the application, but there will be K observation regimes (depending on the value of the state). Thus we can write

$$Y_t|\{x_t = k, x_{1:t-1}, y_{1:t-1}\} \sim f_k(y_t|\theta). \tag{1.1.4}$$

The parameters of this model will be the parameters of (1.1.4) and the parameters of the transition matrix P.

Examples of such models include models of Ion-channels (Ball and Rice, 1992; Hodgson, 1999), DNA sequences (Boys et al., 2000), and speech (Juang and Rabiner, 1991).

Example 3: Changepoint Model

Changepoint models partition the data into homogeneous regions. The model for the data is the same within each region, but differs across regions. Changepoint models have been used for modelling stock prices (Chen and Gupta, 1997), climatic time-series (Beaulieu et al., 2007; Lund and Reeves, 2002), DNA sequences (Didelot et al., 2007; Fearnhead, 2008) and neuronal activity in the brain (Ritov et al., 2002), amongst many other applications.

A simple changepoint model can be described as a state-space model with the following state equation:

$$X_t|\{x_{1:t-1}, y_{1:t-1}\} = \begin{cases} x_{t-1} & \text{with probability } 1-p \\ Z_t & \text{otherwise,} \end{cases}$$

where the Z_ts are iid random variables with density function $p_Z(\cdot|\phi)$. Initially $X_1 = Z_1$, and the observation equation is given by

$$Y_t|\{x_{1:t}, y_{1:t-1}\} \sim p(y_t|x_t).$$

The parameters of this model are $\theta = (p, \phi)$, where p governs the expected number of changepoints in the model, and ϕ the marginal distribution for the state at any time.

We will focus on models for which we can calulate, for any $t < s$

$$Q(t,s) = \int \left(\prod_{i=t}^{s} p(y_i|x) \right) p_Z(x|\phi) \mathrm{d}x. \tag{1.1.5}$$

This is the marginal likelihood of the observations $y_{t:s}$, given that the observations come from a single segment. The functions $Q(t,s)$ depend on ϕ, but for notational convenience we have suppressed this.

1.2 Bayesian analysis and MCMC framework

Our aim is to perform Bayesian inference for a state-space model given data $y_{1:n}$. We assume a prior for the parameters, $p(\theta)$, has been specified, and we wish to obtain the posterior of the parameters $p(\theta|y_{1:n})$, or in some cases we may be interested in the joint distribution of the state and the parameters $p(\theta, x_{1:n}|y_{1:n})$.

How can we design an MCMC algorithm to sample from either of these posterior distributions? In both cases, this can be achieved using data augmentation (Hobert, 2008). That is we design a Markov chain whose state is $(\theta, X_{1:n})$, and whose stationary distribution is $p(\theta, x_{1:n}|y_{1:n})$ (samples from the marginal posterior $p(\theta|y_{1:n})$ can be obtained from samples from $p(\theta, x_{1:n}|y_{1:n})$ just by discarding the $x_{1:n}$ component of each sample). The reason for designing an MCMC algorithm on this state-space is that, for state-space models of the form (1.1.1–1.1.2), we can write down the stationary distribution of the MCMC algorithm up to proportionality:

$$p(\theta, x_{1:n}|y_{1:n}) \propto p(\theta)p(x_1|\theta)\left(\prod_{t=2}^{n} p(x_t|x_{t-1},\theta)\right)\left(\prod_{t=1}^{n} p(y_t|x_t,\theta)\right). \tag{1.2.1}$$

Hence it is straightforward to use standard moves within our MCMC algorithm.

In most applications it is straight-forward to implement an MCMC algorithm with (1.2.1) as its stationary distribution. A common approach is to design moves that update θ conditional on the current values of $X_{1:n}$ and then update $X_{1:n}$ conditional on θ. We will describe various approaches within this framework. We first focus on the problem of updating the state; and to evaluate different methods will consider models where θ is known. Secondly we will consider moves to update the parameters.

1.3 Updating the state

The simplest approach to update the state $X_{1:n}$ is to update its components one at a time. Such a move is called a *single-site update*. While easy to implement, this move can lead to slow mixing if there is strong temporal dependence in the state process. In these cases it is better to update blocks of state components, $X_{t:s}$, or the whole state process $X_{1:n}$ in a single move. (As we will see, in some cases it is possible to update the whole process $X_{1:n}$ directly from its full-conditional distribution $p(x_{1:n}|y_{1:n},\theta)$; in which case these moves are particularly effective.)

We will give examples of single-site moves, and investigate when they do and do not work well; before looking at designing efficient block updates. For notational convenience we drop the conditioning on θ in the notation that we use within this section.

1.3.1 Single-site updates of the state

The idea of single-site updates is to design MCMC moves that update a single value of the state, x_t, conditional on all other values of the state process (and on θ). Repeated application of this move for $t = 1,\dots,n$ will enable the whole state process to be updated.

We introduce the notation that $x_{-t} = (x_1,\dots,x_{t-1},x_{t+1},\dots,x_n)$ denotes the whole state process excluding x_t. So a single-site update will update x_t for fixed x_{-t},θ. The target distribution of such a move is the full-conditional distribution $p(x_t|x_{-t},\theta,y_{1:t})$; which as mentioned above we will write as $p(x_t|x_{-t},y_{1:t})$ – dropping the conditioning on θ in the notation that we use, as we are considering moves for fixed θ. Due to the Markov structure of our model this simplifies to $p(x_t|x_{t-1},x_{t+1},y_t)$ for $t = 2,\dots,n-1$, $p(x_1|x_2,y_1)$ for $t = 1$ and $p(x_n|x_{n-1},y_n)$ for $t = n$. Sometimes we can simulated directly from these full conditional distributions, and such (*Gibbs*) moves will always be accepted. Where this is not possible, then if x_t is low-dimensional we can often implement an efficient *Independence Sampler* (see below).

We now give details of single-site update for Example 2 (Gibbs move) and Example 1

(Independence Sample), and in both cases we investigate the mixing properties of the move in updating $X_{1:n}$.

Example 2: Single-site Gibbs move.

For the HMM model of Example 2, with state transition matrix, P, we have for $t = 2, \dots, n-1$ that

$$\begin{aligned}\Pr(X_t = k|X_{t-1} = i, X_{t+1} = j, y_t) &\propto \Pr(X_t = k|X_{t-1} = i) \Pr(X_{t+1} = j|x_t = k) p(y_t|X_t = k) \\ &= P_{ik} P_{kj} f_k(y_t),\end{aligned}$$

for $k = 1, \dots, K$. Now as X_t has a finite state-space, we can calculate the normalising constant of this conditional distribution, and we get

$$\Pr(X_t = k|X_{t-1} = i, X_{t+1} = j, y_t) = \frac{P_{ik} P_{kj} p_k(y_t)}{\sum_{l=1}^{K} P_{il} P_{lj} f_l(y_t)}.$$

Similarly we obtain $\Pr(X_1 = k|x_2 = j, y_1) \propto \Pr(X_1 = k) P_{kj} f_k(y_1)$ and $\Pr(X_n = k|X_{n-1} = i, y_n) \propto P_{ik} f_k(y_n)$. In both cases the normalising constants of these conditional distributions can be obtained.

Thus for this model we can simulate from the full-conditionals directly, which is the optimal proposal for x_t for fixed x_{-t}. Note that the computational cost of simulation is $O(K)$, due to calculation of the normalising constants. For large K it may be more computationally efficient to use other proposals (such as an independence sample) whose computational cost does not depend on K.

We examine the efficiency of this MCMC move at update the state $X_{1:n}$ by focussing on a HMM model for DNA sequences (see e.g. Boys et al., 2000). The data consists of a sequence of DNA, so $y_t \in \{\text{A,C,G,T}\}$ for all t. For simplicity we consider a two-state HMM, with the likelihood function for $k = 1, 2$ being

$$\Pr(Y_t = y|X_t = k) = \pi_y^{(k)}, \text{ for } y \in \{\text{A,C,G,T}\}.$$

We denote the parameter associated with $X_t = k$ as $\pi^{(k)} = (\pi_{\mathrm{A}}^{(k)}, \pi_{\mathrm{C}}^{(k)}, \pi_{\mathrm{G}}^{(k)}, \pi_{\mathrm{T}}^{(k)})$

We will consider the effect that both the dependence in the state dynamics, and the information in the observations have on the mixing rate of the MCMC move. To do this we will assume that state transition matrix satisfies $P_{12} = P_{21} = \alpha$, and

$$\pi^{(1)} = (1,1,1,1)/4 + \beta(1,1,-1,-1) \qquad \pi^{(2)} = (1,1,1,1)/4 - \beta(1,1,-1,-1),$$

for $0 < \alpha < 1$ and $0 < \beta < 1/4$. Small values of α correspond to large dependence in the state dynamics, and small values of β correspond to less informative observations.

To measure the mixing properties of the single-site MCMC update we (i) simulated data for a given value of (α, β); (ii) ran an MCMC algorithm with single-site updates; and (iii) calculated an autocorrelation function for the MCMC output after discarding a suitable burn-in. For simplicity, we summarised the output based on the autocorrelation at lag-1 (all MCMC runs suggested autocorrelations that decayed approximately exponentially). We calculated the autocorrelation for the number of differences between the true value of the hidden state and the inferred value of the state.

Results are shown in Figure 1.1, where we see that the value of α is the main determinant of the mixing of the MCMC algorithm. Small values of α, which correspond to large dependence, result in poor mixing. Similarly, as β decreases, which relates to less informative observations, the mixing gets worse – though the dependence on β is less than on α. Qualitatively similar results are observed for the two values of n, but for smaller n we see that the value of β has more impact on the mixing properties.

Example 1: Single-site Independence Sampler

Now consider the Stochastic Volatility model of Example 1. We describe an independence sampler that was derived by Shephard and Pitt (1997).

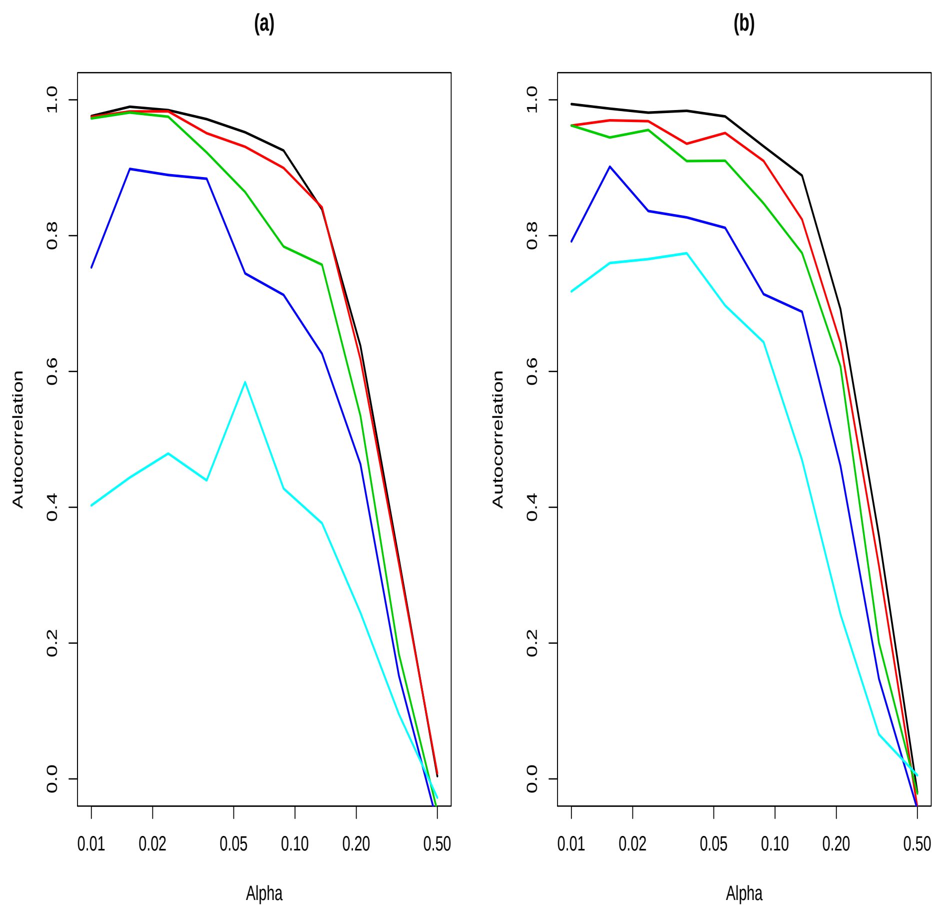

Figure 1.1: Lag-1 autocorrelation values for differing α for a 2-state HMM model: (a) $n = 200$; (b) $n = 500$. In each plot, different lines refer to different values of β; from top to bottom: $\beta = 0.02$ (black); $\beta = 0.065$ (red); $\beta = 0.11$ (green); $\beta = 0.155$ (dark blue); and $\beta = 0.2$ (light blue).

With this model, for $t = 2, \ldots, n-1$ we obtain

$$\begin{aligned} p(x_t|x_{t-1}, x_{t+1}, y_t) &\propto p(x_t|x_{t-1})p(x_{t+1}|x_t)p(y_t|x_t) \\ &\propto \exp\left\{-\frac{1}{2\sigma^2}((x_t - \phi x_{t-1}) + (x_{t+1} - \phi x_t)^2)\right\} \exp\left\{-\frac{x_t}{2}\right\} \exp\left\{-\frac{\exp\{-x_t\}y_t^2}{2\beta^2}\right\} \end{aligned} \tag{1.3.1}$$

where we have removed any constants of proportionality that do not depend on x_t; the first term of the final expression correspond to the two state transition densities, and the final two terms come from the likelihood.

Simulating directly from this conditional distribution is not possible, so we resort to approximation. Our approximation is based on a Taylor expansion of $\log p(x_t|x_{t-1}, x_{t+1}, y_t)$ about an estimate of x_t, which we call $\hat{x}_t$. Now if we define $\mu_t = \phi(x_{t-1} + x_{t+1})/(1 + \phi^2)$ and $\tau^2 = \sigma^2/(1+\phi^2)$, then the first term in (1.3.1) can be re-written as $\exp\{-(x_t - \mu_t)^2/(2\tau^2)\}$ up to a constant of proportionality. Thus without any observation, our conditional distribution of x_t would have a mean μ_t, and this appears a sensible value about which to take a Taylor expansion. Doing this we obtain

$$\log p(x_t|x_{t-1}, x_{t+1}, y_t) \approx -\frac{(x_t - \mu_t)^2}{2\tau^2} - \frac{x_t}{2} - \frac{y_t^2}{2\beta^2}\exp\{-\mu_t\}\left(1 - (x_t - \mu_t) + \frac{1}{2}(x_t - \mu_t)^2\right).$$

As this approximation to the log-density is quadratic, this gives us a Normal approximation to the conditional distribution, which we denote by $q(x_t|x_{t-1}, x_{t+1}, y_t)$. (For full details of the mean and variance of the approximation, see Shephard and Pitt, 1997).Thus we can implement an MCMC move of X_t by using an independence sampler with proposal $q(x_t|x_{t-1}, x_{t+1}, y_t)$.

Similar normal approximations can be obtained for $p(x_1|x_2, y_1)$ and $p(x_n|x_{n-1}, y_n)$, the only difference is in the values of μ_t and τ. Note that better estimates of $\hat{x}_t$ can be found, e.g. by numerically finding the mode of $p(x_t|x_{t-1}, x_{t+1}, y_t)$ (Smith and Santos, 2006), but for single-site updates any increase in acceptance rate is unlikely to be worth the extra computation involved.

We investigate the efficiency of single-site updates for the SV model via simulation. We fix $\beta = 1$ and consider how mixing of the MCMC algorithm depends on the time dependence

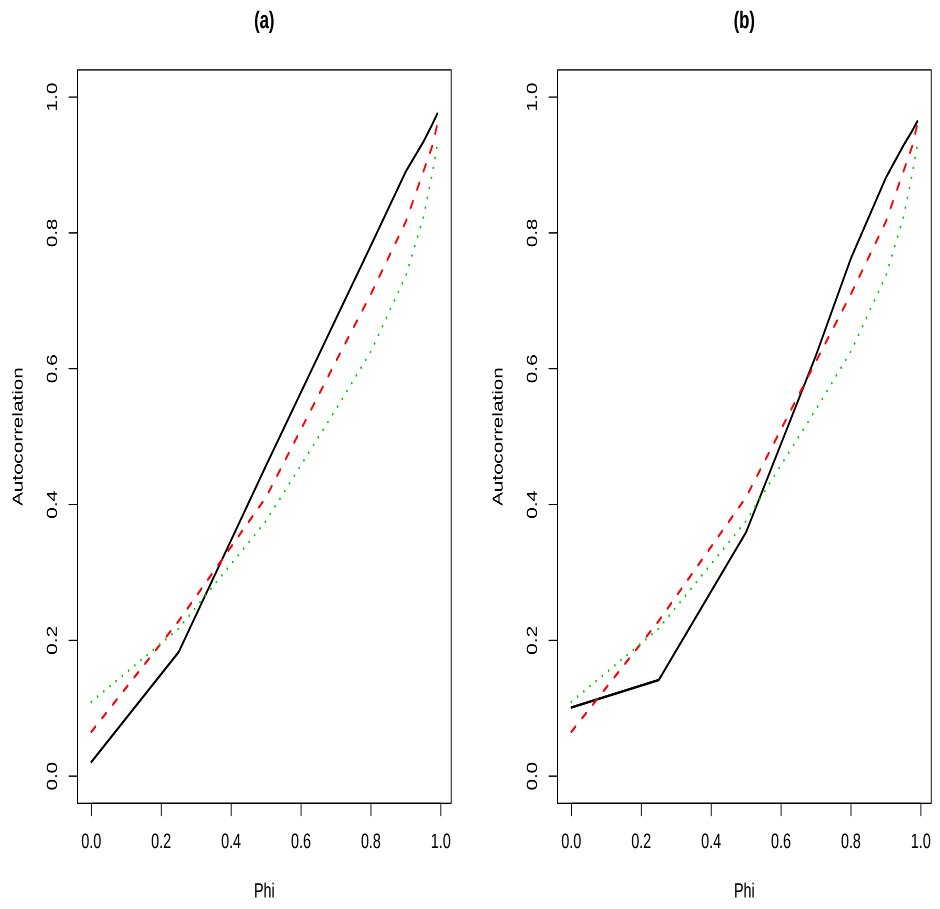

Figure 1.2: Lag-1 autocorrelation values for differing ϕ for the SV model: (a) $n = 200$; (b) $n = 500$. In each plot, different lines refer to different values of τ^2: $\tau^2 = 0.5$ (black, full lines); $\tau^2 = 1$ (red, dashed lines); $\tau^2 = 2.0$ (green, dotted lines).

of the state process, ϕ, and marginal variance of the state process, $\tau^2 = \sigma^2/(1-\phi^2)$. As above, we evaluate mixing by looking at the lag-1 autocorrelation of the mean square error in the estimate of the state process. Results are shown in Figure 1.2, where we see that ϕ has a sizeable effect on mixing – with $\phi \approx 1$, which corresponds to strong correlation in the state process, resulting in poor mixing. By comparison both n and τ^2 have little effect. For all MCMC runs the acceptance rate of the MCMC move was greater than 99%.

1.3.2 Block updates for the state

While the single-site updates of Section 1.3.1 are easy to implement, we have seen that the resulting MCMC algorithms can mix slowly if there is strong depedence in the state-process. This leads to the idea of updating the state at more than one time-point in a single move; which are called *block updates*. Ideally we would update the whole state process in one move, and in some cases it turns out that this is possible to do from the full-conditional, so that moves are always accepted. These include the linear-Gaussian models, where we can use the Kalman Filter (see e.g. Carter and Kohn, 1994; Harvey, 1989); the HMM model of Example 2 and the changepoint model of Example 3. We give details of the methods used for the latter two below.

In situations where it is not possible to update the whole state process from its full conditional, one possibility is to use an independence proposal to update jointly a block of state values. We will describe such an approach for the SV model of Example 1; and then discuss alternative approaches for block updates for models where it is not possible to draw from the full conditional distribution of the state.

Example 2: Updating state from its full conditional

The forward-backward algorithm is a method for sampling from the full conditional of the state-process for discrete HMMs. See Rabiner and Juang (1986) for a review of this method, and Scott (2002) for further examples of its use within Bayesian inference. Here we describe its implementation for the model of Example 2.

The algorithm is based upon a forward recursion which calculates the filtering densities $\Pr(X_t|y_{1:t})$ for $t = 1, \dots, n$; followed by a backward simulation step that simulates from $\Pr(X_n|y_{1:n})$ and then $\Pr(X_t|y_{1:n}, x_{t+1})$ for $t = n-1, \dots, 1$. The forward recursion is initialised with

$$\Pr(X_1 = k|y_1) \propto \Pr(X_1 = k) f_k(y_1), \text{ for } k = 1, \dots, K,$$

where the normalising constant is $p(y_1) = \sum_{l=1}^{K} \Pr(X_1 = l) f_l(y_1)$. Then for $t = 2, \dots, n$ we have

$$\Pr(X_t = k|y_{1:t}) \propto f_k(y_t) \sum_{l=1}^{K} \Pr(X_{t-1} = l|y_{1:t-1}) P_{lk}, \text{ for } k = 1, \dots, K,$$

where the normalising constant is $p(y_t|y_{1:t-1})$. (Note that a byproduct of the forward recursions is that we obtain the likelihood as a product of these normalising constants, as $p(y_{1:n}) = p(y_1) \prod_{t=2}^{n} p(y_t|y_{1:t-1})$.)

Once these filtering densities have been calculated and stored, we then simulate backwards. First X_n is simulated from the filtering density $\Pr(X_n|y_{1:n})$; then for $t = n-1, \dots, 1$ we iteratively simulate X_t given our simulated value for X_{t+1}, from

$$\Pr(X_t = l|y_{1:n}, X_{t+1} = k) = \Pr(X_t = l|y_{1:t}, X_{t+1} = k) \propto \Pr(X_t = l|y_{1:t}) P_{lk}.$$

The computational complexity of the forward-backward algorithm is $O(nK^2)$ for the forward recursion, and $O(nK)$ for the backward simulation. This compares with $O(nK)$ for applying the single-site update to all state-values. Thus, particularly for values large K, it may be computationally more efficient to use single-site updates. As seen above, whether this is the case will depend on the amount of dependence in the state-model.

In the above description, we supressed the dependence on the unknown parameter θ. Standard MCMC algorithms will update $X_{1:n}$ given θ and then θ given $X_{1:n}$ in one iteration. Thus each iteration will (potentially) have a new θ value, and will require the re-application of the forward-backward algorithm to simulate $X_{1:n}$. One approach to reducing the computational cost of using the forward-backward algorithm within MCMC, suggested by Fearnhead (2006) is to (i) obtain a good point estimate of the parameters, $\hat{\theta}$; (ii) apply the forward recursion for this value of the parameter; and (iii) use $\Pr(X_{1:n}|y_{1:n}, \hat{\theta})$ as an independence proposal for updating the state. The advantage of this is that the costly forward-recursion is only required once, as opposed to at every iteration of the MCMC algorithm. Furthermore, Fearnhead (2006) describe an efficient algorithm for simulating large samples of $X_{1:n}$ from the backward simulation step. In applications, providing a good estimate is obtained in (i), this approach has shown to produce efficient MCMC updates. Note that estimation in (i) could be performed in an adaptive manner during the burn-in period of the MCMC algorithm.

Our forward-backward description has focussed on discrete-time processes. It is possible to extend the idea to continous-time (though still discrete valued) HMMs. See for example Fearnhead and Meligkotsidou (2004) and Fearnhead and Sherlock (2006).

Example 3: Updating state from its full conditional

We now show how the forward-backward algorithm can be applied to the changepoint model of Example 3. The idea behind this application dates back to Yao (1984), but see also Barry and Hartigan (1992), Liu and Lawrence (1999) and Fearnhead (2006).

We introduce a new state variable, C_t, which we define to be the time of the most recent changepoint prior to t. Mathematically this is a function of $x_{1:t}$, with

$$C_t = \max\{s : x_s \neq x_{s+1} \text{ for } s < t\},$$

with $C_t = 0$ if there has been no changepoint prior to t (i.e. the set on the right-hand side is empty). Note that $C_t \in \{0, \dots, t-1\}$, and C_t is a Markov process with

$$\Pr(C_t = j | C_{t-1} = i) = \begin{cases} p & \text{if } j = t-1, \\ 1-p & \text{if } i = j, \end{cases}$$

with all other transitions being impossible. Note that these two transitions correspond to there either being or not being a changepoint at time $t-1$.

We can now derive forward-backward algorithm. The forward recursion is initialised with $\Pr(C_1 = 0|y_1) = 1$, and for $t = 2, \dots, n$ we have:

$$\begin{aligned} \Pr(C_t = j|y_{1:t}) &\propto (1-p)\frac{Q(j+1,t)}{Q(j+1,t-1)}\Pr(C_{t-1} = j|y_{1:t-1}) \quad \text{for } j = 0, \dots, t-2, \\ \Pr(C_t = t-1|y_{1:t}) &\propto pQ(t,t). \end{aligned}$$

The first equation corresponds to there not being a changepoint at time $t-1$. This happens with probability $1-p$ and in this case $C_t = C_{t-1}$. The second corresponds to there being a changepoint, which happens with probability p. The $Q(\cdot,\cdot)$ are defined in (1.1.5). In both equations, the term involving $Q(\cdot,\cdot)$ is the likelihood of the observation y_t given C_t and $y_{1:t-1}$.

Once the filtering recursions have been solved, backward simulation proceeds using the conditional distributions

$$\Pr(C_t = j | C_{t+1} = t, y_{1:n}) = \Pr(C_t = j | y_1 : t),$$

where conditioning on $C_{t+1} = t$ is equivalent to conditioning on a changepoint at t. Thus we can simulate the time of the last changepoint from $\Pr(C_n | y_{1:n})$, and then recursively given a changepoint at t simulate the next most recent changepoint from $\Pr(C_t | y_{1:t})$. This simulation continues until we simulate $C_t = 0$, which corresponds to no more changepoints.

The computational complexity of this algorithm is $O(n^2)$. The main cost is in solving the recursions, and one approach to reduce computational cost is to solve these for a specific value of the parameters, and then use the resulting conditional distribution for $X_{1:n}$ as an independence proposal (see Fearnhead, 2006, and the discussion for Example 2 above). Note this forward-backward algorithm can be generalised to allow for different distributions of time between successive changepoints (see e.g. Fearnhead, 2008); and for HMM dependence in the state value for neighbouring segments (Fearnhead and Vasileiou, 2007).

Example 1: Block independence sampler

For the SV model of Example 1, we cannot sample directly from the full conditional distribution $p(x_{1:n} | y_{1:n})$. Instead we follow Shephard and Pitt (1997) and consider an independence sampler for block updating. The proposal distribution for the independence sampler is based on a natural extension of the independence sampler for singe-site updates.

Consider an update for $X_{t:s}$ for $s > t$. For an efficient independence proposal we require a good approximation to $p(x_{t:s} | x_{t-1}, x_{s+1}, y_{t:s})$. (If $t = 1$ we would drop the conditioning on x_{t-1}, and if $s = n$ we would drop the conditioning on x_{s+1} here an in the following.) Now we can write

$$p(x_{t:s} | x_{t-1}, x_{s+1}, y_{t:s}) \propto p(x_{t:s} | x_{t-1}, x_{s+1}) \prod_{j=t}^{s} p(y_j | x_j),$$

where the first term on the right-hand side is a multivariate Gaussin density. Thus if for all $j = t, \ldots, s$, we approximate $p(y_j|x_j)$ by a Gaussian likelihood, we obtain a Gaussian approximation to $p(x_{t:s}|x_{t-1}, x_{s+1}, y_{t:s})$ which can be used as an independence proposal. We can obtain a Gaussian approximation to $p(y_j|x_j)$ by using a quadratic (in x_j) approximation to $\log p(y_j|x_j)$ via a Taylor expansion about a suitable estimate $\hat{x}_j$. The details of this quadratic approximation are the same as for the single-step update described above. Further details can be found in Shephard and Pitt (1997). The resulting quadratic approximation to $p(x_{t:s}|x_{t-1}, x_{s+1}, y_{t:s})$ can be calculated efficiently using the Kalman Filter (Kalman and Bucy, 1961), or efficient methods for Gaussian Markov Random Field models (Rue and Held, 2005), and its complexity is $O(s - t)$.

Implementation of this method requires a suitable set of estimates $\hat{x}_{t:s} = (\hat{x}_t, \ldots, \hat{x}_s)$. If we denote by $q(x_{t:s}|\hat{x}_{t:s})$ to be the Gaussian approximation to $p(x_{t:s}|x_{t-1}, x_{s+1}, y_{t:s})$ obtained by using the estimate $\hat{x}_{t:s}$, then one approach is to: (i) choose an initial estimate $\hat{x}^{(0)}_{t:s}$; and (ii) for $i = 1, \ldots, I$ set $\hat{x}^{(i)}_{t:s}$ to be the mean of $q(x_{t:s}|\hat{x}^{(i-1)}_{t:s})$. In practice choosing $\hat{x}^{(0)}_{t:s}$ to be the mean of $p(x_{t:s}|x_{t-1}, x_{s+1})$ and using small values of I appears to work well.

This approach to designing independence proposals can be extended to other models where the model of the state is linear-Gaussian (see Jungbacker and Koopman, 2007). Using the resulting independence sampler within an MCMC algorithm is straightforward if it is efficient to update the complete state path $X_{1:n}$. If not, we must update the state in smaller blocks. A simplistic approach would be to split the data in to blocks of (approximately) equal size, τ say, and then update in turn $X_{1:\tau}$, $X_{(\tau+1):2\tau}$ etc. However this approach will mean that state values towards the boundaries of each block will mix slowly due to the conditioning on the state values immediately outside the boundary of the blocks. To avoid this Shephard and Pitt (1997) suggest randomly choosing the blocks to be updated for each application of the independence proposal. Another popular alternative is to choose overlapping blocks, for example $X_{1:2\tau}$, $X_{(\tau+1):3\tau}$, $X_{(2\tau+1):4\tau}$ and so on.

A further important consideration in implementation is the choice of block size. Too small and we will obtain poor mixing due to the strong dependence of $X_{t:s}$ on X_{t-1} and X_{s+1}; too large and we will have poor mixing due to low acceptance rates. (One approach is to use

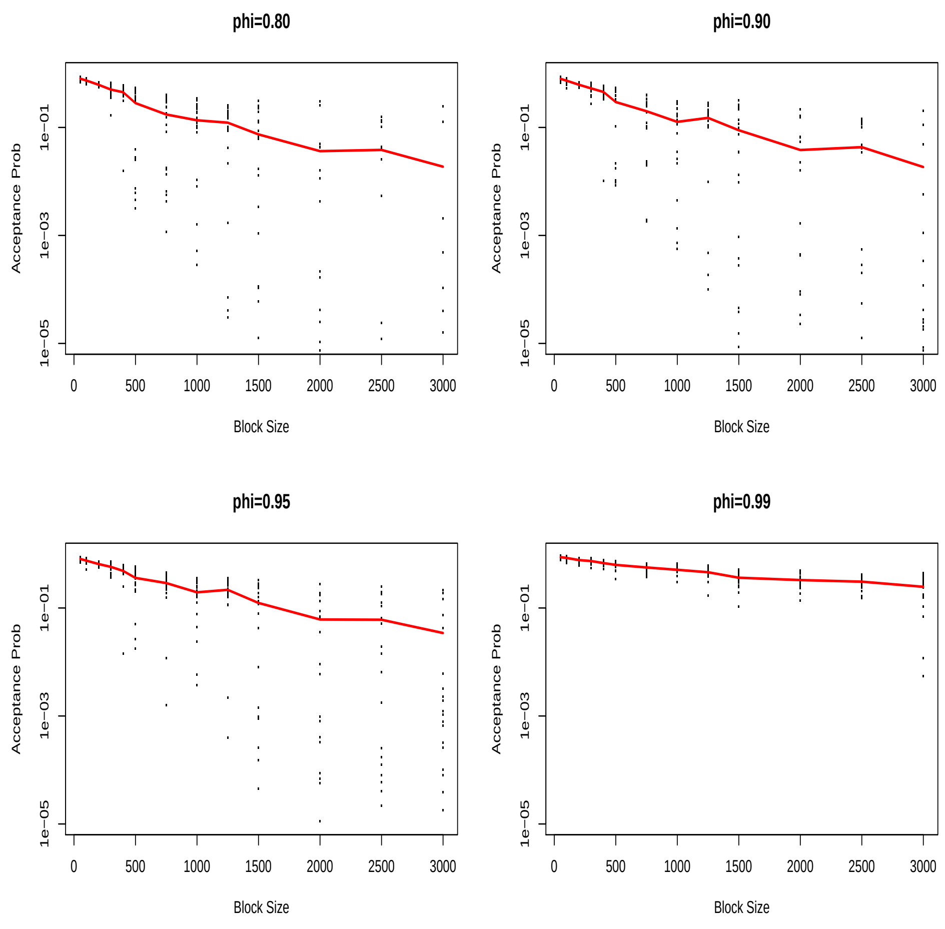

Figure 1.3: Average acceptance rates for different block sizes, and different ϕ values. Black dots show mean acceptance rates for 20 different data-sets for each block size. Red lines show mean acceptance rates for each block size. All runs had $\tau^2 = \sigma^2/(1-\phi^2) = 0.2$. (Some MCMC runs had acceptance rates that are too small to appear on the plot.)

adaptive MCMC methods to choose appropriate block sizes, see Roberts and Rosenthal (2006).) Here we will look at the effect that block size has on acceptance probabilities for the SV model.

Plots of average acceptance rates for different block sizes and different data sets are shown in Figure 1.3. Two features are striking. The first is that efficiency varies substantially with ϕ, with values of $\phi \approx 1$ producing higher average acceptance rates. This is because for $\phi \approx 1$ there is stronger dependence in the state-process, and thus the (Gaussian) $p(x_{t:s}|x_{t-1}, x_{s+1})$ dominates the (non-Gaussian) likelihood $p(y_{t:s}|x_{t:s})$. The second is that there is great variability in acceptance rates across different runs: thus choice of too large block sizes can lead to the chain becoming easily stuck (for example, acceptance probabilities of 10^{-8} or less were observed for blocks of 2,000 or more observations when $\phi = 0.8$). This variability suggests that either randomly choosing block sizes, or adaptively choosing block sizes for a given data set are both sensible strategies.

However, overall we see that the block updates are particularly efficient for the SV model. For block updates, acceptance rates > 0.01 are reasonable, and the average acceptance rate was greater than this for all combinations of ϕ and block size that we considered. Even looking at the worse-case acceptance rates across all runs, we have acceptances rates greater then 0.01 for blocks of size 400 when $\phi = 0.8$; and for $2,500$ for $\phi = 0.99$.

Other approaches

Our examples have shown how to simulate directly from the full conditional of the state; or how to approximate the full conditional for use within an independence proposal. However the former method can only be applied to a limited class of models, and the latter used the linear-Gaussian nature of the state-model. It is possible to obtain good independence proposals for more general state-models, but this can become challenging, particularly for high-dimensional states and models with strong non-linearities.

One general approach to block updates of the states has been recently proposed in Andrieu et al. (2008), which is based upon using sequential Monte Carlo methods (see e.g. Liu and Chen, 1998) within MCMC. Sequential Monte Carlo methods can be efficient for analysing state-space models where parameters are known, and the idea is these are used to generate a proposal distribution for the path of the state within an MCMC algorithm.

1.4 Updating the parameters

We now consider how to update the parameter, θ, within the MCMC algorithm. The natural approach is to update θ conditional on the current value of the state path $x_{1:n}$. Often this is simple to implement as either conjugate priors for θ can be chosen so that we can sample directly from $p(\theta|x_{1:n}, y_{1:n})$, or θ is of sufficiently low-dimension that we can use efficient independence proposals. In some cases we need to update components or blocks of θ at a time, rather than the updating the whole parameter vector in one go.

However, even if we can sample from the full-conditional $p(\theta|x_{1:n}, y_{1:n})$, the overall efficiency of the MCMC algorithm can still be poor if there is strong correlation between θ and $x_{1:n}$. The rate of convergence of an algorithm that alternates between sampling from $p(x_{1:n}|\theta, y_{1:n})$ and $p(\theta|x_{1:n}, y_{1:n})$ is given by Liu (1994) and Roberts and Sahu (1997). If for a square-integrable function f of the parameters, we define the *Bayesian fraction of missing information*:

$$\gamma_f = 1 - \frac{\mathrm{E}\left(\mathrm{Var}\left(f(\theta)|X_{1:n}, y_{1:n}\right)|y_{1:n}\right)}{\mathrm{Var}\left(f(\theta)|y_{1:n}\right)}, \tag{1.4.1}$$

then the geometric rate of convergence of the MCMC algorithm is $\gamma = \sup_f \gamma_f$. Values of $\gamma \approx 1$ suggest a poorly mixing MCMC algorithm. This will occur when, after conditioning on the data, there are functions f for which most of the variation in $f(\theta)$ is explained by the value of the state, $X_{1:n}$.

When there is strong dependence between θ and $X_{1:n}$, there are two techniques for improving mixing. The first is to consider a different parameterisation, with the hope that for this new parameterisation there will be less dependence between the state and the parameter. The second is to use moves that jointly update θ and $X_{1:n}$. We will describe and evaluate approaches for updating θ given $X_{1:n}$, and then consider these two approaches for improving mixing in turn.

1.4.1 Conditional updates of the parameters

Here we focus on Examples 1 and 2, and give outlines of how parameter updates can be made with these models. We will also investigate the mixing properties of the resulting MCMC algorithms.

Example 1: Conditional parameter updates

Following Shephard and Pitt (1997) we will consider indepdent priors for β, σ^2 and ϕ. As

Table 1.1: Lag-1 autocorrelation for β for both Non-centered and Centered parameterisations. Results are for $\sigma^2 = 0.02^2$, $\beta = 1$ and $n = 200$, and different values of ϕ.

ϕ	0.8	0.9	0.95	09.75	0.99
Non-centered	0.11	0.21	0.37	0.62	0.98
Centered	0.89	0.79	0.64	0.43	0.29

β is a scale parameter, we choose the canonical uninformative prior, $p(\beta) \propto 1/\beta$. For σ^2 our prior is $S_0\chi_p^{-2}$. As it is normal to restrict $|\phi| < 1$, we choose a Beta(a, b) prior for $(\phi + 1)/2$. For these choices we have that conditional on $\{x_{1:n}, y_{1:n}\}$, β is independent of ϕ, σ^2, and has distribution

$$\beta^2|\{x_{1:n}, y_{1:n}\} \sim \chi_n^{-2} \sum_{t=1}^{n} y_t^2 \exp\{-x_t\}. \tag{1.4.2}$$

To update ϕ and σ it is simplest to use their conditional distributions

$$\sigma^2|\{x_{1:n}, y_{1:n}, \phi\} \sim \chi_{n+p}^{-2} \left\{ S_0 + x_1^2(1-\phi^2) + \sum_{t=2}^{n}(x_t - \phi x_{t-1})^2 \right\},$$

$$p(\phi|x_{1:n}, y_{1:n}, \sigma) \propto (1+\phi)^{a-1/2}(1-\phi)^{b+1/2} \exp\left\{ -\frac{(1-\phi^2)x_1^2}{2\sigma^2} - \frac{1}{2\sigma^2}\sum_{t=2}^{n}(x_t - \phi x_{t-1})^2 \right\}.$$

The distribution for σ^2 can be sampled from directly. For ϕ, a simple procedure is an independence sampler with Gaussian proposal. The Gaussian proposal is chosen proportional to

$$\exp\left\{ -\frac{(1-\phi^2)x_1^2}{2\sigma^2} - \frac{1}{2\sigma^2}\sum_{t=2}^{n}(x_t - \phi x_{t-1})^2 \right\},$$

which corresponds to a mean of $\sum_{t=2}^{n} x_t x_{t-1} / \sum_{t=2}^{n-1} x_t^2$ and a variance of $\sigma^2 / \sum_{t=2}^{n-1} x_t^2$. (Note that this distribution can proposal values outside $(-1, 1)$, and such values will always be rejected.)

An example of how the mixing of the MCMC algorithm is affected by the dependence within the state-model is shown in the top row of Table 1.1 (labelled non-centered parameterisation). We notice that as ϕ increases, that is the dependence in the state model increases, then the mixing deteriorates. This is because in this limit the amount of information about β contained in the state-path remains roughly constant as ϕ increases, but the amount of information about β contained just in the observations is decreasing. This means that the Bayesian fraction of missing information is increasing, and thus the MCMC algorithm mixes more poorly.

Example 2: Conditional parameter updates

Let P_k denote the kth row of the transition matrix, P. Furthermore consider the case where the parameter vector can be written as $\theta = (P, \phi_1, \ldots, \phi_K)$, with the likelihood function given $X_t = k$ is of the form $f_k(y|\theta) = f_k(y|\phi_k)$. That is we have a disjoint set of parameters for each of the K likelihood models. Further assume first that the distribution of X_1 is independent of θ. In this case, if our priors for the P_ks and ϕ_ks are independent, then the full conditional $p(\theta|x_{1:n}, y_{1:n})$ simplifies. Conditional on $\{x_{1:n}, y_{1:n}\}$, we have independence of $P_1, \ldots, P_K, \phi_1, \ldots, \phi_K$. Thus we can perform independent updates of each of these $2K$ sets of parameter in turn. (If the distribution of X_1 depends on P, then this will introduce weak dependence in the posterior distribution of the P_ks.)

If we choose a Dirichlet prior for the entries of P_k, then the $p(P_k|x_{1:n}, y_{1:n})$ will be a Dirichlet distribution. Updating of ϕ_k will depend on the specific likelihood model and priors used. However, for the DNA model introduced in Section 1.3.1, we have $\phi_k = \pi^{(k)} = (\pi_{\mathrm{A}}^{(k)}, \pi_{\mathrm{C}}^{(k)}, \pi_{\mathrm{G}}^{(k)}, \pi_{\mathrm{T}}^{(k)})$, and if we have a Dirichlet prior then $p(\phi_k|x_{1:n}, y_{1:n})$ will again be Dirichlet.

1.4.2 Reparameterisation of the model

We have seen that dependence between $X_{1:n}$ and θ can result in a MCMC algorithm for $(X_{1:n}, \theta)$ that mixes poorly. One approach to alleviate this is to consider alternative parameterisations.

Papaspiliopoulos et al. (2007) describe two possible general parameterisations for hierarchical models (see also Gelfand et al., 1995; Papaspilopoulos et al., 2003), and these can be used for state-space models. These are *centered parameterisations*, which in our set-up is defined by a model where $p(\theta|x_{1:n}, y_{1:n}) = p(\theta|x_{1:n})$, and *non-centered parameterisations* where a priori θ and $X_{1:n}$ are independent. If we consider Examples 1 and 2 above, then for the Stochastic Volatility model of Example 1 our parameterisation for β is non-centered – as our model for $X_{1:n}$ does not depend on β. By comparison, for Example 2 our parameterisation for P is a centered parameterisation.

While it is non-trivial to introduce a non-centered parameterisation for Example 2 (though Papaspiliopoulos, 2003; Roberts et al., 2004, propose approaches that could be used), it is straightforward to introduce a centered parameterisation for Example 1. We define $\mu = 2\log\beta$ and a new state model $X'_{1:n}$ where

$$X'_t|\{x'_{1:t-1}, y_{1:t-1}\} \sim \mathrm{N}(\mu + \phi(x'_{t-1} - \mu), \sigma^2),$$

with $X'_1 \sim \mathrm{N}(\mu, \sigma^2/(1-\phi^2))$, and

$$Y_t|\{x'_{1:t}, y_{1:t-1}\} \sim \mathrm{N}(0, \exp\{x'_t\}).$$

For this parameterisation we have (Pitt and Shephard, 1999)

$$\mu|\{x'_{1:n}, y_{1:n}\} \sim \mathrm{N}(b/a, \sigma^2/a),$$

where $a = (n-1)(1-\phi)^2 + (1-\phi^2)$ and $b = (1-phi)\{\sum_{t=2}^{n}(x'_t - \phi x'_{t-1}\} + x'_1(1-\phi^2)$.

For large n we can compare γ_f (1.4.1) for $f(\theta) = \mu$ for both centered and non-centered parameterisations. If we conjecture that $\gamma \approx \gamma_f$, then these values will inform us about the relative efficiency of the two parameterisations. To compare γ_f for the two parameterisations we need only compare $\mathrm{E}(\mathrm{Var}(2\log\beta|X_{1:n}, y_{1:n})|y_1 : n)$ and $\mathrm{E}(\mathrm{Var}(\mu|X'_{1:n}, y_{1:n})|y_{1:n})$. If the former is larger, than the centered parameterisation will have a smaller value for γ_f, and we may conjecture will have a better rate of convergence. Otherwise γ_f will be smaller for the non-centered parameterisation.

Now for the non-centered parameterisation we have $\mathrm{Var}(\mu|X'_{1:n}, y_{1:n}) = 1/a \approx \sigma^2/(n(1-$

$\phi)^2)$. Thus as this does not depend on $X'_{1:n}$ we have

$$\mathrm{E}(\mathrm{Var}(\mu|X'_{1:n}, y_{1:n})|y_1 : n) \approx \frac{\sigma^2}{n(1-\phi)^2}.$$

For the centered parameterisation, from (1.4.2), we have that $\mathrm{E}(\mathrm{Var}(2\log\beta|X_{1:n}, y_{1:n})|y_{1:n}) = \mathrm{Var}(\log\chi^2_n)$, thus for large n

$$\mathrm{E}(\mathrm{Var}(2\log\beta|X_{1:n}, y_{1:n})|y_{1:n}) \approx \frac{2}{n}.$$

Thus γ_f is smaller for the centered parameterisation if $2/n > \sigma^2/(n(1-\phi)^2)$ or

$$\phi > 1 - \frac{\sigma}{\sqrt{2}}.$$

This suggests that as $\phi \to 1$ we should prefer using the centered parameterisation, but for small ϕ the non-centered parameterisation would be prefered. This is confirmed by simulation (see Table 1.1). Similarly, when σ is small we should prefer the centered parameterisation.

For the specific model we consider in Example 1, we have centered parameterisations for σ and ϕ. It is possible to extend the non-centered parameterisations for β to one for (β, σ) and even (β, σ, ϕ). For (β, σ) we introduce a state $X'_{1:n}$ where

$$X'_t|\{x'_{1:t-1}, y_{1:t-1}\} \sim \mathrm{N}(\phi x'_{t-1}, 1),$$

with $X'_1 \sim \mathrm{N}(0, 1/(1-\phi^2))$, and

$$Y_t|\{x'_{1:t}, y_{1:t-1}\} \sim \mathrm{N}(0, \beta^2 \exp\{\sigma x'_t\}).$$

For (β, σ, ϕ) we can parameterise the state in terms of the standardised residuals in the AR model, $(X_t - \phi X_{t-1})/\sigma$, and $X_1\sqrt{1-\phi^2}$, which are independent standard normal random variables. This latter idea, and ideas related to it, has been used extensively within continuous time stochastic volatility models (see e.g. Golightly and Wilkinson, 2008; Roberts and Stramer, 2001).

1.4.3 Joint updates of the parameters and state

One way of thinking about why strong correlation between θ and $X_{1:n}$ produces poor mixing, is that large moves of θ are likely to be rejected as they will be inconsistent with the current value of the state. This will happen even if the proposed new value for θ is consistent with the data. This motivates jointly updating θ and $X_{1:n}$, from a proposal $q(\theta', x'_{1:n}|\theta, x_{1:n}) = q(\theta'|\theta)q(x'_{1:n}|\theta')$. Thus $q(\theta'|\theta)$ could propose large moves, and then values of the state-process consistent with θ' will be simulated from $q(x'_{1:n}|\theta')$.

This is most easily and commonly implemented for models where we can simulate directly from $p(x_{1:n}|\theta, y_{1:n})$, in which case we choose $q(x'_{1:n}|\theta') = p(x'_{1:n}|\theta', y_{1:n})$. The resulting acceptance ratio then simplifies to:

$$\min\left\{1, \frac{q(\theta|\theta')p(\theta'|y_{1:n})}{q(\theta'|\theta)p(\theta|y_{1:n})}\right\}.$$

This acceptance ratio does not depend on $x_{1:n}$ or $x'_{1:n}$. The marginal chain for θ is equivalent to a MCMC chain for $p(\theta|y_{1:n})$ with proposal distribution $q(\theta'|\theta)$.

Providing an efficient proposal $q(\theta'|\theta)$ can be found, such an MCMC algorithm will always be more efficient than one that updates θ and $X_{1:n}$ independently. However, the difficulty with implementing this idea is how to choose $q(\theta'|\theta)$. For Markov modulated Poisson processes, Sherlock et al. (2008), found that a Gibbs sampler that updated $X_{1:n}$ given θ and θ given $X_{1:n}$ performed better than this joint update where $q(\theta'|\theta)$ was chosen to be a symmetric random-walk. A further advantage of the Gibbs sampler, is that it avoids tuning $q(\theta'|\theta)$, though this problem can be alleviated by using adaptive MCMC schemes (Andrieu and Thoms, 2008; Sherlock et al., 2008).

A simple extension of this joint updating idea is possible if we have an efficient independence proposal for $x_{1:n}$ given θ – as this proposal could be used as $q(x'_{1:n}|\theta')$. Here the efficiency of the resulting algorithm will depend on both the efficiency of $q(\theta'|\theta)$ as a proposal for an MCMC that explores $p(\theta'|y_{1:n})$, and also the closeness of $q(x'_{1:n}|\theta')$ to $p(x'_{1:n}|\theta', y_{1:n})$. Novel ideas for implementing such moves are given in Andrieu et al. (2008) and Andrieu and Roberts (2007).

Bibliography

Andrieu, C., Doucet, A., and Holenstein, R. (2008). Particle Markov chain Monte Carlo. *Submitted.*

Andrieu, C. and Roberts, G. O. (2007). The pseudo-marginal approach for efficient coputations. *Annals of Statistics*, page In Press.

Andrieu, C. and Thoms, J. (2008). An overview of controlled MCMC. *Submitted to Statistics and Computing.*

Ball, F. G. and Rice, J. A. (1992). Stochastic models for ion channels: Introduction and bibliography. *Mathematical Biosciences*, 112:189–206.

Barry, D. and Hartigan, J. A. (1992). Product partition models for change point problems. *The Annals of Statistics*, 20:260–279.

Beaulieu, C., Ouarda, T. B. M. J., and Seidou, O. (2007). A review of homogenization techniques for climate data and their applicability to precipitation series. *Hydrological Sciences Journal - Journal des Sciences Hyrologiques*, 52:18–37.

Boys, R. J., Henderson, D. A., and Wilkinson, D. J. (2000). Detecting homogeneous segments in DNA sequences by using hidden Markov models. *Journal of the Royal Statistical Society, Series C*, 49:269–285.

Carter, C. K. and Kohn, R. (1994). On Gibbs sampling for state space models. *Biometrika*, 81(3):541–553.

Chen, J. and Gupta, A. K. (1997). Testing and locating changepoints with application to stock prices. *Journal of the American Statistical Association*, 92:739–747.

Didelot, X., Achtman, M., Parkhill, J., Thomson, N. R., and Falush, D. (2007). A bimodal pattern of relatedness between the *salmonella* Paratyphi A and Typhi genomes: Convergence or divergence by homologous recombination? *Genome Research*, 17:61–68.

Fearnhead, P. (2006). Exact and efficient inference for multiple changepoint problems. *Statistics and Computing*, 16:203–213.

Fearnhead, P. (2008). Computational methods for complex stochastic systems: A review of some alternatives to MCMC. *Statistics and Computing*, 18:151–171.

Fearnhead, P. and Meligkotsidou, L. (2004). Exact filtering for partially-observed continuous-time Markov models. *Journal of the Royal Statistical Society, series B*, 66:771–789.

Fearnhead, P. and Sherlock, C. (2006). Bayesian analysis of Markov modulated Poisson processes. *Journal of the Royal Statistical Society, Series B*, 68:767–784.

Fearnhead, P. and Vasileiou, D. (2007). Bayesian analysis of isochores. *Submitted.* available from `www.maths.lancs.ac.uk/∼fearnhea/publications`.

Gelfand, A. E., Sahu, S., and Carlin, B. P. (1995). Efficient parameterisations for normal linear mixed models. *Biometrika*, 82:479–488.

Golightly, A. and Wilkinson, D. J. (2008). Bayesian inference for nonlinear multivariate diffusion models observed with error. *Computational Statistics and Data Analysis*, 52:1674–1693.

Harvey, A. C. (1989). *Forecasting, stuctural time series and the Kalman filter.* Cambridge University Press, Cambridge, UK.

Hobert, J. (2008). *The Handbook of Markov Chain Monte Carlo*, chapter Data Augmentation.

Hodgson, M. E. A. (1999). A Bayesian restoration of an ion channel signal. *Journal of the Royal Statistical Society, Series B*, 61:95–114.

Hull, J. and White, A. (1987). The pricing of options on assets with stochastic volatilities. *Journal of Finance*, 42:281–300.

Juang, B. H. and Rabiner, L. R. (1991). Hidden Markov models for speech recognition. *Technometrics*, 33:251–272.

Jungbacker, B. and Koopman, S. J. (2007). Monte Carlo Estimation for Nonlinear Non-Gaussian State Space Models. *Biometrika*, 94:827–839.

Kalman, R. and Bucy, R. (1961). New results in linear filtering and prediction theory. *Journal of Basic Engineering, Transacation ASME series D*, 83:95–108.

Liu, J. S. (1994). Fraction of missing information and convergence rate of data augmentation. In *Computing Science and Statistics: Proc. 26th Symposium on the Interface*, pages 490–496. Interface Foundation of North America, Fairfax Station, VA.

Liu, J. S. and Chen, R. (1998). Sequential Monte Carlo methods for dynamic systems. *Journal of the American Statistical Association.*, 93:1032–1044.

Liu, J. S. and Lawrence, C. E. (1999). Bayesian inference on biopolymer models. *Bioinformatics*, 15:38–52.

Lund, R. and Reeves, J. (2002). Detection of undocumented changepoints: A revision of the two-phase regression model. *Journal of Climate*, 15:2547–2554.

Papaspiliopoulos, O. (2003). *Non-centered parameterizations for hierarchical models and data augmentation*. PhD thesis, Dept. Mathematics and Statistics, Lancaster University.

Papaspiliopoulos, O., Roberts, G. O., and Sköld, M. (2007). A general framework for the parameterization of hierarchical models. *Statistical Science*, 22:59–73.

Papaspilopoulos, O., Roberts, G. O., and Sköld, M. (2003). Non-centred parameterisations for hierarchical models and data augmentation (with discussion). In Bernardo, J. M., Bayarri, M. J., Berger, J. O., Dawid, A. P., Heckerman, D., Smith, A. F. M., and West, M., editors, *Bayesian statistics 7*, London. Clarendon Press.

Pitt, M. K. and Shephard, N. (1999). Filtering via simulation: auxiliary particle filters. *Journal of the American Statistical Association*, 94:590–599.

Rabiner, L. R. and Juang, B. H. (1986). An introduction to hidden Markov models. *IEEE ASSP Magazine*, pages 4–15.

Ritov, Y., Raz, A., and Bergman, H. (2002). Detection of onset of neuronal activity by allowing for heterogeneity in the change points. *Journal of Neuroscience Methods*, 122:25–42.

Roberts, G. O., Papaspiliopoulos, O., and Dellaportas, P. (2004). Bayesian inference for non-Gaussian Ornstein-Uhlenbeck stochastic volatility processes. *Journal of the Royal Statistical Society, series B*, 66:369–393.

Roberts, G. O. and Rosenthal, J. (2006). Examples of adaptive MCMC. Technical report, Department of Statistics, University of Toronto.

Roberts, G. O. and Sahu, S. K. (1997). Updating schemes, correlation structure, blocking and parameterization for the Gibbs sampler. *Journal of the Royal Statistical Society, Series B*, 59:291–317.

Roberts, G. O. and Stramer, O. (2001). On inference for partially observed nonlinear diffusion models using the Metropolis-Hastings algorithm. *Biometrika*, 88:603–621.

Rue, H. and Held, L. (2005). *Gaussian Markov Random Fields: Theory and Applications.* CRC Press/Chapman and Hall.

Scott, S. L. (2002). Bayesian methods for hidden Markov models: Recursive computing in the 21st century. *Journal of the American Statistical Association*, 97:337–351.

Shephard, N. (1996). Statistical aspects of ARCH and stochastic volatility. In Cox, D. R., Hinkley, D. V., and Barndorff-Nielsen, O. E., editors, *Time Series Models in Econometrics, Finance and Other Fields*, pages 1–67, Chapman and Hall, London.

Shephard, N. and Pitt, M. K. (1997). Likelihood analysis of non-Gaussian measurement time series. *Biometrika*, 84:653–667.

Sherlock, C., Fearnhead, P., and Roberts, G. O. (2008). The random walk Metropolis: linking theory and practice through a case study. *Submitted to Statistical Science.*

Smith, J. Q. and Santos, A. F. (2006). Second order filter distribution approximations for financial time series with extreme outlier. *Journal of Business and Economic Statistics*, 24:329–337.

Yao, Y. (1984). Estimation of a noisy discrete-time step function: Bayes and empirical Bayes approaches. *The Annals of Statistics*, 12:1434–1447.