

A New Approach to Drawing States in State Space Models

William J. McCausland ^{*}Université de Montréal, CIREQ and CIRANO

Shirley Miller [†]Université de Montréal

Denis Pelletier [‡]North Carolina State University

Current version: June 26, 2007

Abstract

We introduce a new method for drawing state variables in Gaussian state space models from their conditional distribution given parameters and observations. Unlike standard methods, our method does not involve Kalman filtering. We show that for some important cases, our method is computationally more efficient than standard methods in the literature. We consider two applications of our method.

Key words: State space models, Stochastic volatility, Count data

1 Introduction

Consider the following Gaussian linear state space model:

$$y_t = X_t\beta + Z_t\alpha_t + G_tu_t, \quad t = 1, \dots, n, \quad (1)$$

^{*}Mailing address: Département de sciences économiques, C.P. 6128, succursale Centre-ville, Montréal QC H3C 3J7, Canada. e-mail: william.j.mccausland@umontreal.ca. Web site: www.cirano.qc.ca/~mccauslw.

[†]e-mail: shirley.miller.lira@umontreal.ca.

[‡]Mailing address: Department of Economics, Campus Box 8110, North Carolina State University, Raleigh, 27695-8110, USA. e-mail: denis_pelletier@ncsu.edu. Web site: <http://www4.ncsu.edu/~dpellet>.

$$\alpha_{t+1} = W_t\beta + T_t\alpha_t + H_tu_t, \quad t = 1, \dots, n-1, \quad (2)$$

$$\alpha_1 \sim N(a_1, P_1), \quad u_t \sim \text{i.i.d. } N(0, I_q), \quad (3)$$

where y_t is a $p \times 1$ vector of dependent variables, α_t is a $m \times 1$ vector of state variables, and β is a $k \times 1$ vector of coefficients. The matrices X_t, Z_t, G_t, W_t, T_t and H_t are known. Equation (1) is the *measurement* equation and equation (2) is the *state* equation. Let $y \equiv (y'_1, \dots, y'_n)'$ and $\alpha \equiv (\alpha'_1, \dots, \alpha'_n)'$.

Frühwirth-Schnatter (1994) and Carter and Kohn (1994) introduce a method for drawing $\alpha|y$ using a recursive approach, for the case $p = 1$ and $q = m + 1$. They use the output of the Kalman filter to draw the α_t in backwards sequence from the conditional distributions $\alpha_t|\alpha_{t+1}, \dots, \alpha_n, y$. de Jong and Shephard (1995) (DeJS hereafter) introduce a procedure for drawing disturbances given y for the case where $p \geq 1$ and q is not necessarily equal to $m + p$. It is easy to construct a draw of α using the disturbances. The method is also recursive and also uses the output of the Kalman filter. Durbin and Koopman (2002) (DK hereafter) introduce another approach, which, like that of de Jong and Shephard, uses the Kalman filter, involves drawing disturbances rather than states, and allows q to be less than $p + m$. The approach is not recursive, however. Rather, disturbances are drawn from their unconditional distribution and a transformation is applied to construct a conditional draw. This approach is more efficient than that of de Jong and Shephard (1995) for large m . Computations after the application of the Kalman filter are particularly fast - there are no matrix-matrix multiplications (although there are matrix-vector multiplications) and no matrix inversions. Furthermore, draws from multivariate normal distributions are diagonal and do not require a Cholesky decomposition.

In this paper, we introduce a new approach to drawing $\alpha|y$ which does not involve the application of the Kalman filter. We first compute the precision Ω and co-vector¹ c of the normal distribution $\alpha|y$. In most cases, this can be done very quickly because of redundant computations. Then we efficiently compute conditional variances $\Sigma_t \equiv \text{Var}[\alpha_t|\alpha_{t+1}, \dots, \alpha_n, y]$ and matrices that can be used to compute the conditional means $E[\alpha_t|\alpha_{t+1}, \dots, \alpha_n, y]$. Repeated draws of $\alpha|y$ are generated particularly efficiently. We also show that just as we can easily use the output of the Kalman filter to compute the likelihood function, we can do the same using our method.

¹If a Gaussian random vector x has mean μ and variance Σ , we will call Σ^{-1} the precision of x and $\Sigma^{-1}\mu$ the co-vector of x .

We will show that in many cases our method is more computationally efficient than that of Durbin and Koopman (2002). In its current state, however, our method works only for the case $q = p + m$.

In Section 2, we describe our new approach to drawing $\alpha|y$. In Section 3, we compare our method with that of Durbin and Koopman (2002), in terms of numbers of various kinds of operations. In Section 4, we compare these methods in an empirical example. We use them to draw stochastic volatility in a Markov chain developed by Kim, Shephard, and Chib (1998) for posterior simulation in a basic stochastic volatility model. In Section 5, we compare these methods in a second empirical example. Here, they are used in an importance sampling application due to Durbin and Koopman (1997) for approximating the likelihood function in semi-Gaussian state space models. In our application, the measurement distribution is Poisson.

2 A New Approach to Drawing States

Instead of specifying the distribution of the time- t innovation in terms of G_t and H_t , we will give its precision A_t :

$$A_t \equiv \left[\text{Var} \left(\begin{bmatrix} G_t \\ H_t \end{bmatrix} u_t \right) \right]^{-1} = \begin{bmatrix} G_t G_t' & G_t H_t' \\ H_t G_t' & H_t H_t' \end{bmatrix}^{-1},$$

which we partition as

$$A_t = \begin{bmatrix} A_{11,t} & A_{12,t} \\ A_{21,t} & A_{22,t} \end{bmatrix},$$

where $A_{11,t}$ is the leading $p \times p$ submatrix of A_t . We let $A_{22,0} \equiv P_1^{-1}$ be the $m \times m$ precision of α_1 and $A_{11,n} \equiv (G_n G_n')^{-1}$ be the $p \times p$ precision of the time $t = n$ innovation $G_n u_n$.

We recognize that the distribution of residuals is usually more easily specified in terms of G_t and H_t rather than A_t . In most cases, however, the A_t are constant or take on one of a small number of values, and so the additional computation required to obtain the A_t is negligible. In some cases, it may even be more natural to specify the precision directly. Fully Gaussian state space models may be used to facilitate estimation in non-linear or non-Gaussian state space models by providing proposal distributions for MCMC methods or importance distributions for importance sampling applications. In these cases, precisions in the approximating Gaussian model may be obtained from Hessian matrices of the log observation

density of the non-linear or non-Gaussian model. For an illustration, see Section 5.

Clearly α and y are jointly Gaussian and therefore the conditional distribution of α given y is also Gaussian. We show in Appendix A that the precision Ω and co-vector c of this conditional distribution are:

$$\Omega \equiv \begin{bmatrix} \Omega_{11} & \Omega_{12} & 0 & \dots & 0 \\ \Omega_{21} & \Omega_{22} & \Omega_{23} & \ddots & \vdots \\ 0 & \Omega_{32} & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \Omega_{n-1,n-1} & \Omega_{n-1,n} \\ 0 & \dots & 0 & \Omega_{n,n-1} & \Omega_{nn} \end{bmatrix} \quad c \equiv \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}, \quad (4)$$

where

$$\Omega_{tt} \equiv Z'_t A_{11,t} Z_t + Z'_t A_{12,t} T_t + T'_t A_{21,t} Z_t + T'_t A_{22,t} T_t + A_{22,t-1}, \quad t = 1, \dots, n-1,$$

$$\Omega_{nn} \equiv Z'_n A_{11,n} Z_n + A_{22,n-1},$$

$$\Omega_{t+1,t} \equiv -A_{21,t} Z_t - A_{22,t} T_t, \quad t = 1, \dots, n-1,$$

$$\Omega_{t,t+1} \equiv -Z'_t A_{12,t} - T'_t A_{22,t}, \quad t = 1, \dots, n-1,$$

$$c_1 \equiv (Z'_1 A_{11,1} + T'_1 A_{21,1})(y_1 - X_1 \beta) - (Z'_1 A_{12,1} + T'_1 A_{22,1})(W_1 \beta) + A_{22,0}(W_0 \beta + T_0 \alpha_0),$$

$$c_t \equiv (Z'_t A_{11,t} + T'_t A_{21,t})(y_t - X_t \beta) - (Z'_t A_{12,t} + T'_t A_{22,t})(W_t \beta) - A_{21,t-1}(y_{t-1} - X_{t-1} \beta) + A_{22,t-1}(W_{t-1} \beta), \quad t = 2, \dots, n-1,$$

$$c_n \equiv Z'_n A_{11,n}(y_n - X_n \beta) - A_{21,n-1}(y_{n-1} - X_{n-1} \beta) + A_{22,n-1}(W_{n-1} \beta).$$

In general, calculation of the Ω_{tt} and $\Omega_{t,t+1}$ is computationally demanding. However, in many cases of interest, A_t , Z_t and T_t are constant, or take on one of a small number of values. In these cases, the computational burden is a constant, not depending on n . We do need to compute each c_t , but provided that the matrix expressions in parantheses can be pre-computed, this involves matrix-vector multiplications, rather than the matrix-matrix multiplications required for later computations.

As in Frühwirth-Schnatter (1994), Carter and Kohn (1994) and de Jong and Shephard (1995), we draw the α_t in reverse order, each α_t from the distribution $\alpha_t | \alpha_{t+1}, \dots, \alpha_n, y$. Standard formulas for conditional Gaussian distributions give

$$\alpha_{1:t} | \alpha_{t+1:n}, y \sim N(\mu_{1:t} - (\Omega_{1:t,1:t})^{-1} \Omega_{1:t,t+1:n} (\alpha_{t+1:n} - \mu_{t+1:n}), (\Omega_{1:t,1:t})^{-1}), \quad (5)$$

where $\mu \equiv \Omega^{-1}c$ and μ , α and Ω are partitioned as

$$\mu = \begin{bmatrix} \mu_{1:t} \\ \mu_{t+1:n} \end{bmatrix} \quad \alpha = \begin{bmatrix} \alpha_{1:t} \\ \alpha_{t+1:n} \end{bmatrix} \quad \begin{bmatrix} \Omega_{1:t,1:t} & \Omega_{1:t,t+1:n} \\ \Omega_{t+1:n,1:t} & \Omega_{t+1:n,t+1:n} \end{bmatrix},$$

with $\mu_{1:t}$, $\alpha_{1:t}$ and $\Omega_{1:t,1:t}$ having dimensions $tm \times 1$, $tm \times 1$ and $tm \times tm$, respectively.

Naive computation of the conditional mean and variance of α_t given $\alpha_{t+1}, \dots, \alpha_n, y$ is very inefficient. However, we have the following result that permits us to compute these conditional moments in time n .

Result 2.1 *If $\alpha | y \sim N(\Omega^{-1}c, \Omega^{-1})$, then*

$$\begin{aligned} E[\alpha_t | \alpha_{t+1}, \dots, \alpha_n, y] &= m_t - \Sigma_t \Omega_{t,t+1} \alpha_{t+1}, \\ \text{Var}[\alpha_t | \alpha_{t+1}, \dots, \alpha_n, y] &= \Sigma_t, \end{aligned}$$

and

$$E[\alpha | y] = (\mu'_1, \dots, \mu'_n)',$$

where

$$\begin{aligned} \Sigma_1 &= (\Omega_{11})^{-1}, & m_1 &= \Sigma_1 c_1, \\ \Sigma_t &= (\Omega_{tt} - \Omega_{t,t-1} \Sigma_{t-1} \Omega_{t-1,t})^{-1}, & m_t &= \Sigma_t (c_t - \Omega_{t,t-1} m_{t-1}), \\ \mu_n &= m_n, & \mu_t &= m_t - \Sigma_t \Omega_{t,t+1} \mu_{t+1}. \end{aligned}$$

The result is based on a Levinson-like algorithm, introduced by Vandebril, Mastronardi, and Van Barel (2007), for solving the equation $Bx = y$, where B is an $n \times n$ symmetric band diagonal matrix and y is a $n \times 1$ vector. We extend their result in two ways. First, we modify the algorithm to work with $m \times m$ submatrices of a block band diagonal matrix rather than individual elements of a band diagonal matrix. Second, we show that the intermediate computations used to solve the equation $\Omega\mu = c$ for the mean $\mu = E[\alpha | y]$ given the precision $\Omega = (\text{Var}[\alpha | y])^{-1}$ and co-vector $c = (\text{Var}[\alpha | y])^{-1} E[\alpha | y]$ can be used to compute the conditional means $E[\alpha_t | \alpha_{t+1}, \dots, \alpha_n, y]$ and conditional variances $\text{Var}[\alpha_t | \alpha_{t+1}, \dots, \alpha_n, y]$. The proof of the result is in Appendix B.

We can now use the following algorithm to draw α from $\alpha | y$ (MMP method hereafter).

1. Compute $\Sigma_1 = (\Omega_{11})^{-1}$, $m_1 = \Sigma_1 c_1$.

2. For $t = 2, \dots, n$, compute

$$\Sigma_t = (\Omega_{tt} - \Omega_{t,t-1}\Sigma_{t-1}\Omega_{t-1,t})^{-1}, \quad m_t = \Sigma_t(c_t - \Omega_{t,t-1}m_{t-1}).$$

3. Draw $\alpha_n \sim N(m_n, \Sigma_n)$.

4. For $t = n - 1, \dots, 1$, draw

$$\alpha_t \sim N(m_t - \Sigma_t\Omega_{t,t+1}\alpha_{t+1}, \Sigma_t).$$

3 Efficiency Analysis

We compare the computational efficiency of various methods for drawing $\alpha|y$. We consider separately the fixed computational cost that is incurred only once, no matter how many draws are needed, and the marginal computational cost required to make an additional draw. We do this because there are some applications, such as Bayesian analysis of state space models using Gibbs sampling, in which only one draw is needed and other applications, such as importance sampling in non-Gaussian models, where many draws are needed.

3.1 Pre-computation

All previous methods we are aware of for drawing $\alpha|y$ as a block involve the Kalman filter. The computations are as follows:

$$e_t = y_t - [X_t\beta] - Z_t a_t, \quad D_t = Z_t P_t Z_t' + [G_t G_t'],$$

$$K_t = (T_t P_t Z_t' + [H_t G_t']) D_t^{-1}, \quad L_t = T_t - K_t Z_t,$$

$$a_{t+1} = [W_t\beta] + T_t a_t + K_t e_t, \quad P_{t+1} = [T_t P_t] L_t' + [H_t H_t'] + [H_t G_t'] K_t$$

Here and elsewhere, we use braces to denote quantities that do not need to be computed for each observation. These include quantities such as $[T_t P_t]$ above that are computed in previous steps, and quantities such as $[H_t H_t']$ that are usually either constant or taking values in a small pre-computable set.

Table 3.1 lists the matrix-matrix multiplications, Cholesky decompositions, and solutions of triangular systems required for an iteration of the Kalman filter,

and those required for the computation of Σ_t and m_t using our method. We represent the solution of triangular systems using notation for the inverse of a triangular matrix, but no actual matrix inversions are performed. It also gives the number of scalar multiplications for each operation as a function of p and m . Terms of less than third order are omitted. All operation counts, here and elsewhere, are per observation. We ignore matrix-vector multiplications, whose costs are mere second order monomials in m and p rather than third order. We take the computational cost of multiplying an $M_1 \times M_2$ matrix by an $M_2 \times M_3$ matrix as $M_1 M_2 M_3$ scalar floating-point multiplications in general. If the result is symmetric or if one of the matrices is triangular, we divide by two. It is possible to multiply matrices more efficiently than using the direct method, but the dimensions required before realizing savings are higher than those usually encountered in state space models. We take the cost of the Cholesky decomposition of an $M \times M$ matrix as $M^3/6$ scalar multiplications, which is the cost using the algorithm in Press, Teukolsky, Vetterling, and Flannery (1992, p. 97). We take the cost of solving a triangular system of M equations as $M^2/2$ scalar multiplications.

The efficiency of our method relative to that of Durbin and Koopman (2002) depends on various features of the application. In some cases, such as those where all the elements of T_t and Z_t are zero or one, certain matrix multiplications do not require any scalar multiplications. In others, certain matrices are diagonal, reducing the number of multiplications by an order. We see that the MMP method has no third order monomials involving p . The coefficient of the m^3 term is $7/6$, compared with 2 for the Kalman filter if $T_t P_t$ is a general matrix multiplication and 1 if T_t is diagonal or composed of zeros and ones.

The de Jong and Shephard (1995) simulation smoother requires an additional Cholesky decomposition of a $m \times m$ matrix and several additional matrix-matrix multiplications.

3.2 Drawing α

Compared with the fixed cost of pre-processing, the marginal computational cost of an additional draw from $\alpha|y$ is negligible for all three methods we consider. In particular, no matrix-matrix multiplications, matrix inversions, or Cholesky decompositions are required. However, when large numbers of these additional draws are required, this marginal cost becomes important. It is here that our method is clearly more efficient than that of Durbin and Koopman (2002).

Using the modified simulation smoothing algorithm in Section 2.3 of Durbin and Koopman (2002), an additional draw from $\alpha|y$ requires the following compu-

Table 1: Scalar multiplications

Method	Operation	Scalar multiplications
Kalman	$P_t Z_t'$	$m^2 p$
	$Z_t [P_t Z_t']$	$m p^2 / 2$
	$T_t [P_t Z_t']$	$m^2 p$
	$D_t = \Upsilon_t \Upsilon_t'$ (Cholesky)	$p^3 / 6$
	$[T_t P_t Z_t' + H_t G_t'] (\Upsilon_t')^{-1} \Upsilon_t^{-1}$	$m p^2$
	$K_t Z_t$	$m^2 p$
	$T_t P_t$	m^3
	$[T_t P_t] L_t'$	m^3
	$[H_t G_t'] K_t$	$m^2 p$
MMP	$(\Omega_{tt} - \Omega_{t,t-1} \Sigma_{t-1} \Omega_{t-1,t}) = \Lambda_t \Lambda_t'$ (Cholesky)	$m^3 / 6$
	$\Lambda_t^{-1} \Omega_{t,t+1}$	$m^3 / 2$
	$\Omega_{t+1,t} \Sigma_t \Omega_{t,t+1} = [\Lambda_t^{-1} \Omega_{t,t+1}]' [\Lambda_t^{-1} \Omega_{t,t+1}]$	$m^3 / 2$

tations. We define $\epsilon_t \equiv G_t u_t$ and $\eta_t \equiv H_t u_t$, and assume $G_t' H_t = 0$ and $X_t \beta = 0$, recognizing that these assumptions can be easily relaxed. The first step is forward simulation using equations (6) and (7) in that article.

$$x_1 \sim N(0, P_1), \quad v_t^+ = Z_t x_t + \epsilon_t^+ \quad x_{t+1} = T_t x_t - K_t v_t^+ + \eta_t^+,$$

where $\epsilon_t^+ \sim N(0, \Xi_t)$ and $\eta_t^+ \sim N(0, Q_t)$. The next step is the backwards recursion of equation (5):

$$r_n = 0, \quad r_{t-1} = [Z_t D_t^{-1}] v_t^+ + L_t' r_t,$$

and the computation of residuals in equation (4):

$$\hat{\eta}_t^+ = Q_t r_t.$$

A draw $\tilde{\eta}$ from the conditional distribution of η given y is given by

$$\tilde{\eta} = \hat{\eta} - \hat{\eta}^+ + \eta^+,$$

where $\hat{\eta}$ is a pre-computed vector. To construct a draw $\tilde{\alpha}$ from the conditional distribution of α given y , we use

$$\tilde{\alpha}_1 = \hat{\alpha}_1 - P_1 r_0 + x_1, \quad \tilde{\alpha}_{t+1} = T_t \tilde{\alpha}_t + \tilde{\eta}_t,$$

where $\hat{\alpha}_1$ is pre-computed.

de Jong and Shephard (1995) draw $\alpha|y$ using the following steps, given in equation (4) of their paper. First ϵ_t is drawn from $N(0, \sigma^2 C_t)$, where the Cholesky factor of $\sigma^2 C_t$ can be pre-computed. Then r_t is computed using the backwards recursion

$$r_{t-1} = [Z'_t D_t^{-1} e_t] + L'_t r_t - [V'_t C_t^{-1}] \epsilon_t.$$

Next, α_{t+1} is computed as

$$\alpha_{t+1} = [W_t \beta] + T_t \alpha_t + \Omega_t r_t + \epsilon_t.$$

In our approach, we draw, for each observation, a vector $v_t \sim N(0, I_m)$ and compute

$$\alpha_t = m_t - [\Sigma_t \Omega_{t,t+1}] \alpha_{t+1} + \Lambda_t^{-1} v_t.$$

Computing $\Lambda_t^{-1} v_t$ using Λ_t (which is triangular) requires $m(m-1)/2$ multiplications and m floating point divisions. If we are making multiple draws, we can compute the reciprocals of the diagonal elements of Λ_t once and convert the divisions into multiplications, which are typically much less costly.

4 Example 1: A Stochastic Volatility Model

Kim, Shephard, and Chib (1998) (KSC hereafter) introduce a Markov chain for posterior simulation of the parameters and state variables of a stochastic volatility model. This model features the measurement equation

$$\tilde{y}_t = \exp(\alpha_t/2) \epsilon_t,$$

and the state equation

$$\alpha_{t+1} = (1 - \phi) \bar{\alpha} + \phi \alpha_t + \sigma_\alpha \eta_t,$$

where $(\epsilon_t, \eta_t) \sim N(0, I_2)$ and the states are stationary. We let $\theta \equiv (\bar{\alpha}, \phi, \sigma_\alpha)$.

KSC linearize the model by taking the log of the square of the measurement equation:

$$\log \tilde{y}_t^2 = \alpha_t + \log \epsilon_t^2.$$

Then they define $y_t \equiv \log(\tilde{y}_t^2 + c)$ as an approximation to $\log \tilde{y}_t^2$. The value of c is chosen just large enough to avoid problems associated with values of \tilde{y}_t close to or equal to zero. Then they approximate the log χ^2 distribution of $e_t \equiv \epsilon_t^2$ as a

mixture of K normals. They tabulate means m_k and variances σ_k^2 , $k = 1, \dots, K$, for a mixture with $K = 7$ components.

KSC then augment the approximate linear model by adding a sequence $s \equiv (s_1, \dots, s_n)$ of component indicator variables. The conditional distribution of (θ, α) given y is preserved and the conditional distribution of (θ, α) given s and y is given by the following linear state space model:

$$y_t = m_{s_t} + \alpha_t + \sigma_{s_t} \epsilon_t,$$

$$\alpha_1 = \bar{\alpha} + \sigma_\alpha / \sqrt{1 - \phi^2}, \quad \alpha_t = (1 - \phi)\bar{\alpha} + \phi\alpha_{t-1} + \sigma_\alpha \eta_t.$$

To simulate the joint posterior distribution of α , θ and s , KSC employ a Markov Chain whose transition distribution is defined by the following sweep.

1. Draw $\theta|s, y$.
2. Draw $\alpha|s, \theta, y$.
3. Draw $s|\alpha, \theta, y$.

They also describe how to reweight a posterior sample in order to correct for the approximation of the distribution of $y_t - \alpha_t$ as a mixture of normals.

KSC use the method of DeJS to implement the draw of $\alpha|s, \theta, y$. This involves a forward pass using the Kalman filter and a backward pass using the DeJS simulation smoother. They also use the output of the Kalman filter to evaluate the density $f(y|\theta, s)$, used for drawing $\theta|s, y$.

Application of our method or that of DK to draw $\alpha|s, \theta, y$ also yields intermediate quantities that can be used to easily compute $f(y|\theta, s)$. Therefore the DeJS, DK and MMP methods can be used interchangeably for sampling the KSC Markov chain.

We now compare in detail the computational efficiency of all three methods by counting computational operations per observation. We count separately the operations required to compute intermediate quantities, to draw α using the intermediate quantities and to evaluate $f(y|\theta, s)$ using the intermediate quantities.

In the DeJS approach, pre-processing involves the application of the Kalman filter and the computation of $D_t^{-1}e_t$. The computations are the following:

$$a_1 = \bar{\alpha}, \quad P_1 = \sigma_\alpha^2 / (1 - \phi^2),$$

$$e_t = [y_t - m_{s_t}] - a_t, \quad D_t = P_t + \sigma_{s_t}^2, \quad K_t = \phi P_t D_t^{-1}, \quad L_t = (\phi - K_t),$$

$$a_{t+1} = [(1 - \phi)\bar{\alpha}] + \phi a_t + K_t e_t, \quad P_{t+1} = [\phi P_t] L_t + \sigma_\alpha^2, \\ (D_t^{-1} e_t).$$

Table 2 gives numbers of operations per observation, under ‘‘DeJS Pre-compute’’.

Drawing $\alpha|y, \theta$ involves applying the DeJS simulation smoother. Given the sequences e_t, D_t^{-1}, K_t, L_t and $D_t^{-1}e_t$, the computations are the following:

$$U_n = 0, \quad r_n = 0, \\ N_t = D_t^{-1} + K_t(K_t U_t), \quad n_t = [D_t^{-1}e_t] - K_t r_t, \quad C_t = \sigma_{s_t}^2 - \sigma_{s_t}^4 N_t, \\ \zeta_t = \sqrt{C_t} N_{0,1}, \quad V_t = \sigma_{s_t}^2 (N_t - \phi[K_t U_t]), \quad r_{t-1} = [D_t^{-1}e_t] + L_t r_t - (V_t/C_t)\zeta_t, \\ U_t = D_t^{-1} + L_t^2 U_t + V_t[V_t/C_t], \quad \alpha_t = [y_t - m_{s_t}] - \sigma_{s_t}^2 n_t - \zeta_t.$$

Numbers of operations per observation are tabulated in Table 2 under ‘‘DeJS Draw’’.

Given the sequences e_t, D_t^{-1}, K_t, L_t and $D_t^{-1}e_t$, we can evaluate $f(y|\theta, s)$ using the following equation:

$$f(y|\theta, s) = (2\pi)^{-n/2} \exp \left[-\frac{1}{2} \sum_{t=1}^n \log D_t + [D_t^{-1}e_t]e_t \right]. \quad (6)$$

Numbers of operations per observation are shown in Table 2 under ‘‘DeJS Evaluate’’.

We now count operations using the DK method. Pre-computation involves running the Kalman filter, but only computing the quantities D_t, D_t^{-1}, K_t, L_t and P_t . Numbers of operations are shown in Table 2 under ‘‘Kalman variances’’.

Using the Kalman variances, a draw of α using the DK method involves the following steps. The first step is to draw α^+ and y^+ from the joint distribution of $(\alpha_1 - \mu, \dots, \alpha_n - \mu, y_1 - m_{s_t} - \mu, \dots, y_n - m_{s_t} - \mu)$, using forward simulation:

$$\alpha_1^+ = [\sigma_h^2/(1 - \phi^2)], \quad \alpha_t^+ = \phi \alpha_{t-1}^+ + \sigma_h N_{0,1}, \quad y_t^+ = \alpha_t^+ + \sigma_{s_t} N_{0,1}.$$

The second step is running the Kalman filter forward, only computing e_t and a_{t+1} , using $y_t - m_{s_t} - \mu - y_t^+$ in place of y_t :

$$a_1 = 0, \quad e_t = [y_t - m_{s_t}] - \mu - y_t^+ - a_t, \quad a_{t+1} = \phi a_t + K_t e_t.$$

The fourth step is running the DK backwards smoother, using equation (5) of DK:

$$r_n = 0, \quad r_{t-1} = D_t^{-1}e_t + L_t r_t.$$

The fifth step is to compute α_t^* using equation (8) of DK:

$$\alpha_1^* = \sigma_h^2 / (1 - \phi^2) r_0, \quad \alpha_t^* = \phi \alpha_{t-1}^* + \sigma_h^2 r_{t-1}.$$

The final step delivers a draw from α :

$$\alpha_t = \alpha_t^+ + \mu + \alpha_t^*.$$

Numbers of operations are shown in Table 2 under ‘‘DK draw’’.

Computing $f(y|\alpha, s)$ using the Kalman variances involves computing the Kalman means e_t and a_t using y_t :

$$a_1 = 0, \quad e_t = y_t - a_t, \quad a_{t+1} = [(1 - \phi)\mu] + \phi a_t + K_t e_t,$$

computing $D_t^{-1} e_t$, and computing $f(y|\theta, s)$ using (6). Numbers of operations are shown in Table 2 under ‘‘DK evaluate’’.

We now turn to the MMP method. We first count the operations required to compute intermediate quantities. The computations are the following:

$$c_t = \begin{cases} [\omega_\alpha(1 - \phi)\bar{\alpha}] + [\omega_{s_t}(y_t - m_{s_t})] & t = 1, n \\ [\omega_\alpha(1 - \phi)^2\bar{\alpha}] + [\omega_{s_t}(y_t - m_{s_t})] & 1 < t < n, \end{cases}$$

$$\Sigma_1 = (\omega_\alpha + \omega_{s_1})^{-1}, \quad \mu_1 = \Sigma_1 c_1,$$

$$\Sigma_t = \{[(1 + \phi^2)\omega_\alpha] + \omega_{s_t} - [\phi^2 \omega_\alpha^2] \Sigma_{t-1}\}^{-1}, \quad \psi_t = [\phi \omega_\alpha] \Sigma_t, \quad \mu_t = \Sigma_t c_t + \psi_t \mu_{t-1}.$$

Numbers of operations are tabulated in Table 2 under ‘‘MMP Pre-compute’’.

The operations required to draw α given y and s are the following:

$$\alpha_n = \mu_n + \sqrt{\Sigma_n} N_{0,1}, \quad \alpha_t = \mu_t - \psi_t \alpha_{t+1} + \sqrt{\Sigma_t} N_{0,1}.$$

Numbers of operations are tabulated in Table 2 under ‘‘MMP Draw’’.

We now consider the computation of $f(y|\theta, s)$ as a function of θ . The laws of conditional probability give

$$f(y|\theta) = \frac{f(\alpha|\theta) f(y|\theta, \alpha)}{f(\alpha|y, \theta)}.$$

This equation holds for all values of α . Also, $f(y|\theta, \alpha)$ does not depend on θ . Therefore, as a function of θ ,

$$f(y|\theta) \propto \frac{f(\alpha|\theta)}{f(\alpha|y, \theta)} \Big|_{\alpha=0}. \quad (7)$$

We have

$$f(\alpha|\theta) \propto \omega_\alpha^{n/2}(1-\phi^2) \exp \left\{ -\frac{\omega_\alpha}{2} \left[(1-\phi^2)(\alpha_1 - \bar{\alpha})^2 + \sum_{t=2}^n (\alpha_t - (1-\phi)\bar{\alpha} - \phi\alpha_{t-1})^2 \right] \right\},$$

and therefore

$$f(\alpha|\theta)|_{\alpha=0} \propto \omega_\alpha^{n/2}(1-\phi^2) \exp \left\{ -\frac{\omega_\alpha}{2} \bar{\alpha}^2 [(1-\phi^2) + (n-1)(1-\phi)^2] \right\},$$

which can be evaluated in constant (not depending on n) time.

Turning to the denominator of (7), we have

$$f(\alpha|y, \theta) = (2\pi)^{-T/2} |\Omega|^{1/2} \exp \left[-\frac{1}{2} (\alpha - \mu)' \Omega (\alpha - \mu) \right],$$

and so

$$f(\alpha|y, \theta)|_{\alpha=0} \propto \exp \left[-\frac{1}{2} \sum_{t=1}^n \log \Sigma_t + \mu_t c_t \right],$$

where $\mu = E[\alpha|y, s]$. We compute μ using the recursion in Result 2.1:

$$\mu_n = m_n, \quad \mu_t = m_t + \psi_t \mu_{t+1}.$$

Numbers of operations are shown in Table 2 under ‘‘MMP evaluate’’.

We profile code for all three methods to see how they perform in practice. We use data on the French Franc exchange rate from January 6, 1994 to January 15, 1999. We construct the series $r_t \equiv \log P_t - \log P_{t-1}$, where P_t is the noon rate for the French Franc in U.S. dollars. Our data source is the web site of the Federal Reserve Bank of New York. We simulated 50000 periods of the KSC chain for each of the three methods. Table 4 gives the computational cost of pre-computing intermediate quantities, drawing α and evaluating $f(y|\theta, s)$, for each of the three methods. We normalize the cost of DeJS pre-computation to unity. In our implementation, each iteration of the KSC chain transition requires two pre-computations, two evaluations and one draw. The total time required to draw the chain using the DK method was 13.5% more than that required using the DeJS method. Using the MMP method, the total time was 17.3% less.

5 Example 2: A Semi-Gaussian State Space Model

Durbin and Koopman (1997) show how to compute an arbitrarily accurate evaluation of the likelihood function for a semi-Gaussian state space model in which

Table 2: Computational costs per observation

Algorithm	+/-	\times	\div	log	$\sqrt{\quad}$	$N_{0,1}$
1 Kalman variances ($D_t, D_t^{-1}, K_t, L_t, P_t$)	3	3	1			
2 Kalman means (e_t, a_t)	3	2				
3 $D_t^{-1}e_t$		1				
4 $f(y \theta, s)$ using $D_t^{-1}, e_t, D_t^{-1}e_t$	2	1		1		
5 DeJS pre-compute (1,2,3)	6	6	1			
6 DeJS draw ($N_t, n_t, C_t, V_t, r_{t-1}, U_t, \alpha_t$)	10	13	1		1	1
7 DeJS evaluate (4)	2	1		1		
8 DK pre-compute (1)	3	3	1			
9 DK draw ($\alpha_t^+, y_t^+, e_t, a_t, r_{t-1}, \alpha_t^*, \alpha_t$)	10	9				2
10 DK evaluate (2,3,4)	5	4		1		
11 MMP pre-compute ($c_t, \Sigma_t, \psi_t, m_t$)	4	4	1			
12 MMP draw	2	2			1	1
13 MMP evaluate	3	2		1		

Table 3: Time costs relative to DeJS pre-compute

	DeJS	DK	MMP
pre-compute	1.000	0.679	0.843
draw	1.114	1.492	0.428
evaluate	0.149	0.398	0.252

the state evolves according to equation (2), but the conditional distribution of observations given states is given by a general distribution with density (or mass) function $p(y|\alpha)$. To simplify, we suppress notation for the dependence on θ , the vector of parameters.

The approach is as follows. The first step is to construct a fully Gaussian state space model with the same state dynamics as the semi-Gaussian model but with a Gaussian measurement equation of the following form:

$$y_t = \mu_t + Z_t \alpha_t + \epsilon_t, \quad (8)$$

where the ϵ_t are independent $N(0, \Xi_t)$ and independent of the state equation innovations. The Z_t are matrices such that the distribution of y given α depends only on the $Z_t \alpha_t$. They choose μ_t and Ξ_t such that the implied conditional density $g(y|\alpha)$ approximates $p(y|\alpha)$ as a function of α near the mode of $p(\alpha|y)$. The next step is to draw a sample of size N from the conditional distribution of α given y for the fully Gaussian state space model. The final step is to use this sample as an importance sample to approximate the likelihood for the semi-Gaussian model.

The Gaussian measurement density $g(y|\alpha)$ is chosen such that $\log g(y|\alpha)$ is a quadratic approximation of $\log p(y|\alpha)$, as a function of α , at the mode $\hat{\alpha}$ of the density $p(\alpha|y)$. Durbin and Koopman (1997) find this density by iterating the following steps until convergence to obtain μ_t and Ξ_t :

1. Using the current values of the μ_t and Ξ_t , compute $\hat{\alpha} = E_g[\alpha|y]$, where E_g denotes expectation with respect to the Gaussian density $g(\alpha|y)$. Durbin and Koopman (1997) use routine Kalman filtering and smoothing with the fully Gaussian state space model to find $\hat{\alpha}$.
2. Using the current $\hat{\alpha}$, compute the μ_t and Ξ_t such that $\log p(y|\alpha)$ and $\log g(y|\alpha)$ have the same gradient and Hessian (with respect to α) at $\hat{\alpha}$. Durbin and Koopman (1997) show that μ_t and Ξ_t solve the following two equations:

$$\frac{\partial \log p(y_t|\hat{\alpha}_t)}{\partial (Z_t \alpha_t)} - \Xi_t^{-1} (y_t - Z_t \hat{\alpha}_t - \mu_t) = 0, \quad (9)$$

$$\frac{\partial^2 \log p(y_t|\hat{\alpha}_t)}{\partial (Z_t \alpha_t) \partial (Z_t \alpha_t)'} + \Xi_t^{-1} = 0. \quad (10)$$

It is interesting to note that this delivers the specification of the measurement equation error of the fully Gaussian model directly in terms of the precision Ξ_t^{-1} rather than the variance Ξ_t directly.

The likelihood function $L(\theta)$ we wish to evaluate is

$$L(\theta) = p(y) = \int p(y, \alpha) d\alpha = \int p(y|\alpha)p(\alpha) d\alpha. \quad (11)$$

Durbin and Koopman (1997) employ importance sampling to efficiently and accurately approximate the above integrals. The likelihood for the approximating Gaussian model is

$$L_g(\theta) = g(y) = \frac{g(y, \alpha)}{g(\alpha|y)} = \frac{g(y|\alpha)p(\alpha)}{g(\alpha|y)}. \quad (12)$$

Substituting for $p(\alpha)$ from (12) into (11) gives

$$L(\theta) = L_g(\theta) \int \frac{p(y|\alpha)}{g(y|\alpha)} g(\alpha|y) d\alpha = L_g(\theta) E_g[w(\alpha)], \quad (13)$$

where

$$w(\alpha) \equiv \frac{p(y|\alpha)}{g(y|\alpha)}.$$

One can generate a random sample $\alpha^{(1)}, \dots, \alpha^{(N)}$ from the density $g(\alpha|y)$ using any of the methods for drawing states in fully Gaussian models. An unbiased Monte Carlo estimate of $L(\theta)$ is

$$\hat{L}_1(\theta) = L_g(\theta) \bar{w}, \quad (14)$$

where $\bar{w} = N^{-1} \sum_{i=1}^N w(\alpha^{(i)})$.

It is usually more convenient to work with the log-likelihood, and we can write

$$\log \hat{L}_1(\theta) = \log L_g(\theta) + \log \bar{w}. \quad (15)$$

However, $E[\log \bar{w}] \neq \log E_g[w(\alpha^{(i)})]$, so (15) is a biased estimator of $\log L(\theta)$.

Durbin and Koopman (1997) propose an approximately unbiased estimator of $\log L(\theta)$ given by

$$\log \hat{L}_2(\theta) = \log L_g(\theta) + \log \bar{w} + \frac{s_w^2}{2N\bar{w}^2}, \quad (16)$$

where s_w^2 is an estimator of the variance of the $w(\alpha^{(i)})$ given by

$$s_w^2 = \frac{1}{N-1} \sum_{i=1}^N (w(\alpha^{(i)}) - \bar{w})^2.$$

5.1 Modifications to the Algorithm for Approximating $L(\theta)$

We propose here three modifications of the Durbin and Koopman (1997) method for approximating $L(\theta)$. The modified method does not involve Kalman filtering.

First, we use the MMP algorithm to draw α from its conditional distribution given y .

Second, we compute $L_g(\theta)$ as the extreme right hand side of equation (12). The equation holds for any value of α ; convenient choices which simplify computations include the prior mean and the posterior mean. See our stochastic volatility example for an example.

Finally, in the rest of this section we present a method for computing the μ_t and Ξ_t of the fully Gaussian state space model. It is based on a multivariate normal approximation of $p(\alpha|y)$ at its mode $\hat{\alpha}$ and the application of Result 2.1. It is computationally more efficient than Kalman filtering and smoothing.

We first compute $\hat{\alpha}$ by iterating the following steps until convergence.

1. Using the current value of $\hat{\alpha}$, find the precision $\bar{\bar{H}}$ and co-vector $\bar{\bar{c}}$ of a Gaussian approximation to $p(\alpha|y)$ based on a second-order Taylor expansion of $\log p(\alpha) + \log p(y|\alpha)$ around the point $\hat{\alpha}$.
2. Using the current values of $\bar{\bar{H}}$ and $\bar{\bar{c}}$, compute $\hat{\mu} = \bar{\bar{H}}^{-1}\bar{\bar{c}}$, the mean of the Gaussian approximation, using Result 2.1.

We then use equations (9) and (10) to compute the μ_t and Ξ_t .

We compute the precision $\bar{\bar{H}}$ as $\bar{H} + \tilde{H}$, and the co-vector $\bar{\bar{c}}$ as $\bar{c} + \tilde{c}$, where \bar{H} and \bar{c} are the precision and co-vector of the marginal distribution of α (detailed formulations are provided for our example in the next section), and \tilde{H} and \tilde{c} are the precision and co-vector for a Gaussian density approximating $p(y|\alpha)$ as a function of α up to a multiplicative constant. Since \tilde{H} is block-diagonal and \bar{H} is block-band-diagonal, $\bar{\bar{H}}$ is also block-band-diagonal.

We compute \tilde{H} and \tilde{c} as follows. Let $a(\alpha_t) \equiv -2 \log[p(y_t|\alpha_t)]$. We approximate $a(\alpha_t)$ by $\tilde{a}(\alpha_t)$, consisting of the first three terms of the Taylor expansion of $a(\alpha_t)$ around $\hat{\alpha}_t$:

$$a(\alpha_t) \approx \tilde{a}(\alpha_t) = a(\hat{\alpha}_t) + \frac{\partial a(\hat{\alpha}_t)}{\partial \alpha_t}(\alpha_t - \hat{\alpha}_t) + \frac{1}{2}(\alpha_t - \hat{\alpha}_t)' \frac{\partial^2 a(\hat{\alpha}_t)}{\partial \alpha_t \partial \alpha_t'} (\alpha_t - \hat{\alpha}_t).$$

If we complete the square, we obtain

$$\tilde{a}(\alpha_t) = (\alpha_t - h_t^{-1}c_t)' h_t (\alpha_t - h_t^{-1}c_t) + k,$$

where

$$h_t = \frac{1}{2} \frac{\partial^2 a(\hat{\alpha}_t)}{\partial \alpha_t \partial \alpha_t'},$$

$$c_t = h_t \hat{\alpha}_t - \frac{1}{2} \frac{\partial a(\hat{\alpha}_t)}{\partial \alpha_t},$$

and k is an unimportant term not depending on α_t . Note that h_t and c_t are the precision and co-vector of a multivariate normal distribution with density proportional to $\exp[-\frac{1}{2}\tilde{a}(\alpha_t)]$.

Since $\log p(y|\alpha)$ is additively separable in the elements of α , it means that it is reasonably well approximated, as a function of α , by $\prod_{t=1}^n \exp[-\frac{1}{2}\tilde{a}(\alpha_t)]$, which is proportional to a multivariate normal distribution with precision \tilde{H} and co-vector \tilde{c} , given by

$$\tilde{H} \equiv \begin{bmatrix} h_1 & 0 & \cdots & 0 \\ 0 & h_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & h_n \end{bmatrix} \quad \text{and} \quad \tilde{c} \equiv \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix}.$$

5.2 A Multivariate Poisson Model with Time-Varying Intensities

As an example of a semi-Gaussian state space model, let us consider a case where $y_t \equiv (y_{t1}, \dots, y_{tp})$ is a process describing multivariate count data. To model y_t , we assume that conditionally on some time varying and stochastic count intensity vector $\lambda_t \equiv (\lambda_{t1}, \dots, \lambda_{tp})$, they are independent Poisson. Thus the conditional distribution of y_t given λ_t is given by

$$p(y_{t1}, \dots, y_{tp} | \lambda_{t1}, \dots, \lambda_{tp}) = \prod_{i=1}^p \frac{\exp(-\lambda_{ti}) \lambda_{ti}^{y_{ti}}}{y_{ti}!}, \quad (17)$$

The latent count intensities $\lambda_{t1}, \dots, \lambda_{tp}$ are assumed to follow

$$\lambda_{ti} = \exp\left(\sum_{j=1}^m z_{ij} \alpha_{tj}\right), \quad i = 1, \dots, n, \quad (18)$$

$$\alpha_{t+1,j} = (1 - \phi_j) \bar{\alpha}_j + \phi_j \alpha_{tj} + \eta_{tj}, \quad j = 1, \dots, m, \quad (19)$$

where the η_{tj} are independent $N(0, Q_j)$. Denote by Q the diagonal matrix with the Q_j 's on the diagonal: $Q = \text{diag}(Q_1, \dots, Q_m)$. We assume that given the process $\{\eta_t\}$, the y_t are conditionally independent, with conditional probability mass function given by (17). The parameters of the model are $\theta \equiv (\bar{\alpha}_j, \phi_j, Q_j, \gamma, z_{ij})_{i \in \{1, \dots, p\}, j \in \{1, \dots, m\}}$.

We now turn to the problem of estimating the likelihood $L(\theta)$ of this particular semi-Gaussian model using the approach of Durbin and Koopman (1997). We first need to determine the matrix Z_t in the measurement equation (8) of the fully Gaussian model. For cases like this one where the measurement distribution is in the exponential family, they provide a choice for Z_t , which in our case is $Z_t \equiv (z_{ij})_{i=1, \dots, p; j=1, \dots, m}$. See Section 4.1 and especially equation (24) in Durbin and Koopman (1997) for details. Also, for this example, the precision \bar{H} and co-vector \bar{c} , are given by

$$\bar{H} = \begin{bmatrix} \bar{H}_{11} & \bar{H}_{12} & 0 & \cdots & 0 & 0 \\ \bar{H}_{21} & \bar{H}_{22} & \bar{H}_{23} & \cdots & 0 & 0 \\ 0 & \bar{H}_{32} & \bar{H}_{33} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \bar{H}_{n-1, n-1} & \bar{H}_{n-1, n} \\ 0 & 0 & 0 & \cdots & \bar{H}_{n, n-1} & \bar{H}_{nn} \end{bmatrix}, \quad \bar{c} = \begin{bmatrix} \bar{c}_1 \\ \bar{c}_2 \\ \vdots \\ \bar{c}_{n-1} \\ \bar{c}_n \end{bmatrix}$$

where

$$\begin{aligned} \bar{H}_{11} &= \bar{H}_{nn} = Q^{-1}, \\ \bar{H}_{jj} &= \begin{bmatrix} (1 + \phi_1^2)/Q_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & (1 + \phi_m^2)/Q_m \end{bmatrix}, \quad j = 2, \dots, n-1, \\ \bar{H}_{j, j+1} &= \bar{H}_{j+1, j} = \begin{bmatrix} -\phi_1/Q_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & -\phi_m/Q_m \end{bmatrix}, \quad j = 1, \dots, n-1, \\ \bar{c}_1 &= \bar{c}_n = \begin{bmatrix} \bar{\alpha}_1(1 - \phi_1)/Q_1 \\ \vdots \\ \bar{\alpha}_m(1 - \phi_m)/Q_m \end{bmatrix}, \\ \bar{c}_j &= \begin{bmatrix} \bar{\alpha}_1(1 - \phi_1)^2/Q_1 \\ \vdots \\ \bar{\alpha}_m(1 - \phi_m)^2/Q_m \end{bmatrix}, \quad j = 2, \dots, n-1. \end{aligned}$$

We compare the computational efficiency of all three methods for estimating the likelihood for this semi-Gaussian state space model. We do so by counting computational operations and profiling code.

Since a large number of draws from $g(\alpha|y)$ is required for a good approximation of $L(\theta)$, we focus on the marginal computational cost of an additional draw. We will see that for a typical number of draws, the computational overhead associated with the first draw is small.

In the DeJS approach, one additional draw α_t requires the following computations per observation [see equation (5) of their paper]:

$$\begin{aligned} n_t &= [D_t^{-1}e_t] - K_t' r_t, & \epsilon_t &= [C_t^{1/2}]N(0, I_p), & \xi_t &= \Gamma_t n_t + \epsilon_t, \\ Z\alpha_t &= [y_t - \mu_t] - \xi_t, & r_{t-1} &= [Z'D_t^{-1}e_t] + L_t' r_t - [V_t' C_t^{-1}]\epsilon_t. \end{aligned}$$

In the DK approach, one additional draw requires the following computations per observation. (Here we do not simulate α but rather the $Z\alpha_t$, which we obtain more easily by simulating the disturbances ϵ_t according to Section 2.3 of Durbin and Koopman (2002).) There is a forward pass to simulate v_t^+ :

$$v_t^+ = \mu_t + Zx_t^+ + \epsilon_t^+, \quad x_{t+1}^+ = T_t x_t^+ + \eta_t^+ - K_t v_t^+,$$

where $\epsilon_t^+ \sim N(0, \Xi_t)$ and $\eta_t^+ \sim N(0, Q)$. This is followed by a backward pass [see equations (4) and (5) and Algorithm 1 of their paper]:

$$\begin{aligned} \hat{\epsilon}_t^+ &= \Xi_t(D_t^{-1}v_t - K_t' r_t), & r_{t-1} &= [Z'D_t^{-1}]v_t^+ + L_t' r_t, \\ \tilde{\epsilon}_t &= \hat{\epsilon}_t - \hat{\epsilon}_t^+ + \epsilon_t^+, & Z\alpha_t &= [y_t - \mu_t] - \tilde{\epsilon}_t, \end{aligned}$$

where $\hat{\epsilon}_t$ is pre-computed.

In the MMP approach, one additional draw requires the following computations per observation²:

$$\alpha_t = m_t - [\Sigma_t \Omega_{t,t+1}]\alpha_{t+1} + [\Sigma_t^{1/2}]N(0, I_m).$$

The computational costs per observation for an additional draw of α_t are summarized in Table 4.

We profile code for all three methods to see how they perform in practice. We use data on the number of transactions over consecutive two minute intervals for four different stocks in the same industry. For one business day (November 6,

²Adding $p \times m$ multiplications for each of the $Z\alpha_t$, which are required to evaluate $p(y|\alpha)$.

Table 4: Computational costs per observation per additional draw of α_t

Algorithm	+/-	\times	$N_{0,1}$
DeJS	$3p + 2m$	$(3p^2 + p)/2 + 2mp + m^2$	p
DK	$6p + 3m$	$(5p^2 + p)/2 + 4mp + 2m + m^2$	$p + m$
MMP	$2m$	$(3m^2 + m)/2 + pm$	m

2003), we look at all the transactions for four different gold-mining companies: Agnico-Eagle Mines Limited, Barrick Gold Corporation, Gold Fields Limited and Goldcorp Inc. We use all the transactions recorded during normal trading hours on the New York Stock Exchange Trade and Quote database. This gives 195 observations for each series. The data are plotted in Figure 1.

We take the number of factors to be equal to the number of observed series. That is, $m = p = 4$. To ensure identification, we impose $z_{ii} = 1$ and $z_{ij} = 0$ for $j > i$.

For all three methods, we compute $\hat{\alpha}$ using the fast method presented in Section 5.1. This puts all methods for drawing states on an equal footing. We point out, though, that this gives a small advantage to the estimation of $L(\theta)$ using either the DeJS or DK methods, relative to the case where the Kalman filter and simulation smoother are used to compute $\hat{\alpha}$.

For various values of the size N of the importance sample, Table 5.2 gives the ratio of the time cost in 100ths of seconds of (i) generating N draws of $\alpha^{(i)}$ and (ii) the total cost of evaluating the log-likelihood once, which consists of the former plus some overhead, including the computation of $\mu_t, \Xi_t, \hat{\alpha}$ and the $w(\alpha^{(i)})$. For the latter, we report costs for two different approximations of $\hat{\alpha}$: one using a single iteration of steps 1 and 2 in Section 5.1, the other using five iterations. All times are averaged over 100 replications³.

First, we see that the cost of evaluating the log-likelihood over and above the cost of drawing the states is around 0.1 second (one iteration for $\hat{\alpha}$) or 0.3 second (five iterations) and that it is the major cost for small number of draws. Second, except for the case $N = 1$, DeJS is computationally more efficient than DK, by a factor of about 2 with $N > 50$. Third, MMP is much more computationally effi-

³The simulations were performed on an AMD Athlon(tm) 64 X2 5600+ cpu with Matlab R2006a. Note that the reported time costs are in the case where matrix multiplications involving triangular matrices are performed with Matlab's built-in matrix product, which does not take advantage of the triangular structure. We tried dynamically loading a function written in C for triangular matrix multiplication, but the additional overhead exceeded the savings.

cient than DeJS and DK for any number of draws, with the efficiency increasing with N . As a point of reference, Durbin and Koopman (1997) consider $N = 200$ (combined with antithetic and control variables) as an acceptable value in an empirical example they consider.

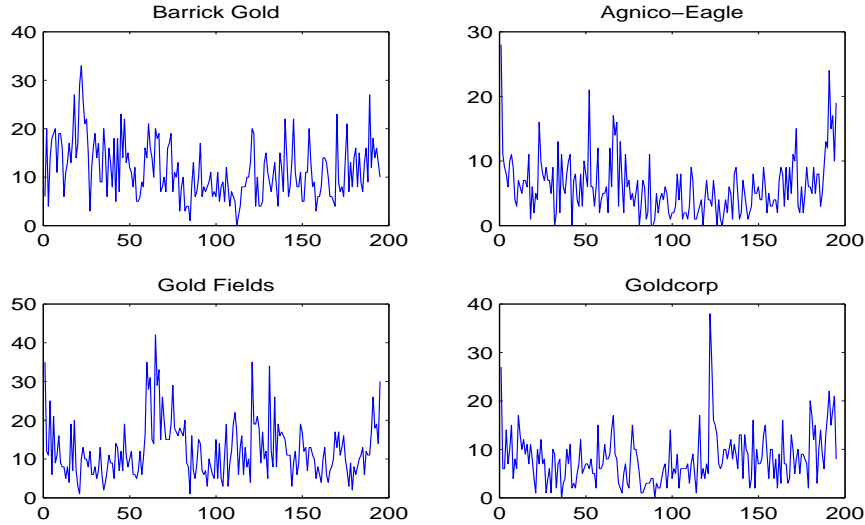


Figure 1: Transactions data

6 Conclusion

In this paper we introduce a new method for drawing state variables in Gaussian state space models from their conditional distribution given parameters and observations. Unlike standard methods, such as de Jong and Shephard (1995) and Durbin and Koopman (2002), our method does not involve Kalman filtering. It is instead based on a Levinson-like algorithm, introduced by Vandebriel, Mastronardi, and Van Barel (2007), for solving the equation $Bx = y$, where B is an $n \times n$ symmetric band diagonal matrix and y is a $n \times 1$ vector. We extend their result in two ways. First, we modify the algorithm to work with $m \times m$ submatrices of a block band diagonal matrix rather than individual elements of a band diagonal matrix. Second, we show that the intermediate computations used to solve the equation $\Omega\mu = c$ for the mean $\mu = E[\alpha|y]$ given the precision $\Omega = (\text{Var}[\alpha|y])^{-1}$ and

Table 5: Time cost of drawing $\alpha^{(i)}$ and the total cost of evaluating the likelihood, as a function of the number of draws N . For the total time cost, numbers are reported when performing one and five iterations to obtain $\hat{\alpha}$. Figures are in 100ths of seconds.

Method		$N = 1$	$N = 10$	$N = 50$	$N = 150$
DeJS	α draw	9.7	22.0	78.3	215.1
	total	(19.2–38.7)	(31.9–51.7)	(89.6–108.9)	(229.2–253.6)
DK	α draw	7.1	34.6	156.6	462.9
	total	(16.7–36.6)	(45.1–64.9)	(168.1–185.7)	(477.8–491.0)
MMP	α draw	4.6	10.0	34.9	103.5
	total	(14.6–33.9)	(20.3–40.4)	(46.5–65.5)	(118.2–136.3)

co-vector $c = (\text{var}[\alpha|y])^{-1}E[\alpha|y]$ can be used to compute the conditional means $E[\alpha_t|\alpha_{t+1}, \dots, \alpha_n, y]$ and conditional variances $\text{Var}[\alpha_t|\alpha_{t+1}, \dots, \alpha_n, y]$.

We show that for some important cases, our method is computationally more efficient than standard methods in the literature. These methods use Kalman filtering, which involves solving systems of p equations in p unknowns, requiring $O(p^3)$ scalar multiplications. If the A_t can be pre-computed, or take on only a constant number of values, our method requires no operations of higher order than p^2 , in p . If the Z_t and T_t can also be pre-computed, or take on only a constant number of values, the order drops to p .

Our method is also particularly efficient for applications in which several draws of α are required for each value of the parameters of the model.

We consider two applications of our methods. The first is posterior simulation of the parameters and state variables of a stochastic volatility model. The second is evaluation of the log-likelihood of a multivariate Poisson model with latent count intensities.

A Derivation of Ω and c

In this appendix, we derive the expression (4) for the precision and co-vector of the conditional distribution of the state variable α given the dependent variable y .

We can express the joint density of y and α as follows:

$$f(y, \alpha) \propto \exp \left[-\frac{1}{2}g(y, \alpha) \right],$$

where

$$\begin{aligned} g(y, \alpha) &= (\alpha_1 - a_1)' P_1^{-1} (\alpha_1 - a_1) & (20) \\ &+ \sum_{t=1}^{n-1} \begin{bmatrix} y_t - X_t \beta - Z_t \alpha_t \\ \alpha_{t+1} - W_t \beta - T_t \alpha_t \end{bmatrix}' \begin{bmatrix} G_t G_t' & G_t H_t' \\ H_t G_t' & H_t H_t' \end{bmatrix}^{-1} \begin{bmatrix} y_t - X_t \beta - Z_t \alpha_t \\ \alpha_{t+1} - W_t \beta - T_t \alpha_t \end{bmatrix} \\ &+ (y_n - X_n \beta - Z_n \alpha_n)' (G_n G_n')^{-1} (y_n - X_n \beta - Z_n \alpha_n). \end{aligned}$$

We then expand the time t term of (20) and organize terms, obtaining

$$\begin{aligned} &(y_t - X_t \beta)' A_{11,t} (y_t - X_t \beta) - (y_t - X_t \beta)' A_{12,t} (W_t \beta) & (21) \\ &- (W_t \beta)' A_{21,t} (y_t - X_t \beta) + (W_t \beta)' A_{22,t} (W_t \beta) \\ &- [(y_t - X_t \beta)' (A_{11,t} Z_t + A_{12,t} T_t) - (W_t \beta)' (A_{21,t} Z_t + A_{22,t} T_t)] \alpha_t \\ &- \alpha_t' [(Z_t' A_{11,t} + T_t' A_{21,t}) (y_t - X_t \beta) - (Z_t' A_{12,t} + T_t' A_{22,t}) (W_t \beta)] \\ &+ \alpha_t' [Z_t' A_{11,t} Z_t + Z_t' A_{12,t} T_t + T_t' A_{21,t} Z_t + T_t' A_{22,t} T_t] \alpha_t \\ &- [-(y_t - X_t \beta)' A_{12,t} + (W_t \beta)' A_{22,t}] \alpha_{t+1} \\ &- \alpha_{t+1}' [-A_{21,t} (y_t - X_t \beta) + A_{22,t} (W_t \beta)] \\ &+ \alpha_{t+1}' A_{22,t} \alpha_{t+1} \\ &+ \alpha_t' [-Z_t' A_{12,t} - T_t' A_{22,t}] \alpha_{t+1} \\ &+ \alpha_{t+1}' [-A_{21,t} Z_t - A_{22,t} T_t] \alpha_t \end{aligned}$$

The conditional density $f(\alpha|y)$ must be proportional to $f(y, \alpha)$ as a function α . We can write this as

$$f(\alpha|y) \propto \exp \left[-\frac{1}{2}(\alpha - \Omega^{-1}c)' \Omega (\alpha - \Omega^{-1}c) \right],$$

where Ω is the conditional precision of α and c is the conditional co-vector. That is, $\Omega = \Sigma^{-1}$ and $c = \Sigma^{-1}\mu$, where μ and Σ are the conditional mean and variance.

Equating $(\alpha - \Omega^{-1}c)' \Omega (\alpha - \Omega^{-1}c)$ with expression (20) and matching terms of the form $\alpha_t' A$ and $\alpha_t B \alpha_s$, with the help of (21), yields (4).

B Proof of Result 2.1

Suppose $\alpha|y \sim N(\Omega^{-1}c, \Omega^{-1})$ and define

$$\begin{aligned}\Sigma_1 &= (\Omega_{11})^{-1}, & m_1 &= \Sigma_1 c_1, \\ \Sigma_t &= (\Omega_{tt} - \Omega_{t,t-1}\Sigma_{t-1}\Omega_{t-1,t})^{-1}, & m_t &= \Sigma_t(c_t - \Omega_{t,t-1}m_{t-1}).\end{aligned}$$

Now let $\mu_n \equiv m_n$ and for $t = n-1, \dots, 1$, let $\mu_t = m_t - \Sigma_t \Omega_{t,t+1} \mu_{t+1}$. Let $\mu = (\mu'_1, \dots, \mu'_n)'$.

We first show that $\Omega\mu = c$, which means that $\mu = E[\alpha|y]$:

$$\begin{aligned}\Omega_{11}\mu_1 + \Omega_{12}\mu_2 &= \Omega_{11}(m_1 - \Sigma_1\Omega_{12}\mu_2) + \Omega_{12}\mu_2 \\ &= \Omega_{11}((\Omega_{11})^{-1}c_1 - (\Omega_{11})^{-1}\Omega_{12}\mu_2) + \Omega_{12}\mu_2 = c_1.\end{aligned}$$

For $t = 2, \dots, n-1$,

$$\begin{aligned}\Omega_{t,t-1}\mu_{t-1} + \Omega_{tt}\mu_t + \Omega_{t,t+1}\mu_{t+1} &= \Omega_{t,t-1}(m_{t-1} - \Sigma_{t-1}\Omega_{t-1,t}\mu_t) + \Omega_{tt}\mu_t + \Omega_{t,t+1}\mu_{t+1} \\ &= \Omega_{t,t-1}m_{t-1} + (\Omega_{tt} - \Omega_{t,t-1}\Sigma_{t-1}\Omega_{t-1,t})\mu_t + \Omega_{t,t+1}\mu_{t+1} \\ &= \Omega_{t,t-1}m_{t-1} + \Sigma_t^{-1}\mu_t + \Omega_{t,t+1}\mu_{t+1} \\ &= \Omega_{t,t-1}m_{t-1} + \Sigma_t^{-1}(m_t - \Sigma_t\Omega_{t,t+1}\mu_{t+1}) + \Omega_{t,t+1}\mu_{t+1} \\ &= \Omega_{t,t-1}m_{t-1} + (c_t - \Omega_{t,t-1}m_{t-1}) = c_t.\end{aligned}$$

$$\begin{aligned}\Omega_{n,n-1}\mu_{n-1} + \Omega_{nn}\mu_n &= \Omega_{n,n-1}(m_{n-1} - \Sigma_{n-1}\Omega_{n-1,n}\mu_n) + \Omega_{nn}\mu_n \\ &= \Omega_{n,n-1}m_{n-1} + \Sigma_n^{-1}\mu_n \\ &= \Omega_{n,n-1}m_{n-1} + \Sigma_n^{-1}m_n \\ &= \Omega_{n,n-1}m_{n-1} + (c_n - \Omega_{n,n-1})m_{n-1} = c_n.\end{aligned}$$

We will now prove that $E[\alpha_t|\alpha_{t+1}, \dots, \alpha_n, y] = m_t - \Sigma_t \Omega_{t,t+1} \alpha_{t+1}$ and that $\text{Var}[\alpha_t|\alpha_{t+1}, \dots, \alpha_n, y] = \Sigma_t$. We begin with (5) and note that the only non-zero elements of $\Omega_{1:t,t+1:n}$ come from $\Omega_{t,t+1}$. We can therefore write the univariate conditional distribution $\alpha_t|\alpha_{t+1:n}$ as

$$\alpha_t|\alpha_{t+1:n} \sim N(\mu_t - ((\Omega_{1:t,1:t})^{-1})_{tt}\Omega_{t,t+1}(\alpha_{t+1} - \mu_{t+1}), ((\Omega_{1:t,1:t})^{-1})_{tt}).$$

The following inductive proof establishes the result $\text{Var}[\alpha_t|\alpha_{t+1}, \dots, \alpha_n, y] = \Sigma_t$:

$$(\Omega_{11})^{-1} = \Sigma_1$$

$$\begin{aligned}
((\Omega_{1:t,1:t})^{-1})_{tt} &= (\Omega_{tt} - \Omega_{t,1:t-1}(\Omega_{1:t-1,1:t-1})^{-1}\Omega_{1:t-1,t})^{-1} \\
&= (\Omega_{tt} - \Omega_{t,t-1}\Sigma_{t-1}\Omega_{t-1,t})^{-1} = \Sigma_t.
\end{aligned}$$

As for the conditional mean,

$$E[\alpha_t | \alpha_{t+1}, \dots, \alpha_n, y] = \begin{cases} \mu_t - \Sigma_t \Omega_{t,t+1} (\alpha_{t+1} - \mu_{t+1}) & t = 1, \dots, n-1 \\ \mu_n & t = n. \end{cases}$$

By the definition of μ_t , $m_t = \mu_t + \Sigma_t \Omega_{t,t+1} \mu_{t+1}$, so we obtain

$$E[\alpha_t | \alpha_{t+1}, \dots, \alpha_n, y] = \begin{cases} m_t - \Sigma_t \Omega_{t,t+1} \alpha_{t+1} & t = 1, \dots, n-1 \\ m_n & t = n. \end{cases}$$

References

- Carter, C. K., and Kohn, R. (1994). ‘On Gibbs Sampling for State Space Models’, *Biometrika*, 81(3): 541–553.
- de Jong, P., and Shephard, N. (1995). ‘The Simulation Smoother for Time Series Models’, *Biometrika*, 82(1): 339–350.
- Durbin, J., and Koopman, S. J. (1997). ‘Monte Carlo maximum likelihood estimation for non-Gaussian state space models’, *Biometrika*, 84(3): 669–684.
- (2002). ‘A Simple and Efficient Simulation Smoother for State Space Time Series Analysis’, *Biometrika*, 89(3): 603–615.
- Frühwirth-Schnatter, S. (1994). ‘Data augmentation and Dynamic Linear Models’, *Journal of Time Series Analysis*, 15: 183–202.
- Kim, S., Shephard, N., and Chib, S. (1998). ‘Stochastic Volatility: Likelihood Inference and Comparison with ARCH Models’, *Review of Economic Studies*, 65(3): 361–393.
- Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (1992). *Numerical recipes in C*. Cambridge University Press, Cambridge, second edn., The art of scientific computing.

Vandebril, R., Mastronardi, N., and Van Barel, M. (2007). 'A Levinson-like algorithm for symmetric strongly nonsingular higher order semiseparable plus band matrices', *Journal of Computational and Applied Mathematics*, 198(1): 74–97.