Parallel Batch Kalman Filtering

Pete Bunch

July 30, 2012

1 Introduction

Sometimes we want to run a Kalman filter offline, for example when using Rao-Blackwellisation in an MCMC algorithm, such as the MCMC-DA system for target tracking. In this case, the sequential nature of the Kalman filter prevents it from being easily parallelisable. Here we try to fix that.

2 An Offline Formulation of the Kalman Filter

Consider a standard discrete-time linear-Gaussian state-space model,

$$x_n = Ax_{n-1} + w_n \tag{1}$$
$$y_n = Cx_n + v_n \tag{2}$$

$$w_n \sim \mathcal{N}(.|0, Q) \tag{3}$$
$$v_n \sim \mathcal{N}(.|0, R) \tag{4}$$

$$x_0 \sim \mathcal{N}(.|m_0, P_0). \tag{5}$$

We can stack up all N of the state and observation equations into matrix equations,

$$\underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_N \end{bmatrix}}_{X} = \underbrace{\begin{bmatrix} A \\ A^2 \\ A^3 \\ \vdots \\ A^N \end{bmatrix}}_{F} x_0 + \underbrace{\begin{bmatrix} I & 0 & 0 & \dots & 0 \\ A & I & 0 & \dots & 0 \\ A^2 & A & I & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ A^N & A^{N-1} & A^{N-2} & \dots & I \end{bmatrix}}_{G} \underbrace{\begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_N \end{bmatrix}}_{W} \tag{6}$$

$$\underbrace{\begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix}}_{Y} = \underbrace{\begin{bmatrix} C & & \\ & \ddots & \\ & & C \end{bmatrix}}_{H} \underbrace{\begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix}}_{X} + \underbrace{\begin{bmatrix} v_1 \\ \vdots \\ v_N \end{bmatrix}}_{V}. \tag{7}$$

$$W \sim \mathcal{N}\left(.|0, \underbrace{\begin{bmatrix} Q & & \\ & \ddots & \\ & & Q \end{bmatrix}}_{S}\right) \tag{8}$$

$$C \sim \mathcal{N}\left(.|0, \underbrace{\begin{bmatrix} R & & \\ & \ddots & \\ & & R \end{bmatrix}}_{T}\right). \tag{9}$$

This gives us the following distributions,

$$p(X|x_0) = \mathcal{N}(X|Fx_0, GSG^T) \tag{10}$$
$$p(Y|X) = \mathcal{N}(Y|HX, T) \tag{11}$$
$$p(x_0) = \mathcal{N}(x_0|m_0, P_0). \tag{12}$$

Now its time for Bayes and some marginalisation,

$$\begin{aligned} p(X|Y) &= \frac{p(Y|X)p(X)}{p(Y)} \\ &= \frac{p(Y|X)\int p(X|x_0)p(x_0)}{p(Y)} \\ &= \mathcal{N}(X|\mu, \Sigma). \end{aligned} \tag{13}$$

This makes use of the two following little Gaussian identities (arbitrary notation),

$$\int \mathcal{N}(y|Cx, R)\mathcal{N}(x|\mu, \Sigma)dx = \mathcal{N}(y|C\mu, C\Sigma C^T + R)$$

$$\mathcal{N}(y|Cx, R)\mathcal{N}(x|\mu, \Sigma) \propto \mathcal{N}(x|\left[C^T R^{-1} C + \Sigma^{-1}\right]^{-1}\left[C^T R^{-1} y + \Sigma^{-1}\mu\right], \left[C^T R^{-1} C + \Sigma^{-1}\right]^{-1}).$$

The moments of the posterior Gaussian are given by.

$$\Sigma = \left[(GSG^T + FP_0F^T)^{-1} + H^T T^{-1} H\right]^{-1} \tag{14}$$
$$\mu = \Sigma\left[(GSG^T + FP_0F^T)^{-1} Fx_0 + H^T T^{-1} Y\right]. \tag{15}$$

Good. μ is the vector of concatenated posterior means for each state. Σ is the complete covariance matrix for all states over time. To replicate a Kalman smoother, we only need the blocks on the diagonal of this matrix. The other blocks are the covariances between states at different times, and are less interesting.

Σ^{-1} is highly structured, and in fact we can right is out explicitly. G is square and clearly full rank (because A has to be full rank for a valid HMM),

and it turns out that its inverse has the following wizard form,

$$G^{-1} = \begin{bmatrix} I & 0 & 0 & \dots & 0 & 0 \\ -A & I & 0 & \dots & 0 & 0 \\ 0 & -A & I & \vdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & I & 0 \\ 0 & 0 & 0 & \dots & -A & I \end{bmatrix}. \tag{16}$$

Hence,

$$(GSG^T)^{-1} = G^{-T}S^{-1}G^{-1} \tag{17}$$
$$= \begin{bmatrix} Q^{-1}+A^TQ^{-1}A & -A^TQ^{-1} & 0 & \dots & 0 & 0 \\ -Q^{-1}A & Q^{-1}+A^TQ^{-1}A & -A^TQ^{-1} & \dots & 0 & 0 \\ 0 & -Q^{-1}A & Q^{-1}+A^TQ^{-1}A & \vdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & Q^{-1}+A^TQ^{-1}A & -A^TQ^{-1} \\ 0 & 0 & 0 & \dots & -Q^{-1}A & Q^{-1} \end{bmatrix}.$$

Now expand using the Woodbury identity,

$$(GSG^T + FP_0F^T)^{-1} = (GSG^T)^{-1} - (GSG^T)^{-1}F(P_0^{-1} + F^T(GSG^T)^{-1}F)^{-1}F^T(GSG^T)^{-1} \tag{18}$$

This looks horrible, but it turns out that it all collapses deliciously to almost nothing,

$$(GSG^T)^{-1}F = \begin{bmatrix} Q^{-1}A \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{19}$$

$$F^T(GSG^T)^{-1}F = A^TQ^{-1}A \tag{20}$$

$$(GSG^T)^{-1}F(P_0^{-1} + F^T(GSG^T)^{-1}F)^{-1}F^T(GSG^T)^{-1}$$
$$= \begin{bmatrix} Q^{-1}A(P_0^{-1}+A^TQ^{-1}A)^{-1}A^TQ^{-1} & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix}. \tag{21}$$

Thus we can write,

$$
\begin{aligned}
\Sigma^{-1} &= \left[(GSG^T + FP_0F^T)^{-1} + H^TT^{-1}H\right] \\
&= \begin{bmatrix}
Q^{-1} + A^TQ^{-1}A & -A^TQ^{-1} & 0 & \dots & 0 & 0 \\
-Q^{-1}A & Q^{-1} + A^TQ^{-1}A & -A^TQ^{-1} & \dots & 0 & 0 \\
0 & -Q^{-1}A & Q^{-1} + A^TQ^{-1}A & \vdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \dots & Q^{-1} + A^TQ^{-1}A & -A^TQ^{-1} \\
0 & 0 & 0 & \dots & -Q^{-1}A & Q^{-1} + A^TQ^{-1}A
\end{bmatrix} \\
&\quad + \begin{bmatrix}
-Q^{-1}A(P_0^{-1} + A^TQ^{-1}A)^{-1}A^TQ^{-1} & 0 & \dots & 0 & 0 \\
0 & 0 & \dots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \dots & 0 & 0 \\
0 & 0 & \dots & 0 & -A^TQ^{-1}A
\end{bmatrix} \\
&\quad + \begin{bmatrix}
C^TR^{-1}C & 0 & 0 & \dots & 0 & 0 \\
0 & C^TR^{-1}C & 0 & \dots & 0 & 0 \\
0 & 0 & C^TR^{-1}C & \vdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \dots & C^TR^{-1}C & 0 \\
0 & 0 & 0 & \dots & 0 & C^TR^{-1}C
\end{bmatrix}.
\end{aligned} \tag{22}
$$

Thus Σ^{-1} has a nice sparse, tridiagonal structure, composed of a Toeplitz matrix, and two diagonal matrixes.

The blocks of the inverse covariance matrix represent conditional coprecisions. The diagonal terms represent the precision of an estimate given the preceding state (which contributes Q^{-1}) the following state (which contributes $A^TQ^{-1}A$) and the corresponding observation (which contributes $C^TR^{-1}C$). This explains why all diagonal terms are the same except the final one (for which there is no next state) and the first one (for which the previous state has a different covariance). Note that if the initial state is unknown (large diagonal entries in P_0), then initial condition correction becomes $-Q^{-1}$, cancelling out the term corresponding to information from the previous state. On the other hand, if the initial state is known (small diagonal entries in P_0), then the correction term disappears.

We can also simplify the mean calculation. Consider the vector ξ:

$$\xi = \left[(GSG^T + FP_0F^T)^{-1}Fx_0 + H^TT^{-1}Y\right].$$

For $n \neq 1, N$,

$$\begin{aligned}\xi_n &= -Q^{-1}A\left(A^{n-1}m_0\right) + \left(Q^{-1} + A^TQ^{-1}A\right)\left(A^nm_0\right) - A^TQ^{-1}\left(A^{n+1}m_0\right) + C^TR^{-1}y_n \\ &= \left[-Q^{-1} + Q^{-1} + A^TQ^{-1}A - A^TQ^{-1}A\right]A^nm_0 + C^TR^{-1}y_n \\ &= C^TR^{-1}y_n. \end{aligned} \tag{23}$$

For $n = N$,

$$\begin{aligned}\xi_N &= -Q^{-1}A\left(A^{N-1}m_0\right) + Q^{-1}A^Nm_0 + C^TR^{-1}y_N \\ &= \left[-Q^{-1} + Q^{-1}\right]A^Nm_0 + C^TR^{-1}y_N \\ &= C^TR^{-1}y_N. \end{aligned} \tag{24}$$

For $n = 1$,

$$\begin{aligned}\xi_1 &= \left(Q^{-1} + A^TQ^{-1}A - Q^{-1}A(P_0^{-1} + A^TQ^{-1}A)^{-1}A^TQ^{-1}\right)(Am_0) - A^TQ^{-1}\left(A^2m_0\right) + C^TR^{-1}y_1 \\ &= \left[Q^{-1} - Q^{-1}A(P_0^{-1} + A^TQ^{-1}A)^{-1}A^TQ^{-1} + A^TQ^{-1}A - A^TQ^{-1}A\right]Am_0 + C^TR^{-1}y_1 \\ &= \left[Q^{-1} - Q^{-1}A(P_0^{-1} + A^TQ^{-1}A)^{-1}A^TQ^{-1}\right]Am_0 + C^TR^{-1}y_1 \\ &= (Q + AP_0A^T)^{-1}Am_0 + C^TR^{-1}y_1. \end{aligned} \tag{25}$$

Hence, ξ simplifies to,

$$\xi = \begin{bmatrix} C^TR^{-1}y_1 \\ C^TR^{-1}y_2 \\ \vdots \\ C^TR^{-1}y_N \end{bmatrix} + \begin{bmatrix} (Q + AP_0A^T)^{-1}Am_0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{26}$$

3 Problem Summary

We have the following matrixes,

$$
\begin{aligned}
\Sigma^{-1} &= \left[(GSG^T + FP_0F^T)^{-1} + H^TT^{-1}H\right] \\
&= \begin{bmatrix}
Q^{-1} + A^TQ^{-1}A & -A^TQ^{-1} & 0 & \dots & 0 & 0 \\
-Q^{-1}A & Q^{-1} + A^TQ^{-1}A & -A^TQ^{-1} & \dots & 0 & 0 \\
0 & -Q^{-1}A & Q^{-1} + A^TQ^{-1}A & \vdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \dots & Q^{-1} + A^TQ^{-1}A & -A^TQ^{-1} \\
0 & 0 & 0 & \dots & -Q^{-1}A & Q^{-1} + A^TQ^{-1}A
\end{bmatrix} \\
&\quad + \begin{bmatrix}
-Q^{-1}A(P_0^{-1} + A^TQ^{-1}A)^{-1}A^TQ^{-1} & 0 & \dots & 0 & 0 \\
0 & 0 & \dots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \dots & 0 & 0 \\
0 & 0 & \dots & 0 & -A^TQ^{-1}A
\end{bmatrix} \\
&\quad + \begin{bmatrix}
C^TR^{-1}C & 0 & 0 & \dots & 0 & 0 \\
0 & C^TR^{-1}C & 0 & \dots & 0 & 0 \\
0 & 0 & C^TR^{-1}C & \vdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \dots & C^TR^{-1}C & 0 \\
0 & 0 & 0 & \dots & 0 & C^TR^{-1}C
\end{bmatrix}
\end{aligned} \tag{27}
$$

$$
\xi = \begin{bmatrix} C^TR^{-1}y_1 \\ C^TR^{-1}y_2 \\ \vdots \\ C^TR^{-1}y_N \end{bmatrix} + \begin{bmatrix} (Q + AP_0A^T)^{-1}Am_0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{28}
$$

The tasks that remain are as follows:

- Calculate the diagonal blocks of Σ.
- Solve the system of equations $\Sigma^{-1}\mu = \xi$ to give us μ.

These will give us the same covariance matrixes and means generated by the Kalman filter.

4 Direct Method

We write the inverse covariance matrix as,

$$\Sigma^{-1} = \begin{bmatrix} \gamma & \beta & & & & \\ \beta^T & \alpha & \beta & & & \\ & \beta^T & \alpha & \ddots & & \\ & & \ddots & \ddots & \beta & \\ & & & \beta^T & \alpha & \beta \\ & & & & \beta^T & \delta \end{bmatrix}. \tag{29}$$

The matrix equation, $\Sigma^{-1}\mu = \xi$, can then be written as N simultaneous equations,

$$\begin{aligned} \gamma\mu_1 + \beta\mu_2 &= \xi_1 \\ \beta^T\mu_1 + \alpha\mu_2 + \beta\mu_3 &= \xi_2 \\ \beta^T\mu_2 + \alpha\mu_3 + \beta\mu_4 &= \xi_3 \\ &\vdots \\ \beta^T\mu_{N-2} + \alpha\mu_{N-1} + \beta\mu_N &= \xi_{N-1} \\ \beta^T\mu_{N-1} + \delta\mu_N &= \xi_N. \end{aligned}$$

These may be solved numerically. The first equation defines μ_2 in terms of μ_1. Using this, the second equation defines μ_3 in terms of μ_1. Progressing sequentially through the list, the $(N-1)$th and Nth equations both define μ_N in terms of μ_1. Thus we have two equations in terms of two unknowns (μ_1 and μ_N) which may be solved completely. The value for μ_1 is then substituted into the intermediate expressions to yield the entire μ vector.

This method superficially resembles a Kalman smoother, consisting of a forward pass through the data followed by a second, back-substitution phase. Unlike a Kalman smoother, the second phase is easily and fully parallelisable (i.e. it may be conducted in $\mathcal{O}(1)$ time with $> N$ processors). The first step, however, is innately sequential, and a dumb implementation would still require $\mathcal{O}(N)$ time.

We devise a parallel implementation of the first phase by repeatedly partitioning the data in a similar manner to the FFT, QuikSort, etc. Consider dividing the data into 2 sets, from $1, \ldots, K$ and from $K+1, \ldots, N$. Using the first K equations as before, we can write down relations numerically defining μ_K and μ_{K+1} in terms of μ_1. Meanwhile, we use the remaining $N-K$ equations to write down two relations numerically defining μ_N in terms of μ_K and μ_{K+1}. These two sets may be processed entirely in parallel. It is then a small task to substitute the former into the latter and hence calculate a value for μ_1. The time for the first, parallel part would be $\mathcal{O}(N/2)$ and that for the second, joining part $\mathcal{O}(1)$.

Instead of dividing the data into two sets, we could divide it into M sets. The time for the parallel part would then be $\mathcal{O}(N/M)$ and that for the second

part $\mathcal{O}(M)$. By choosing $M \approx \sqrt{N}$, we achieve a total complexity of $\mathcal{O}(\sqrt{N})$. Alternatively, we could recursively bisect the interval, leading to a complexity of $\mathcal{O}(\log_2(N))$ (I haven't checked this rigorously, but i'm pretty sure it is).

Things don't get too much more complicated for the covariance matrix,

$$\Sigma^{-1}\Sigma = I \tag{30}$$

$$\begin{bmatrix} \gamma & \beta & & & & \\ \beta^T & \alpha & \beta & & & \\ & \beta^T & \alpha & \ddots & & \\ & & \ddots & \ddots & \beta & \\ & & & \beta^T & \alpha & \beta \\ & & & & \beta^T & \delta \end{bmatrix} \begin{bmatrix} \sigma_1 & \sigma_{1,2} & \sigma_{1,3} & \dots & \sigma_{1,N} \\ \sigma_{1,2} & \sigma_2 & \sigma_{2,3} & \dots & \sigma_{2,N} \\ \sigma_{1,3} & \sigma_{2,3} & \sigma_3 & \dots & \sigma_{3,N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sigma_{1,N} & \sigma_{2,N} & \sigma_{3,N} & \dots & \sigma_{N,N} \end{bmatrix} = I.$$

We use a subset of $3N-2$ of the available equations,

$$\begin{aligned} \gamma\sigma_1 + \beta\sigma_{1,2} &= I \\ \gamma\sigma_{1,2} + \beta\sigma_2 &= 0 \\ \\ \beta^T\sigma_1 + \alpha\sigma_{1,2} + \beta\sigma_{1,3} &= 0 \\ \beta^T\sigma_{1,2} + \alpha\sigma_2 + \beta\sigma_{2,3} &= I \\ \beta^T\sigma_{1,3} + \alpha\sigma_{2,3} + \beta\sigma_3 &= 0 \\ \\ \beta^T\sigma_2 + \alpha\sigma_{2,3} + \beta\sigma_{2,4} &= 0 \\ \beta^T\sigma_{2,3} + \alpha\sigma_3 + \beta\sigma_{3,4} &= I \\ \beta^T\sigma_{2,4} + \alpha\sigma_{3,4} + \beta\sigma_4 &= 0 \\ &\vdots \\ \beta^T\sigma_{N-2} + \alpha\sigma_{N-2,N-1} + \beta\sigma_{N-2,N} &= 0 \\ \beta^T\sigma_{N-2,N-1} + \alpha\sigma_{N-1} + \beta\sigma_{N-1,N} &= I \\ \beta^T\sigma_{N-2,N} + \alpha\sigma_{N-1,N} + \beta\sigma_N &= 0 \\ \\ \beta^T\sigma_{N-1} + \delta\sigma_{N-1,N} &= 0 \\ \beta^T\sigma_{N-1,N} + \delta\sigma_N &= I. \end{aligned}$$

In the same manner as the μs, we can cascade through these numerically to get two equations in two unknowns (σ_1 and σ_N). In the above, the nth block corresponds to the nth row of Σ^{-1}, and allows us to calculate $\sigma_{n-1,n+1}$, $\sigma_{n,n+1}$ and σ_{n+1} (excepting the first and last blocks). The last block gives us a second constraint for σ_N. A similar back substitution phase then follows.

This procedure is $\mathcal{O}(N)$ (or rather $\mathcal{O}(3N)$ to be precise) but it may be parallelised in the same way as the mean calculation. To start at the $(K+1)$th step, we need to assume σ_K, σ_{K+1} and $\sigma_{K,K+1}$ are known.

5 Fourier Transform Method

Rich has details of this. Its efficacy depends on whether the parallel FFT is $\mathcal{O}(N)$ or $\mathcal{O}(\log_2(N))$, which we need to investigate.