Fast Kalman filtering via a low-rank perturbative approach

Liam Paninski, Kamiar Rahnama Rad, and Jonathan Huggins
Department of Statistics and Center for Theoretical Neuroscience
Columbia University
http://www.stat.columbia.edu/∼liam

September 23, 2011

Abstract

Kalman filtering is a fundamental tool in statistical time series analysis: it is computationally tractable in many real-world situations, implements the optimal Bayesian filter in the linear-Gaussian setting, and serves as a key step in the inference algorithms for a wide variety of nonlinear and non-Gaussian models. However, standard implementations of the Kalman filter require $O(N^3)$ time and $O(N^2)$ space per timestep, where N is the dimension of the state variable, and are therefore impractical in many high-dimensional problems. In this paper we note that if a relatively small number of observations with low signal-to-noise (SNR) are available per time step, the Kalman equations may be approximated in terms of a low-rank perturbation of the prior state covariance matrix in the absence of any observations. In many cases this approximation may be computed and updated very efficiently (often in just $O(N)$ or $O(N \log N)$ time and space per timestep), using fast methods from numerical linear algebra. This opens up the possibility of real-time adaptive experimental design and optimal control in systems of much larger dimension than was previously feasible. We describe an application involving smoothing of spatiotemporal neuroscience data.

Introduction

Understanding the dynamics of large systems for which limited, noisy observations are available is a fundamental and recurring scientific problem. A key step in any such analysis involves data assimilation: we must incorporate incoming observations and update our beliefs about the dynamical state of the system accordingly. The Kalman filter may be considered the canonical method for data assimilation (Durbin and Koopman, 2001); this method provides a conceptually simple recursive framework for online Bayesian inference in the context of linear and Gaussian dynamics and observation processes. Furthermore, the Kalman filter serves as the underlying computational engine in a wide variety of more complicated non-Gaussian and nonlinear statistical models.

However, these methods face a major limitation: standard implementations of the Kalman filter require $O(N^3)$ time and $O(N^2)$ space per timestep, where N denotes the dimension of the system state variable, and are therefore impractical for applications involving very high-dimensional systems. The bottleneck is in the representation and computation of the forward covariance matrix $C_t = C(q_t|Y_{1:t})$: this is the posterior covariance of the N-dimensional state vector q_t, given the sequence of observations $Y_{1:t}$ up to the current time t. Two natural ideas for reducing the computational burden of storing and computing this $N \times N$ matrix have been

explored. First, if C_t is sparse (i.e., consists of mostly zeros), then we can clearly store and perform matrix-vector computations with C_t with $o(N^2)$ complexity. In many examples C_t has a nearly banded, or strongly tapered, structure (i.e., most of the large components of C_t are near the diagonal), and sparse approximate matrix updates can be exploited (Pnevmatikakis et al., 2011). This approach has been shown to be extremely effective in some cases (Furrer and Bengtsson, 2007; Khan and Moura, 2008; Bickel and Levina, 2008; Kaufman et al., 2008; El Karoui, 2008), but in many settings there is no a priori reason to expect C_t to have any useful sparse structure, and therefore this idea can not be applied generally.

Second, we could replace C_t with a low-rank approximation. For example, a major theme in the recent literature on numerical weather prediction (where the system of interest is the atmosphere discretized in a spatial grid, leading in many cases to a state dimension in the tens or hundreds of millions) has been the development of the theory of the "ensemble Kalman filter" (Verlaan, 1998; Treebushny and Madsen, 2005; Gillijns et al., 2006; Chandrasekar et al., 2008; Evensen, 2009), which implements a Monte Carlo-based, low-rank approximation of the full Kalman filter. Randomized low-rank approximations of large matrices have also received increasing attention in the applied math literature (Liberty et al., 2007; Halko et al., 2009). Low-rank approximations for C_t are typically justified on computational grounds — in some cases it is hard to think of any alternative approaches for storing (even approximately) a general large matrix without any sparse structure — but may also be justified statistically in the case that many high-signal-to-noise (high-SNR) observations are available: in this setting, we can argue that our posterior uncertainty C_t will be approximately restricted to a subspace of dimension significantly less than N, as discussed, e.g., in (Solo, 2004). Alternatively, we may impose a low-rank structure on the posterior covariance C_t directly by choosing our prior covariance matrix to be of low rank (Wikle and Cressie, 1999; Wood, 2006; Cressie and Johannesson, 2008; Banerjee et al., 2008; Cressie et al., 2010); however, our focus in this work is on approximating C_t given a prior covariance matrix which is of full rank.

The low-SNR setting, where a relatively small number of noisy observations are available per time step, has been explored less thoroughly. One exception is the neuronal dendritic application discussed in (Paninski, 2010), where we noted that C_t could be approximated very accurately in terms of a low-rank perturbation of C_0, the prior equilibrium covariance of the state variable q_t in the absence of any observations Y. (Note that this approximation is very different from the high-SNR case, where we approximate C_t as a low-rank perturbation of the zero matrix, not of C_0.) To efficiently update this low-SNR approximation to C_t, (Paninski, 2010) exploited the special structure of the dynamics in this application: dendritic voltage dynamics are governed by a cable equation on a tree (Koch, 1999), which may be solved using symmetric sparse matrix methods in $O(N)$ time (Hines, 1984). In turn, this implied that C_t could be updated in $O(n^3 + nN)$ time, where n is the rank of the perturbation of C_0 used to represent C_t. Since empirically $n \ll N$ sufficed to accurately approximate C_t in this application, this approach resulted in a much faster implementation of the Kalman filter, with linear instead of cubic complexity in N.

In this paper we note that this basic idea can be applied much more generally. We describe a number of examples where special features of the system dynamics allow us to compute and update the low-SNR approximation to C_t very efficiently (often in just $O(n^3+nN)$ or $O(n^3+nN\log N)$ time and $O(nN)$ space per timestep), using fast methods from numerical linear algebra. One particularly simple setting involves spatiotemporal smoothing applications; as a concrete example, we describe how to apply the proposed methods to efficiently smooth certain kinds of high-dimensional spatiotemporal neuroscience data.

Basic setup

We begin by briefly reviewing the Kalman filter and establishing notation. Again, let q_t denote our state variable, and y_t the observation at time t. We assume that q_t and y_t satisfy the following linear-Gaussian dynamics and observation equations:

$$q_{t+1} = Aq_t + u_t + \epsilon_t,\ \epsilon_t \sim \mathcal{N}(0, V) \quad (1)$$

$$y_t = B_t q_t + \eta_t,\ \eta_t \sim \mathcal{N}(\mu_t^\eta, W_t). \quad (2)$$

Here A represents the system dynamics matrix; u_t is a deterministic input to the system at time t, and ϵ_t is an i.i.d. Gaussian vector with mean zero and covariance V. B_t denotes the observation gain matrix, W_t the observation noise covariance, and μ_t^η an offset mean in the observation. Our methods are sufficiently general that the dimension of y_t (and therefore that of B_t, W_t, and μ_t^η) can vary with time. (Nonlinear and non-Gaussian observations may also be incorporated in some cases, as we will discuss further below.) However, neither A nor V in the dynamics equation are allowed to vary with time.

Now the focus of this paper is the efficient implementation of the Kalman filter recursion (Durbin and Koopman, 2001) for computing the forward mean

$$\mu_t = E(q_t | Y_{1:t})$$

and covariance

$$C_t = Cov(q_t | Y_{1:t}),$$

where $Y_{1:t}$ denotes all of the observed data $\{y_s\}$ up to time t. The Kalman recursions may be written as:

$$C_t = \left[(AC_{t-1}A^T + V)^{-1} + B^T W^{-1} B\right]^{-1}$$

$$\mu_t = C_t \left[(AC_{t-1}A^T + V)^{-1} (A\mu_{t-1} + u_t) + B^T W^{-1}(y_t - \mu_t^\eta)\right].$$

We have suppressed the possible time-dependence of B and W in the observation equation for notational clarity; the extension of these equations to the general case is standard (Durbin and Koopman, 2001). Note that the computation of the inverses in the recursion for C_t requires $O(N^3)$ time in general, or $O(N^2)$ time via the Woodbury lemma (Golub and Van Loan, 1996) if the observation matrix B is of low rank (i.e., if $rank(B) \ll N$). In either case, $O(N^2)$ space is required to store C_t.

These recursions are typically initialized with the marginal equilibrium covariance (i.e., the steady-state covariance of q_t in the absence of any observations):

$$C_0 = \lim_{t \to \infty} Cov(q_t).$$

Here we may state our first basic assumption, namely that the dynamics matrix A is stable (Anderson and Moore, 1979); otherwise the existence of C_0 is not guaranteed. Note that we restrict our attention in this paper to the case of stationary processes; extensions to nonstationary models are possible (Pnevmatikakis and Paninski, 2011), but will not be discussed here.

This equilibrium covariance C_0 satisfies the discrete Lyapunov equation

$$AC_0A^T + V = C_0; \quad (3)$$

this is just the Kalman recursion for C_t above, solving for $C_t = C_{t-1}$ in the special case that $B = 0$ (i.e., no observations are available). This equation can be solved explicitly in many cases (Anderson and Moore, 1979), as we discuss briefly now.

Applying the standard moving-average recursion (Brockwell and Davis, 1991) for the autoregressive model q_t leads to

$$C_0 = \sum_{i=0}^{\infty} A^i V (A^T)^i \tag{4}$$

(again, under suitable conditions to guarantee that this series has a finite limit). If the dynamics equation A commutes with the dynamics noise covariance V, and A is normal (i.e., $AA^T = A^T A$), this reduces to the explicit solution

$$C_0 = V \sum_{i=0}^{\infty} (AA^T)^i = V(I - AA^T)^{-1}.$$

More generally, if V and A do not commute then we can employ the (linear) whitening change of variables $x_t = V^{-1/2} q_t$ (assuming V is of full rank). Abbreviating the symmetric matrix square root $E = V^{1/2}$ and defining the reparameterized covariance matrix C_0' via $C_0 = EC_0'E$, we rewrite the Lyapunov equation as

$$EE^{-1}AEC_0'EA^TE^{-1}E + EE = EC_0'E.$$

Pre- and post-multiplying by E^{-1} gives

$$(E^{-1}AE)C_0'(E^{-1}AE)^T + I = C_0';$$

if we define A_V through the similarity transformation $A_V = E^{-1}AE$, assume A_V is normal, and argue as above, we find that

$$C_0' = \left(I - (E^{-1}AE)(E^{-1}AE)^T \right)^{-1} = \left(I - A_V A_V^T \right)^{-1},$$

so

$$C_0 = E \left(I - A_V A_V^T \right)^{-1} E.$$

The case that V is of reduced rank, or that the resulting A_V is non-normal, appears to be more difficult, as noted in more detail in the Discussion section below.

Fast method

Now the basic idea is that, in low-SNR conditions, C_t should be close to C_0: i.e., we should be able to represent the time-varying covariance C_t as a small perturbation about the steady-state solution C_0, in some sense. Thus, more concretely, we will approximate C_t as

$$C_t \approx C_0 + U_t D_t U_t^T, \tag{5}$$

where $U_t D_t U_t^T$ is a low-rank matrix we will update directly. We will now show that it is straightforward to compute and update the perturbations U_t and D_t efficiently whenever fast methods are available to solve linear equations involving A and C_0.

But first, why does the approximation in eq. (5) make sense? It is easy to see, using the Woodbury matrix lemma, that if we make k observations at time $t = 1$ then eq. (5) will hold exactly, for U_1 of rank k. If we make no further observations, then C_t follows the simple update rule

$$\begin{aligned} C_t &= AC_{t-1}A^T + V \\ &= A[C_0 + U_{t-1}D_{t-1}U_{t-1}^T]A^T + V \\ &= C_0 + AU_{t-1}D_{t-1}U_{t-1}^T A^T; \end{aligned}$$

the first equality is just the standard Kalman update in the case that $B = 0$, and the third equality follows from eq. (3). Iterating, we see that

$$C_t = C_0 + A^{t-s}U_s D_s U_s^T (A^{t-s})^T,$$

where s denotes the time of the last available observation. Since A is assumed to be stable, this implies that the perturbation to C_t around the equilibrium covariance C_0 caused by the observations up to time s will decay exponentially; for $t - s$ sufficiently large, we can discard some dimensions of the perturbation $A^{t-s}U_s D_s U_s^T (A^{t-s})^T$ without experiencing much error in C_t. In the case that additional observations become available with each timestep t, a similar phenomenon governs the behavior of C_t: long-ago observations are eventually "forgotten," due to the exponential decay caused by the double multiplication AC_tA^T. We may exploit this exponential decay by discarding some dimensions of $C_t - C_0$ as they become sufficiently small, and if the observations are sufficiently low-rank and low-SNR relative to the decay rate imposed by A, then the effective rank of $C_t - C_0$ will remain small. Conversely, if the observation information matrix $B^T W^{-1} B$ is large relative to the decay rate imposed by A, then the effective rank of $C_t - C_0$ can become large, and in this case the approximations developed here will no longer be useful[1].

This intuition is quantified in Fig. 1. If we compare C_t to C_0 by computing the spectrum of $C_0^{-1/2} C_t C_0^{-1/2}$, we see that only a small fraction of the eigenvalues of $C_0^{-1/2} C_t C_0^{-1/2}$ are significantly different from one. Thus $C_0^{-1/2} C_t C_0^{-1/2}$ may be approximated as a low-rank perturbation of the identity matrix, or equivalently, C_t may be approximated as a low-rank perturbation of C_0. The effective rank of this perturbation depends on the decay rate imposed by A and on the SNR of the observations, specifically on the scale of the observation noise W and on the dimension of the observation vector y_t: a larger $B^T W^{-1} B$ leads to a larger effective rank of $I - C_0^{-1/2} C_t C_0^{-1/2}$. In some cases we can quantify this dependence explicitly. For example, in the appendix we note that the solution to the Riccati equation for the limiting covariance in the presence of time-invariant observation matrices B and W may be computed in terms of an expansion in the observation information matrix $B^T W^{-1} B$, in the low-SNR limit that $B^T W^{-1} B$ is small. For example, in the simplest case, that A, V, and B commute, $C_\infty = C_0 + C^1 + o(B^T W^{-1} B)$, where the matrix C^1 has the same rank as B, is of the same order as $(B^T W^{-1} B)$, and may be computed explicitly. See the appendix for further details and discussion.

Now we can describe a method for efficiently updating U_t and D_t. We will use A and C_0 in what follows; it is easy to substitute the transformed matrices A_V and C_0' (defined in the

[1] It seems reasonable to expect that this qualitative explanation in terms of exponential decay of error could be developed further into a more quantitative stability theory that could provide bounds on the error generated by discarding some dimensions of $C_t - C_0$; we leave such a development for future work.

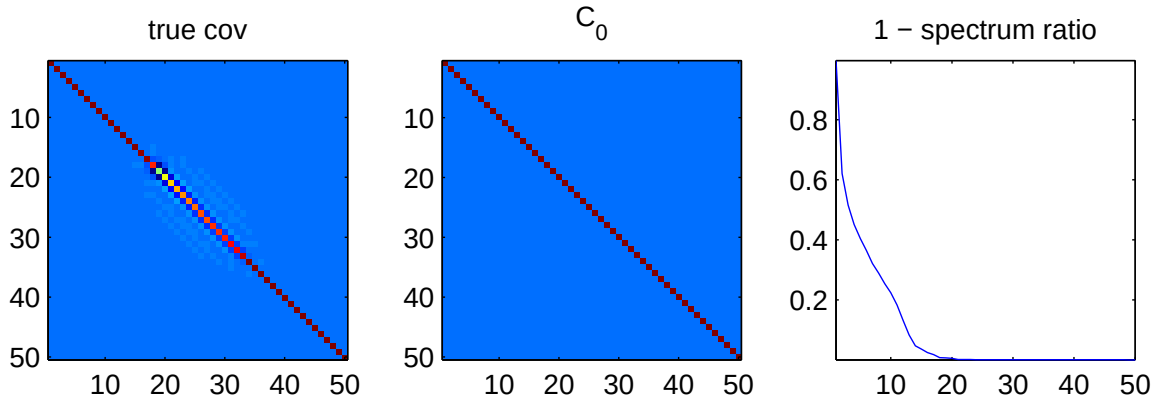

Figure 1: C_t is fairly close to C_0; in particular, $I - C_0^{-1/2} C_t C_0^{-1/2}$ has low effective rank. Left: true C_t. Middle: C_0. Both C_0 and C_t are plotted on the same colorscale, to facilitate direct comparison. Right: eigenvalue spectrum of $I - C_0^{-1/2} C_t C_0^{-1/2}$; an approximation of rank about 20 would seem to suffice for $I - C_0^{-1/2} C_t C_0^{-1/2}$ here. This C_t and C_0 were extracted from $t = 200$ in the "place field" simulation discussed in more depth in Figs. 2 and 3, below.

previous section) if necessary. First, as above, write

$$\begin{aligned} (AC_{t-1}A^T + V)^{-1} &= (A[C_0 + U_{t-1}D_{t-1}U_{t-1}^T]A^T + V)^{-1} \\ &= (C_0 + AU_{t-1}D_{t-1}U_{t-1}^T A^T)^{-1}. \end{aligned}$$

Next apply the Woodbury matrix lemma:

$$\begin{aligned} (C_0 + AU_{t-1}D_{t-1}U_{t-1}^T A^T)^{-1} &= C_0^{-1} - C_0^{-1}AU_{t-1}(D_{t-1}^{-1} + U_{t-1}^T A^T C_0^{-1} AU_{t-1})^{-1} U_{t-1}^T A^T C_0^{-1} \\ &= C_0^{-1} - R_t Q_t R_t^T, \end{aligned}$$

where we have abbreviated

$$R_t = C_0^{-1} AU_{t-1}$$

and

$$Q_t = (D_{t-1}^{-1} + U_{t-1}^T A^T C_0^{-1} AU_{t-1})^{-1}.$$

Now plug this into the covariance update:

$$\begin{aligned} C_t &= \left[(AC_{t-1}A^T + V)^{-1} + B^T W^{-1} B\right]^{-1} \\ &= \left[C_0^{-1} - R_t Q_t R_t^T + B^T W^{-1} B\right]^{-1}. \end{aligned}$$

We see that the update is of low-rank form. To apply Woodbury again, we just need to simplify the term $-R_t Q_t R_t^T + B^T W^{-1} B$. Choose an orthogonal basis

$$O_t = orth([R_t \;\; B])$$

and then write

$$-R_t Q_t R_t^T + B^T W^{-1} B = O_t M_t O_t^T,$$

with

$$M_t = -O_t^T R_t Q_t R_t^T O_t + O_t^T B^T W^{-1} B O_t.$$

Now, finally, apply Woodbury again[2]:

$$\begin{aligned} C_t &= \left[C_0^{-1} - R_t Q_t R_t^T + B^T W^{-1} B\right]^{-1} \\ &= \left[C_0^{-1} + O_t M_t O_t^T\right]^{-1} \\ &= C_0 - C_0 O_t (M_t^{-1} + O_t^T C_0 O_t)^{-1} O_t^T C_0. \end{aligned} \tag{6}$$

In practice, we find that occasionally M_t is itself poorly-conditioned (i.e., it is effectively of low rank). In this case, we approximate $M_t \approx G_t H_t G_t^T$, where H_t is a square matrix containing the eigenvalues of M_t that are above some tolerance value, and G_t is the corresponding eigenvector matrix; then we apply the Woodbury lemma directly to $C_0^{-1} + (O_t G_t) H_t (O_t G_t)^T$, instead of $C_0^{-1} + O_t M_t O_t^T$.

We obtain U_t and D_t by truncating the SVD of the expression on the right-hand side of equation (6): in Matlab, for example, we write

$$[U', D'] = svd(C_0 O_t (M_t^{-1} + O_t^T C_0 O_t)^{-1/2}, `econ'),$$

then choose U_t as the first n columns of U' and D_t as the negative square of the first n diagonal elements D', where n is chosen to be large enough (for accuracy) and small enough (for computational tractability). We have found that a reasonable choice of n is as the least solution of the inequality:

$$\sum_{i \le n} |D_{ii}| \ge c \sum_i |D_{ii}|; \tag{7}$$

i.e., choose n to capture at least a large fraction c of the variance in the right-hand term perturbing C_0 in equation (6).

Now the update for μ_t is relatively standard:

$$\begin{aligned} \mu_t &= C_t \left[(A C_{t-1} A^T + V)^{-1} m_t + B^T W^{-1} (y_t - \mu_t^\eta)\right] \\ &= (P_t^{-1} + B^T W^{-1} B)^{-1} \left[P_t^{-1} m_t + B^T W^{-1} (y_t - \mu_t^\eta)\right] \\ &= (P_t - P_t B^T (W + B P_t B^T)^{-1} B P_t) \left[P_t^{-1} m_t + B^T W^{-1} (y_t - \mu_t^\eta)\right] \\ &= m_t - P_t B^T (W + B P_t B^T)^{-1} B \left[s_t + m_t\right] + s_t, \end{aligned}$$

where we have made the abbreviations

$$P_t = C_0 + A U_{t-1} D_{t-1} U_{t-1}^T A^T, \tag{8}$$

$$m_t = A \mu_{t-1} + u_t, \tag{9}$$

and

$$s_t = P_t B^T W^{-1} (y_t - \mu_t^\eta).$$

[2]It is well-known that the Woodbury formula can be numerically unstable when the observation covariance W is small (i.e., the high-SNR case). It should be straightforward to derive a low-rank square-root filter (Howard and Jebara, 2005; Treebushny and Madsen, 2005; Chandrasekar et al., 2008) to improve the numerical stability here, though we have not yet pursued this direction. Meanwhile, a crude but effective method to guarantee that C_t remains positive definite is to simply shrink D_t slightly if any negative eigenvalues are detected. This can be done easily in $O(N)$ time by restricting attention to the subpsace spanned by U_t.

Note that we update the mean μ_t first, then truncate U_t and D_t.

To review, we have introduced some simple low-rank recursions for U_t, D_t, and μ_t in terms of C_0 and A. These recursions may be defined in the original parameterization or in the whitened representation, in which case we use A_V in place of A and C_0' in place of C_0. The key point is that C_0 or C_0^{-1} need never be computed explicitly; instead, all we need is to multiply by A and multiply and divide by C_0 or C_0^{-1}, whichever is easiest (by "divide," we mean to solve equations of the form $C_0 v = r$ for the unknown vector v and known vector r). The orthogonalization and SVD steps require $O(n^3)$ time, assuming $n \ll N$, while all the other steps involve $O(n)$ matrix-vector multiplications or divisions by C_0. Thus, if $K(N)$ denotes the cost of a single matrix-vector multiplication or division by C_0, the computational complexity of each low-rank update is $O(n^3 + nK(N))$. In many cases of interest (see below) $K(N) = o(N^2)$, and therefore the low-rank method is significantly faster than the standard Kalman recursion for large N.

We close this section by noting that the posterior marginal variance difference $[C_t - C_0]_{ii}$ can be computed in $O(nN)$ time given U_t and D_t, since computing the diagonal of $C_t - C_0$ just requires us to sum the squared elements of $(-D_t)^{1/2} U_t$. This quantity is useful in a number of contexts (Huggins and Paninski, 2010). In addition, the method can be sped up significantly in the special case that B and W are time-invariant: in this case, C_t will converge to a limit (as an approximate solution of the corresponding Riccati equation), and we can stop recomputing U_t and D_t on every time step. More generally, if B_t and W_t are time-varying in a periodic manner, then C_t will also be periodic, and we can store a period's worth of U_t and D_t in memory, instead of continuing to recompute these on each timestep.

Full forward-backward smoothing

So far we have focused on the forward problem of computing estimates of q_t given the data available up to time t, i.e., $E(q_t|Y_{1:t})$ and $Cov(q_t|Y_{1:t})$. To incorporate all of the available information $Y_{1:T}$ (not just $Y_{1:t}$), we need to perform a backward recursion. Two methods are available: we can use the Kalman backward smoother (Shumway and Stoffer, 2006), which provides both $E(q_t|Y_{1:T})$ and $Cov(q_t|Y_{1:T})$, or a version of the Thomas recursion for solving block-tridiagonal systems (Press et al., 1992), which is slightly faster but only provides the estimated mean $E(q_t|Y_{1:T})$.

Both recursions can be adapted to our low-rank setting. In the Kalman backward smoother we can approximate $Cov(q_t|Y_{1:T}) \approx C_0 + U_t^s D_t^s (U_t^s)^T$, for an appropriately chosen low-rank matrix $U_t^s D_t^s (U_t^s)^T$, which can be updated efficiently using methods similar to those we have described here for the forward low-rank approximation $C_0 + U_t D_t U_t^T$; see (Huggins and Paninski, 2010) for full details. To derive an efficient low-rank block-Thomas approach, first we recall that the output of Kalman filter-smoother, $E(q_t|Y_{1:T})$, may be written as the solution to a block-tridiagonal linear system (Fahrmeir and Kaufmann, 1991; Paninski et al., 2010). Close inspection of the standard Thomas recursion applied to this block-tridiagonal system reveals that the key step involves repeated multiplications by a large matrix which turns out to be the identity in the limit that the observation information matrix $B^T W^{-1} B$ tends to zero. More generally, in the case that $B^T W^{-1} B$ is nonzero but small and low-rank, we can replace this identity matrix with an approximate matrix of the form $I + Z_t$, where Z_t is a low-rank matrix which can again be updated efficiently using methods similar to those discussed here. See (Huggins and Paninski, 2011) for full details on this low-rank block-Thomas approach. For either approach (backward Kalman or block-Thomas), the computational complexity scales

as $O(n^3 + nK(N))$, as in the forward Kalman case discussed above.

Examples for which the proposed fast methods are applicable

There are many examples where the required manipulations with A and C_0 are relatively easy. The following list is certainly non-exhaustive. As before, we assume A is normal.

First, if A or its inverse is banded (or tree-banded, in the sense that $A_{ij} \neq 0$ only if i and j are neighbors on a tree) then so is C_0^{-1}, and multiplying and dividing by C_0 costs just $O(N)$ time and space per timestep, via either the junction tree algorithm from the theory of Markov random fields (Jordan, 1999; Weiss and Freeman, 2001; Shental et al., 2008) or approximate minimum degree reparameterizations of C_0 (Rue and Held, 2005; Davis, 2006), as explained in (Paninski, 2010).

Second, in many cases A is defined in terms of a partial differential operator. (Again, the example discussed in (Paninski, 2010) falls in this category; the evolution of the voltage on the dendritic tree is governed by a cable equation on a tree (Hines, 1984; Koch, 1999).) A in these cases is typically sparse and has a specialized local structure; multiplication by A and C_0^{-1} requires just $O(N)$ time and space. In many of these cases multigrid methods or other specialized PDE solvers can be used to divide by C_0^{-1} in $O(N)$ time and space (Briggs et al., 2000). As one specific example, multigrid methods are well-established in electroencephalographic (EEG) and magnetoencephalographic (MEG) analysis (Wolters, 2007; Lew et al., 2009), and therefore could potentially be utilized to significantly speed up the Kalman-based analyses described in (Long et al., 2006; Galka et al., 2008; Freestone et al., 2011).

Third, A will have a Toeplitz (or block-Toeplitz) structure in many physical settings, for example whenever the state variable q_t has a spatial structure and the dynamics are spatially-invariant in some sense. Multiplication by A and C_0^{-1} via the fast Fourier transform (FFT) requires just $O(N \log N)$ time and space in these cases. Similarly, division by C_0^{-1} can be performed iteratively via preconditioned conjugate gradient descent, which in many cases again requires $O(N \log N)$ time and space (Chan and Ng, 1996). Of course, if A is circulant then FFT methods may be employed directly to multiply and divide by C_0 in $O(N \log N)$ time and space (Press et al., 1992).

In all of these cases, block structure in A may be exploited easily, since the transpose and product involved in the construction of C_0 will preserve this block structure. Kronecker structure in AA^T may often be exploited easily, by the mixed-product and distributive properties of the Kronecker product. Of course, there are many other specialized matrix forms (sparse H-matrices (Hackbusch and Khoromskij, 2000), multipole operators (Memarsadeghi et al., 2008), etc.) for which fast numerical methods are available. Finally, it is worth noting that parallelization is a major theme in modern numerical analysis; many specialized parallel algorithms, with even faster scaling (depending on the number of available processing cores) have been devised for the cases discussed above.

Application to smoothing

Now for the main statistical examples we have in mind. In many statistical settings, the dynamics matrix A and noise covariance V are not directly defined; the analyst has some flexibility in choosing these matrices according to criteria including physical realism and computational tractability. Perhaps the simplest approach is to use a separable prior, defined

most easily as follows. Let $A = aI$, $0 < a < 1$. Now

$$C_0 = (1 - a^2)^{-1} V;$$

thus it is clear that when it is easy to multiply and divide by V, we may apply the fast methods discussed above with no modifications. Note that in this case the prior covariance of the vector $Q = [q_1 \; q_2 \; \ldots \; q_T]$ is separable:

$$cov(Q) = C_0 \otimes C_{AR},$$

where $\otimes$ denotes the Kronecker product and C_{AR} denotes the covariance of the standardized one-dimensional AR(1) process, $q_{t+1} = aq_t + \sqrt{1 - a^2}\epsilon_t, \epsilon_t \sim \mathcal{N}(0, 1)$. Note that the posterior covariance $cov(Q|Y)$ is not separable in general, which complicates exact inference.

It is straightforward to construct more interesting nonseparable examples. For example, in many cases we may choose a basis so that V and A are diagonal and the transformation back to the "standard" basis is fast. Examples include the discrete Fourier basis, common spline bases (Green and Silverman, 1994; Wood, 2006), and wavelet bases (Daubechies, 1992; Walnut, 2002). Now the interpretation is that each basis element is endowed with an AR(1) prior: the (i, i)-th element of A defines the temporal autocorrelation width of the i-th process, while the elements of the diagonal matrix $(I - A^2)^{-1}V$ set the processes' prior variance (and therefore $(I - A^2)^{-1}V$ expressed in the "standard" basis sets the prior covariance C_0). The difficulty in applying the standard Kalman recursion in this setting is that if B is not also diagonal in this representation, then direct implementations of the Kalman filter require $O(N^3)$ time per timestep, since C_t does not remain diagonal in general. Nonetheless, the fast low-rank smoother may be applied in a straightforward manner in this setting: computing $E(q_t|Y)$ and $Cov(q_t|Y)$ requires $O(n^3 + nN)$ time, to which we add the time necessary to transform back into the standard basis.

A further speedup is possible in this diagonal case, if the observation matrices B_t are sparse; i.e., if each observation y_t only provides information about a few elements of the state vector q_t. This setting arises frequently in environmental applications, for example, where a just a few sampling stations are often available to take spatially-localized samples of large spatiotemporal processes of interest (Stroud et al., 2001). Another example, from neuroscience, will be discussed in the following section. If I_t denotes the set of indices for which B_s is nonzero for $s \leq t$, then it is easy to show that the forward covariance C_t matrix need only be evaluated on the $|I_t| \times |I_t|$ submatrix indexed by I_t; if i or j are not in I_t, then $[C_t]_{ij} = [C_0]_{ij}$. Thus, we need only update the low-rank matrix U_t at the indices I_t, reducing the computational complexity of each update from $O(n^3 + nN)$ to $O(n^3 + n|I_t|)$. Clearly, with each new update at time t, we will add some elements to I_t, but we can also discard some elements as we go because our low-rank updates will effectively "forget" information as time progresses, as discussed above. (In particular, the indices for which the recent observations provide no information will eventually be dropped.) Thus in practice $|I_t|$ often remains much smaller than N, leading to a significant speedup.

The fast low-rank methods can also greatly facilitate the selection of hyperparameters in the smoothing setting: typically the data analyst will need to set the scale over which the data are smoothed, both temporally and spatially, and we would often like to do this in a data-dependent manner. There are a number of standard approaches for choosing hyperparameters, including cross-validation, generalized cross-validation, expectation-maximization, and maximum marginal likelihood or empirical Bayes methods. In all of these cases, it is

clearly beneficial to be able to compute the estimate more rapidly for a variety of hyperparameter settings. In addition, the output of the filter-smoother is often a necessary ingredient in hyperparameter selection. For example, the standard expectation-maximization method of (Shumway and Stoffer, 2006) can be easily adapted to the low-rank setting: we have already discussed the computation of the sufficient statistics $E(q_t|Y)$ and $Cov(q_t|Y)$, and the remaining sufficient statistics $E(q_t q_{t+1}^T|Y)$ follow easily. Similarly, a straightforward application of the low-rank determinant lemma allows us to efficiently compute the marginal log-likelihood $\log p(Y)$, via a simple adaptation of the standard forward recursion for the loglikelihood in the Kalman filter model (Rabiner, 1989).

Two neuroscience examples

To make these ideas more concrete, we now examine two examples from neuroscience. For our first example we consider neurons in the rodent hippocampal brain region; many of these neurons respond selectively depending on the animal's current location. This spatial dependency can be summarized in terms of a "place field" $f(x)$, where $f(x)$ is the expected response of the neuron (quantified by the number of action potentials emitted by the neuron in a fixed time interval), given that the animal is located at position x. It is known that these place fields can in some cases change with time; in this case we might replace $f(x)$ with $f(x,t)$. These time-varying place fields $f(x,t)$ are often represented as a sum of some fixed spatial basis functions (Brown et al., 2001; Frank et al., 2002; Czanner et al., 2008):

$$f(x,t) = \sum_i q_{it} f_i(x). \tag{10}$$

For example, the basis $\{f_i(x)\}$ could consist of spline functions defined on the spatial variable x. Now we place a prior on how the weights q_{it} evolve with time. In the simplest case, q_{it} could evolve according to independent AR(1) processes; as emphasized above, this means that the dynamics matrix A is diagonal. Now the observation model in this setting may be taken to be $y_t = f(x_t,t) + \eta_t$, with η_t denoting an i.i.d. Gaussian noise source, or we can use a slightly more accurate Poisson model, $y_t \sim Poiss\{f(x_t,t)\}$, where in either case x_t represents the (known) location of the animal as a function of time t, and $Poiss\{\lambda_t\}$ denotes a Poisson process with rate λ_t. (More detailed models are possible, of course (Czanner et al., 2008; Rahnama Rad and Paninski, 2010), but this basic formulation is sufficient to illustrate the key points here.) So the observation matrix B_t is just a N-dimensional vector, $B_{it} = f_i(x_t)$, if we use N basis vectors to represent the place field f. Computing B_t requires at most $O(N)$ time; if the basis functions f_i have compact support, then B_t will be sparse (i.e., computable in $O(1)$ time), and we can employ the speedup based on the sparse index vector I_t described above.

A second example comes from sensory systems neuroscience. The activity of a neuron in a sensory brain region depends on the stimulus which is presented to the animal. The activity of a visual neuron, for example, is typically discussed using the notion of a "receptive field," which summarizes the expected response of the neuron as a function of the visual stimulus presented to the eye. We can use a similar model structure to capture these stimulus-dependent responses; for example, we might model $y_t = s_t^T f^t + \eta_t$ in the Gaussian case, or $y_t \sim Poiss\{\exp[s_t^T f^t]\}$ in the Poisson case, where s_t is the sensory stimulus presented to the neuron at time t, $s_t^T f^t = \sum_x s(x,t) f(x,t)$ denotes the linear projection of the stimulus s_t onto the receptive field f^t at time t, and $f(x,t)$ is proportional to the expectation of y_t given that

a light of intensity $s(x,t)$ was projected onto the retina at location x. (The exponential link function exp[.] in this Poisson regression model can be replaced easily in this context with any function which is convex and whose logarithm is concave; see (Paninski, 2004; Paninski et al., 2007) for further discussion.) As indicated by the notation f^t, these receptive fields can in many cases themselves vary with time, and to capture this temporal dependence it is common to use a weighted sum of basis functions model, as in equation (10). This implies that the observation matrix B_t can be written as $B_t = s_t^T F$, where the i-th column of the basis matrix F is given by f_i. If the basis functions f_i are Fourier or wavelet functions, then the matrix-vector multiplications $s_t^T F$ can be performed in $O(N \log N)$ time per timestep; if f_i are compactly supported, F will be sparse, and computing $s_t^T F$ requires just $O(N)$ time.

Now in each of these settings the low-SNR Kalman filter is easy to compute. In the case of Gaussian observation noise η_t we proceed exactly as described above; in the Poisson case we can employ well-known extensions of the Kalman filter described, for example, in (Fahrmeir and Kaufmann, 1991; Fahrmeir and Tutz, 1994; Brown et al., 1998; Paninski et al., 2010); see the appendix for details. In either case, the filtering requires $O(n^3 + n|I_t|)$ time per timestep t. When the filtering is complete (i.e., $E(q_t|Y)$ has been computed for each desired t), we typically want to transform from the q_t space to represent $E(f|Y)$; again, if the basis functions f_i correspond to wavelet or Fourier functions, this costs $O(N \log N)$ time per timestep, or $O(N)$ time if the f_i functions are compactly supported.

Figures 2 and 3 illustrate the output of the fast filter-smoother applied to simulated place field data. The spatial variable x is chosen to be one-dimensional here, for clarity. We chose the true place field $f(x,t)$ to be a Gaussian bump (as a function of x) whose mean varied sinusoidally in time but whose height and width were held constant (see the upper left panel of Fig. 2). The basis matrix F consisted of 50 equally-spaced bump functions with compact support (specifically, spatial Gaussians truncated at $\sigma \approx 4$, with each bump located one standard deviation σ apart from the next.) The dynamics coefficient a (in the diagonal dynamics matrix $A = aI$) was about 0.97, which corresponds to a temporal correlation time of $\tau = 30$ timesteps; the simulation shown in Fig. 2 lasted for $T = 1000$ timesteps. To explore the behavior of the filter in two regimes, we let x_t begin by sampling a wide range of locations (see Fig. 2 for $t < 200$ or so), but then settling down to a small spatial subset for larger values of t. We used the Gaussian noise model for y_t in this simulation.

We find that, as expected, the filter does a good job of tracking $f(x,t)$ for locations x near the observation points x_t, where the observations y_t carry a good deal of information, but far from x_t the filter defaults to its prior mean value, significantly underestimating $f(x,t)$. The posterior uncertainty $V(f(x,t)|Y) = diag[FCov(q_t|Y)F^T]$ remains near the prior uncertainty $diag[FC_0F^T]$ in locations far from x_t, as expected. Figures 1 and 3 illustrate that the low-rank approximation works well in this setting, despite the fact that (at least for t sufficiently large) only a few singular values are retained in our low-rank approximation (c.f. Fig. 2, lower left panel). We set the variance fraction coefficient $c = 0.99$ in equation (7) for this simulation; the results do not depend qualitatively on the precise value of c in this case (data not shown).

We have also applied the filter to real neuronal data, recorded from single neurons in the mouse hippocampal region by Dr. Pablo Jercog. In these experiments the mouse was exploring a two-dimensional cage, and so we estimated the firing rate surface $f(x,t)$ as a function of time t and a two-dimensional spatial variable x. The results are most easily viewed in movie form; see http://www.stat.columbia.edu/∼liam/research/abstracts/fast-low-SNR-Kalman-abs.html for details.

Finally, Figure 4 illustrates an application of the fast filter-smoother to the second con-

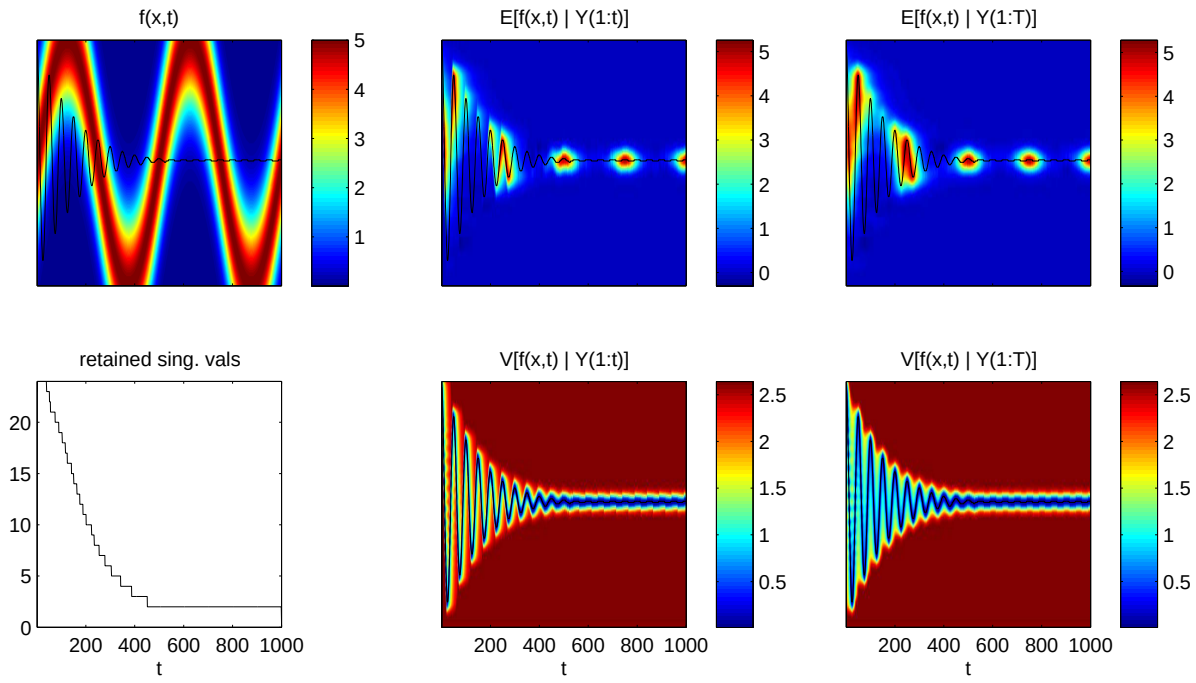

Figure 2: Output of the filter-smoother applied to simulated one-dimensional place field data. The superimposed black trace in all but the lower left panel indicates the simulated path x_t of the animal; x_t begins by sampling a wide range of locations for $t < 200$, but settles down to a small spatial subset for larger values of t. Upper left: true simulated place field $f(x,t)$ is shown in color; $f(x,t)$ has a Gaussian shape as a function of x, and the center of this Gaussian varies sinusoidally as a function of time t. Top middle and right panels: estimated place fields, forward ($E(f(x,t)|Y_{1:t})$) and forward-backward ($E(f(x,t)|Y_{1:T})$), respectively. Here (in a slight abuse of notation) we use $E(f^t|Y)$ to denote the projected mean $FE(q_t|Y)$, where F is the basis matrix corresponding to the basis coefficients q. Note that the estimated place fields are accurate near the observed positions x_t, but revert to the mean when no information is available. Bottom middle and right panels: marginal variance of the estimated place fields, forward ($V(f(x,t)|Y_{1:t})$) and forward-backward ($V(f(x,t)|Y_{1:T})$), respectively. Again, note that the filter output is most confident near x_t. Lower left panel: effective rank of $C_0 - C_t^s$ as a function of t in the forward-backward smoother; the effective rank is largest when x_t samples many locations in a short time period.

text described above. We simulated neuronal responses of the form $y_t = s_t^T f^t + \eta_t$, where the sensory stimulus s_t was taken to be a spatiotemporal Gaussian white noise process and the response noise η_t was also modeled as Gaussian and white, for simplicity. As discussed above, we represented f^t as a time-varying weighted sum of fixed basis functions f_i. In this case the basis F consisted of real-valued Fourier functions (sines and cosines), and multiplication by this basis matrix was implemented via the fast Fourier transform. As in the previous example, we chose the dynamics matrix A to be proportional to the identity; the effective autocorrelation time was $\tau = 50$ time steps here. The dynamics noise covariance V was diagonal (and therefore so was the prior covariance C_0), with the diagonal elements chosen so that the

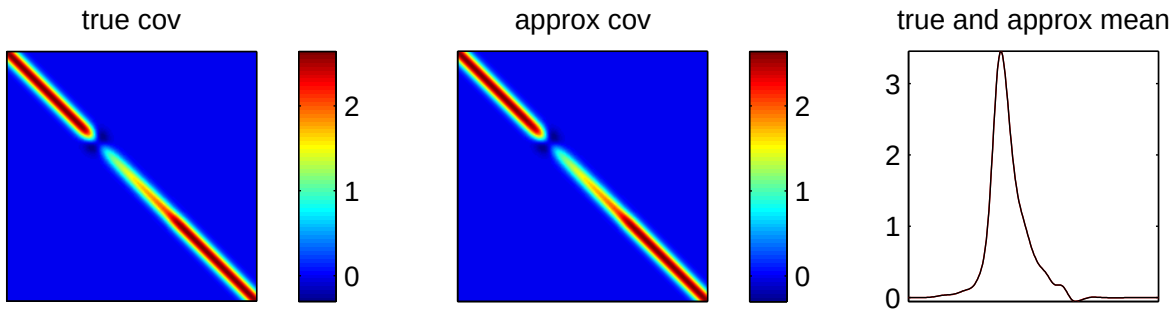

Figure 3: Comparison of the true vs. approximate projected covariance FC_tF^T and mean $F\mu_t$ at $t = 200$. Simulation is as in Fig. 2. Left panel: true forward projected covariance FC_tF^T. Middle panel: approximate forward covariance $F(C_0 + U_tD_tU_t)F^T$. The maximal pointwise error between these two matrices is about 1%. Right panel: true and approximate forward mean $F\mu_t$. The black trace indicates the true mean and the red trace (barely visible) the approximate mean. Figure 1 shows C_t from the same simulation at the same time, $t = 200$.

prior variance of the ω-th frequency basis coefficient falls off proportionally to ω^{-2}; this led to an effective smoothing prior. Figure 4 provides a one-dimensional example, where the full spatiotemporal output of the filter-smoother can be visualized directly. We have also applied the filter to higher-dimensional examples; a two-dimensional example movie is available at http://www.stat.columbia.edu/∼liam/research/abstracts/fast-low-SNR-Kalman-abs.html.

Discussion

We have presented some simple but effective methods for more efficiently computing the Kalman filter and smoother recursions. The basic idea is that, in many cases, fast methods are available for multiplying and dividing by the prior equilibrium state covariance C_0, and the posterior state covariance C_t can be well-approximated by forming a low-rank perturbation of the prior C_0. These low-rank perturbations, in turn, can be computed and updated in an efficient recursive manner.

There are a number of clear opportunities for application of this basic idea. Some exciting examples involve optimal control and online experimental design in high-dimensional settings; for instance, optimal online experimental design requires us to choose the observation matrix B_t adaptively, in real time, to reduce the posterior uncertainty optimally, in some sense (Fedorov, 1972; Mackay, 1992; Krause et al., 2008; Lewi et al., 2009). In the linear-Gaussian case, the posterior covariance C_t is independent of the observations Y, so we can precompute the optimal sequence of B_t, though more generally (in the case of nonlinear or non-Gaussian observations) the optimal B_t can only be computed after observing the data $Y_{1:t-1}$. A wide variety of objective functions based on the posterior covariance C_t have been employed in the experimental design literature (Fedorov, 1972); the fast methods we have introduced in this paper can be adapted to compute many of these objective functions, including those based on the posterior state entropy, or weighted sums of the marginal posterior state variance. See (Huggins and Paninski, 2010) for an application of these ideas to the neuronal dendritic setting.

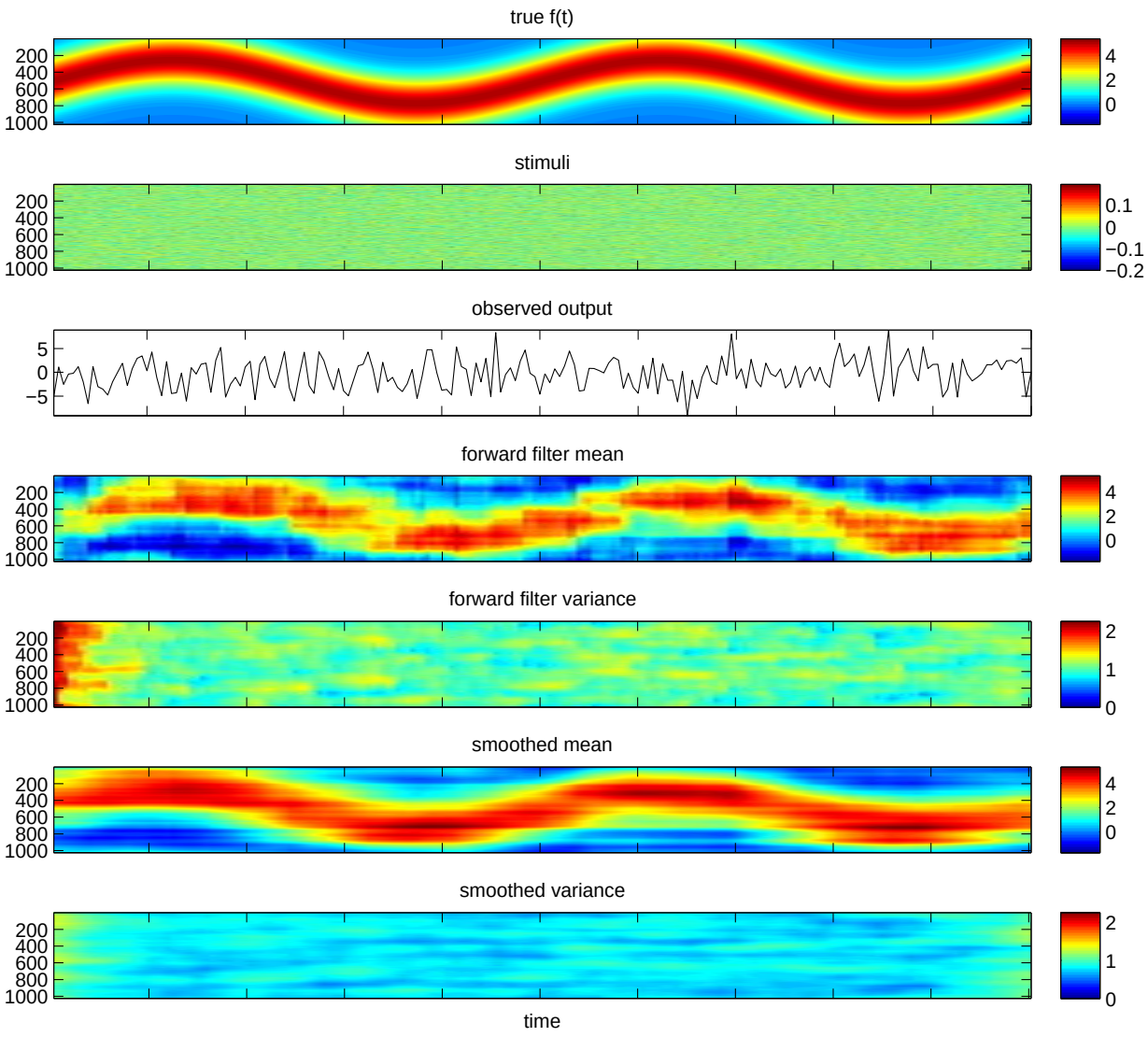

Figure 4: Tracking a time-varying one-dimensional receptive field. Top panel: the true receptive field f^t was chosen to be a spatial Gaussian bump whose center varied sinusoidally as a function of time t. Second panel: the stimulus s_t was chosen to be spatiotemporal white Gaussian noise. Third panel: simulated output observed according to the Gaussian model $y_t = s_t^T f^t + \eta_t$. Lower four panels: the forward filter mean $E(f^t|Y_{1:t})$ and marginal variance $Var(f^t(x)|Y_{1:t})$ and the full forward-backward smoother mean and marginal variance $E(f^t|Y_{1:T})$ and marginal variance $Var(f^t(x)|Y_{1:T})$. The dimension of the state variable f^t here was 2^{10}; inference required tens of seconds on a laptop. Time units are arbitrary here; the assumed prior autocorrelation time was $\tau = 50$ timesteps, while the total length of the experiment $T = 200$ timesteps.

We have seen that the prior covariance is especially easy to compute in the case that the dynamics matrix A is normal: here C_0 may be computed analytically, assuming the dynamics

noise covariance V can be transformed via a convenient whitening transformation. A key direction for future work will be to extend these methods to the case that A is a non-normal matrix. While there are a number of methods for solving the Lyapunov equation in this non-normal case (Anderson and Moore, 1979; Higham, 2008), it seems harder to find general efficient methods for multiplying or dividing by the solution C_0 in $o(N^2)$ time and space, as required by our fast method. The dynamics matrix A is normal (indeed, symmetric) in many applications — e.g., electrostatic applications, where the interactions between compartments i and j are symmetric (as in the neuronal cable case discussed in (Paninski, 2010)), and many mechanical applications (Tipireddy et al., 2009)), but non-normal dynamics matrices also arise quite frequently in practice. For example, weather prediction applications involve dynamics with strong drift (not just diffusion) terms, making A non-symmetric and perhaps non-normal in many cases. Standard direct methods for solving the Lyapunov equation given a non-normal dynamics matrix A (e.g., the Bartels-Stewart algorithm (Antoulas, 2005)) require an orthogonalization step that takes $O(N^3)$ time in general. There is a large applied mathematics literature on the approximate solution of Lyapunov equations with sparse dynamics (see e.g. (Sabino, 2007) for a nice review), but the focus of this literature is on the case that the noise covariance matrix is of low rank, which may be of less relevance in some statistical applications. Further research is needed into how best to adapt modern methods for solving the Lyapunov equation (e.g., those based on the matrix sign function (Higham, 2008)) to this fast Kalman filter setting.

Another important direction for future research involves generalizations beyond the simple Kalman setting explored here. The smoothers we have discussed are all based on a simple vector AR(1) framework. It is natural to ask if similar methods can be employed to handle the AR(p) case, or if other temporal smoothing methodologies (e.g., penalized spline methods (Green and Silverman, 1994; DiMatteo et al., 2001; Wood, 2006)) might benefit from a similar approach, since all of these techniques rely heavily on solving linear equations for which the corresponding matrices are banded in the temporal domain. One promising idea is to develop methods for directly solving these banded matrix equations, without using the Kalman recursions per se (Fahrmeir and Kaufmann, 1991; Paninski et al., 2010); see (Huggins and Paninski, 2011) for one implementation of this approach.

Finally, another major limitation of the methods discussed here is that we assume the underlying dynamics model is stationary, in order to compute the equilibrium state covariance C_0. One way to generalize this idea is to interpret C_0 simply as the prior covariance, and not the equilibrium solution; then C_0 can vary as a function of time. In some cases this time-varying C_0 can be computed efficiently, and suitable low-rank approximations for C_t follow directly (Pnevmatikakis and Paninski, 2011). This opens up some interesting applications involving the incorporation of non-Gaussian priors (Park and Casella, 2008); we are currently in the process of pursuing these directions further.

Appendix: nonlinear observations

We would like to incorporate observations y_t obeying some arbitrary conditional density $p(y_t|q_t)$ into our filter equations. This is of course difficult in general, since if $p(y_t|q_t)$ is chosen maliciously it is clear that our posterior distribution $p(q_t|Y_{1:t})$ may be highly non-Gaussian, and our basic Kalman recursion will break down. However, if $\log p(y_t|q_t)$ is a smooth, concave function of q_t, it is known that a Gaussian approximation to $p(q_t|Y_{1:t})$ will often be fairly accurate (Fahrmeir and Tutz, 1994; Brown et al., 1998; Paninski et al., 2010),

and our Kalman recursion may be adapted in a fairly straightforward manner.

For simplicity, we will focus on the case that the observations y_{it} are independent samples from $p(y_{it}|B_i q_t)$, where B_i denotes the i-th row of the observation matrix B. (The extension to the case that y_t depends in a more general manner on the projection Bq_t may be handled similarly.) We approximate the posterior mean μ_t with the maximum a posteriori (MAP) estimate,

$$\begin{aligned}\mu_t &\approx \arg\max_{q_t} \left[\log p(q_t|Y_{0:t-1}) + \log p(y_t|q_t)\right] \\ &\approx \arg\max_{q_t} \left[-\frac{1}{2}(q_t - m_t)^T P_t^{-1}(q_t - m_t) + \sum_i \log p(y_{it}|B_i q_t)\right].\end{aligned}$$

(Recall that the one-step covariance matrix P_t and mean m_t were defined in eqs. (8-9).) This MAP update is exact in the linear-Gaussian case (and corresponds exactly to the Kalman filter), but is an approximation more generally.

To compute this MAP estimate, we use Newton-Raphson. We need the gradient and Hessian of the log-posterior with respect to q_t,

$$\nabla = -P_t^{-1}(q_t - m_t) + B^T f_1(q_t)$$

and

$$H = -P_t^{-1} + B^T diag[f_2(q_t)]B,$$

respectively. Here $f_1(q_t)$ and $f_2(q_t)$ are the vectors formed by taking the first and second derivatives, respectively, of $\log p(y_{it}|u)$ at $u = B_i q_t$, with respect to u. Now we may form the Newton step:

$$\begin{aligned}q_{new} &= q_{old} - H^{-1}\nabla \\ &= q_{old} - \left(-P_t^{-1} + B^T diag[f_2(q_{old})]B\right)^{-1}\left(-P_t^{-1}(q_{old} - m_t) + B' f_1(q_{old})\right) \\ &= q_{old} - (P_t - P_t B^T(-diag[f_2(q_{old})^{-1}] + BP_t B^T)^{-1} BP_t)\left[P_t^{-1}(q_{old} - m_t) - B^T f_1(q_{old})\right] \\ &= m_t + P_t B^T(-diag[f_2(q_{old})^{-1}] + BP_t B^T)^{-1} B\left[q_{old} - m_t - P_t B^T f_1(q_{old})\right] + P_t B^T f_1(q_{old})\end{aligned}$$

We iterate, using a backstepping linesearch to guarantee that the log-posterior increases on each iteration, until convergence (i.e., when $q_{new} \approx q_{old}$, set $\mu_t = q_{old}$). Then, finally, we update the covariance C_t by replacing W^{-1} with $-diag[f_2(q_t)]$ in the original derivation. Since multiplication by P_t is assumed fast (and we need to compute $P_t B^T$ just once per timestep), all of these computations remain tractable. Finally, we note that it is also straightforward to adapt these fast methods in the context of the filter-forward sample-backward approach discussed in (Jungbacker and Koopman, 2007) for sampling from the posterior $p(Q|Y)$ once the MAP path for Q is obtained; however, we have not yet pursued this direction extensively.

Appendix: Low-SNR approximations of the solution to the Riccati equation

We would like to better understand the solution C of the Riccati equation in the low-SNR regime:

$$C^{-1} = (ACA^T + I) + B^T W^{-1} B,$$

with $B^T W^{-1} B$ small and constant. (For simplicity, note that we have standardized via the usual whitening transformation so that the dynamics noise covariance matrix is just the

identity; in addition, we will assume that the dynamics noise covariance is full-rank.) The Riccati equation is difficult to solve in general. Since we are interested in the low-SNR case, we may replace the information matrix $B^T W^{-1} B$ with ϵJ, where ϵ is understood to be a small parameter. We know that $C = C_0$ when $\epsilon = 0$. Now we can search for a solution in terms of a series expansion in ϵ near $\epsilon = 0$.

For ease of interpretation, we will assume that every matrix in sight (namely A, A^T, and J) commutes. Then $C_0 = (I - AA^T)^{-1}$, as discussed in the main text. To first order, we seek a solution for C in the form $C = C_0 + \epsilon Q$. We expand both sides of the equation

$$C^{-1} = (ACA^T + I)^{-1} + \epsilon J$$

to first order in ϵ:

$$\begin{aligned} C^{-1} &= (C_0 + \epsilon Q)^{-1} \\ &= C_0^{-1}(I - \epsilon Q C_0^{-1}) + o(\epsilon) \end{aligned}$$

and

$$\begin{aligned} (ACA^T + I)^{-1} + \epsilon J &= (C_0 + \epsilon AQA^T)^{-1} + \epsilon J \\ &= C_0^{-1}(I - \epsilon AQA^T C_0^{-1}) + \epsilon J + o(\epsilon). \end{aligned}$$

Matching terms up to first order in ϵ gives

$$Q = -JC_0^3.$$

(If A is normal but does not commute with J, then we can use a similar approach and obtain Q as the solution to a discrete Lyapunov equation; we omit the details.) Thus, in this case, up to first order in ϵ, $rank(C - C_0) = rank(J)$, providing further quantitative justification for our low-rank approximation. More generally, this result can provide at least a rough sense of how large the perturbation $C - C_0$ will be as a function of the observation information matrix J and the dynamics matrix A (through $C_0 = (I - AA^T)^{-1}$). This, in turn, could be useful in determining how large the perturbation rank n will have to be to maintain a given accuracy in the approximate filter.

Acknowledgments

LP is supported by a McKnight Scholar award and an NSF CAREER award. JHH is supported by the Columbia College Rabi Scholars Program. We thank E. Pnevmatikakis for helpful discussions and P. Jercog for kindly sharing his hippocampal data with us.

References

Anderson, B. and Moore, J. (1979). *Optimal Filtering*. Prentice Hall.

Antoulas, A. (2005). *Approximation of large-scale dynamical systems*. Cambridge University Press.

Banerjee, S., Gelfand, A. E., Finley, A. O., and Sang, H. (2008). Gaussian predictive process models for large spatial data sets. *Journal Of The Royal Statistical Society Series B*, 70:825–848.

Bickel, J. and Levina, E. (2008). Regularized estimation of large covariance matrices. *Annals of Statistics*, 36:199–227.

Briggs, W. L., Henson, V. E., and McCormick, S. F. (2000). *A multigrid tutorial (2nd ed.)*. SIAM.

Brockwell, P. and Davis, R. (1991). *Time Series: Theory and Methods*. Springer.

Brown, E., Frank, L., Tang, D., Quirk, M., and Wilson, M. (1998). A statistical paradigm for neural spike train decoding applied to position prediction from ensemble firing patterns of rat hippocampal place cells. *Journal of Neuroscience*, 18:7411–7425.

Brown, E., Nguyen, D., Frank, L., Wilson, M., and Solo, V. (2001). An analysis of neural receptive field plasticity by point process adaptive filtering. *PNAS*, 98:12261–12266.

Chan, R. H. and Ng, M. K. (1996). Conjugate gradient methods for toeplitz systems. *SIAM Review*, 38:427–482.

Chandrasekar, J., Kim, I., Bernstein, D., and Ridley, A. (2008). Cholesky-based reduced-rank square-root Kalman filtering. *American Control Conference*, pages 3987–3992.

Cressie, N. and Johannesson, G. (2008). Fixed rank kriging for very large spatial data sets. *Journal Of The Royal Statistical Society Series B*, 70:209–226.

Cressie, N., Shi, T., and Kang, E. L. (2010). Fixed rank filtering for spatio-temporal data. *Journal of Computational and Graphical Statistics*, 19:724–745.

Czanner, G., Eden, U., Wirth, S., Yanike, M., Suzuki, W., and Brown, E. (2008). Analysis of between-trial and within-trial neural spiking dynamics. *Journal of Neurophysiology*, 99:2672–2693.

Daubechies, I. (1992). *Ten Lectures on Wavelets*. SIAM.

Davis, T. (2006). *Direct Methods for Sparse Linear Systems*. SIAM.

DiMatteo, I., Genovese, C., and Kass, R. (2001). Bayesian curve fitting with free-knot splines. *Biometrika*, 88:1055–1073.

Durbin, J. and Koopman, S. (2001). *Time Series Analysis by State Space Methods*. Oxford University Press.

El Karoui, N. (2008). Operator norm consistent estimation of large-dimensional sparse covariance matrices. *Annals of Statistics*, 36:2717–2756.

Evensen, G. (2009). *Data assimilation: the Ensemble Kalman Filter*. Springer.

Fahrmeir, L. and Kaufmann, H. (1991). On Kalman filtering, posterior mode estimation and fisher scoring in dynamic exponential family regression. *Metrika*, 38:37–60.

Fahrmeir, L. and Tutz, G. (1994). *Multivariate Statistical Modelling Based on Generalized Linear Models*. Springer.

Fedorov, V. (1972). *Theory of Optimal Experiments*. Academic Press, New York.

Frank, L., Eden, U., Solo, V., Wilson, M., and Brown, E. (2002). Contrasting patterns of receptive field plasticity in the hippocampus and the entorhinal cortex: An adaptive filtering approach. *J. Neurosci.*, 22(9):3817–3830.

Freestone, D., Aram, P., Dewar, M., Scerri, K., Grayden, D., and Kadirkamanathan, V. (2011). A data-driven framework for neural field modeling. *NeuroImage*, In Press.

Furrer, R. and Bengtsson, T. (2007). Estimation of high-dimensional prior and posterior covariance matrices in Kalman filter variants. *J. Multivar. Anal.*, 98:227–255.

Galka, A., Ozaki, T., Muhle, H., Stephani, U., and Siniatchkin, M. (2008). A data-driven model of the generation of human EEG based on a spatially distributed stochastic wave equation. *Cognitive Neurodynamics*, 2(2):101–13.

Gillijns, S., Bernstein, D., and De Moor, B. (2006). The reduced rank transform square root filter for data assimilation. *Proc. of the 14th IFAC Symposium on System Identification*.

Golub, G. H. and Van Loan, C. F. (1996). *Matrix Computations*. The Johns Hopkins University Press.

Green, P. and Silverman, B. (1994). *Nonparametric Regression and Generalized Linear Models*. CRC Press.

Hackbusch, W. and Khoromskij, B. N. (2000). A sparse H-matrix arithmetic. Part II: application to multi-dimensional problems. *Computing*, 64:21–47.

Halko, N., Martinsson, P.-G., and Tropp, J. A. (2009). Finding structure with randomness: Stochastic algorithms for constructing approximate matrix decompositions. *Arxiv*, page 0909.4061.

Higham, N. (2008). *Functions of matrices: theory and computation*. SIAM.

Hines, M. (1984). Efficient computation of branched nerve equations. *International Journal of Bio-Medical Computing*, 15(1):69 – 76.

Howard, A. and Jebara, T. (2005). Square root propagation. *Columbia University Computer Science Technical Reports*, 040-05.

Huggins, J. and Paninski, L. (2010). Optimal experimental design for sampling voltage on dendritic trees. *Under review*.

Huggins, J. and Paninski, L. (2011). A fast method for detecting synapse locations on dendritic trees. *In preparation*.

Jordan, M. I., editor (1999). *Learning in graphical models*. MIT Press, Cambridge, MA, USA.

Jungbacker, B. and Koopman, S. (2007). Monte Carlo estimation for nonlinear non-Gaussian state space models. *Biometrika*, 94:827–839.

Kaufman, C. G., Schervish, M. J., and Nychka, D. W. (2008). Covariance tapering for likelihood-based estimation in large spatial data sets. *Journal of the American Statistical Association*, 103(484):1545–1555.

Khan, U. A. and Moura, J. M. F. (2008). Distributing the Kalman filter for large-scale systems. *IEEE Transactions on Signal Processing*, 56:4919–4935.

Koch, C. (1999). *Biophysics of Computation*. Oxford University Press.

Krause, A., Singh, A., and Guestrin, C. (2008). Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9:235–284.

Lew, S., Wolters, C. H., Dierkes, T., Röer, C., and MacLeod, R. S. (2009). Accuracy and run-time comparison for different potential approaches and iterative solvers in finite element method based EEG source analysis. *Appl. Numer. Math.*, 59:1970–1988.

Lewi, J., Butera, R., and Paninski, L. (2009). Sequential optimal design of neurophysiology experiments. *Neural Computation*, 21:619–687.

Liberty, E., Woolfe, F., Martinsson, P.-G., Rokhlin, V., and Tygert, M. (2007). Randomized algorithms for the low-rank approximation of matrices. *Proceedings of the National Academy of Sciences*, 104:20167–20172.

Long, C. J., Purdon, R. L., Temereanca, S., Desai, N. U., Hämäläinen, M., and Brown, E. N. (2006). Large scale Kalman filtering solutions to the electrophysiological source localization problem–a MEG case study. *Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 1.

Mackay, D. (1992). Information-based objective functions for active data selection. *Neural Computation*, 4:589–603.

Memarsadeghi, N., Raykar, V., Duraiswami, R., and Mount, D. (2008). Efficient kriging via fast matrix-vector products. In *Aerospace Conference, 2008 IEEE*, pages 1 –7.

Paninski, L. (2004). Maximum likelihood estimation of cascade point-process neural encoding models. *Network: Computation in Neural Systems*, 15:243–262.

Paninski, L. (2010). Fast Kalman filtering on quasilinear dendritic trees. *Journal of Computational Neuroscience*, 28:211–28.

Paninski, L., Ahmadian, Y., Ferreira, D., Koyama, S., Rahnama, K., Vidne, M., Vogelstein, J., and Wu, W. (2010). A new look at state-space models for neural data. *Journal of Computational Neuroscience*, 29:107–126.

Paninski, L., Pillow, J., and Lewi, J. (2007). Statistical models for neural encoding, decoding, and optimal stimulus design. In Cisek, P., Drew, T., and Kalaska, J., editors, *Computational Neuroscience: Progress in Brain Research*. Elsevier.

Park, T. and Casella, G. (2008). The Bayesian Lasso. *Journal of the American Statistical Association*, 103:681–686.

Pnevmatikakis, E., Kelleher, K., Chen, R., Josic, K., Saggau, P., and Paninski, L. (2011). Fast nonnegative spatiotemporal calcium smoothing in dendritic trees. *COSYNE*.

Pnevmatikakis, E. and Paninski, L. (2011). Fast interior-point inference in high-dimensional sparse, penalized state-space models. *Submitted*.

Press, W., Teukolsky, S., Vetterling, W., and Flannery, B. (1992). *Numerical recipes in C*. Cambridge University Press.

Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77:257–286.

Rahnama Rad, K. and Paninski, L. (2010). Efficient estimation of two-dimensional firing rate surfaces via Gaussian process methods. *Network*, 21:142–168.

Rue, H. and Held, L. (2005). *Gaussian Markov random fields: theory and applications*. CRC Press.

Sabino, J. (2007). *Solution of large-scale Lyapunov equations via the block modified Smith method*. PhD thesis, Rice University.

Shental, O., Bickson, D., Siegel, P. H., Wolf, J. K., and Dolev, D. (2008). Gaussian belief propagation for solving systems of linear equations: Theory and application. *arXiv:0810.1119v1*.

Shumway, R. and Stoffer, D. (2006). *Time Series Analysis and Its Applications*. Springer.

Solo, V. (2004). State estimation from high-dimensional data. *ICASSP*, 2:685–688.

Stroud, J. R., Muller, P., and Sanso, B. (2001). Dynamic models for spatiotemporal data. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 63:pp. 673–689.

Tipireddy, R., Nasrellah, H., and Manohar, C. (2009). A Kalman filter based strategy for linear structural system identification based on multiple static and dynamic test data. *Probabilistic Engineering Mechanics*, 24:60–74.

Treebushny, D. and Madsen, H. (2005). On the construction of a reduced rank square-root Kalman filter for efficient uncertainty propagation. *Future Gener. Comput. Syst.*, 21:1047–1055.

Verlaan, M. (1998). *Efficient Kalman filtering algorithms for hydrodynamic models*. PhD thesis, TU Delft.

Walnut, D. (2002). *An introduction to wavelet analysis*. Springer.

Weiss, Y. and Freeman, W. T. (2001). Correctness of belief propagation in Gaussian graphical models of arbitrary topology. *Neural Computation*, 13(10):2173–2200.

Wikle, C. and Cressie, N. (1999). A dimension-reduced approach to space-time Kalman filtering. *Biometrika*, 86(4):815–829.

Wolters, C. (2007). The finite element method in EEG/MEG source analysis. *SIAM News*, 52(2).

Wood, S. N. (2006). Low-rank scale-invariant tensor product smooths for generalized additive mixed models. *Biometrics*, 62:1025–36.