Approximating the Bayesian inverse for nonlinear dynamical systems

To cite this article: B M Bell and G Pillonetto 2008 J. Phys.: Conf. Ser. 124 012009

View the article online for updates and enhancements.

Related content
- Fast Track Communication
  Dorje C Brody, Anna C T Gustavsson and Lane P Hughston
- State vector evolution localized over the edges of a square tight-binding lattice
  He Liang-Ming and Shi Duan-Wen
- Quantum noise and stochastic reduction
  Dorje C Brody and Lane P Hughston

Recent citations
- Distributed Kalman smoothing in static Bayesian networks
  Gianluigi Pillonetto et al
- A distributed Kalman smoother
  Bradley M. Bell and Gianluigi Pillonetto
Approximating the Bayesian inverse for nonlinear dynamical systems

Bradley M. Bell, Gianluigi Pillonetto

1 submitted to AIP 2007-11-15, this version 2008-02-11
2 Applied Physics Laboratory, University of Washington, Seattle USA
3 Department of Information Engineering, University of Padova, Italy
E-mail: bradbell@washington.edu

Abstract. We are given a sequence of measurement vectors and a possibly nonlinear relation to a corresponding sequence of state vectors. We are also given a possibly nonlinear model for the dynamics of the state vectors. The goal is to estimate (invert) for the state vectors when there is noise in both the measurements and the dynamics of the state. In general, obtaining the minimum variance (Bayesian) estimate of the state vectors is difficult because it requires evaluations of high dimensional integrals with no closed analytic form. We use a block tridiagonal Cholesky algorithm to simulate from the Laplace approximation for the posterior of the state vectors. These simulations are used as a proposal density for the random-walk Metropolis algorithm to obtain samples from the actual posterior. This provides a means to approximate the minimum variance estimate, as well as confidence intervals, for the state vector sequence. Simulations of a fed-batch bio-reactor model are used to demonstrate that this approach obtains better estimates and confidence intervals than the iterated Kalman smoother.

1. Introduction
We use \( x_t : \Omega \rightarrow \mathbb{R}^n \) to denote the state of a stochastic dynamical system and \( z_t : \Omega \rightarrow \mathbb{R}^m \) a stochastic measurement that is related to the state, where \( t \) is time and \( \Omega \) is a probability space. We are given a fixed sequence of times \( \{t_k : k = 1, \ldots, N\} \) at which the corresponding measurement values \( \{z_k\} \) are known. Here and below, given a random function \( u_t : \Omega \rightarrow \mathbb{R}^q \), we use \( u_k \) to denote \( u_t \) at \( t = t_k \) and \( \{u_k\} \) to denote \( \{u_k : k = 1, \ldots, N\} \). Our goal is to estimate the state sequence \( \{x_k\} \) from the measurement sequence \( \{z_k\} \). The forward model expresses the conditional probability density for the measurement sequence given the state sequence \( p(\{z_k\} | \{x_k\}) \). The prior model expresses the probability density for the state sequence independent of the relation between the measurements and the state \( p(\{x_k\}) \). The joint density \( p(\{z_k, x_k\}) \) corresponds to a specific value for both the measurement and state sequences. Using the definition of conditional probability, we have

\[
p(\{z_k, x_k\}) = p(\{z_k\} | \{x_k\}) p(\{x_k\})
\]

The maximum a posteriori estimate is the value of the state sequence \( \{x_k\} \) that maximizes the joint density (minimizes the joint negative log density) and can be computed by means of an Iterated Kalman Smoother (IKS) [1, Figure 2]. The inverse model expresses the posterior
distribution for the state sequence given the measurement sequence using Bayes’ Theorem, i.e.,

$$p(\{x_k\} \mid \{z_k\}) = p(\{z_k\}, \{x_k\}) / \int_{-\infty}^{+\infty} p(\{z_k\}, \{x_k\}) \, d\{x_k\}$$

(1)

The minimum variance estimate (expected value) for the state sequence given the measurement sequence is

$$E[\{x_k\} \mid \{z_k\}] = \int_{-\infty}^{+\infty} \{x_k\} \, p(\{x_k\} \mid \{z_k\}) \, d\{x_k\}$$

Computing this integral may not be analytically intractable. To overcome this difficulty, several stochastic sampling approaches have been introduced in recent years. Monte Carlo strategies relying on sequential importance sampling (particle filters) are one approach, see the seminal work [2]. In this approach, stochastic samples are used to represent the state posterior distribution by generating adaptive grids whose values are updated randomly according to a simulation-based rule, e.g., [3, Section III.B]. Smoothing methods that use this approach to approximate the posterior marginal distribution of the state at each time point $$x_k$$ have been presented in [4, 5, 6]. A method for sampling trajectories from the posterior joint distribution of the entire state sequence $$\{x_k\}$$ is described in [7].

Markov chain Monte Carlo (MCMC) represents another powerful approach for generating samples from posterior densities. This approach uses the Metropolis-Hastings algorithm to generate a Markov chain that converges to the target density, e.g., see [8, 9]. This approach to smoothing for nonlinear dynamical systems has had limited application. The difficulty is in designing a proposal density for which the number of samples required is not excessive (required so that the posterior distribution is accurately approximated). We present such a proposal density together with an algorithm for generating its samples. The IKS is used to find local maximizers of the joint density $$p(\{z_k\}, \{x_k\})$$. Then the posterior $$p(\{x_k\} \mid \{z_k\})$$, near each local maximizer, is approximated by a normal that is second order accurate, see equation (5). The state space structure, defined in Section 2, enables us to do this with a number of floating point operations that scales linearly with the number of time points $$N$$.

This paper is organized as follows: Section 2 defines our statistical model for the dynamical system and the measurements. Section 3 presents the corresponding joint negative log-likelihood as a function of the measurement and state sequence values. It is shown in this section that the Hessian of the negative log-likelihood is block tridiagonal. Section 4 presents a Cholesky factorization method for positive definite block tridiagonal systems. Section 5 presents an MCMC algorithm that approximates the posterior distribution of the state sequence. Section 6 presents a simulated experiment which demonstrates that the minimum variance estimate (as approximated by our MCMC algorithm) can be superior to the maximum a posterior estimate (which can be computed using an IKS).

2. Model Formulation

The measurement vectors are related to the state vectors by

$$z_k = h_k(x_k) + v_k$$

$$v_k \sim \mathcal{N}(0, R_k)$$

where each $$h_k : \mathbb{R}^n \to \mathbb{R}^m$$ is a known model for the mean of the measurements at time index $$k$$ given the state vector at time index $$k$$ and each $$R_k \in \mathbb{R}^{m \times m}$$ is a known positive definite covariance matrix.
We also have a model for the dynamics of the system that is expressed as a prior distribution on the state sequence. To be specific, for \( k = 1, \ldots, N \)

\[
x_k = g_k(x_{k-1}) + w_k
\]

where each \( g_k : \mathbb{R}^n \rightarrow \mathbb{R}^m \) is a known model for the mean of the state vector at time index \( k \) given the value of the state vector at time index \( k - 1 \) and each \( Q_k \in \mathbb{R}^{m \times m} \) is a known positive definite covariance matrix. In addition, the set of random variables \( \{ w_k \} \cup \{ w_k \} \) are independent.

Kalman filter and smoother models usually include a known prior state estimate \( \hat{x}_1 \) where

\[
x_1 - \hat{x}_1 \sim \mathcal{N}(0, P_1)
\]

\( P_1 \in \mathbb{R}^{m \times m} \) is a known positive definite covariance matrix, and \( x_1 - \hat{x}_1 \) is independent of the other random variables. In order to avoid a special case for the initial state estimate, we define

\[
w_1 = x_1 - \hat{x}_1 , \quad g_1(x_0) = \hat{x}_1 , \quad \text{and} \quad Q_1 = P_1
\]

This includes the initial state estimate as part of our prior model for the state vector sequence. Note that the function \( g_1(x_0) \) is constant, i.e., it does not depend on the value of \( x_0 \). In order to avoid another special case, we also define

\[
x_{N+1} = 0 , \quad g_{N+1}(x_N) = 0 , \quad \text{and} \quad Q_{N+1} = I_n
\]

where \( I_n \) is the \( n \times n \) identity matrix.

3. **Likelihood Function**

It follows from our problem formulation that, for \( k = 1, \ldots, N \), the negative log probability density for the state vector at time \( k \) given the state vector at time index \( k - 1 \) is

\[
-\log p(x_k \mid x_{k-1}) = \frac{1}{2} \log \det(2\pi Q_k) + \frac{1}{2} [x_k - g_k(x_{k-1})]^T Q_k^{-1} [x_k - g_k(x_{k-1})]
\]

(Note that \( \det(2\pi Q_k) \) is equal to \( (2\pi)^n \det(Q_k) \)). It also follows that, for \( k = 1, \ldots, N \), the negative log probability density for the measurement vector \( z_k \) given the state vector \( x_k \) is

\[
-\log p(z_k \mid x_k) = \frac{1}{2} \log \det(2\pi R_k) + \frac{1}{2} [z_k - h_k(x_k)]^T R_k^{-1} [z_k - h_k(x_k)]
\]

Using the independence property, it follows that the negative log likelihood of the state sequence \( \{ x_k \} \), and the measurement sequence \( \{ z_k \} \), is given by

\[
L(x_1, \ldots, x_N) = \sum_{k=1}^{N} -\log p(z_k \mid x_k) - \log p(x_k \mid x_{k-1})
\]

\[
= \sum_{k=1}^{N} \frac{1}{2} \log \det(2\pi Q_k) + \frac{1}{2} [x_k - g_k(x_{k-1})]^T Q_k^{-1} [x_k - g_k(x_{k-1})]
\]

\[
+ \sum_{k=1}^{N} \frac{1}{2} \log \det(2\pi R_k) + \frac{1}{2} [z_k - h_k(x_k)]^T R_k^{-1} [z_k - h_k(x_k)]
\]

The joint density is expressed in terms of the negative log likelihood by

\[
p(\{ z_k \}, \{ x_k \}) = \exp[-L(x_1, \ldots, x_N)] \tag{2}
\]

We use \( \partial_k L(x_1, \ldots, x_N) \) to denote the partial derivative of \( L(x_1, \ldots, x_N) \) with respect to \( x_k \).

The following lemma follows from [10, equation (5)]:

\[
\frac{\partial}{\partial x_k} L(x_1, \ldots, x_N) = -\sum_{k'=1}^{N} \frac{1}{2} \frac{\partial}{\partial x_k} \log \det(2\pi Q_k) + \frac{1}{2} [x_k - g_k(x_{k-1})]^T Q_k^{-1} [x_k - g_k(x_{k-1})]
\]

\[
+ \sum_{k'=1}^{N} \frac{1}{2} \frac{\partial}{\partial x_k} \log \det(2\pi R_k) + \frac{1}{2} [z_k - h_k(x_k)]^T R_k^{-1} [z_k - h_k(x_k)]
\]

\[
= \sum_{k'=1}^{N} \mathbb{E} \left[ \frac{\partial}{\partial x_k} \log p(x_{k'} \mid x_{k-1}) \right] + \sum_{k'=1}^{N} \mathbb{E} \left[ \frac{\partial}{\partial x_k} \log p(z_{k'} \mid x_{k-1}) \right]
\]
Lemma 1 For \( k = 1, \ldots, N \),
\[
\partial_k L(x_1, \ldots, x_N) = [x_k - g_k(x_{k-1})]^T Q_k^{-1} - [x_{k+1} - g_{k+1}(x_k)]^T Q_{k+1}^{-1} h_{k+1}^{(1)}(x_k)
\]
\[
- [z_k - h_k(x_k)]^T R_k^{-1} h_k^{(1)}(x_k)
\]

Given a matrix \( A \in \mathbb{R}^{p \times q} \) we use the notation \( \text{rvec}(A) \in \mathbb{R}^{pq} \) for the corresponding column vector in row major order, i.e., for \( i = 1, \ldots, p \) and \( j = 1, \ldots, q \),
\[
\text{rvec}(A)_{(i-1)q+j} = A_{i,j}
\]

Given another matrix \( B \in \mathbb{R}^{r \times s} \) we use the the notation \( A \otimes B \) for the Kronecker product defined by
\[
A \otimes B = \begin{pmatrix}
A_{1,1}B & \cdots & A_{1,q}B \\
\vdots & \ddots & \vdots \\
A_{p,1}B & \cdots & A_{p,q}B
\end{pmatrix}
\]

Given a matrix valued function \( C(x) \) where \( C : \mathbb{R}^n \to \mathbb{R}^{p \times q} \), its derivative \( C^{(1)} : \mathbb{R}^n \to \mathbb{R}^{pq \times n} \) is defined by
\[
C^{(1)}(x) = \frac{d}{dx} \{ \text{rvec}[C(x)] \}
\]

A discussion of this notation for derivatives of matrix valued functions can be found in [11, Section 6.3]. Using this notation above we have the following lemma [10, equations (6)-(8)]:

Lemma 2 Using the notation \( I_n \) for the \( n \times n \) identity matrix, for \( k = 1, \ldots, N \),
\[
\partial_k \partial_k L(x_1, \ldots, x_N) = Q_k^{-1} + g_{k+1}^{(1)}(x_k) [x_{k+1} - g_{k+1}(x_k)]^T Q_{k+1}^{-1} h_{k+1}^{(1)}(x_k)
\]
\[
- \left( \left( [x_{k+1} - g_{k+1}(x_k)]^T Q_{k+1}^{-1} \right) \otimes I_n \right) g_{k+1}^{(2)}(x_k)
\]
\[
- \left( \left( [z_k - h_k(x_k)]^T R_k^{-1} \right) \otimes I_n \right) h_k^{(2)}(x_k)
\]

and for \( k = 2, \ldots, N \)
\[
\partial_{k-1} \partial_k L(x_1, \ldots, x_N) = -Q_k^{-1} \partial_{k-1} g_k(x_{k-1})
\]

Given a sequence \( \{ B_k \in \mathbb{R}^{n \times n} \} \) for \( k = 1, \ldots, N \), and a sequence \( \{ C_k \in \mathbb{R}^{n \times n} \} \) for \( k = 2, \ldots, N \), we define the notation
\[
\text{tridiag}(B_1, C_2, \ldots, C_N, B_N) = \begin{pmatrix}
B_1 & C_2^T & 0 & \cdots & 0 \\
C_2 & B_2 & C_3^T & \ddots & \vdots \\
0 & C_3 & B_3 & \ddots & 0 \\
\vdots & \ddots & \ddots & \ddots & C_N^T \\
0 & \cdots & 0 & C_N & B_N
\end{pmatrix}
\]

Using this notation, we have the following theorem [10, equation (10)]

Theorem 3 We define the sequences \( \{ B_k : \mathbb{R}^n \to \mathbb{R}^{n \times n} \} \) and \( \{ C_k : \mathbb{R}^n \to \mathbb{R}^{n \times n} \} \) by the first two equations below. It follows that the Hessian of \( L(x_1, \ldots, x_N) \) is given by the third equation below.
\[
B_k(x_k) = \partial_k \partial_k L(x_1, \ldots, x_N)
\]
\[
C_k(x_{k-1}) = \partial_{k-1} \partial_k L(x_1, \ldots, x_N)
\]
\[
L^{(2)}(x_1, \ldots, x_N) = \text{tridiag}[B_1(x_1), C_2(x_1), \ldots, C_N(x_{N-1}), B_N(x_N)]
\]
4. Block Tridiagonal Cholesky Inversion

Given a positive definite matrix \( A \in \mathbb{R}^{n \times n} \), we use \( A^{-\top} \) to denote the inverse of the transpose of \( A \) and \( \text{chol}(A) \) to denote a Cholesky factor of \( A \), i.e., \( \text{chol}(A) \) is a lower triangular matrix such that

\[
I_n = A^{-\top}A^\top \\
A = \text{chol}(A)\text{chol}(A)^\top
\]

Given a sequence of vectors \( \{a_k \in \mathbb{R}^n\} \) we use the notation \( \text{vec}\{a_k\} \in \mathbb{R}^{nN} \) to denote the corresponding column vector, i.e.,

\[
\text{vec}\{a_k\} = \begin{pmatrix}
a_1 \\
\vdots \\
a_N
\end{pmatrix}
\]

We are given the following inputs to our block tridiagonal Cholesky algorithm: a sequence of symmetric matrices \( \{B_k \in \mathbb{R}^{n \times n}\} \), a sequence of matrices \( \{C_k \in \mathbb{R}^{n \times n}\} \) (\( C_1 \) is not used), and a sequence of vectors \( \{e_k \in \mathbb{R}^n\} \). The algorithm computes a sequence of vectors \( \{s_k \in \mathbb{R}^n\} \) such that

\[
\text{vec}\{s_k\} = \text{chol}\{\text{tridiag}(B_1, C_2, \cdots, C_N, B_N)\}^{-\top}\text{vec}\{e_k\}
\] (3)

**Algorithm 4** Block tridiagonal Cholesky inversion

(i) Set \( F_1 = \text{chol}(B_1) \). For \( k = 2, \cdots, N \) set \( G_k = C_kF_{k-1}^{-\top} \) and \( F_k = \text{chol}(B_k - G_kG_k^\top) \).

(ii) Set \( s_N = F_N^{-\top}e_N \). For \( k = N-1, \cdots, 1 \) set \( s_k = F_k^{-\top}(e_k - G_k^\top s_{k+1}) \).

The following theorem and corollary are a restatement of [10, Proposition 6 and Lemma 7]:

**Theorem 5** If the matrix \( \text{tridiag}(B_1, C_2, \cdots, C_N, B_N) \) in Algorithm 4 is positive definite, all of the arguments to \( \text{chol} \) in the algorithm are positive definite and the output sequence \( \{s_k\} \) satisfies equation (3). Furthermore, the Algorithm has computational complexity \( O(Nn^3) \) floating point operations.

**Corollary 6** Suppose the matrix \( \text{tridiag}(B_1, C_2, \cdots, C_N, B_N) \) in Algorithm 4 is positive definite and \( \{e_k\} \) is a sequence of independent random vectors with \( e_k \sim \mathcal{N}(0, I_n) \). It follows that

\[
\text{vec}\{s_k\} \sim \mathcal{N}\left[0, \text{tridiag}(B_1, C_2, \cdots, C_N, B_N)^{-1}\right]
\]

For \( i = 1, \ldots, Nn \) we define the \( i \)-th elementary vector \( E^i \in \mathbb{R}^{nN} \) by

\[
E^i_j = \begin{cases} 
1 & \text{if } i = j \\
0 & \text{otherwise}
\end{cases}
\]

where \( E^i_j \) denotes the \( j \)-th component of \( E^i \).

**Corollary 7** Suppose the matrix \( \text{tridiag}(B_1, C_2, \cdots, C_N, B_N) \) in Algorithm 4 is positive definite, the output \( \{s_k^i\} \) corresponding to the input satisfying \( \text{vec}\{e_k\} = E^i \) and the output \( \{s_k^j\} \) corresponding to the input satisfying \( \text{vec}\{e_k\} = E^j \). It follows that

\[
(\text{vec}\{s_k^i\})^\top\text{vec}\{s_k^j\}
\]

is equal to the \((i, j)\) component of \( \text{tridiag}(B_1, C_2, \cdots, C_N, B_N)^{-1} \).
5. MCMC Approximation of the Posterior Distribution

The Iterated Kalman Smooth (IKS), e.g. [1, Figure 2], can be used to find a local minimizer \( \{ \hat{x}_k \} \) for \( L(x_1, \ldots, x_N) \). For this paper, we assume that there is only one local minimizer of \( L(x_1, \ldots, x_N) \) (the more general case leads to complications that are considered in [10]). We also assume that the second order sufficiency condition holds, i.e., \( L^{(2)}(\hat{x}_1, \ldots, \hat{x}_N) \) is positive definite. We define

\[
\hat{\Sigma} = L^{(2)}(\hat{x}_1, \ldots, \hat{x}_N)^{-1}
\]

\[
\hat{L}(x_1, \ldots, x_N) = L(\hat{x}_1, \ldots, \hat{x}_N) + (\text{vec}\{x_k\} - \text{vec}\{\hat{x}_k\})^T\hat{\Sigma}^{-1}(\text{vec}\{x_k\} - \text{vec}\{\hat{x}_k\})
\]

Using the fact that \( L^{(1)}(\hat{x}_1, \ldots, \hat{x}_N) = 0 \) we have

\[
L(x_1, \ldots, x_N) = \hat{L}(x_1, \ldots, x_N) + o\left(\|\text{vec}\{x_k\} - \text{vec}\{\hat{x}_k\}\|^2\right)
\]

Given equation (2), this leads us to the approximation for \( p(\{x_k\}, \{z_k\}) \) defined by

\[
\hat{p}(\{x_k\}, \{z_k\}) = \exp[-\hat{L}(x_1, \ldots, x_N)]
\]

Replacing \( p(\{x_k\}, \{z_k\}) \) by this approximation in equation (1), we obtain the following approximation for \( p(\{x_k\} \mid \{z_k\}) \)

\[
\hat{p}(\{x_k\} \mid \{z_k\}) = [\det(2\pi\hat{\Sigma})]^{-1/2} \exp\left[-\frac{1}{2}(\text{vec}\{x_k\} - \text{vec}\{\hat{x}_k\})^T\hat{\Sigma}^{-1}\frac{1}{2}(\text{vec}\{x_k\} - \text{vec}\{\hat{x}_k\})\right]
\]

which is the density for an \( \mathbf{N}(\{\hat{x}_k\}, \hat{\Sigma}) \) distribution. The following corollary is a direct application of the Corollary 6 and Theorem 3:

**Theorem 8** Suppose that \( \alpha > 0 \) and that in Algorithm 4 the input matrices are given by \( B_k = \alpha^{-1}\partial_k\partial_k L(\hat{x}_1, \ldots, \hat{x}_N) \) and \( C_k = \alpha^{-1}\partial_k \partial_k L(\hat{x}_1, \ldots, \hat{x}_N) \). Further suppose \( \{e_k\} \) is a sequence of independent random vectors with each \( e_k \sim \mathbf{N}(0, I_n) \). It follows that \( \text{vec}\{s_k\} \sim \mathbf{N}(0, \alpha\hat{\Sigma}) \) where \( \hat{\Sigma} \) is given by equation (4).

We are given the following inputs to our MCMC algorithm for sampling from the posterior distribution \( p(\{x_k\} \mid \{z_k\}) \): the measurements \( \{z_k\} \), the mean measurement model functions \( \{b_k(x_k)\} \), the measurement variances \( \{Q_k\} \), the mean transition model functions \( \{g_k(x_{k-1})\} \), the transition variances \( \{R_k\} \), the variance scaling factor \( \alpha > 0 \) (see Remark 10), and the number of state sequences to simulate \( M \).

The following is an application of the random-walk Metropolis algorithm [12, equation (1.7)]. It generates a sequence of state vectors \( \{\{x_k^\ell\} : \ell = 1, \ldots, M\} \) such that for any function of the state sequence \( f(\{x_k\}) \)

\[
\mathbf{E}[f(\{x_k\}) \mid \{z_k\}] = \int_{-\infty}^{+\infty} f(\{x_k\}) p(\{x_k\} \mid \{z_k\}) d\{x_k\} \approx \frac{1}{M} \sum_{\ell=1}^{M} f(\{x_k^\ell\})
\]

We use \( \{\{x_k^\ell\}\} \) to denote this sequence of sequence of state vectors.

**Algorithm 9** \textit{MCMC sampling the state sequence posterior distribution}

(i) Compute a local minimizer \( \{\hat{x}_k\} \) of \( L(x_1, \ldots, x_N) \) and set \( \hat{\Sigma} \) to \( L^{(2)}(\hat{x}_1, \ldots, \hat{x}_N)^{-1} \).

(ii) Set \( x_k^0 \) to \( \{\hat{x}_k\} \) and set \( \ell = 1 \).

(iii) Sample \( \{s_k^\ell\} \) from \( \mathbf{N}(\{x_k^{\ell-1}\}, \alpha\hat{\Sigma}) \).
(iv) Sample \( u^\ell \) from a uniform distribution on \([0, 1]\).

(v) Set \( r^\ell = \frac{\mathbf{p}(\{s_k^\ell\} \mid \{z_k\})}{\mathbf{p}(\{x_{k-1}^\ell\} \mid \{z_k\})} \).

(vi) If \( u^\ell \leq r^\ell \), set \( \{x_k^\ell\} = \{s_k^\ell\} \), otherwise \( \{x_k^\ell\} = \{x_{k-1}^\ell\} \).

(vii) If \( \ell < M \), set \( \ell \) to \( \ell + 1 \) and go to step iii.

**Remark 10** We define the acceptance rate for the algorithm above as the number of indices \( \ell \) for which \( u^\ell \leq r^\ell \) divided by \( M \) (the total number of iterations). A pilot run of the algorithm is repeated, and the value of \( \alpha \) is adjusted, until the corresponding acceptance rate is between 20 and 30 percent (the optimal value is 23.4\% [13, Theorem 1]).

**Remark 11** It follows from equations (2) and (1) that \( r^\ell \) can be computed using the identity
\[
\frac{\mathbf{p}(\{s_k^\ell\} \mid \{z_k\})}{\mathbf{p}(\{x_{k-1}^\ell\} \mid \{z_k\})} = \exp \left[ L(x_{k-1}^{\ell-1}, \ldots, x_N^{\ell-1}) - L(s_1^\ell, \ldots, s_N^\ell) \right]
\]

**Remark 12** This is the same as [10, Algorithm 9] for the case where there is only one local minimizer of \( L(x_1, \ldots, x_N) \). In both algorithms, the proposal distribution for this case is symmetric, i.e., the density for \( \mathbf{N}(\{x_k^{\ell-1}\}, \alpha \Sigma) \) evaluated at \( \{s_k^\ell\} \) is equal to the density for \( \mathbf{N}(\{s_k^\ell\}, \alpha \Sigma) \) evaluated at \( \{x_k^{\ell-1}\} \). For this reason, the ratio of these two densities is not be included in the computation of \( r^\ell \) above.

### 6. Simulated Experiment
We consider the fed-batch bio-reactor model in [14, Section 4.2 Example 1]. At each time index \( k \), the state vector \( x_k \) has three components: \( x_{k,1} = a_k \) is the biomass concentration, \( x_{k,2} = b_k \) is the substrate concentration, and \( x_{k,3} = c_k \) is the volume. The biomass growth rate as a function of substrate concentration \( \mu : \mathbb{R} \to \mathbb{R} \) is given by
\[
\mu(S) = \frac{\mu_m S}{\beta_m + S + S^2/\beta_i}
\]
where \( \mu_m = 0.5 \) is the maximum growth rate, \( \beta_m = 0.5 \) is the substrate saturation constant, and \( \beta_i = 0.5 \) is the substrate inhibition constant [15, page 58]. The following discrete approximation of the stochastic dynamical system is used for this example
\[
\begin{align*}
a_k &= a_{k-1} + \left[ +\mu(b_{k-1})a_{k-1} - \frac{F a_{k-1}}{c_{k-1}} \right] \Delta t + w_{k,1} \\
b_k &= b_{k-1} + \left[ -\frac{\mu(b_{k-1})a_{k-1}}{Y} + \frac{F(S_F - b_{k-1})}{c_{k-1}} \right] \Delta t + w_{k,2} \\
c_k &= c_{k-1} + F \Delta t + w_{k,3}
\end{align*}
\]
where \( F = 0.5 \) is the rate at which feed is flowing into the bio-reactor, \( Y = 0.5 \) is the yield coefficient (amount of biomass per unit substrate), \( S_F = 10 \) is the concentration of the substrate in the feed, \( \Delta t = 0.4 \) is the time between index \( k \) and index \( k+1 \), \( w_k \sim \mathbf{N}(0, Q_k) \) is the stochastic component of the dynamics, and \( Q_k = 0.2 I_3 \) for all \( k \). For \( k = 2, \ldots, N \), the corresponding mean transition model \( g_k : \mathbb{R}^3 \to \mathbb{R}^3 \) is
\[
g_k \left( \begin{array}{c} a_{k-1} \\ b_{k-1} \\ c_{k-1} \end{array} \right) = \left( \begin{array}{c} a_{k-1} \\ b_{k-1} \\ c_{k-1} \end{array} \right) + \left( \begin{array}{c} \frac{\mu(b_{k-1})a_{k-1}}{Y} - \frac{F a_{k-1}}{c_{k-1}} \\ -\frac{\mu(b_{k-1})a_{k-1}}{Y} + \frac{F(S_F - b_{k-1})}{c_{k-1}} \\ 0 \end{array} \right) \Delta t
\]
The actual initial state vector for the simulations \( x_1 \), and its estimate \( g_1(x_0) \), are
\[
g_1(x_0) = x_1 = (a_1, b_1, c_1)^T = (4, 5, 3)^T
\]
At each time index $k$, the measurement vector $z_k$ has two components,

\[
\begin{align*}
    z_{k,1} &= a_k + b_k + v_{k,1} \\
    z_{k,1} &= b_k + c_k + v_{k,1}
\end{align*}
\]  

where $v_k \sim \mathcal{N}(0, R_k)$ is the measurement noise and $R_k = I_2$ for all $k$. For $k = 1, \ldots, N$, the corresponding mean measurement model $h_k : \mathbb{R}^3 \to \mathbb{R}^2$ is

\[
h_k\left(\begin{array}{c}
    a_k \\
    b_k \\
    c_k
\end{array}\right) = \left(\begin{array}{c}
    a_k + b_k \\
    b_k + c_k
\end{array}\right)
\]

**Remark 13** Note that all the mean measurement model functions $\{h_k(x_k)\}$ are affine. On the other hand and except for $k = 1$, the mean transition model functions $\{g_k(x_{k-1})\}$ are not affine. If they were also affine, the minimizer of $L(x_1, \ldots, x_N)$ would be equal to the minimum variance estimate $E(\{x_k\} | \{z_k\})$.

The number of MCMC simulated sequences to generate $M$ was determined using the Raftery-Lewis criteria [16]. For each function $f$ that was integrated using the average of the MCMC samples (see approximation 6), the Raftery-Lewis input parameters were the quantiles 0.025, 0.25, 0.5, 0.75, 0.975, with corresponding precisions 0.02, 0.05, 0.01, 0.05, 0.02, and with requested probability 0.95. It was thereby determined that the value $M = 20,000$ was sufficiently large for all the integrations in this simulated experiment.

For this simulated experiment there are $N = 100$ time indices. The true state sequence $\{x_k\}$ was generated using the stochastic difference equation (7). Then the measurement sequence $\{z_k\}$ was generated using equation (8). Algorithm 9 was run with the inputs specified above. We use $\{\{x'_k\}\}$ to denote the corresponding sequence of sequence of state vectors generated by the algorithm. The MCMC estimate for the state sequence $\{x_k\}$ is given by

\[
\bar{x}_k = \frac{1}{M} \sum_{\ell=1}^{M} x'_k
\]

We define the indicator $\chi : \{\text{false, true}\} \to \{0, 1\}$ by $\chi(\text{false}) = 0$ and $\chi(\text{true}) = 1$. For $k = 1, \ldots, N$ and $j = 1, 2, 3$, the corresponding MCMC lower $\gamma$-confidence limit is defined as the minimum value $\bar{\lambda}_{k,j}$ such that

\[
\frac{1}{2} \gamma \leq \frac{1}{M} \sum_{\ell=1}^{M} \chi\left(x'_{k,j} \leq \bar{\lambda}_{k,j}\right)
\]

The corresponding MCMC upper $\gamma$-confidence limit is defined as the maximum value $\bar{\rho}_{k,j}$ such that

\[
\frac{1}{2} \gamma \leq \frac{1}{M} \sum_{\ell=1}^{M} \chi\left(\bar{\rho}_{k,j} \leq x'_{k,j}\right)
\]

In Figure 1, the measurement sequence $\{z_k\}$ is displayed using circles, the model for the measurement mean given the state sequence $\{h_k(x_k)\}$ is displayed using thick lines, and the model for the measurement mean given the estimated state sequence $\{h_k(\bar{x}_k)\}$ is displayed using thin lines. The upper plot in the figure corresponds to the first measurement component $z_{k,1}$ and the lower plot corresponds to the second measurement component $z_{k,2}$.

In Figure 2 the true state sequence $\{x_k\}$ is displayed using thick lines, the estimated state sequence $\{\bar{x}_k\}$ is displayed using thin lines, the lower 95%-confidence limits $\{\bar{\lambda}_k\}$ is displayed...
Figure 1. Upper plot is for $z_{k,1}$ lower plot is for $z_{k,2}$, circles correspond to measurement sequence $\{z_k\}$, thick lines correspond to true state measurement mean sequence $\{h_k(x_k)\}$, thin lines correspond to estimated state measurement mean sequence $\{h_k(\bar{x}_k)\}$.

using dash-dot lines, and the upper 95%-confidence limits $\{\hat{\rho}_k\}$ is displayed using dash-dot lines. The average root mean square error

$$\sqrt{\frac{1}{N} \sum_{k=1}^{N} (x_{k,j} - \bar{x}_{k,j})^2}$$

was 1.44, 1.34, and 1.3 for $j$ equal to 1, 2 and 3 respectively.

For comparison, we use $\{\hat{x}_k\}$ to denote the maximum a posteriori estimate of the state sequence, i.e., the minimizer of $L(x_1, \ldots, x_N)$ which can be computed using an Iterated Kalman Smoother (IKS). The variance corresponding the IKS is the matrix $\Sigma^{-1}$ in equation (5). The entries of this matrix can be computed using Corollary 7. Under the normal approximation, this variance is converted to 95%-confidence limits for each component of the state sequence. For $k = 1, \ldots, N$ and $j = 1, 2, 3$, we denote the corresponding lower and upper confidence limit by $\hat{\lambda}_{k,j}$ and $\hat{\rho}_{k,j}$ respectively.

In Figure 3 the true state sequence $\{x_k\}$ is displayed using thick lines, the estimated state sequence $\{\hat{x}_k\}$ is displayed using thin lines, the lower 95%-confidence limits $\{\hat{\lambda}_k\}$ is displayed using dash-dot lines, and the upper 95%-confidence limits $\{\hat{\rho}_k\}$ is displayed using dash-dot lines. The average root mean square error

$$\sqrt{\frac{1}{N} \sum_{k=1}^{N} (x_{k,j} - \hat{x}_{k,j})^2}$$
Figure 2. Upper plot is for $a_k = x_{k,1}$, middle plot is for $b_k = x_{k,2}$, lower plot is for $c_k = x_{k,3}$. Thick lines correspond to true state sequence $\{x_k\}$, thin lines correspond to MCMC estimate state sequence $\{\bar{x}_k\}$, dash-dot lines correspond to MCMC 95%-confidence limits.

was 2.13, 2.14, and 2.0 for $j$ equal to 1, 2 and 3 respectively.

7. Conclusions
We presented an MCMC method for sampling state sequences from their posterior density. The method exploits the special structure of a state-space model to efficiently generate a Markov Chain that approximates the posterior density. This is done using a random-walk proposal density that is a second order approximation for the posterior. As opposed to methods based on particle filters [4, 5, 6], the proposed method approximates posterior of the entire state sequence and not just the marginal posterior at each of the time point. The method in [7] also approximates the entire posterior but it exploits a different methodology that relies upon sequential sampling and particulate representations. Another advantage of MCMC over particle filters is the availability of convergence tests that determine the number of samples required to reconstruct the posterior with a certain accuracy. Simulated data show the effectiveness of this approach and demonstrate that the proposed method can be superior to optimization estimates such as the iterated Kalman smoother.

Acknowledgments
This research was partially supported by the Italian Ministry of University and Research through the PRIN Project entitled 'Tecniche ed applicazioni innovative di identificazione e controllo adattativo' and by the NIH grant P41 EB-001975 entitled 'Resource Facility for Population Kinetics'.
Figure 3. Upper plot is for $a_k = x_{k,1}$, middle plot is for $b_k = x_{k,2}$, lower plot is for $c_k = x_{k,3}$. thick lines correspond to true state sequence $\{x_k\}$, thin lines correspond to IKS estimate state sequence $\{\hat{x}_k\}$, dash-dot lines correspond to IKS 95%-confidence limits.