Journal of Data Science 20 (4), 461–474
October 2022
DOI: 10.6339/22-JDS1071
Statistical Data Science

High-Dimensional Nonlinear Spatio-Temporal Filtering by Compressing Hierarchical Sparse Cholesky Factors

Anirban Chakraborty[1] and Matthias Katzfuss[1,*]
[1]*Department of Statistics, Texas A&M University, 3143 TAMU, College Station, TX 77843, USA*

Abstract

Spatio-temporal filtering is a common and challenging task in many environmental applications, where the evolution is often nonlinear and the dimension of the spatial state may be very high. We propose a scalable filtering approach based on a hierarchical sparse Cholesky representation of the filtering covariance matrix. At each time point, we compress the sparse Cholesky factor into a dense matrix with a small number of columns. After applying the evolution to each of these columns, we decompress to obtain a hierarchical sparse Cholesky factor of the forecast covariance, which can then be updated based on newly available data. We illustrate the Cholesky evolution via an equivalent representation in terms of spatial basis functions. We also demonstrate the advantage of our method in numerical comparisons, including using a high-dimensional and nonlinear Lorenz model.

Keywords *basis functions; data assimilation; hierarchical Vecchia approximation; Lorenz model; unscented Kalman filter*

1 Introduction

State-space models (SSM) represent observed systems that evolve over time. We can think of SSMs as a combination of two mathematical equations: one equation describes the evolution of the system state over time, while the other specifies the relationship between the state and the observations (e.g., West and Harrison, 1997; Shumway and Stoffer, 2000). Here, we concern ourselves with spatio-temporal SSMs describing the evolution of a discretized spatial field over time. These SSMs have seen widespread implementation in various domains, including geosciences and biomedical applications. Filtering or data assimilation (e.g., Nychka and Anderson, 2010) refers to sequentially inferring the posterior distribution of the state distribution at a point in time given all available data up to that time point. Filtering can be challenging for spatio-temporal SSMs, as they are often nonlinear and high-dimensional.

There is an enormous literature on filtering for SSMs in low to moderate dimensions. For linear Gaussian SSMs, the Kalman filter (Kalman, 1960) computes the Gaussian filtering distributions in closed form. For nonlinear SSMs, the extended Kalman filter (EKF; Grewal and Andrews, 1993, Ch. 5) relies on a linearization of the evolution operator. In contrast, the unscented Kalman filter (UKF; Wan and Van Der Merwe, 2000) applies the potentially nonlinear evolution to so-called sigma points computed based on singular value or Cholesky decomposition of the filtering covariance matrix. Several authors (e.g., Wan and Van Der Merwe, 2000; St-Pierre and Gingras, 2004; Khazraj et al., 2016) have shown that in many situations UKF per-

*Corresponding author. Email: katzfuss@gmail.com.

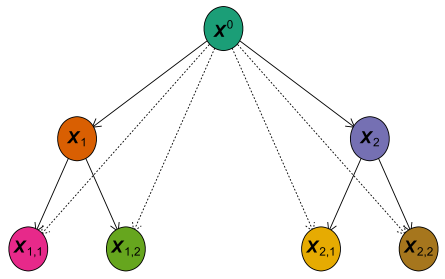
Received August 1, 2022; Accepted September 28, 2022

forms better than EKF. However, EKF, UKF, and many extensions (Arasaratnam and Haykin, 2009; Wang et al., 2013; Meng et al., 2018; Fang et al., 2020) rely on inversion or decomposition of the covariance matrices involved in the filtering distributions. Since these operations scale cubically in the dimension of the matrix, it becomes increasingly infeasible when the system is high-dimensional. Particle filters (e.g., Gordon et al., 1993; Liu and Chen, 1998; Pitt and Shephard, 1999) are a class of sequential Monte Carlo methods that rely on propagating samples or particles via the evolution, which are then re-weighted according to the observations at each time point. While such methods can in principle obtain accurate filtering distributions in almost any SSMs, the required number of particles to do so increases exponentially with the state dimension.

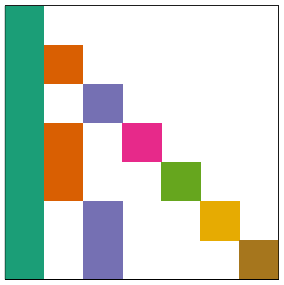
For filtering in high-dimensional nonlinear spatio-temporal SSMs, the extended Kalman Vecchia Laplace (EKVL) filter (Jurek and Katzfuss, 2022) combines the EKF with the Vecchia approximation (e.g., Vecchia, 1988; Stein et al., 2004; Datta et al., 2016; Guinness, 2018; Katzfuss and Guinness, 2021; Katzfuss et al., 2020; Schäfer et al., 2021) for scalability and with a Laplace approximation for handling non-Gaussian observations. However, this approach relies on a linearization of the evolution like the EKF, which can be inaccurate and computationally expensive. Ensemble Kalman filter (EnKF) methods propagate an ensemble of particles similar to particle filters, but they linearly shift the ensemble members to avoid weight collapse in the update step. This requires estimating the forecast covariance matrix from the often small ensemble, which usually relies on spatial localization or tapering. As a result, long-range forecast correlation is often ignored, and the EnKF can be less accurate than the EKVL filter (Jurek and Katzfuss, 2022). Chandrasekar et al. (2008) introduced a reduced-rank UKF (RRUKF) that only propagates a small number of sigma points that are then used to estimate the forecast covariance matrix. This can lead to a severe loss in accuracy, and the method is only applicable under additive system noise to avoid singularity of the forecast covariance matrix.

Here we propose a novel approach by combining the ideas of the UKF and hierarchical Vecchia (HV) approximation used in Jurek and Katzfuss (2022). After computing an HV Cholesky factor of the initial covariance matrix, at every time point we compress the sparse Cholesky factor to preserve its non-zero entries into a tall and skinny matrix with a small number of columns. Next, we directly apply the equation of system dynamics on the columns of the aforementioned rectangular matrix, similar to sigma points in the UKF, and store all the transformed columns in a new rectangular matrix. Finally, we decompress the new rectangular matrix back into a sparse square matrix, which we treat as the Cholesky factor of the forecast covariance matrix. We illustrate that these operations can be interpreted as operating on spatial basis functions and can also be justified from that perspective. We demonstrate the advantage of our methods using numerical comparisons. Our method is highly scalable, as its matrix operations rely on sparse matrices and because the (often very expensive) evolution operator only has to be evaluated a small number of times at each time point; the evolution can be viewed as a black box and no linearization is required.

The remainder of this article is organized as follows. In Section 2, we review the HV approximation and we propose a specific ordering and grouping of the spatial grid locations. In Section 3, we introduce our filtering scheme. In Section 4, we demonstrate the accuracy of our method in a comparison on Lorenz data. Section 5 concludes.



(a) Hierarchical DAG



(b) Sparsity pattern of Cholesky factor

Figure 1: Illustration of a HV assumption for a vector $\mathbf{x}$ split into seven subsets: (a) Dependence structure between the subsets or nodes visualized using a directed acyclic graph (DAG) arranged in an upside-down tree. Some arrows are shown using solid lines to emphasize the tree structure. (b) The corresponding block-sparsity pattern of the approximate Cholesky factor, with nonzero entries in each column in the same color as the corresponding node in the DAG.

2 Hierarchical Vecchia (HV) Approximation of a Spatial Field

2.1 Review of the HV Approximation

Consider a Gaussian vector $\mathbf{x} = (x_1, x_2, \dots, x_n)^\top$ with covariance matrix $\mathbf{\Sigma}$, corresponding to a spatial process $x(\cdot)$ discretized on a spatial grid $\mathcal{S} = \{\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_n\}$, with $\mathbf{s}_i \in \mathcal{D} \subset \mathbb{R}^d$. The hierarchical Vecchia (HV) approximation (Jurek and Katzfuss, 2022) can be used to approximate the Cholesky factor of $\mathbf{\Sigma}$ by a sparse matrix $\mathbf{L}$. We briefly review this approach here.

As described in more detail in Section 2.2, the entries of $\mathbf{x}$ are grouped into subsets $X_{i_1,\dots,i_m}$, whose conditional dependence structure is assumed to be described by a tree-like directed acyclic graph as illustrated in Figure 1a. Assuming an ordering of the $X_{i_1,\dots,i_m}$ by increasing number of subscripts, this corresponds to a triangular block-sparsity pattern $\mathbf{S}$ illustrated in Figure 1b, with $\mathbf{S}_{ij} = 1$ if x_i is a descendant of x_j in the DAG (i.e., they are on the same branch of the tree), and $\mathbf{S}_{ij} = 0$ otherwise. For example, this means that $\mathbf{S}_{i1} = 1$ for all i, because all nodes are descendants of the stem node. We can approximate the Cholesky factor of $\mathbf{\Sigma}$ by simply applying an incomplete Cholesky factorization of $\mathbf{\Sigma}$ that skips all operations on matrix elements not in $\mathbf{S}$: $\mathbf{L} = IC0(\mathbf{\Sigma}, \mathbf{S})$. This operation only requires $\mathcal{O}(nR^2)$ time, where R is the maximum number of nonzero entries per row in $\mathbf{S}$. Further, as this is equivalent to a Vecchia approximation with appropriately chosen conditioning sets, the resulting $\mathbf{L}$ is the optimal approximate Cholesky factor for this sparsity pattern, in the sense that it achieves the minimum Kullback-Leibler divergence between $\mathcal{N}_n(\mathbf{0}, \mathbf{\Sigma})$ and $\mathcal{N}_n(\mathbf{0}, \hat{\mathbf{L}}\hat{\mathbf{L}}^\top)$ among all $\hat{\mathbf{L}}$ with sparsity pattern $\mathbf{S}$ (e.g., Katzfuss and Guinness, 2021; Jurek and Katzfuss, 2022; Schäfer et al., 2021).

2.2 Hierarchical Partitioning of Spatial Locations

To arrange the elements of $\mathbf{x}$ in a hierarchical tree structure as in Figure 1a, we consider a hierarchical partitioning of the associated spatial grid $\mathcal{S}$, as illustrated in Figure 2. To do so, we partition the spatial domain $\mathcal{D}$ into two halves and assign the r_0 grid points closest to the partition boundary to the first set X^0. Next, we split each of the resulting two subregions in half, selecting the r_1 nearest not-yet-assigned points to each partition boundary as X_1 and X_2, respectively. We continue the partitioning up until resolution M, where all remaining points (at

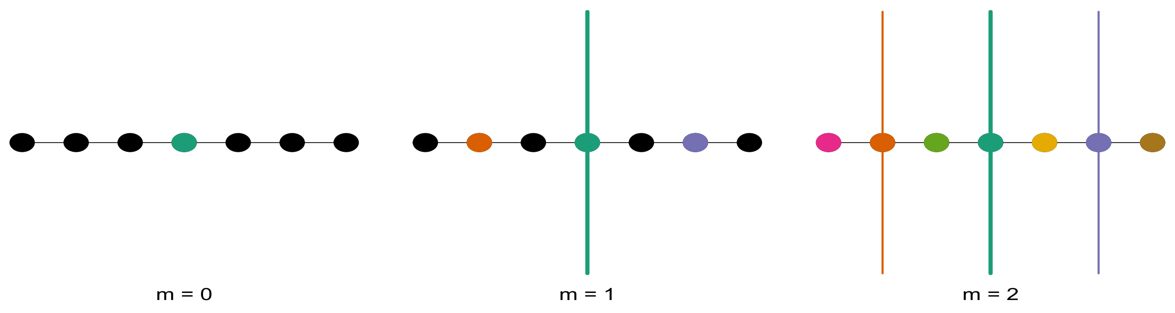


Figure 2: Hierarchical partitioning of $n = 7$ locations on a one-dimensional domain. At resolution $m = 0$, we select the $r_0 = 1$ middle point (in green) as X^0 and put it first in the ordering. At $m = 1$, we split the spatial region into two subregions, select $r_1 = 1$ point in the center of each of the two subregions and put them next in ordering. Continuing thusly for $m = 1, 2, \ldots, M$, we obtain an HV partitioning as illustrated in Figure 1, with $M = 2$ and $r_m = 1$ for $m = 0, 1, 2$.

most r_M per subregion) are assigned to sets $X_{i_1,\ldots,i_M}$. Figure 3 illustrates that the HV approximation based on this arrangement scheme can be highly accurate in some settings; often, it is more accurate than an arrangement based on maximum-minimum-distance ordering of the points suggested in Jurek and Katzfuss (2022).

Now consider the choice of tuning parameters M and $\mathbf{r} = (r_0, r_1, \ldots, r_M)^\top$. First, we recommend choosing r_m such that r_m points or knots are enough to capture (most of) the dependence between the two subregions at resolution m; r_m will typically increase with the dimension of the considered spatial domain. Then, M has to be large enough such that all n grid points can be assigned to a tree node without having to make r_M too large; in other words, we choose M as the smallest integer such that $\sum_{m=0}^{M} 2^m r_m \approx n$. Since this expression increases exponentially in M, a small increase in M is often sufficient to accommodate a large increase in n. A more detailed discussion of the choice of tuning parameters for a closely related spatial model can be found in Katzfuss (2017, Sect. 2.5).

Castrillón-Candás et al. (2013, 2016) developed a hierarchical domain partitioning mechanism to develop sparse basis representations of covariance matrices using interpolating points within the unit hypercube, which is similar to our domain partitioning mechanism. However, it is not clear if or how this framework can be extended to our nonlinear sequential filtering setting.

3 Nonlinear Filtering based on Compressed Cholesky Factors

3.1 State-Space Models and Filtering

Consider a nonlinear non-Gaussian spatio-temporal state-space model (SSM) for discrete time points $t = 1, 2, \ldots,$

$$y_{ti}|x_{ti} \sim g_{ti}(.|x_{ti}), \quad i \in \mathcal{I}_t \tag{1}$$

$$\mathbf{x}_t = \mathcal{E}_t(\mathbf{x}_{t-1}), \tag{2}$$

where y_{ti} is the ith observation at time point t, $\mathcal{I}_t \subset \{1, \ldots, n\}$ is an index set that contains all the spatial locations where we observe the process at time t, and $\mathbf{x}_t$ is the n dimensional

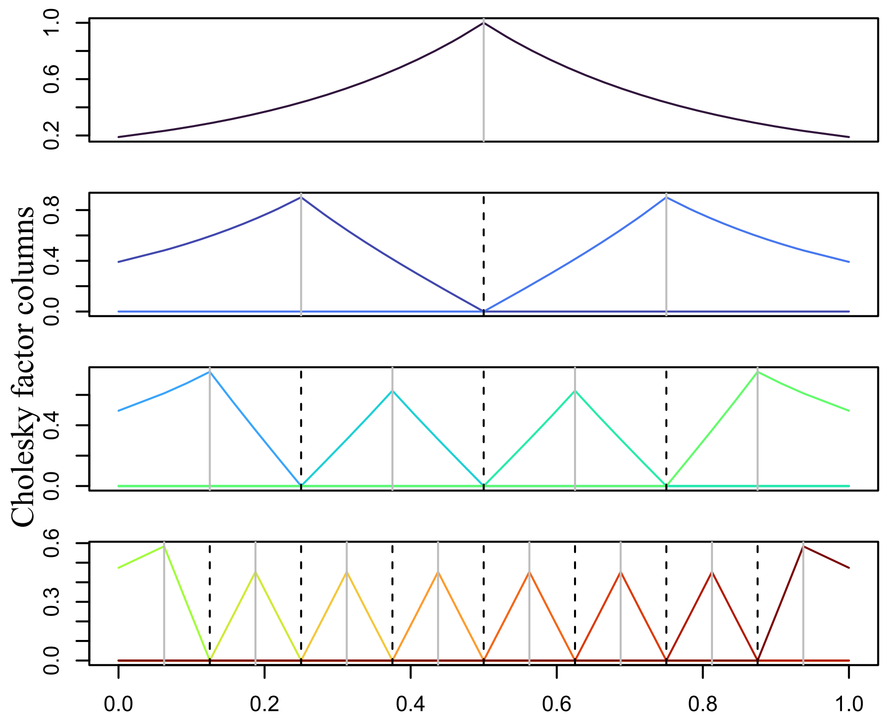


Figure 3: Basis-function representation of an HV approximation of an exponential covariance $\mathbf{\Sigma}$ evaluated at $n = 32$ regularly spaced locations on a one-dimensional domain. From top to bottom, the four panels correspond to resolutions $m = 0, 1, 2, 3$ of a hierarchical domain partitioning with $M = 4$ resolutions, each with $r_m = 1$ basis functions (in different colors) and $r_m = 1$ knots (grey vertical solid lines) per subregion (black vertical dashed lines). Each of the basis functions is obtained by interpolating the entries of one of the columns of the HV Cholesky factor $\mathbf{L}$ (see Figure 4a). In this setting, the HV approximation is exact (i.e., $\mathbf{\Sigma} = \mathbf{L}\mathbf{L}^\top$).

unobserved state of interest consisting of a spatial field evaluated at n locations. We assume that the initial state $\mathbf{x}_0$ of the above state-space model follows $\mathcal{N}_n(\boldsymbol{\mu}_{0|0}, \mathbf{\Sigma}_{0|0})$. The evolution operator $\mathcal{E}_t$ specifies how the process state evolves from one time point to the next. We assume that the evolution is local, in that the process at location, say, $\mathbf{s}$ at time t depends mostly on the process at locations close to $\mathbf{s}$ at time $t-1$; in most applications, this assumption is at least approximately satisfied as long as the time steps are short enough. The special case of linear evolution can be described by an $n \times n$ matrix, $\mathcal{E}_t(\mathbf{x}_{t-1}) = \mathbf{E}_t \mathbf{x}_{t-1}$.

Filtering refers to computing the posterior distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t})$, sequentially for $t = 1, 2, \ldots$. At each time point, this requires two steps. The forecast step computes the forecast distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t-1})$ from the previous filtering distribution $p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})$ by applying the evolution in (2): $p(\mathbf{x}_t|\mathbf{y}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{x}_{t-1}$. The update step then computes the new filtering distribution at time t by updating the forecast distribution based on new data $\mathbf{y}_t$:

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1})}{\int p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1})d\mathbf{x}_t}. \tag{3}$$

For linear Gaussian SSMs, the forecast and filtering distributions are Gaussian and can be computed in closed form using the Kalman filter. For nonlinear and high-dimensional SSMs as we consider here, the filtering distributions can generally not be computed exactly and hence require approximations.

We will now describe the forecast and update steps for our proposed scalable filtering approach. At the initial time point $t = 0$, we obtain an HV approximation of $\mathbf{\Sigma}_{0|0}$, such that its Cholesky factor has the sparsity pattern described in Section 2. Our filtering operations preserve this hierarchical block-sparsity structure, meaning that the Cholesky factor of the filtering and forecast covariance at every time point exhibits this sparsity. This ensures that the low computational cost is maintained over time.

3.2 Forecast Step

Assume that the filtering distribution at time $t-1$ is $\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1} \sim \mathcal{N}_n(\boldsymbol{\mu}_{t-1}, \mathbf{\Sigma}_{t-1})$, where $\mathbf{\Sigma}_{t-1} = \mathbf{L}_{t-1}\mathbf{L}_{t-1}^\top$. The goal is to compute the forecast distribution $\mathbf{x}_t|\mathbf{y}_{1:t-1}$.

To motivate our forecast step, temporarily disregard the mean $\boldsymbol{\mu}_{t-1}$, and so $\tilde{\mathbf{x}}_{t-1} = \mathbf{L}_{t-1}\mathbf{z}$ follows the filtering distribution, where the columns of $\mathbf{L}_{t-1}$ can be viewed as spatial basis functions (see Figure 3) and $\mathbf{z} \sim \mathcal{N}_n(\mathbf{0}, \mathbf{I}_n)$ is the vector of corresponding weights. We know that $\tilde{\mathbf{x}}_t = \mathcal{E}_t(\tilde{\mathbf{x}}_{t-1})$ follows the desired forecast distribution. If the evolution were linear, then $\tilde{\mathbf{x}}_t = \mathbf{E}_t\tilde{\mathbf{x}}_{t-1} = (\mathbf{E}_t\mathbf{L}_{t-1})\mathbf{z} = \tilde{\mathbf{L}}_t\mathbf{z}$, where the jth column of the new basis-function matrix $\tilde{\mathbf{L}}_t$ is obtained by applying the evolution matrix $\mathbf{E}_t$ to the jth column of $\mathbf{L}_{t-1}$. The UKF can be viewed as an extension to nonlinear evolution, which is applied to so-called sigma points that are also essentially columns of $\mathbf{L}_{t-1}$. However, a limitation of both of these approaches is that $\mathbf{L}_{t-1}$ is an $n \times n$ matrix, and so the evolution must be applied n times; as applying the evolution operator is often very computationally expensive, this is not feasible for large n.

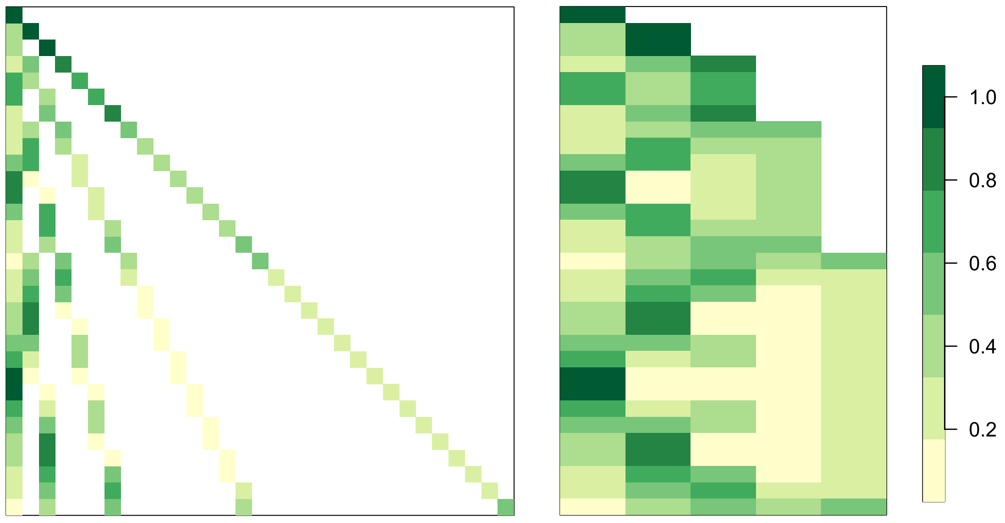
To get around this limitation, assume now that $\mathbf{L}_{t-1}$ exhibits a hierarchical block-sparsity structure as in Section 2, and the basis functions represented by its columns at each resolution have complementary support, as illustrated in Figure 3. Our idea is to combine basis functions with complementary support into a single basis function or, equivalently, compressing the large and sparse $n \times n$ matrix $\mathbf{L}_{t-1}$ into a tall, skinny, and dense $n \times R$ matrix, say $\mathbf{L}_{t-1}^C$, where $R = \sum_{m=0}^{M} r_m$. To be precise, the compression proceeds sequentially across the columns, moving each nonzero entry of the column to the leftmost zero position in the same row. This is illustrated in Figure 4b, where $n = 32$ and $R = 5$.

We then apply the evolution operator $\mathcal{E}_t$ separately to each of the R columns of $\mathbf{L}_{t-1}^C$, and then decompress the resulting matrix back to an $n \times n$ matrix $\tilde{\mathbf{L}}_t$ with the same sparsity pattern as $\mathbf{L}_t$, reversing the compression operation above. This means that we only need to apply the evolution $\mathcal{E}_t$ (as opposed to needing to linearize it), and we only have to do so $R \ll n$ times. While the compression incurs an approximation error near the partition boundaries, this error can be small when the evolution is sufficiently local, as illustrated in Figures 5 and 6. Roughly speaking, for low resolution m the basis functions are large in magnitude and range but there are few partition boundaries; for large m, there are many boundaries but the basis functions are small in magnitude and highly local, hence incurring relatively small error. We have seen similar results in numerical experiments with different covariance models (e.g., Whittle) and larger range parameters (not shown).

To summarize, we approximate the forecast distribution as

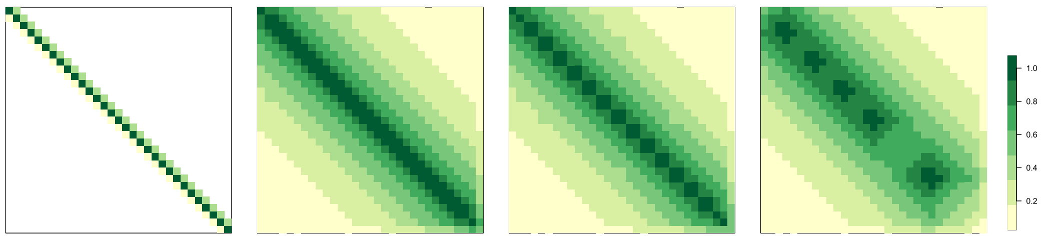
$$\mathbf{x}_t|\mathbf{y}_{1:t-1} \sim \mathcal{N}_n(\tilde{\boldsymbol{\mu}}_t, \tilde{\mathbf{L}}_t\tilde{\mathbf{L}}_t^\top), \tag{4}$$

where $\tilde{\boldsymbol{\mu}}_t = \mathcal{E}_t(\boldsymbol{\mu}_{t-1})$, and $\tilde{\mathbf{L}}_t = \text{decompress}(\mathcal{E}_t(\mathbf{L}_{t-1}^C))$ is computed as described above. The cost of our forecast step is essentially that of R evaluations of the evolution operator, which is highly



(a) Cholesky factor $\mathbf{L}$ (b) Compressed Chol. factor $\mathbf{L}^C$

Figure 4: Left: Sparse $n \times n$ Cholesky factor $\mathbf{L}$ of the exponential covariance matrix $\mathbf{\Sigma}$ mentioned in Figure 3. Right: Compressed Cholesky factor $\mathbf{L}^C$ of size $n \times R$. Here, $n = 32$, $M = 4$, $r_m \equiv 1$, and hence $R = \sum_{m=0}^{M} r_m = 5$.



(a) Evolution $\mathbf{E}_t$ (b) Exact forecast cov (c) Our forecast cov (d) RRUKF forecast

Figure 5: For the 32×32 exponential covariance $\mathbf{\Sigma} = \mathbf{\Sigma}_{t-1}$ on the unit interval also considered in Figures 3–4: (a): A linear advection-diffusion evolution operator $\mathbf{E}_t$ with advection parameter 0.01 and diffusion parameter 0.0001. (b)–(d): The forecast covariance $\tilde{\mathbf{\Sigma}}_t$ obtained using three different approaches based on $\mathbf{\Sigma}_{t-1}$ and $\mathbf{E}_t$. (b): Using exact evolution: $\tilde{\mathbf{\Sigma}}_t = \mathbf{E}_t \mathbf{\Sigma}_{t-1} \mathbf{E}_t^\top$. (c): Using our compression forecast: $\tilde{\mathbf{\Sigma}}_t = \tilde{\mathbf{L}}_t \tilde{\mathbf{L}}_t^\top$ with $\tilde{\mathbf{L}}_t = \text{decompress}(\mathcal{E}_t(\mathbf{L}_{t-1}^C))$, where the 32×5 $\mathbf{L}_{t-1}^C$ is shown in Figure 4b. (d): Using a reduced-rank unscented Kalman filter (RRUKF) with rank 5. RRUKF is considerably less accurate than our approach.

application-specific but always much smaller than the n evolution evaluations in the standard UKF.

3.3 Update Step

The forecast distribution in (4) can be viewed as the prior distribution at time t, and the goal of the update step is to obtain the posterior or filtering distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t}) \propto p(\mathbf{x}_t|\mathbf{y}_{1:t-1})p(\mathbf{y}_t|\mathbf{x}_t)$ based on new data $\mathbf{y}_t$. To do so, we use the Vecchia-Laplace approximation (Zilber and Katzfuss,

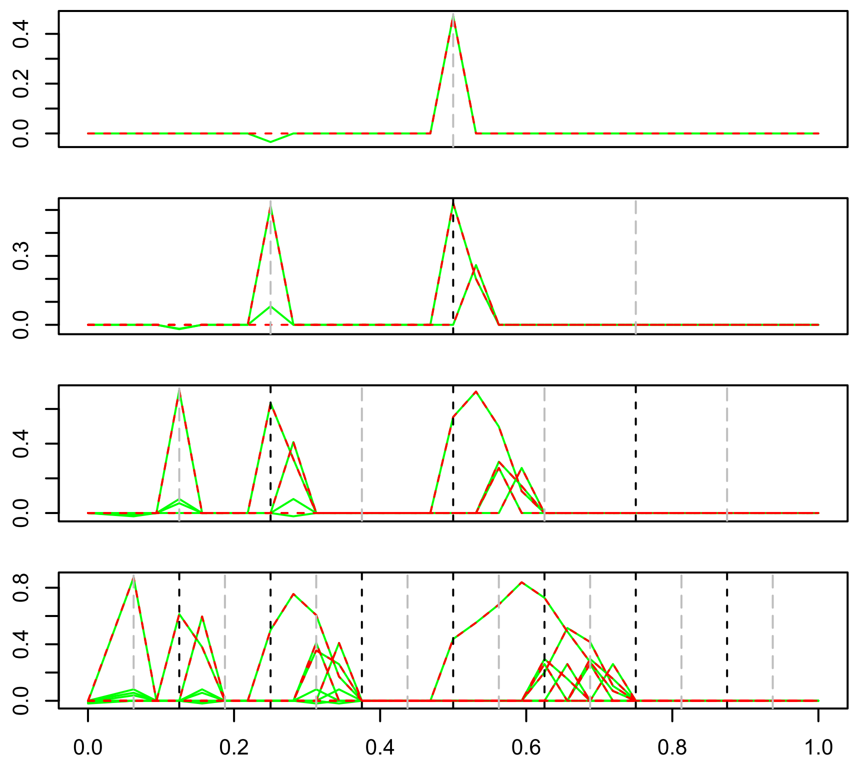


Figure 6: Basis-function representation of the Cholesky of the forecast covariance in the setting of Figures 3–5: Exact Cholesky $\mathbf{E}_t\mathbf{L}_{t-1}$ in solid green and our compressed forecast $\tilde{\mathbf{L}}_t = \text{decompress}(\mathcal{E}_t(\mathbf{L}^C_{t-1}))$ in dashed red.

2021) as in the update step of the EKVL filter (Jurek and Katzfuss, 2022), which is scalable and preserves the Cholesky sparsity. We review this update step here.

The Laplace approximation is a second-order (i.e., Gaussian) approximation of the log-posterior at its mode, which can be obtained using the Newton-Raphson algorithm. Starting with an initial guess $\mathbf{x}_t^{(0)}$ (e.g., $\mathbf{x}_t^{(0)} = \tilde{\boldsymbol{\mu}}_t$), we update the state estimate as $\mathbf{x}_t^{(l+1)} = h(\mathbf{x}_t^{(l)})$ with

$$\begin{aligned} h(\mathbf{x}_t) &= \mathbf{x}_t - [\tfrac{\partial^2}{\partial \mathbf{x}_t \mathbf{x}_t'} \log(\hat{p}(\mathbf{y}_t|\mathbf{x}_{1:t}))]^{-1} \tfrac{\partial}{\partial \mathbf{x}_t} \log(\hat{p}(\mathbf{y}_t|\mathbf{x}_{1:t})) \\ &= \mathbf{x}_t + \mathbf{W}_{\mathbf{x}_t}^{-1}\mathbf{D}_{\mathbf{x}_t}^{-1}(\mathbf{v}_{\mathbf{x}_t} - \mathbf{x}_t), \end{aligned} \tag{5}$$

where $\mathbf{W}_{\mathbf{x}_t} = [\tilde{\mathbf{L}}_t\tilde{\mathbf{L}}_t^\top]^{-1} + \mathbf{D}_{\mathbf{x}_t}^{-1}$, $\mathbf{v}_{\mathbf{x}_t} = \mathbf{x}_t + \mathbf{D}_{\mathbf{x}_t}\mathbf{u}_{\mathbf{x}_t}$,

$$u_i(x_{ti}) = \mathbf{1}_{\{i\in\mathcal{I}_t\}} \tfrac{\partial}{\partial x_{ti}} \log(g_{ti}(y_{ti}|x_{ti})), \quad \text{and} \quad \mathbf{u}_{\mathbf{x}_t} = (u_1(x_{t1}), \ldots, u_n(x_{tn}))^\top,$$

$$d_i(x_{ti}) = -\mathbf{1}_{\{i\in\mathcal{I}_t\}} [\tfrac{\partial^2}{\partial x_{ti}^2} \log(g_{ti}(y_{ti}|x_{ti}))]^{-1}, \quad \text{and} \quad \mathbf{D}_{\mathbf{x}_t}^{-1} = \text{diag}(d_1(x_{t1}), \ldots, d_n(x_{tn})).$$

For Gaussian observation distributions, $\mathbf{v}_{\mathbf{x}_t} = \mathbf{y}_t$ and $\mathbf{D}_{\mathbf{x}_t}$ do not depend on the current estimate $\mathbf{x}_t$, and the Newton-Raphson procedure converges in a single iteration. For other observation distributions, typically only a small number of iterations are required for this second-order optimization method to converge such that $\mathbf{x}_t^{(l+1)} \approx \mathbf{x}_t^{(l)}$. We then set the filtering distribution as

$$\mathbf{x}_t|\mathbf{y}_{1:t} \sim \mathcal{N}_n(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t), \tag{6}$$

where $\boldsymbol{\mu}_t = \mathbf{x}_t^{(l+1)}$ and $\boldsymbol{\Sigma}_t = \mathbf{W}_{\boldsymbol{\mu}_t}^{-1} = \mathbf{L}_t\mathbf{L}_t^\top$.

A crucial and unique property of the hierarchical Cholesky sparsity structure is that it is preserved under inversion, and so $\tilde{\mathbf{L}}_t^{-1}$ and $\mathbf{W}_{\mathbf{x}_t}$ can be computed quickly. Further, the Cholesky factor $\mathbf{L}_t$ of $\mathbf{W}_{\mathbf{x}_t}$ also has the same sparsity structure as $\tilde{\mathbf{L}}_t$, and so this sparsity is preserved throughout the update step. Our update step requires only $\mathcal{O}(nR^2k)$ time, where k is the number of Newton-Raphson iterations; we have $k = 1$ for Gaussian observation distributions and often $k \leqslant 10$ otherwise.

4 Filtering for Lorenz Model

4.1 Comparison to EKVL

Lorenz models (e.g., Lorenz, 2005) are probably the most common testbed for data-assimilation techniques (e.g., Ott et al., 2004) and represent realistic features of atmospheric evolution. We consider a complex and high-dimensional model (Lorenz, 2005, Sect. 3) whose dynamics are described by

$$\tfrac{\partial}{\partial t}\tilde{x}_i = \tfrac{1}{K^2}\textstyle\sum_{l=-K/2}^{K/2}\sum_{j=-K/2}^{K/2} -\tilde{x}_{i-2K-l}\tilde{x}_{i-K-j} + \tilde{x}_{i-K+j-l}\tilde{x}_{i+K+j} - \tilde{x}_i + F, \tag{7}$$

where $\tilde{x}_{-i} = \tilde{x}_{n-i}$ to reflect that the model is defined on a circle. We set $K = 35$ and $F = 10$. The evolution operator $\mathcal{E}_t$ in our state-space model is obtained by solving (7) on a regular grid of size $n = 768$ on a circle using a 4-th order Runge-Kutta scheme with thirty internal steps of size $dt = 0.0005$, and setting $x_i = b\tilde{x}_i$ with $b = 0.2$, following Jurek and Katzfuss (2022). Our method scales quasi-linearly in n, and so it can in principle be applied to much higher dimensions. We chose $n = 768$ here for ease of comparison to an existing method that can be expensive for large n, but brief experiments with our method in higher dimensions (e.g., $n = 1{,}280$) were also successful (not shown).

At every time point t, we randomly select $|\mathcal{I}_t| = 77$ (i.e., roughly 10%) of the n grid locations to be observed. We take the initial parameters $\boldsymbol{\mu}_{0|0}$ and $\boldsymbol{\Sigma}_{0|0}$ as the sample mean and covariance matrix from a very long run of the Lorenz model (7). For $t = 1, \ldots, T = 40$, we simulate five datasets from (2) with the nonlinear evolution operator $\mathcal{E}_t$ as (7). We consider two simulation settings for the data model (1), one with Gaussian observations with variance $\tau^2 = 0.1$ and one with Gamma observations, $y_{ti}|x_{ti} \sim \mathcal{G}(2, 2\exp(-x_{ti}))$.

We compare our proposed compressed HV (CHV) filter to the EKVL filter (Jurek and Katzfuss, 2022), whose forecast step relies on approximating the evolution function $\mathcal{E}_t$ by its first-order Taylor approximation, $\mathbf{E}_t \approx \frac{\partial\mathcal{E}_t(\mathbf{x}_{t-1})}{\partial\mathbf{x}_{t-1}}\big|_{\mathbf{x}_{t-1}=\boldsymbol{\mu}_{t-1}}$, where $\boldsymbol{\mu}_{t-1}$ is the filtering mean at time $t-1$. The EKVL filter is a natural competitor in our setting, as it strongly outperformed the EnKF on the Lorenz (2005, Sect. 3) model (see Jurek and Katzfuss, 2022), and it has the same update step as our CHV, allowing us to study the impact of our proposed compressed-Cholesky forecast step. For both competitors, we use the same hierarchical domain partitioning shown in Figure 7 with $M = 7$, $r_0 = 6$, and $r_m = 3$ for $m = 1, \ldots, M$, and hence $R = 27$.

An example of a Lorenz trajectory and the corresponding CHV filtering result is shown in Figure 8. The CHV filter tracks the true state well.

Figure 9 shows the comparison results in terms of the log-score (e.g., Gneiting and Katzfuss, 2014) given by the average log filtering density evaluated at the true state vector, $\log\mathcal{N}_n(\mathbf{x}_t|\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$. The log-score for our CHV method is stable over time, whereas the EKVL becomes increasingly inaccurate as time progresses. In addition, our method does not require linearization

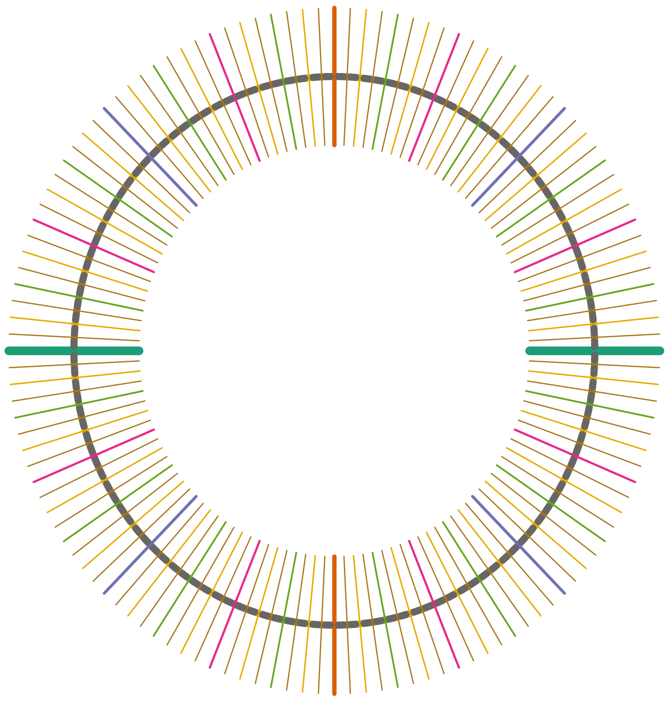
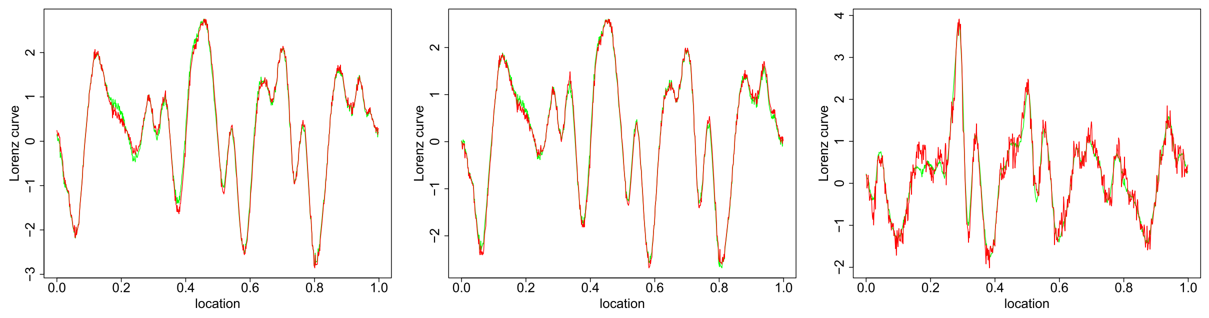


Figure 7: Hierarchical multi-resolution decomposition of $n = 768$ locations on a circle with $M = 7$, $r_0 = 6$, and $r_m = 3$ for $1 \leqslant m \leqslant M$. Lines denote domain partition boundaries, with decreasing line width for increasing resolution. For example, the first two splits are denoted by the horizontal green and vertical red lines, respectively. Because of the density of the locations, the points representing them (in black) seemingly blend together into a solid circle line.



(a) $t = 1$ (b) $t = 3$ (c) $t = 30$

Figure 8: For time points $t = 1, 3, 30$, example of true Lorenz states $\mathbf{x}_t$ (green) and the corresponding CHV filtering means $\boldsymbol{\mu}_t$.

of the evolution model and hence is significantly faster; CHV completed in less than 8 minutes for $T = 40$ time points, which is only approximately 20% of the time required by the EKVL filter.

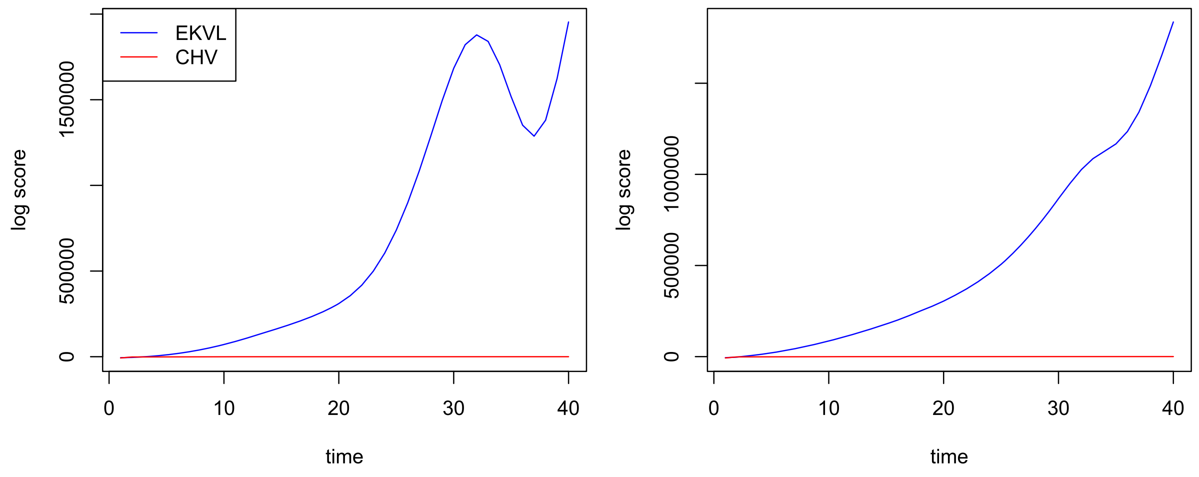


Figure 9: For the Lorenz setting, accuracy comparison in terms of log-scores (lower is better) of the filtering distributions of our CHV filter and the existing EKVL filter, for Gaussian (left) and Gamma (right) observation distributions in (1).

4.2 Comparison to RRUKF

We also considered a comparison to the UKF (Wan and Van Der Merwe, 2000), but this approach is not feasible for large n, because it requires roughly $2n$ evaluations of the evolution operator at each time point and requires $\mathcal{O}(n^3)$ time for the update step. The RRUKF (Chandrasekar et al., 2008) is more comparable in cost to our CHV approach if the RRUKF rank is equal to the R used in our CHV. As this RRUKF is only applicable to SSMs with additive innovation error, we carry out a separate comparison to the RRUKF in this setting here.

We consider the same setting as in Section 4.1, except that the evolution equation is now given by,

$$\mathbf{x}_t = \mathcal{E}_t(\mathbf{x}_{t-1}) = \mathcal{L}(\mathbf{x}_{t-1}) + \mathbf{w}_t, \tag{8}$$

where $\mathcal{L}$ denotes the Lorenz model (7), and we assume $\mathbf{w}_t \sim \mathcal{N}_n(\mathbf{0}, \tau^2\mathbf{I}_n)$ with $\tau^2 = 10^{-8}$.

We apply our CHV approach using the same partitioning scheme as in Section 4.1, and compare it to RRUKF with rank 76, which is much larger than the R used for CHV. As shown in Figure 10, our CHV method is much more accurate than RRUKF, as may have been expected from the poor accuracy of RRUKF in Figure 5d. In addition, CHV is 8 times faster than the RRUKF, which makes our method a great filtering tool.

5 Conclusion

We proposed a scalable filtering algorithm for nonlinear and non-Gaussian state-space models using a novel forecast technique relying on compression and decompression of Cholesky factors with a hierarchical sparsity structure. Our algorithm preserves the sparsity structure of the Cholesky factor over time, resulting in fast computation and low memory requirements. No linearization of the evolution operator is required, as the operator is simply applied to a small number of vectors that can be viewed as basis functions representing spatial structure at various resolutions. We showed that the filter can be considerably more accurate than state-of-the-art competitors on the popular Lorenz model.

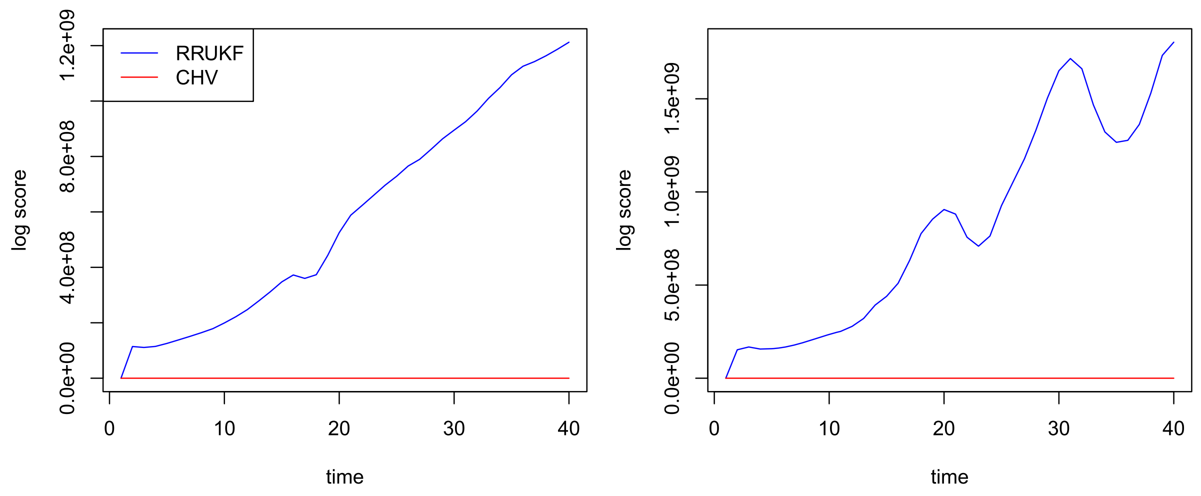


Figure 10: For the Lorenz setting with additive model error, accuracy comparison in terms of log-scores (lower is better) of the filtering distributions of our CHV filter and the RRUKF filter, for Gaussian (left) and Gamma (right) observation distributions in (1).

While we focused on spatio-temporal state-space models, our method can also be applied in more general settings where the assumption of locality of the underlying dynamics discussed in Section 3.2 is reasonable and where a hierarchical partitioning of the input space as discussed in Section 2.1 is feasible, for example based on a correlation distance (Kang and Katzfuss, 2021).

Supplementary Material

`R` code to reproduce our results and figures is available at https://github.com/katzfuss-group/CHVfilter.

Funding

MK was partially supported by National Science Foundation (NSF) Grants DMS–1654083 and DMS–1953005, and by the National Aeronautics and Space Administration (80NM0018F0527).

References

Arasaratnam I, Haykin S (2009). Cubature Kalman filters. *IEEE Transactions on Automatic Control*, 54(6): 1254–1269.

Castrillón-Candás JE, Genton MG, Yokota R (2016). Multi-level restricted maximum likelihood covariance estimation and kriging for large non-gridded spatial datasets. *Spatial Statistics*, 18: 105–124. Spatial Statistics Avignon: Emerging Patterns.

Castrillón-Candás JE, Li J, Eijkhout V (2013). A discrete adapted hierarchical basis solver for radial basis function interpolation. *BIT*, 53(1): 57–86.

Chandrasekar J, Kim IS, Bernstein DS, Ridley AJ (2008). Reduced-rank unscented Kalman filtering using Cholesky-based decomposition. In: *2008 American Control Conference*, 1274–1279.

Datta A, Banerjee S, Finley AO, Gelfand AE (2016). Hierarchical nearest-neighbor Gaussian process models for large geostatistical datasets. *Journal of the American Statistical Association*, 111(514): 800–812.

Fang C, Liu J, Ye S, Zhang J (2020). The geometric unscented Kalman filter. arXiv preprint: https://arxiv.org/abs/2009.13079.

Gneiting T, Katzfuss M (2014). Probabilistic forecasting. *Annual Review of Statistics and Its Application*, 1(1): 125–151.

Gordon N, Salmond D, Smith A (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings. Part F. Radar and Signal Processing*, 140(2): 107–113.

Grewal MS, Andrews AP (1993). *Kalman Filtering: Theory and Applications*. Prentice Hall.

Guinness J (2018). Permutation and grouping methods for sharpening Gaussian process approximations. *Technometrics*, 60(4): 415–429.

Jurek M, Katzfuss M (2022). Hierarchical sparse Cholesky decomposition with applications to high-dimensional spatio-temporal filtering. *Statistics and Computing*, 32: 15.

Kalman R (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1): 35–45.

Kang M, Katzfuss M (2021). Correlation-based sparse inverse Cholesky factorization for fast Gaussian-process inference. arXiv preprint: https://arxiv.org/abs/2112.14591.

Katzfuss M (2017). A multi-resolution approximation for massive spatial datasets. *Journal of the American Statistical Association*, 112(517): 201–214.

Katzfuss M, Guinness J (2021). A general framework for Vecchia approximations of Gaussian processes. *Statistical Science*, 36(1): 124–141.

Katzfuss M, Guinness J, Gong W, Zilber D (2020). Vecchia approximations of Gaussian-process predictions. *Journal of Agricultural, Biological, and Environmental Statistics*, 25(3): 383–414.

Khazraj H, Faria da Silva F, Bak CL (2016). A performance comparison between extended Kalman Filter and unscented Kalman Filter in power system dynamic state estimation. In: *2016 51st International Universities Power Engineering Conference (UPEC)*, 1–6.

Liu JS, Chen R (1998). Sequential Monte Carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93(443): 1032–1044.

Lorenz EN (2005). Designing chaotic models. *Journal of the Atmospheric Sciences*, 62(5): 1574–1587.

Meng D, Miao L, Shao H, Shen J (2018). A seventh-degree cubature Kalman filter. *Asian Journal of Control*, 20(1): 250–262.

Nychka DW, Anderson JL (2010). Data assimilation. In: *Handbook of Spatial Statistics* (AE Gelfand, PJ Diggle, M Fuentes, P Guttorp, eds.), 477–494. CRC Press. Chapter 27.

Ott E, Hunt BR, Szunyogh I, Zimin AV, Kostelich EJ, Corazza M, et al. (2004). A local ensemble Kalman filter for atmospheric data assimilation. *Tellus A*, 56: 415–428.

Pitt MK, Shephard N (1999). Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, 94(446): 590–599.

Schäfer F, Katzfuss M, Owhadi H (2021). Sparse Cholesky factorization by Kullback-Leibler minimization. *SIAM Journal on Scientific Computing*, 43(3): A2019–A2046.

Shumway RH, Stoffer DS (2000). *Time Series Analysis and Its Applications*. Springer.

St-Pierre M, Gingras D (2004). Comparison between the unscented Kalman filter and the extended Kalman filter for the position estimation module of an integrated navigation information system. In: *IEEE Intelligent Vehicles Symposium, 2004*, 831–835. IEEE.

Stein ML, Chi Z, Welty L (2004). Approximating likelihoods for large spatial data sets. *Journal* *of the Royal Statistical Society, Series B*, 66(2): 275–296.

Vecchia A (1988). Estimation and model identification for continuous spatial processes. *Journal of the Royal Statistical Society, Series B*, 50(2): 297–312.

Wan E, Van Der Merwe R (2000). The unscented Kalman filter for nonlinear estimation. In: *Adaptive Systems for Signal Processing, Communications, and Control. Lake Louise, Canada*, 153–158.

Wang S, Feng J, Chi KT (2013). Spherical simplex-radial cubature Kalman filter. *IEEE Signal Processing Letters*, 21(1): 43–46.

West M, Harrison J (1997). *Bayesian Forecasting and Dynamic Models. Springer Series in Statistics*. Springer-Verlag.

Zilber D, Katzfuss M (2021). Vecchia-Laplace approximations of generalized Gaussian processes for big non-Gaussian spatial data. *Computational Statistics & Data Analysis*, 153: 107081.