Bayesian Learning of Degenerate Linear Gaussian State Space Models Using Markov Chain Monte Carlo

Pete Bunch, James Murphy, and Simon Godsill

***Abstract*—Linear Gaussian state-space models are ubiquitous in signal processing, and an important procedure is that of estimating system parameters from observed data. Rather than making a single point estimate, it is often desirable to conduct Bayesian learning, in which the entire posterior distribution of the unknown parameters is sought. This can be achieved using Markov chain Monte Carlo. On some occasions it is possible to deduce the form of the unknown system matrices in terms of a small number of scalar parameters, by considering the underlying physical processes involved. Here we study the case where this is not possible, and the entire matrices must be treated as unknowns. An efficient Gibbs sampling algorithm exists for the basic formulation of linear model. We extend this to the more challenging situation where the transition model is possibly degenerate, i.e., the transition covariance matrix is singular. Appropriate Markov kernels are devised and demonstrated with simulations.**

***Index Terms*—Covariance matrices, linear systems, markov processes, monte carlo methods, parameter estimation, time series analysis.**

I. Introduction

STATE space models are frequently used to describe time-varying systems, and Linear Gaussian models in particular are ubiquitous throughout signal processing. The physical phenomena modelled within this framework include the kinematics of moving targets [1], audio signals [2], genetic networks [3], and many others. The popularity of linear Gaussian state space models stems in part from their analytic tractability. Inference tasks can be performed in closed form using algorithms such as the Kalman filter [4] and Rauch-Tung-Striebel (RTS) smoother [5].

A linear Gaussian state space model is specified by a number of system matrices which govern the latent state and observation processes. An important consideration is how to learn these fixed matrices from observed data. Sometimes it is possible to deduce the form of the system matrices in terms of a small number of scalar parameters by considering the physical mechanisms underlying the system, and then attempt to learn these. (See e.g., [6], [7].) However, in many systems there may not be such an obvious parameterization, and it may be better to treat each system matrix in its entirety as an unknown variable to be learned; this is the case studied in this paper. There is a large body of research on this topic, which has generally concentrated on point estimate methods. These include subspace identification approaches [8], [9], and maximum likelihood estimates using either gradient-based algorithms (see e.g., [10], [11]) or Expectation-Maximisation (EM) [12]–[16].

More recently, Bayesian approaches have been developed for learning linear Gaussian state space models, in which full posterior distributions for the system parameters are calculated. Since this is not possible analytically (see discussion in [17]), approximations must be employed. One possibility is to take a variational approach, approximating the full joint posterior distribution as a product of independent marginals [17]–[19]. Alternatively, the true joint posterior can be targeted using Markov chain Monte Carlo (MCMC). This has the advantage of providing consistent estimates not only of the system matrices but also of any function of these quantities (e.g., phase margins, system poles). The price we pay is in computation; it is necessary to allow the Markov chains to run until convergence in order to form good estimates. Generic Metropolis-Hastings strategies such as those adopted in [20] are liable to mix very slowly and exact too great a computational demand. However, if conjugate priors are assumed for the various system matrices, then a Gibbs sampler may be implemented which attains substantially improved efficiency and requires almost no algorithm tuning [21].

In some systems, the components of the latent state may not be free to vary independently of each other. Such systems are best modelled by a degenerate transition model, in which the transition covariance matrix is singular. This is the case, for example, when the latent state comprises the positions and velocities of multiple points on the same rigid body, meaning that their motion is deterministically coupled. In [22], [23], the authors consider the use of degenerate transition models for various tracking applications. The Gibbs sampler of [21] is not able to handle degenerate models. In this paper we extend the MCMC learning framework to encompass this scenario. This is achieved by appropriate factorizations of the covariance matrix which allow us still to exploit the fast-mixing Gibbs kernel. In addition, reversible jump moves [24], [25] are introduced which enable the sampler to learn explicitly the rank of the transition covariance matrix. The new algorithm is important because it allows us to perform Bayesian learning with a broader class of linear state space models, leading to better descriptions of real systems and improved estimation performance.

Manuscript received September 23, 2015; revised January 18, 2016 and March 30, 2016; accepted April 14, 2016. Date of publication May 11, 2016; date of current version June 30, 2016. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Tongtong Li. The authors are supported by the EPSRC BTaRoT grant.

The authors are with the Department of Engineering, University of Cambridge, CB2 1PZ Cambridge, U.K. (e-mail: pb404@cam.ac.uk; jm362@cam.ac.uk; sjg30@cam.ac.uk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2016.2566598

The paper is structured as follows. In Section II we review the Gibbs sampler for basic linear Gaussian state space systems, focusing on the transition model. In Section III we describe modifications for degenerate transitions, and in Section IV we show how the rank of the transition covariance may be learned. Simulations on a toy model, and a motion capture application are presented in Section VI with discussion and conclusions in Section VII.

II. Basic Linear Gaussian Models

A. Definition

State space models are used to represent time-varying systems, and consist of a latent state process $\{x_t\}_{t=1:T}$ and a related observation process $\{y_t\}_{t=1:T}$. The challenge is to infer the sequence of unknown latent states, and learn parameters of the model, from the sequence of known observations. It is usually assumed that the state process is Markovian (i.e., $x_t|x_{t-1}$ is independent of $x_{1:t-2}$) and that each observation depends only on the current state (i.e., $y_t|x_t$ is independent of $x_{1:t-1}$ and $x_{t+1:T}$).

A basic linear state space model therefore obeys the following recursive system equations,

$$x_t = Fx_{t-1} + \epsilon_t^x \tag{1}$$

$$y_t = Hx_t + \epsilon_t^y, \tag{2}$$

where $x_t \in \mathbb{R}^{d_x}, y_t \in \mathbb{R}^{d_y}$. $F \in \mathbb{R}^{d_x \times d_x}$ and $H \in \mathbb{R}^{d_y \times d_x}$ are the transition and observation matrices. $\epsilon_t^x \in \mathbb{R}^{d_x}$ and $\epsilon_t^y \in \mathbb{R}^{d_y}$ are the state disturbance and observation noise variables and we additionally assume that these are drawn from a Gaussian distribution,

$$\epsilon_t^x \sim \mathcal{N}(0, Q) \tag{3}$$

$$\epsilon_t^y \sim \mathcal{N}(0, R), \tag{4}$$

where Q and R are positive definite covariance matrices. This model can be written equivalently in terms of transition and observation densities,

$$p(x_t|x_{t-1}, F, Q) = \mathcal{N}(x_t|Fx_{t-1}, Q) \tag{5}$$

$$p(y_t|x_t, H, R) = \mathcal{N}(y_t|Hx_t, R). \tag{6}$$

The initial state x_1 may be known or may be assigned a Gaussian prior.

B. Identifiability

The system is parameterized by four matrices: F, H, Q, R. In some applications, it may be appropriate to treat all of these as completely unknown. However, this scenario can be troublesome for learning algorithms, because the system is not identifiable—we obtain the same likelihood if the latent state is multiplied by an arbitrary invertible matrix, along with corresponding multiplications of F, Q and H. (See e.g., [26].) To resolve the parameter ambiguity, it is necessary to constrain the values of one or more of the system matrices. One option (that taken in [26], for example) is to set Q to the identity matrix. This entails that the nature of the latent states and their relationship to the observations are entirely unknown a priori. In this paper we consider a contrasting approach, where we resolve the ambiguity by specifying the nature of the latent states and setting H to a pre-determined value. This approach has the advantage that the resulting model is more easily interpreted, since we can choose the latent states to be physically meaningful quantities. For example, in the motion capture application studied in Section VI, we choose the latent states to be the positions and velocities of the markers.

For the rest of the paper we will focus on learning the matrices of the transition model, i.e., F and Q. It is straightforward include learning of the observation covariance matrix R by adding sampling steps to the MCMC scheme which draw from the appropriate conditional distribution, and this is demonstrated in the simulations in Section VI.

C. Inference

Conditional on the system matrices, state inference may be carried out analytically. Posterior filtering and smoothing densities may be calculated using the Kalman filter [4] and Rauch-Tung-Striebel (RTS) smoother [5]. Furthermore, the Kalman filter may also be used to evaluate the marginal likelihood $p(y_{1:T}|F, Q)$, and it is possible to draw posterior samples from the state posterior $p(x_{1:T}|y_{1:T}, F, Q)$ using the forward-filtering-backward-sampling method [27]. The purpose of a learning algorithm is to estimate unknown values of $\{F, Q\}$ from a sequence of observations. Within the Bayesian framework this means calculating or approximating the posterior distribution $p(F, Q|y_{1:T})$, which we can achieve using MCMC.

For Bayesian learning, we need to define prior distributions for the unknown system matrices F and Q. Although these should be selected to reflect prior belief about the system in question, there is substantial benefit in using conjugate priors, as these enable us to use Gibbs sampling moves. Typically, we will not have strong prior beliefs about the parameters, so it is reasonable to use an uninformative conjugate prior. However, if we do have prior knowledge to take into account which means that a conjugate prior is not appropriate, then we can treat each Gibbs move as a proposal in a Metropolis-Hastings scheme and use an accept/reject stage to account for the difference between the true prior and the conjugate prior used for the sampling [21].

D. System Matrix Priors

The conjugate prior for F, Q is a matrix normal-inverse Wishart distribution [21],

$$Q \sim \mathcal{IW}(\nu_0, \Psi_0) \tag{7}$$

$$F|Q \sim \mathcal{MN}(M_0, Q, \Omega_0), \tag{8}$$

with the following density function,

$$p(F, Q) = p(F|Q)\, p(Q) \tag{9}$$

$$p(Q) = \frac{|\Psi_0|^{\frac{\nu_0}{2}}}{2^{\frac{\nu_0}{2}}\Gamma\left(\frac{\nu_0}{2}\right)}|Q|^{-\frac{\nu_0 + d_x + 1}{2}} \exp\left(-\frac{1}{2}\mathrm{Tr}\left[Q^{-1}\Psi_0\right]\right) \tag{10}$$

$$p(F|Q) = (2\pi)^{-\frac{d_x^2}{2}} |Q|^{-\frac{d_x}{2}} |\Omega_0|^{-\frac{d_x}{2}} \times \exp\left(-\frac{1}{2}\mathrm{Tr}\left[Q^{-1}(F-M_0)\Omega_0^{-1}(F-M_0)^T\right]\right) \tag{11}$$

$\nu_0 \in \mathbb{R}, \nu_0 > d_x - 1$. Ψ_0 and Ω_0 are $d_x \times d_x$ positive definite matrices. $M_0 \in \mathbb{R}^{d_x \times d_x}$.

E. Gibbs Sampling for Basic Linear Gaussian Models

Our principal interest is to learn the system matrices F and Q from a sequence of observations $y_{1:T}$. The posterior distribution over these variables $\pi(F,Q) = p(F,Q|y_{1:T})$ is not amenable to MCMC sampling because there is no analytic expression for the density. Instead we introduce the latent state sequence $x_{1:T}$ as a nuisance variable and target the joint posterior,

$$\pi(F,Q,x_{1:T}) = p(F,Q,x_{1:T}|y_{1:T}) \propto p(y_{1:T}|x_{1:T})\, p(x_{1:T}|F,Q)\, p(F|Q)\, p(Q). \tag{12}$$

An appropriate Gibbs sampler can be constructed by sampling alternately from the following conditional posterior distributions,

$$\pi(x_{1:T}|F,Q)$$
$$\pi(F,Q|x_{1:T}).$$

The distribution of the sampled values will then converge to the target posterior distribution [28]. The state conditional $\pi(x_{1:T}|F,Q)$ can be sampled using the forward-filtering-backward-sampling algorithm [21], [27], which works by first running a Kalman filter forwards through the data, then running backwards in time sampling each state conditional on those in the future. See Appendix A. The parameter conditional may also be sampled directly. This depends on the state sequence probability,

$$p(x_{1:T}|F,Q) \propto \prod_{t=2}^{T} p(x_t|x_{t-1},F,Q) \propto |Q|^{-\frac{T-1}{2}} \exp\left(-\frac{1}{2}\sum_{t=2}^{T}(x_t - Fx_{t-1})^T Q^{-1}\ (x_t - Fx_{t-1})\right). \tag{13}$$

It is straightforward to show that the required conditional distribution is also a matrix normal-inverse Wishart distribution [21],

$$Q|x_{1:T} \sim \mathcal{IW}(\nu,\Psi) \tag{14}$$

$$F|Q,x_{1:T} \sim \mathcal{MN}(M,Q,\Omega), \tag{15}$$

with the following updated hyperparameters,

$$\begin{aligned} \Omega^{-1} &= \Omega_0^{-1} + S_1 \\ M\Omega^{-1} &= M_0\Omega_0^{-1} + S_2 \\ \nu &= \nu_0 + S_0 \\ \Psi &= \Psi_0 + S_3 + M_0\Omega_0^{-1}M_0^T - M\Omega^{-1}M^T, \end{aligned} \tag{16}$$

in which the following sufficient statistics are used,

$$S_0 = T-1 \qquad S_1 = \sum_{t=2}^{T} x_{t-1}x_{t-1}^T$$
$$S_2 = \sum_{t=2}^{T} x_t x_{t-1}^T \qquad S_3 = \sum_{t=2}^{T} x_t x_t^T. \tag{17}$$

Both matrix normal and inverse Wishart distributions may be sampled using standard methods. Hence we have all the components necessary to implement a Gibbs sampler for this model.

III. Degenerate Transition Models

A commonly encountered variation on the basic linear Gaussian state space model is as follows,

$$x_t = Fx_{t-1} + G\epsilon_t^x \tag{18}$$

$$\epsilon_t^x \sim \mathcal{N}(\epsilon_t^x|0,I), \tag{19}$$

in which ϵ_t^x as dimension r, and G is $d_x \times r$. This is almost equivalent to the model in the previous section, with $Q = GG^T$. However, in the previous model Q was positive definite by definition, whereas here it may not be full rank. When this occurs, the conventional transition density $p(x_t|x_{t-1},F,G)$ is not defined, and the sampling operations underlying our Gibbs sampler are no longer valid. Degenerate models such as these arise in tracking applications (see [22] for a discussion of this phenomenon); when non-Markovian models, such as autoregressive process, are converted to state space form; and when it is natural to parameterize a system with more latent states than there are degrees of freedom (see Section VI.C). In this section, we introduce suitable parameterizations for such degenerate transition models, and show how MCMC moves may be implemented for learning. Here we treat r as fixed and known; in the following section mechanisms will be introduced to learn this too.

The matrix G is not uniquely identifiable from the state sequence, because the distribution of $G\epsilon_t^x$ is identical to that of $G\Xi\epsilon_t^x$ where Ξ is an arbitrary orthogonal matrix. This ambiguity does not affect Q, since,

$$(G\Xi)(G\Xi)^T = GG^T = Q. \tag{20}$$

For this reason, we treat Q as fundamental, as before. G may be defined to be any matrix square root of Q (e.g., the Cholesky factor or the principal matrix square root). In this new setting, Q is a positive semi-definite matrix with rank r.

There are two difficulties with extending the methodology from the basic model to the degenerate case. First, the conjugate matrix normal—inverse Wishart distribution becomes singular. This places an undesirable constraint on the transition matrix which needs to be removed. Second, each transition, $(x_t - Fx_{t-1})$, is now constrained to lie in a particular r-dimensional subspace within the state space; specifically, within the column space of G. This subspace is fixed conditional on any sampled state trajectory $x_{1:T}$ (provided that $T > r$). This makes standard Gibbs sampling from $\pi(F,Q|x_{1:T})$ insufficient, since there are values of F and Q which can never be reached.

There is an obvious way to avoid being constrained to a particular subspace by the sampled state trajectory; simply return to targeting the marginal posterior $\pi(F, Q)$ and fall back to using any generic Metropolis-Hastings approach. However, with a complex and high dimensional parameter space, designing efficient proposal distributions for Metropolis-Hastings is difficult. We would prefer to attain the automated, fast-mixing properties displayed by the Gibbs sampler on the full rank model.

The solution we propose here is to use Metropolis-Hastings steps targeting the marginal to ensure irreducibility of the sampler, combined with constrained Gibbs sampling steps, which can only modify a component of the parameters, to accelerate the mixing of the chain. As in the full rank case, the target distribution will be the posterior over parameters and the state trajectory $\pi(F, Q, x_{1:T})$. This means that each of the Metropolis-Hastings steps targeting the marginal posterior $\pi(F, Q)$ must be followed by a draw from the state conditional $\pi(x_{1:T}|F, Q)$ in order to complete the iteration. The two steps together comprise a collapsed Gibbs move [29]. Note that the forward-filtering-backward-sampling procedure can be used without modification to draw from the state conditional. See Appendix A.

A. Covariance Decompositions

In formulating an MCMC algorithm for degenerate transition models, we will make use of two factorizations of the transition covariance matrix. First, an eigendecomposition,

$$Q = V \Lambda V^T, \tag{21}$$

in which Λ is a diagonal $r \times r$ matrix with $r = \mathrm{Rank}(Q)$, and V is a $d_x \times r$ matrix of orthonormal columns from the appropriate Stiefel manifold. This is the "non-singular" part of the factorization. In order to resolve ambiguities in this representation, we follow [30] and require that the first element in each column of V should be positive, and that eigenvalues in Λ be arranged in decreasing order of magnitude. The transformation from Q to $\{\Lambda, V\}$ is then bijective.

Note that we can find a set of $d_x - r$ additional vectors orthonormal to V using the Gram-Schmidt procedure. We denote the matrix of these vectors $V_\perp$. This matrix is not unique in general.

A second decomposition will also be useful. Any positive semi-definite can be factorized in the following way,

$$Q = U D U^T, \tag{22}$$

where D is an $r \times r$ positive definite matrix and U is a $d_x \times r$ matrix of orthonormal vectors. In general, this factorisation is not unique. However, here we employ a particular procedure for generating D and U which does yield a unique result, ensuring that the transformation from Q to $\{D, U\}$ is also bijective. This factorisation is constructed using Givens rotations and thus we refer to it as the Givens decomposition. A Givens rotation matrix has the following structure,

$$\left[\Gamma_{i,j}(\gamma) - I\right]_{k,l} = \begin{cases} \cos(\gamma) - 1 & k = l = i \text{ or } k = l = j \\ \sin(\gamma) & k = i, l = j \\ -\sin(\gamma) & k = j, l = i \\ 0 & \text{otherwise,} \end{cases} \tag{23}$$

Algorithm 1: Givens decomposition.

Input: Eigvenvalues and eigenvectors Λ, V of positive semi-definite matrix Q.

Set $U_C \leftarrow I_{d_x \times d_x}$, $U_R \leftarrow I_{r \times r}$, $E \leftarrow V$.
for $i = r, \ldots, 1$ **do**
  for $j = 1, \ldots, i-1$ **do**
    Calculate the rotation $\gamma_{i,j} = \tan^{-1}(E_{i,j}/E_{i,i})$.
    Construct the Givens matrix $\Gamma_{i,j}(\gamma_{i,j})$.
    Set $E \leftarrow E\,\Gamma_{i,j}(\gamma_{i,j})$, $U_R \leftarrow \Gamma_{i,j}(\gamma_{i,j})^T U_R$.
  end for
end for
for $j = 1, \ldots, r$ **do**
  for $i = d_x, \ldots, r+1$ **do**
    Calculate the rotation $\gamma_{i,j} = \tan^{-1}(E_{i,j}/E_{j,j})$.
    Construct the Givens matrix $\Gamma_{i,j}(\gamma_{i,j})$.
    Set $E \leftarrow \Gamma_{i,j}(\gamma_{i,j})\,E$, $U_C \leftarrow U_C \Gamma_{i,j}(\gamma_{i,j})^T$.
  end for
end for
Extract $\tilde{E}$, the first r rows of E, and U, the first r columns of U_C.
Calculate $D = \tilde{E} U_R \Lambda U_R^T \tilde{E}^T$.

Output: U, D.

where $\gamma \in [-\pi/2, \pi/2]$ is a rotation in the plane of the i and j coordinate directions. The procedure for the Givens decomposition is shown in Algorithm 1

The Givens decomposition is important because U is uniquely determined by the state trajectory, whereas D is free to vary. The computational complexity of factorizing the eigenvector matrix is $\mathcal{O}(r \times d_x)$. (Note that multiplication by a Givens matrix is $\mathcal{O}(1)$ due to sparsity.) Details and further discussion is included in Appendix B.

B. Degenerate Transition Priors

The matrix normal-inverse Wishart conjugate prior will need some modifications in order to apply it to degenerate models.

An appropriate prior distribution for the singular transition covariance matrix may be defined by setting the degrees of freedom of an inverse Wishart distribution equal to the desired rank, $\nu_0 = r$. This results in a singular inverse Wishart distribution over the rank-r positive semi-definite matrices [31],

$$Q|r \sim \mathcal{I}W(r, \Psi_0). \tag{24}$$

The density (with respect to the Lesbegue measure on the distinct elements) is,

$$p(Q|r) = \frac{|\Psi_0|^{\frac{r}{2}}}{2^{\frac{1}{2}rd_x}\pi^{\frac{1}{2}r(d_x-r)}\Gamma_r\left(\frac{r}{2}\right)} \times |\Lambda|^{-\frac{1}{2}(3d_x - r + 1)} \exp\left(-\frac{1}{2}\mathrm{Tr}\left[Q^+ \Psi_0\right]\right), \tag{25}$$

in which Q^+ denotes the Moore-Penrose pseudoinverse of Q.

The matrix normal component of the prior is more problematic. Although the distribution is still well-defined, the fact that Q is singular means that the space of possible values is restricted to a subspace of $\mathbb{R}^{d_x \times d_x}$, placing an unwanted constraint on our

model. (See [31] for a discussion of the singular matrix normal distribution.)

In order to relax this unwanted constraint, the following non-singular prior distribution may be used,

$$F|Q \sim \mathcal{MN}\left(M_0, Q + V_\perp \Lambda_\perp V_\perp^T, \Omega_0\right) \tag{26}$$

in which $\Lambda_\perp$ is a diagonal matrix of positive eigenvalues. The matrix $Q + V_\perp \Lambda_\perp V_\perp^T$ is then guaranteed to be positive definite.

Using this prior, we need a mechanism for choosing $\Lambda_\perp$. These extra eigenvalues control the rate at which probability decays as the matrix moves away from the subspace defined by Q. Since $V_\perp$ can be arbitrarily rotated, it will generally be most suitable to use $\Lambda_\perp = \alpha I$ where α is a prior estimate of a "typical" eigenvalue. Alternatively, α could be made dependent on Λ; for example it could be set to the average (arithmetic or geometric) of the current eigenvalues. In practice, uninformative priors will often be used. If $\Omega_0 \to \infty$, then the effect of α becomes negligible.

C. Constrained Gibbs Sampling for Degenerate Transitions

As discussed, standard Gibbs sampling where we alternately sample from the state and parameter conditionals does not allow the Markov chain to fully explore the parameter space because of the constraint that each transition $(x_t - Fx_{t-1})$ must lie in the r-dimensional column space of G. Nevertheless, we can sample from these distributions exactly, and doing so allows the chain to efficiently explore values of the transition matrix and transition covariance within the permitted regions. These constrained Gibbs sampling steps rely on the Givens decomposition (22).

1) Likelihood and Subspace Constraints: There is no transition density over $\mathbb{R}^{d_x}$ for the degenerate model. However, a density does exist with respect to an appropriate measure on the reachable subspace, which provides us with a likelihood function [31]. We can derive this from the underlying state transition equation,

$$\begin{aligned} x_t &= Fx_{t-1} + UD^{\frac{1}{2}}\epsilon_t^x \\ &\Rightarrow U^T\left(x_t - Fx_{t-1}\right) \sim \mathcal{N}\left(0, D\right) \\ p\left(x_t|x_{t-1}, F, Q\right) &\propto |D|^{-\frac{1}{2}} \\ &\quad \times \exp\left(-\frac{1}{2}(x_t - Fx_{t-1})^T UD^{-1}U^T\left(x_t - Fx_{t-1}\right)\right). \end{aligned} \tag{27}$$

This is defined over the subspace,

$$\left\{x_t : x_t = Fx_{t-1} + Uz, z \in \mathbb{R}^r\right\}. \tag{28}$$

The likelihood function for the entire state trajectory is thus,

$$\begin{aligned} p\left(x_{1:T}|F, Q\right) &\propto \prod_{t=2}^{T} p\left(x_t|x_{t-1}, F, Q\right) \\ &\propto |D|^{-\frac{1}{2}(T-1)} \exp\left(-\frac{1}{2}\sum_{t=2}^{T}(x_t - Fx_{t-1})^T U \right. \\ &\quad \left. \times D^{-1}U^T\left(x_t - Fx_{t-1}\right)\right) \\ &= |D|^{-\frac{1}{2}S_0} \exp\left(-\frac{1}{2}\mathrm{Tr}\left[D^{-1}\left(F_U S_1 F_U^T - F_U S_2^T U \right.\right.\right. \\ &\quad \left.\left.\left. - U^T S_2 F_U^T + U^T S_3 U\right)\right]\right) \end{aligned} \tag{29}$$

in which the sufficient statistics are as before and,

$$F_U = U^T F. \tag{30}$$

Writing all the subspace constraints concisely in a single equation,

$$X_{2:T} = FX_{1:T-1} + UZ \tag{31}$$

where $Z \in \mathbb{R}^{r \times (T-1)}$ and

$$X_{2:T} = \left[x_2\ x_3\ \cdots\ x_T\right]. \tag{32}$$

Now introduce,

$$F_\perp = \left(I - UU^T\right)F, \tag{33}$$

such that the transition matrix may be decomposed into orthogonal components,

$$F = UF_U + F_\perp. \tag{34}$$

Using this, the constraint equation becomes,

$$\begin{aligned} rX_{2:T} &= \left(UF_U + F_\perp\right)X_{1:T-1} + UZ \\ r\Rightarrow X_{2:T} &= F_\perp X_{1:T-1} + UZ', \end{aligned}$$

where Z' is a second arbitrary matrix in the same space as Z. Notice that F_U and D do not appear in this equation, and thus may be freely altered by the sampler. In contrast, $F_\perp$ and U are uniquely determined. Sampling from $\pi\left(F, Q|x_{1:T}\right)$ thus entails drawing new values from $\pi\left(F_U, D|F_\perp, U, x_{1:T}\right)$.

2) Transforming the Prior: We will need the distribution on $\{F_U, D\}$ implied by the prior we chose for $\{F, Q\}$. These follow from simple properties of the Wishart and matrix normal distributions (see [30]). For the matrix normal part,

$$F_U|U, D \sim \mathcal{MN}\left(U^T M_0, D, \Omega_0\right). \tag{35}$$

For the inverse Wishart part,

$$D|U \sim \mathcal{IW}\left(r, \left(U^T\Psi_0^{-1}U\right)^{-1}\right). \tag{36}$$

Notice that this Inverse Wishart distribution is no longer singular, since D is $r \times r$.

3) Sampling the Conditional: Using the constrained likelihood function and the transformed prior, we now obtain the required posterior conditional distribution $\pi\left(F_U, D|F_\perp, U, x_{1:T}\right)$. This follows the same algebraic steps as in the full rank case and leads to a similar matrix normal-inverse Wishart distribution,

$$\begin{aligned} D|U, x_{1:T} &\sim \mathcal{IW}\left(\nu, \Psi\right) \\ F_U|D, U, x_{1:T} &\sim \mathcal{MN}\left(M, Q, \Omega\right), \end{aligned} \tag{37}$$

with the following updated hyperparameters,

$$\Omega^{-1} = \Omega_0^{-1} + S_1 \tag{38}$$

$$M\Omega^{-1} = U^T \left(M_0\Omega_0^{-1} + S_2\right) \tag{39}$$

$$\nu = r + S_0 \tag{40}$$

$$\Psi = \left(U^T \Psi_0^{-1} U\right)^{-1} + U^T \left(S_3 + M_0\Omega_0^{-1}M_0{}^T\right) U - M\Omega^{-1}M^T \tag{41}$$

Once new values of F_U and D have been sampled, the corresponding values of F and Q may be calculated using the existing $F_\perp$ and U.

D. Metropolis-Hastings for the Posterior Marginal

In this section we introduce two Metropolis-Hastings kernels which target the marginal posterior $\pi(F, Q)$. Sampling from each of these should be followed by a draw of the state trajectory from $\pi(x_{1:T}|F, Q)$. Together, this constitutes a collapsed Gibbs move for the target joint posterior distribution [29]. These moves are needed in order to ensure that the sampler can attain any possible values in the parameter space.

1) Covariance Random Walk: The covariance matrix is specified by its eigenvalues and eigenvectors. The sampler is already able to explore the possible eigenvalues using the constrained Gibbs moves, but the eigenvector matrix is partially fixed during this process. We can allow the sampler to reach any possible eigenvector matrix using a rotational random walk. Suppose the current value of the transition covariance is Q', a new value is proposed using the following procedure:

1) Sample an orthogonal matrix Ξ from some distribution ς.
2) Set $Q^* = \Xi Q' \Xi^T$.

The transformation is invertible and has a Jacobian of 1. If in addition we require that $\varsigma(\Xi) = \varsigma\left(\Xi^T\right)$, then the proposal distribution is symmetric and the acceptance probability is simply,

$$\alpha(Q' \to Q^*) = \min\left\{1, \frac{\pi(F, Q^*)}{\pi(F, Q')}\right\} = \min\left\{1, \frac{p(y_{1:T}|F, Q^*)}{p(y_{1:T}|F, Q')} \times \frac{p(F, Q^*)}{p(F, Q')}\right\}. \tag{42}$$

There are numerous ways to sample the rotation matrix Ξ from a suitable proposal distribution ς. For example, we could use the Cayley transform [32], a bijective mapping from the skew-symmetric matrices to the rotation matrices, defined by,

$$P(S) = (I - S)^{-1}(I + S). \tag{43}$$

To sample from ς, we draw $\frac{1}{2}d_x(d_x - 1)$ independent scalar random variables $\{s_{i,j}\}_{0<i<j<d_x}$ from any zero-mean distribution; a suitable choice is,

$$s_{k,l} \sim \mathcal{N}\left(0, \sigma_Q^2\right). \tag{44}$$

Use these to construct a skew-symmetric matrix S,

$$S_{k,l} = \begin{cases} s_{k,l} & k < l \\ -s_{l,k} & k > l \\ 0 & k = l, \end{cases} \tag{45}$$

and then set $\Xi = P(S)$. The Cayley transform has the property that $P(-S) = P(S)^{-1} = P(S)^T$, which implies that $\varsigma(\Xi) = \varsigma\left(\Xi^T\right)$ as required. An alternative is to use Givens rotations. First sample $i \in [1, d_x], j \in [1, d_x] \setminus i$, and $\gamma \in [-\pi/2, \pi/2]$ from some zero-mean distribution. Form the Givens matrix $\Gamma_{i,j}(\gamma)$ according to (23) and use $\Xi = \Gamma_{i,j}(\gamma)$. This also has the property that $\Gamma_{i,j}(-\gamma) = \Gamma_{i,j}(\gamma)^T$, implying a symmetric proposal.

2) Transition Matrix Random Walk: A more conventional Metropolis-Hastings kernel can be used for the transition matrix; for example, using a simple Gaussian random walk as the proposal distribution. If the current value is F', a new matrix is sampled from a matrix normal distribution,

$$F^* \sim \mathcal{MN}\left(F', \sigma_F^2 I, I\right), \tag{46}$$

and accepted with probability,

$$\alpha(F' \to F^*) = \min\left\{1, \frac{\pi(F^*, Q)}{\pi(F', Q)}\right\} = \min\left\{1, \frac{p(y_{1:T}|F^*, Q)}{p(y_{1:T}|F', Q)} \times \frac{p(F^*, Q)}{p(F', Q)}\right\}. \tag{47}$$

IV. Unknown Covariance Matrix Rank

Since it is unlikely that the rank of Q will be known a priori, a mechanism is needed which will allow the sampler to learn this parameters as well. Changing the rank of Q changes the number of independent elements, and therefore the dimension of the target probability distribution. Thus, we will need to use reversible jump MCMC for this task [24], [25].

A. Priors With Unknown Rank

We must take care in setting Ψ_0 in $p(Q)$. In general, the scale of an inverse Wishart random variable depends on both the scale matrix and the degrees of freedom. If the same Ψ_0 were used for all r then the prior would favour different scales with different possible values of the rank. In order to make the prior scale invariant across r, we replace Ψ_0 with $\Psi_0^{[r]} = r\Psi_0^{[1]}$. $\Psi_0^{[1]}$ sets the scale of Q globally.

We also introduce a prior distribution for the rank of Q,

$$p(r) = \sum_{i=1}^{d_x} w_i \delta_i(r), \tag{48}$$

where $\delta_i(j)$ is a probability point mass at $j = i$, and $\sum_i w_i = 1$. The target distribution is now,

$$\pi(F, Q, r) \propto p(y_{1:T}|F, Q)\, p(F, Q|r)\, p(r). \tag{49}$$

B. Reversible Jump for Rank Learning

A reversible jump sampler is an MCMC algorithm on a family of probability distributions on spaces of different dimensions. Sampling operations are designed in pairs which allow the sampler to move between the different distributions. Each move begins by sampling additional variables (if required) and then applying a reversible transformation. Finally the newly proposed

state is accepted or rejected according to a Metropolis-Hastings acceptance probability.

For this problem, there is a distribution for each possible value of the rank. Two moves are required, to increase and decrease the rank. The moves suggested here are by no means unique, but are intuitive, simple to implement, and have proven effective in simulations.

The reversible transformation can be accomplished in terms of eigenvectors and eigenvalues. Suppose the current value of the transition covariance is $Q = V\Lambda V^T$, which has rank r. To increase the rank, first we sample an additional eigenvalue $\lambda_{r+1} \in [0, \lambda_r]$ (recall that the elements of Λ are sorted in decreasing order of magnitude, so $\lambda_{r+1} < \lambda_r < \cdots < \lambda_1$) and a corresponding eigenvector $v_{r+1} \in \mathcal{V}_{r+1}$, from the set of vectors which could legitimately be appended to V (i.e., orthonormal to the existing eigenvectors with positive sign of the first element),

$$\mathcal{V}_{r+1} = \left\{ v : |v| = 1,\ V^T v = 0,\ \mathrm{Sign}\left(v_{r+1,1}\right) = +1 \right\}. \quad (50)$$

A new covariance matrix is then generated using the transformation,

$$Q^* = Q + \lambda_{r+1} v_{r+1} v_{r+1}^T \quad (51)$$

$$= \begin{bmatrix} V & v_{r+1} \end{bmatrix} \begin{bmatrix} \Lambda & 0 \\ 0 & \lambda_{r+1} \end{bmatrix} \begin{bmatrix} V^T \\ v_{r+1}^T \end{bmatrix} \quad (52)$$

The inverse transformation, from Q^* to $\{Q, \lambda_{r+1}, v_{r+1}\}$ simply requires identifying and removing the smallest eigenvalue and its corresponding eigenvector. This is an effective choice because the component with the smallest eigenvalue has the least effect on the likelihood of the observed data, and will thus have the highest probability of being accepted.

For the sake of simplicity we suggest sampling the new eigenvalue and eigenvector independently from uniform distributions. For the eigenvalue,

$$q_\lambda\left(\lambda_{r+1}|Q\right) = \begin{cases} \frac{1}{\lambda_r} & 0 < \lambda_{r+1} < \lambda_r \\ 0 & \text{otherwise.} \end{cases} \quad (53)$$

For the eigenvector,

$$q_v\left(v_{r+1}|Q\right) = \begin{cases} \frac{1}{C_{r+1}} & v_{r+1} \in \mathcal{V}_{r+1} \\ 0 & \text{otherwise.} \end{cases} \quad (54)$$

C_{r+1} is the volume of $\mathcal{V}_{r+1}$, which is half of a $(d - r - 1)$-sphere (half because of the requirement that the sign of the first element be positive) [30]. Hence,

$$C_{r+1} = \frac{\pi^{\frac{1}{2}(d_x - r)}}{\Gamma\left(\frac{d_x - r}{2}\right)}. \quad (55)$$

In order to generate such a vector, first draw a $(d_x - r)$-dimensional sample from $\mathcal{N}(0, I)$, normalize it, and then multiply by $V_\perp$.

The acceptance probability for this move and its reverse respectively are [24],

$$\alpha\left(Q, \lambda_{r+1}, v_{r+1} \to Q^*\right) = \min\left\{1, \beta\right\} \quad (56)$$

$$\alpha\left(Q^* \to Q, \lambda_{r+1}, v_{r+1}\right) = \min\left\{1, \beta^{-1}\right\} \quad (57)$$

$$\beta = \frac{\pi\left(F, Q^*, r+1\right) J\left(Q, \lambda_{r+1}, v_{r+1} \to Q^*\right)}{\pi\left(F, Q, r\right) q_\lambda\left(\lambda_{r+1}|Q\right) q_v\left(v_{r+1}|Q\right)} \times \frac{P\left(r+1 \to r\right)}{P\left(r \to r+1\right)}, \quad (58)$$

where $J\left(Q, \lambda_{r+1}, v_{r+1} \to Q^*\right)$ is the Jacobian of the transformation and $P\left(r \to r+1\right)$ is the chosen probability at each step of performing a move which changes the rank from r to $r+1$.

The Jacobian of the eigendecomposition for a positive semi-definite matrix is [31],

$$J\left(\Lambda, V \to Q\right) = 2^{-r} \times \prod_{i=1}^{r} \lambda_i^{d_x - r} \times \prod_{j=1}^{r} \prod_{i=1}^{j-1} \left(\lambda_i - \lambda_j\right). \quad (59)$$

Repeating this for Q^* and taking the ratio of the two, we obtain,

$$J\left(Q, \lambda_{r+1}, v_{r+1} \to Q^*\right) = 2^{-1} \lambda_{r+1}^{d_x - r - 1} \prod_{i=1}^{r} \lambda_i^{-1} \prod_{i=1}^{r} \left(\lambda_i - \lambda_{r+1}\right). \quad (60)$$

V. The Complete Algorithm

There are many ways in which the various Markov kernels could be combined to produce a complete MCMC sampler. Our suggested scheme is shown in Algorithm 2. A Markov chain is generated by repeatedly sampling according to this procedure and storing the values returned each iteration. Steps 2 to 4 constitute the collapsed Gibbs move, and steps 5 to 8 comprise a draw from the constrained parameter conditional. Each iteration requires two passes of the Kalman filter, in order to calculate the marginal likelihoods $p\left(y_{1:T}|F'', Q''\right)$ for the Metropolis-Hastings acceptance probability.

We have so far assumed that R is fixed and known a priori. In practice, R will often be unknown. We can learn it within the MCMC scheme by targeting the posterior distribution $\pi\left(F, Q, R, x_{1:T}\right)$. An additional Gibbs move is introduced to draw from $\pi\left(R|F, Q, x_{1:T}\right)$, which can be sampled exactly if we use a conjugate inverse Wishart prior.

An alternative, which we employ in the following simulations, is to assume the following structure,

$$R = \xi_y I, \quad (61)$$

in which $\xi_y \in \mathbb{R}^+$ is an unknown scalar variance. Sampling from the appropriate conditional distribution $\pi\left(\xi_y|F, Q, x_{1:T}\right)$ can also be achieved exactly, this time using an inverse gamma prior.

VI. Simulations

In this section we seek to demonstrate the efficacy of MCMC for learning degenerate linear state space models. The algorithm is applied to two toy models with simulated data in order

Algorithm 2: MCMC kernel for degenerate linear state space models.

Input: $F', Q', x'_{1:T}$.

1: Randomly select one of the Metropolis-Hastings kernels targeting $\pi(F, Q)$, detailed in Sections III.D2, III.D1, and IV.B.
2: Propose new parameter values F^*, Q^*.
3: With the appropriate acceptance probability, set $F' \leftarrow F^*$ and $Q' \leftarrow Q^*$.
4: Sample $x'_{1:T} \sim \pi(x_{1:T}|F', Q')$.
5: Calculate D', U', the Givens decomposition of Q', using Algorithm 1.
6: Calculate $F'_U, F'_\perp$, the orthogonal components of F', using (30) & (33).
7: Sample $F''_U, D'' \sim \pi(F_U, D|F'_\perp, U', x'_{1:T})$, the constrained parameter conditional given in (37).
8: Calculate F'' and Q'' from $F''_U, F'_\perp, D'', U'$ using (22) & (34).

Output: $F'', Q'', x''_{1:T}$.

to demonstrate that the sampler converges and generates a plausible posterior distribution, and also that the results are more accurate than those achieved by simply approximating the covariance matrix as full rank and using a standard Gibbs sampler. Then we use the new algorithm to learn a degenerate model for interpolation in a motion capture application, and demonstrate improved accuracy over competing methods.

All the simulations use the random walk Metropolis-Hastings kernels to alter F and Q, using the Cayley transformation method to propose changes to Q. During each iteration, one of the three kernels is chosen, each with probability $\frac{1}{3}$. For the reversible jump moves, the probability of either adding or removing an eigenvalue is $\frac{1}{2}$. Each of the random walk kernels has a single scalar algorithm parameter which determines the size of the proposed moves, σ_Q^2 in (44) and σ_F^2 in (46). In order to accelerate mixing, and reduce the need for algorithm tuning, these are adjusted adaptively based on the acceptance rate using the method introduced by [33].

When learning a degenerate transition model, the Metropolis-Hastings moves are less efficient at exploring the parameter space than the Gibbs moves. In order to reduce the length of the required burn in, we initialise the sampler by running a small number of iterations (100 in both simulations) with a full rank model. This puts the parameters in broadly the right region of the parameter space before the rank is allowed to reduce and the less efficient Metropolis-Hastings steps are required.

A. Toy Model 1

First we consider a 4-dimensional model, with transition matrix,

$$F = \begin{bmatrix} 0.95 & 0.8 & 0.8 & 0 \\ 0 & 0.95 & -0.5 & 0.1 \\ 0 & 0 & 1.6 & -0.8 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

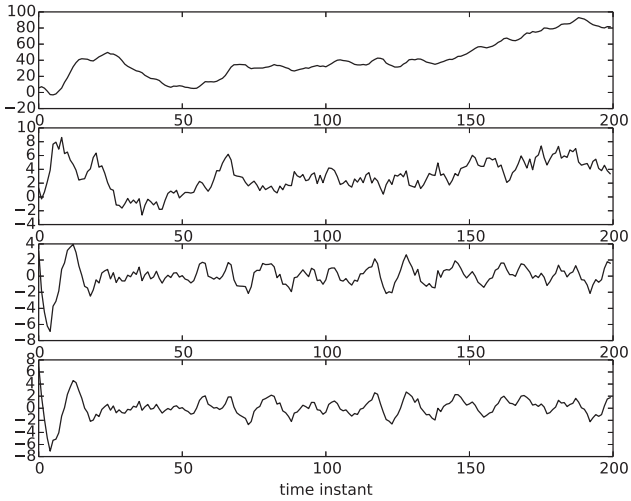


Fig. 1. An example data set simulated from toy model 1, showing evolution of the four state components over time.

The transfer function associated with F has two real and two complex poles, all on or close to the unit circle. The transition covariance matrix is,

$$Q = \begin{bmatrix} \frac{1}{2} & \frac{1}{\sqrt{2}} \\ \frac{1}{2} & -\frac{1}{\sqrt{2}} \\ \frac{1}{2} & 0 \\ \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} \frac{3}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 & 0 \end{bmatrix},$$

which has a rank of 2. For the observation model, $H = I$ and $R = 0.03\,I$.

A data set with $T = 200$ time instants simulated from the model is shown in Fig. 1.

MCMC was used to learn models with both full rank and singular covariance matrices, using 10000 iterations, of which 5000 are discarded as burn in.

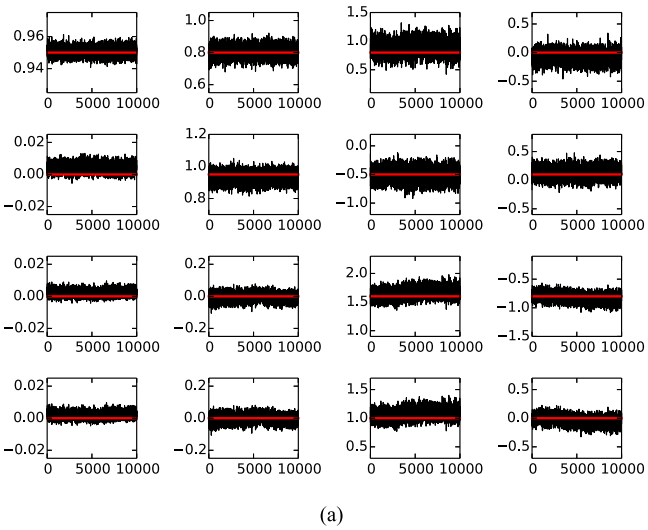
The hyperparameters were set to generally uninformative values. For the full rank model,

$$\begin{aligned} \nu_0 &= d_x - 1 \quad \Psi_0 = \nu_0 I \\ M_0 &= 0 \qquad \Omega_0 = 100\,I. \end{aligned} \tag{62}$$

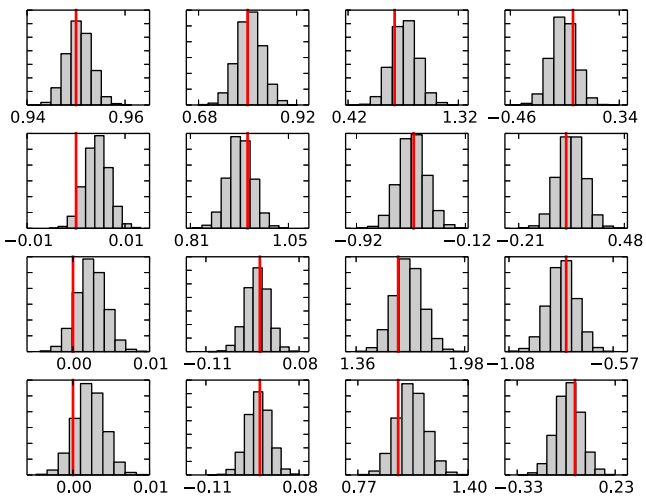
Similarly, for the degenerate model,

$$\begin{aligned} \Psi_0^{[1]} &= I \qquad M_0 = 0 \\ \Omega_0 &= 100I \qquad \alpha = 1 \\ w_i &= \frac{1}{d_x}. \end{aligned} \tag{63}$$

Trace plots and posterior histograms for F and Q learned using the degenerate model are shown in Figs. 2 and 3. These seem to demonstrate that the Markov chain mixes reasonably well, taking a few thousand iterations to converge. Furthermore, the resulting approximations of the posterior distribution are consistent with the true values.

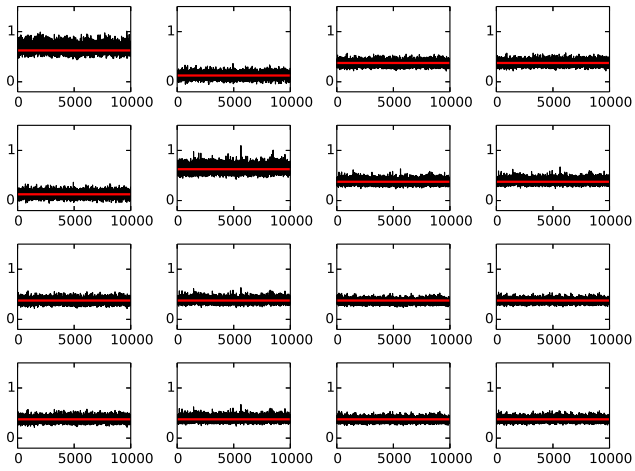


(a)

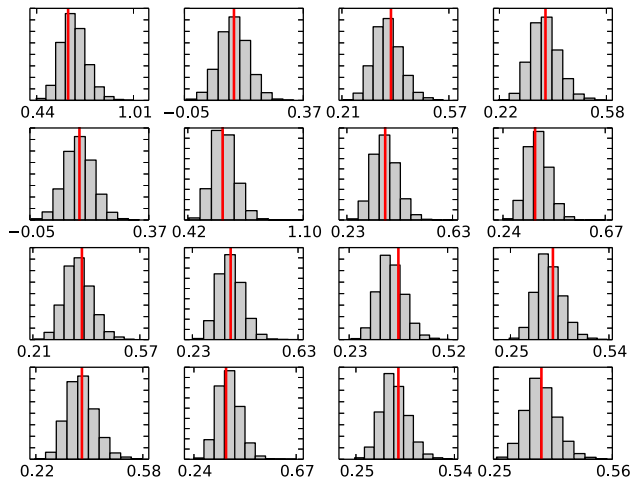


(b)

Fig. 2. Toy Model 1: MCMC (a) trace plot and (b) histogram for the components of F, for the degenerate transition model.



(a)



(b)

Fig. 3. Toy Model 1: MCMC (a) trace plot and (b) histogram for the components of Q, for the degenerate transition model.

Next we compare the results from full rank and degenerate models. The state trajectory is estimated more accurately using the degenerate model, with an RMSE of 3.15, compared to 3.81 for the full rank model. However, the Markov chain for the degenerate model mixes more slowly, as illustrated by the autocorrelation plots in Fig. 4. It also takes longer to execute, with our python implementation taking 0.18 s per iteration compared to 0.12 s for the full rank model.

The algorithm correctly identifies the rank of the covariance matrix. However, its behavior may be cause for concern. The sampler settles on the true value within a few tens of iterations and then never changes. With a discrete variable such as rank this may simply indicate that the posterior distribution highly concentrated on this value. However, it may also be a symptom of poor mixing for this variable.

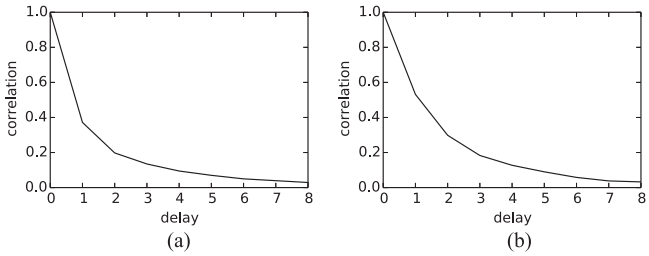


Fig. 4. Toy Model 1: MCMC autocorrelation plots for ξ_y for (a) the full rank transition model and (b) the degenerate transition model.

B. Toy Model 2

We consider a second toy example in which the covariance matrix rank is less precisely determined by the observations in order to investigate the ability of the sampler to explore the posterior distribution of this variable. The latent state is 6-dimensional with $F = 0.99I$, and a covariance matrix of rank

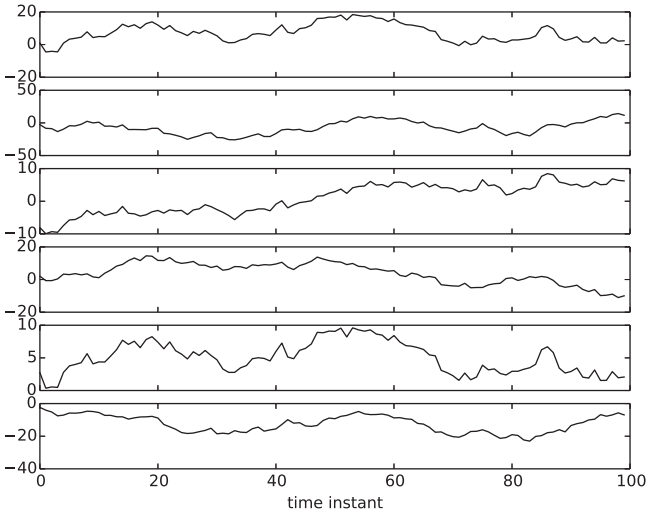


Fig. 5. An example data set simulated from toy model 2, showing evolution of the six state components over time.

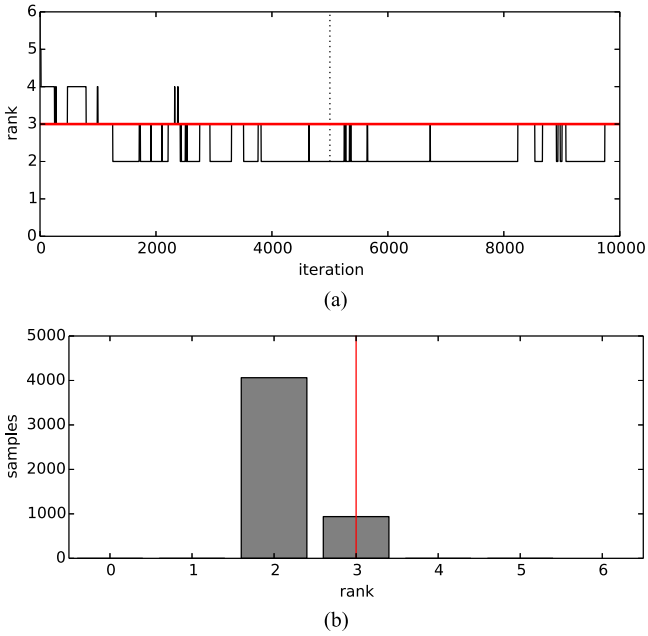


Fig. 6. Toy Model 2: MCMC (a) trace plot and (b) histogram for $\mathrm{Rank}\,(Q)$ with the degenerate transition model.

3 sampled from the prior,

$$Q = AA^T$$

$$A = \begin{bmatrix} 1.76 & 0.40 & 0.98 \\ 2.24 & 1.87 & -0.98 \\ 0.95 & -0.15 & -0.10 \\ 0.41 & 0.14 & 1.45 \\ 0.76 & 0.12 & 0.44 \\ 0.33 & 1.49 & -0.21 \end{bmatrix}. \tag{64}$$

For the observation model, $H = I$ and $R = \xi_y I$, with $\xi_y = 1$.

A data set with $T = 100$ time instants simulated from the model is shown in Fig. 5.

The MCMC algorithm was used to learn a degenerate model. All algorithm parameters were the same as the previous example. The trace plot for the covariance matrix rank and the posterior histogram are shown in Fig. 6. The posterior approximation places significant weight on the true value. Reducing the value of ξ_y leads to an increased posterior weight on the correct value of the rank. As noise variance is increased, so does the uncertainty in the rank, because the noise masks the effect of the transition covariance corresponding to the smaller eigenvalue components.

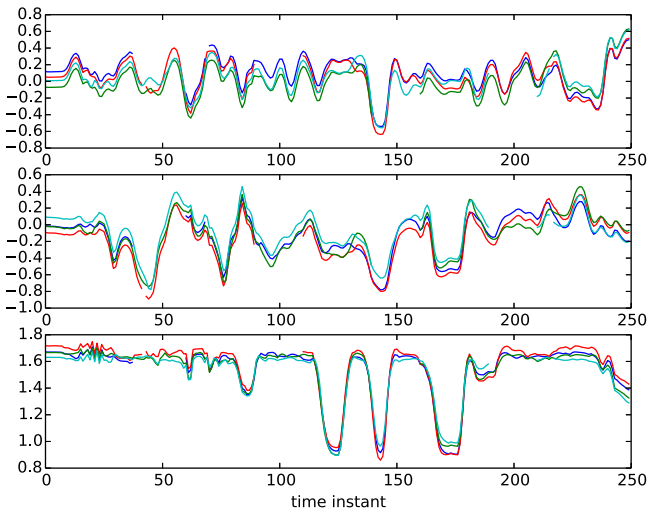


Fig. 7. Motion capture data. Showing the three position coordinates over time for each of the four head markers.

C. Motion Capture Interpolation

In a practical example, the MCMC algorithms are applied to the problem of marker interpolation in a motion capture application. The data used is from [34]. A motion capture system was used to measure the 3-dimensional coordinates of a number of markers attached to a subject's body. Here we consider the four markers attached to the head. Since these are all attached to the same rigid body, we expect there to be substantial correlation in their motion. Sometimes markers are occluded meaning that position measurements are missing. If a system model for the motion of the markers can be learned, then the missing marker locations can be inferred by sampling or Kalman smoothing.

The original data is down-sampled to 5 Hz, and consists of 250 time instants with occasional measurements missing due to occlusion. Observations from one marker at a time are discarded in 4 sections each of 20 time instants (i.e., 4 s each). The remaining observations are used to learn various system models with MCMC, and performance is assessed by comparing the root-mean-square error (RMSE) of the inferred positions in the missing sections. The data set is shown in Fig. 7.

We assume that the latent state consists of 3-dimensional position and velocity for each marker in Cartesian coordinates,

$$x_t = \begin{bmatrix} \mathbf{r}_t \\ \mathbf{v}_t \end{bmatrix}.$$

TABLE I
ROOT-MEAN-SQUARE ERROR (RMSE) FOR ESTIMATED MISSING MARKER POSITIONS, IN METRES

Model	RMSE
NCVM	4.692
Basic	0.154
Degenerate	0.086
MSVD	0.926

The observation model is defined by the following matrices,

$$H = \begin{bmatrix} I & 0 \end{bmatrix} \tag{65}$$

$$R = \xi_y I. \tag{66}$$

Three transition models are trained using MCMC:

1) A near constant velocity motion (NCVM) model [35] in which the motion of each marker is assumed independent from the rest.
2) A standard linear model, with full rank Q.
3) A degenerate linear model, with Q singular and of unknown rank.

The near constant velocity model has the following system matrices,

$$F = \begin{bmatrix} I & I \\ 0 & I \end{bmatrix} \tag{67}$$

$$Q = \xi_x \begin{bmatrix} \frac{1}{3}I & \frac{1}{2}I \\ \frac{1}{2}I & I \end{bmatrix}. \tag{68}$$

ξ_x is an unknown which may be learned within the MCMC scheme using Gibbs moves.

The hyperparameters are set to the same uninformative values as for the toy model, with the following exceptions,

$$\begin{aligned} \Psi_0 &= 0.001\nu_0 I \\ \Psi_0^{[1]} &= 0.001 I \\ \alpha &= 0.01. \end{aligned} \tag{69}$$

Each MCMC algorithm is run for 20000 iterations, of which the first 10000 are discarded as burn-in. The sampler for the degenerate model displays very similar behavior as with the toy model. In the first few hundred iterations the parameters alter rapidly, and the covariance rank is reduced to a value of 9, where it remains thereafter. The Metropolis-Hastings moves continue to alter the system matrices until the chain appears to converge after about 5000 iterations.

For each model, an estimate of the position of the missing markers is made by taking the mean of the sampled state trajectories from the Markov chain. The resulting RMSE is then calculated. For comparison, a missing value singular value decomposition (MSVD) algorithm is also tested [16], [36], [37], in which we use components sufficient to capture 95% of the energy in each iteration. The RMSE results are shown in Table I.

VII. DISCUSSION AND CONCLUSIONS

Markov chain Monte Carlo algorithms are an effective method for Bayesian learning of linear state space systems. In this paper, we have introduced new MCMC kernels for learning linear state space models which may have a singular transition covariance matrix of unknown rank. Such degenerate models may provide a better mathematical description of some systems, when the number of independent components in the state transitions is lower than the number of state dimensions. When a degenerate model is appropriate, the state estimates it produces can be more accurate than those obtained from an equivalent full rank model, as we have shown in simulations. Furthermore, with the rank of the transition covariance matrix included in the model, the sampler provides an explicit estimate of the posterior distribution over this parameter.

The price for using a degenerate transition model is an increased computational cost of the MCMC simulation. The computational complexity is the same as the full rank algorithm, dominated by the Kalman filtering and backward sampling operations which are $\mathcal{O}\left(d_x^3 T\right)$. However, the need for additional Metropolis-Hastings steps increases the cost per iteration. Moreover, the Markov chain mixes more slowly and thus requires a longer chain to be simulated to achieve the same effective sample size. An important avenue for future research will be designing more sophisticated Metropolis-Hastings moves to explore the parameter space more efficiently than the simple random walks used here.

The standard linear Gaussian state space model, with $\mathrm{Rank}\,(Q) = d_x$, discussed in Section II is a special case of the more general class with $\mathrm{Rank}\,(Q)$ unknown. Correspondingly, the MCMC algorithm for the general case reduces to the standard Gibbs sampler of Section II if we re-impose the constraint that $\mathrm{Rank}\,(Q) = d_x$. In this case $D = Q, U = I$, the "constrained" Gibbs moves are no longer constrained, and the Metropolis-Hastings moves become redundant.

APPENDIX A
SAMPLING THE STATE TRAJECTORY

The Kalman filter recursively calculates the state predictive and filtering distributions,

$$p\left(x_t | y_{1:t-1}, F, Q\right) = \mathcal{N}\left(x_t | \hat{m}_t, \hat{P}_t\right) \tag{70}$$

$$p\left(x_t | y_{1:t}, F, Q\right) = \mathcal{N}\left(x_t | m_t, P_t\right). \tag{71}$$

The recursion equations for the means and variances are well known. See e.g., [11].

The forward-filtering-backward-sampling procedure draws a sample from $p(x_{1:T} | y_{1:T}, F, Q)$. This is achieved by first sampling from the final filtering distribution $p(x_T | y_{1:T})$ and then for $t = T-1, \ldots, 1$ sampling from $p\left(x_t | x_{t+1}, y_{1:t}, F, Q\right)$. Each of these factors is also a Gaussian,

$$\begin{aligned} & p\left(x_t | x_{t+1}, y_{1:t}, F, Q\right) \\ & \propto p\left(x_{t+1} | x_t, F, Q\right) p\left(x_t | y_{1:t}, F, Q\right) \\ &= \mathcal{N}\left(x_t | m_t + K_t\left(x_{t+1} - F m_t\right), \hat{P}_t - K_t F P_t\right), \end{aligned} \tag{72}$$

where $K_t = P_t F^T \left(F P_t F^T + Q\right)^{-1}$. The proof is straightforward using standard identities for the Gaussian distribution [11].

This procedure is still valid when Q is singular, provided the matrix $\left(F P_t F^T + Q\right)$ is invertible. A sufficient condition for this is that both the initial covariance P_1 and the transition matrix F are both non-singular. With a matrix normal prior on F, the probability of it being singular is almost surely 0.

To derive the expression for $p\left(x_t | x_{t+1}, y_{1:t}, F, Q\right)$ in the case where the transition covariance is singular, replace Q with $Q + hI$ where I is the identity matrix and h a non-negative real scalar such that a transition density can be written down. Then the same algebraic steps can be followed as in the non-singular case. At the end, set $h = 0$ and obtain the same formula.

Appendix B
A Matrix Factorization for Positive Semi-Definite Matrices

A Givens rotation matrix has the following structure,

$$\left[\Gamma_{i,j}\left(\gamma\right) - I\right]_{k,l} = \begin{cases} \cos\left(\gamma\right) - 1 & k = l = i \text{ or } k = l = j \\ \sin\left(\gamma\right) & k = i, l = j \\ -\sin\left(\gamma\right) & k = j, l = i \\ 0 & \text{otherwise,} \end{cases} \tag{73}$$

where $\gamma \in [-\pi/2, \pi/2]$ is a rotation in the plane of the i and j coordinate directions. Any orthogonal $d \times d$ matrix may be written as a product of $\frac{1}{2} d\left(d-1\right)$ such rotations in the following way [38],

$$\begin{aligned} V = \tilde{E} &\times \left[\Gamma_{1,2}(\gamma_{1,2})\right] \times \cdots \\ &\times \left[\Gamma_{1,d-1}(\gamma_{1,d-1}) \cdots \Gamma_{d-2,d-1}(\gamma_{d-2,d-1})\right] \\ &\times \left[\Gamma_{1,d}(\gamma_{1,d}) \cdots \Gamma_{d-1,d}(\gamma_{d-1,d})\right], \end{aligned} \tag{74}$$

where $\tilde{E}$ is a diagonal matrix in which the diagonal elements are ± 1. This factorisation may be derived (and also calculated) by repeatedly left-multiplying the original matrix by a (transposed) Givens rotation matrix such that one element is set to 0. The matrix remaining once all the elements below the diagonal have been eliminated is $\tilde{E}$, and the factorisation above results from a straightforward rearrangement. See [38] for details. This factorisation is not unique. Any order of Givens matrices may be used provided each operation does not interfere with the elements zeroed by the preceding operations. Furthermore, we can also zero an element of the matrix by right multiplication by a Givens matrix, so the rotations may be split into two blocks, one each side of the sign matrix.

Now consider instead a $d \times r$ matrix of orthogonal columns. This can be factorized using the same mechanism. First right multiply by Givens matrices to zero the elements on the first r rows below the main diagonal. Then left multiply by Givens matrices to zero all the elements on rows $r+1$ to d. After rearranging, the resulting factorisation is,

$$V = U_C E U_R, \tag{75}$$

where

$$\begin{aligned} U_R = &\left[\Gamma_{1,2}(\gamma_{1,2})\right] \times \cdots \\ &\times \left[\Gamma_{1,r}(\gamma_{1,r}) \ldots \Gamma_{r-1,r}(\gamma_{r-1,r})\right] \end{aligned} \tag{76}$$

$$\begin{aligned} U_C = &\left[\Gamma_{1,r+1}(\gamma_{1,r+1}) \ldots \Gamma_{1,d}(\gamma_{1,d})\right] \times \cdots \\ &\times \left[\Gamma_{r,r+1}(\gamma_{r,r+1}) \ldots \Gamma_{r,d}(\gamma_{r,d})\right], \end{aligned} \tag{77}$$

and E has the following structure,

$$E = \begin{bmatrix} \tilde{E} \\ 0_{(d-r)\times r} \end{bmatrix}.$$

where $\tilde{E}$, as before, is diagonal with elements ± 1.

Finally consider a rank-r positive semi-definite matrix Υ for which the non-singular part of the eigendecomposition is,

$$\Upsilon = V \Lambda V^T. \tag{78}$$

Factorizing the matrix of eigenvectors using the Givens method, we find,

$$\begin{aligned} \Upsilon &= U_C E U_R \Lambda U_R^T E^T U_C^T \\ &= U_C \begin{bmatrix} \tilde{E} U_R \Lambda U_R^T \tilde{E}^T & 0 \\ 0 & 0 \end{bmatrix} U_C^T \\ &= U D U^T, \end{aligned}$$

in which U consists of the first r columns of U_C, and,

$$D = \tilde{E} U_R \Lambda U_R^T \tilde{E}^T. \tag{79}$$

Comparing with (74), we see that $\tilde{E} U_R$ can represent any $r \times r$ orthogonal matrix and hence D is a unique positive definite matrix.

Acknowledgment

The authors thank Michael Burke for advice and assistance with the motion capture application and Rich Wareham for his geometric insights.

References

[1] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation With Applications to Tracking and Navigation: Theory, Algorithms and Software*. New York, NY, USA: Wiley Online Library, 2002.

[2] S. Godsill and P. Rayner, *Digital Audio Restoration*. New York, NY, USA: Springer, 1998.

[3] M. J. Beal, F. Falciani, Z. Ghahramani, C. Rangel, and D. L. Wild, "A Bayesian approach to reconstructing genetic regulatory networks with hidden factors," *Bioinformatics*, vol. 21, no. 3, pp. 349–356, 2005.

[4] R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.*, vol. 82, pp. 35–45, 1960.

[5] H. E. Rauch, F. Tung, and C. T. Striebel, "Maximum likelihood estimates of linear dynamic systems," *AIAA J.*, vol. 3, no. 8, pp. 1445–1450, 1965.

[6] N. Kantas, A. Doucet, S. Singh, and J. Maciejowski, "An overview of sequential Monte Carlo methods for parameter estimation in general state-space models," in *Proc. IFAC Symp. Syst. Identif. (SYSID)*, 2009, pp. 774–785.

[7] C. Andrieu, A. Doucet, and R. Holenstein, "Particle Markov chain Monte Carlo methods," *J. Roy. Statist. Soc. B, Statist. Methodol.*, vol. 72, pp. 269–342, 2010.

[8] P. Van Overschee and B. De Moor, "Subspace algorithms for the stochastic identification problem," in *Proc. 30th IEEE Conf. Decision Control*, 1991, pp. 1321–1326.

[9] M. Viberg, "Subspace-based methods for the identification of linear time-invariant systems," *Automatica (Trends in System Identification)*, vol. 31, no. 12, pp. 1835–1851, 1995.
[10] O. Cappé, E. Moulines, and T. Rydén, *Inference in Hidden Markov Models*. New York, NY, USA: Springer, 2005, vol. 6.
[11] S. Särkkä, *Bayesian Filtering and Smoothing*. Cambridge, U.K.: Cambridge Univ. Press, 2013.
[12] R. H. Shumway and D. S. Stoffer, "An approach to time series smoothing and forecasting using the EM algorithm," *J. Time Series Anal.*, vol. 3, no. 4, pp. 253–264, 1982.
[13] V. Digalakis, J. R. Rohlicek, and M. Ostendorf, "ML estimation of a stochastic linear system with the EM algorithm and its application to speech recognition," *IEEE Trans. Speech Audio Process.*, vol. 1, no. 4, pp. 431–442, Oct. 1993.
[14] Z. Ghahramani and G. E. Hinton, "Parameter estimation for linear dynamical systems," Comput. Sci. Dept., Univ. of Toronto, Toronto, Canada ON, Tech. Rep. Tech. Rep. crg-tr-96-2, 1996.
[15] S. Gibson and B. Ninness, "Robust maximum-likelihood estimation of multivariable dynamic systems," *Automatica*, vol. 41, no. 10, pp. 1667–1682, 2005.
[16] L. Li, J. McCann, N. S. Pollard, and C. Faloutsos, "DynaMMo: Mining and summarization of coevolving sequences with missing values," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min. (KDD'09)*, New York, NY, USA, 2009, pp. 507–516.
[17] M. J. Beal, *Variational algorithms for approximate Bayesian inference*, Ph.D. dissertation, Univ. of London, London, U.K., 2003.
[18] Z. Ghahramani and M. J. Beal, "Propagation algorithms for variational Bayesian learning," *Adv. Neural Inf. Process. Syst.*, pp. 507–513, 2001.
[19] D. Barber and S. Chiappa, "Unified inference for variational Bayesian linear Gaussian state-space models," *Advances in Neural Information Processing Systems 19*. B. Schölkopf, J. C. Platt, and T. Hoffman, Eds., Cambridge, MA, USA: MIT Press, 2007, pp. 81–88.
[20] B. Ninness and S. Henriksen, "Bayesian system identification via Markov chain Monte Carlo techniques," *Automatica*, vol. 46, no. 1, pp. 40–51, 2010.
[21] A. Wills, T. B. Schön, F. Lindsten, and B. Ninness, "Estimation of linear systems using a Gibbs sampler," in *Proc. 16th IFAC Symp. Syst. Identif. (SYSID)*, 2012, pp. 203–208.
[22] S. Maskell, *Sequentially structured Bayesian solutions*, Ph.D. dissertation, Engineering Dept., , Cambridge Univ., Cambridge, U.K., 2004.
[23] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund, "Particle filters for positioning, navigation, and tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 425–437, Feb. 2002.
[24] P. J. Green, "Reversible jump Markov chain Monte Carlo computation and Bayesian model determination," *Biometrika*, vol. 82, no. 4, p. 711, 1995.
[25] P. J. Green and D. I. Hastie, "Reversible jump MCMC," *Genetics*, vol. 155, no. 3, pp. 1391–1403, 2009.
[26] S. Roweis and Z. Ghahramani, "A unifying review of linear Gaussian models," *Neural Comput.*, vol. 11, no. 2, pp. 305–345, Feb. 1999.
[27] S. Chib, "Calculating posterior distributions and modal estimates in Markov mixture models," *J. Econometr.*, vol. 75, no. 1, pp. 79–97, 1996.
[28] G. O. Roberts and A. F. M. Smith, "Simple conditions for the convergence of the Gibbs sampler and Metropolis-Hastings algorithms," *Stoch. Process. Appl.*, vol. 49, no. 2, pp. 207–216, 1994.
[29] D. A. van Dyk and T. Park, "Partially collapsed Gibbs samplers," *J. Amer. Statist. Assoc.*, vol. 103, no. 482, pp. 790–796, 2008.
[30] R. J. Muirhead, *Aspects of Multivariate Statistical Theory*. New York, NY, USA: Wiley, 1982.
[31] J. A. Díaz-García and R. Gutiérrez-Jáimez, "Distribution of the generalised inverse of a random matrix and its applications," *J. Statist. Planning Infer.*, vol. 136, no. 1, pp. 183–192, 2006.
[32] C. A. León, J.-C. Massé, and L.-P. Rivest, "A statistical model for random rotations," *J. Multivar. Anal.*, vol. 97, no. 2, pp. 412–430, 2006.
[33] G. O. Roberts and J. S. Rosenthal, "Examples of adaptive MCMC," *J. Comput. Graph. Statist.*, vol. 18, no. 2, pp. 349–367, 2009.
[34] A. Aristidou, "Marker prediction and skeletal reconstruction in motion capture technology," Univ. of Cyprus, Nicosia, Cyprus, Tech. Rep. Tech. Rep., 2013.
[35] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking. Part I: Dynamic models," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 39, no. 4, pp. 1333–1364, 2003.
[36] N. Srebro and T. Jaakkola, "Weighted low-rank approximations," in *Proc. Int. Conf. on Mach. Learn.*, vol. 3, 2003, pp. 720–727.
[37] G. Liu and L. McMillan, "Estimation of missing markers in human motion capture," *Vis. Comput.*, vol. 22, no. 9–11, pp. 721–728, 2006.
[38] T. W. Anderson, I. Olkin, and L. G. Underhill, "Generation of random orthogonal matrices," *SIAM J. Scientif. Statist. Comput.*, vol. 8, no. 4, pp. 625–629, 1987.

Pete Bunch received a Ph.D. degree from Cambridge University in 2014, and an M.Eng. degree from the same institution in 2010. Subsequently, he has worked as a Research Associate with the Signal Processing and Communications Laboratory in the Cambridge University Engineering Department. His research interests include sequential inference, Monte Carlo methods and machine learning. He recently left academia to see what the rest of the world is up to.

James Murphy completed his Ph.D. from Cambridge University in 2013. He received an undergraduate degree in computer science from Cambridge University and an M.Sc. in mathematical modelling from Oxford University. He is a Research Associate with the Signal Processing Group within the Engineering Department at Cambridge University. His research interests include Monte Carlo methods for inference in univariate and multivariate time series problems.

Simon Godsill is a Professor of Statistical Signal Processing with the Engineering Department, Cambridge University, Cambridge, U.K. He is also a Professorial Fellow and tutor with Corpus Christi College Cambridge.

He coordinates an active research group in Signal Inference and its Applications within the Signal Processing and Communications Laboratory at Cambridge University, specializing in Bayesian computational methodology, multiple object tracking, audio and music processing, and financial time-series modeling. A particular methodological theme over recent years has been the development of novel techniques for optimal Bayesian filtering, using Sequential Monte Carlo or Particle Filtering methods. He has published extensively in journals, books, and international conference proceedings.