Received: 30 March 2022 | Revised: 21 July 2022 | Accepted: 30 July 2022

DOI: 10.1002/env.2757

RESEARCH ARTICLE

WILEY

Scalable spatio-temporal smoothing via hierarchical sparse Cholesky decomposition

Marcin Jurek[1] | **Matthias Katzfuss**[2]

[1]Department of Statistics and Data Science, University of Texas at Austin, Austin, Texas, USA

[2]Department of Statistics, Texas A&M University, College Station, Texas, USA

Correspondence
Matthias Katzfuss, Department of Statistics, Texas A&M University, College Station, TX, USA.
Email: katzfuss@gmail.com

Funding information
National Aeronautics and Space Administration, Grant/Award Number: 80NM0018F0527; National Science Foundation, Grant/Award Numbers: DMS-1654083, DMS-1953005, CCF-1934904

Abstract

We propose an approximation to the forward filter backward sampler (FFBS) algorithm for large-scale spatio-temporal smoothing. FFBS is commonly used in Bayesian statistics when working with linear Gaussian state-space models, but it requires inverting covariance matrices which have the size of the latent state vector. The computational burden associated with this operation effectively prohibits its applications in high-dimensional settings. We propose a scalable spatio-temporal FFBS approach based on the hierarchical Vecchia approximation of Gaussian processes, which has been previously successfully used in spatial statistics. On simulated and real data, our approach outperformed a low-rank FFBS approximation.

KEYWORDS

data assimilation, smoothing, spatio-temporal statistics, state-space model, Vecchia approximation

1 | INTRODUCTION

Developments in data collection and storage technologies over the past decade have led to an unprecedented influx of data across scientific disciplines. Environmental sciences in particular have profited immensely from these advances. For example, frequent and high-resolution measurements of carbon dioxide acquired by the Orbiting Carbon Observatory (Sun et al., 2017) helped to increase the understanding of CO_2 sinks and sources. Massive remotely sensed data were demonstrated to be of help in determining the concentration of volcanic ash in the atmosphere (Bugliaro et al., 2021), which is crucial for air traffic control and weather forecasting. Not all big datasets are collected using satellites however. Recently, Argo, a large system of autonomous floats, was deployed worldwide to collect data used in studying ocean temperature changes and the water cycle (Jayne et al., 2017).

Datasets of this kind are often spatio-temporal in nature and typically measure some scientifically interesting phenomenon. This leads the researchers to analyze them using a "mechanistic" approach. Within this paradigm, changes in time are represented by a (possibly discretized) differential equation, while the residual variation in space is captured using a purely statistical model (e.g., Wikle et al., 2019). Using this framework, data can be used to estimate the true value of the variable of interest, filling in the gaps where the observations are missing or inaccurate due to measurement errors, as well as to infer the unknown parameters. The first of these objectives is traditionally accomplished using the Kalman filter (Kalman, 1960) and smoother (also known as the Rauch–Tung–Striebel smoother, Rauch et al., 1965), while parameter inference is possible using a Gibbs sampler, often based on the forward filter backward sampler (FFBS, Carter & Kohn, 1994; Durbin & Koopman, 2002; Frühwirth-Schnatter, 1994).

A major challenge in using these existing techniques with big environmental data is their poor scalability as the number of observations or grid points grows. Specifically, the computational cost of the canonical versions of filtering and

smoothing methods is cubic in the number of observations at each time point. Countless approximations have been developed to address these problems, many of them being focused on filtering inference (see, e.g., Jurek & Katzfuss, 2022, and the citations therein). A particularly promising class of methods, which have recently gained prominence, are algorithms using an ensemble to represent the distribution of the state vector (Evensen et al., 2022; Grudzien & Bocquet, 2021), most notably the ensemble Kalman filter (Evensen, 1994; Katzfuss et al., 2020). Variational approaches led to the development of the so-called 4D-VAR algorithm (see, e.g., Evensen et al., 2022, for a comprehensive introduction), which has found mission-critical operational applications (see, e.g., ECMWF, 2021).

Relatively little attention has been devoted to smoothing. Among the existing works, Katzfuss and Cressie (2012) propose a method based on a low-rank approximation of the latent Gaussian random field, which scales well but may not be able to reproduce fine-scale features. Stroud et al. (2010) suffer from somewhat of the opposite problem, because it relies on tapering the sample covariance matrix and thus may struggle with smooth covariance functions (see numerical experiments in Jurek & Katzfuss, 2021). Sigrist et al. (2015) propose an approach based on spectral methods which are limited to observations on a regular grid. Another technique for approximate smoothing inference uses particle-based methods (Carvalho et al., 2010), but such methods cannot be used when the dimension of the latent space exceeds several hundred because of particle collapse. A method that is perhaps the closest to our in spirit is based on the ensemble Kalman smoother, which is reviewed and extended in Katzfuss et al. (2020). However, it also requires additional approximations such as tapering, and the number of distinct samples that it produces is always equal to the size of the ensemble, which can be inefficient.

We propose a scalable algorithm for generating samples from the smoothing distribution, directly approximating the FFBS algorithm, based on the hierarchical Vecchia (HV) approximation that has previously been used for spatio-temporal filtering (Jurek & Katzfuss, 2022). We summarize the previous results developed in the context of filtering and extend them to approximate smoothing inference. This is not straightforward because matrix approximations used in previous work cannot be easily applied in the context of smoothing. We conducted numerical experiments showing that our sampler outperformed a low-rank approximation and showing how our method can be used to estimate unknown parameters using a Gibbs sampler. We also applied our method to a real data set and showed that it performed better than a competing approach. The code and data needed to reproduce our results can be found at https://github.com/marcinjurek/scalable-FFBS.

This article is organized as follows. Section 2 introduces notation and briefly describes the linear Gaussian state space model and the canonical methods used for filtering, smoothing and sampling. Section 3 presents sparse Cholesky factorization and the HV approximation. In Section 4, we propose approximations to the canonical methods from Section 2 and conclude with a scalable version of the FFBS algorithm. Section 5 contains numerical experiments which demonstrate excellent performance of our approximate methods. Section 6 discusses an application to a real data set. Section 7 concludes and proposes directions for future research.

2 | SPATIO-TEMPORAL STATE-SPACE MODEL

Consider a Gaussian process $x(\cdot)$ defined over a domain $[1, 2, \dots, T] \times \mathcal{D} \subset \mathbb{R}^2$. Let $\mathcal{S} = \{\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_{n_c}\}$ be a grid over $\mathcal{D}$ and let $\mathbf{x}_t = \left[x(t, \mathbf{s}_1), x(t, \mathbf{s}_2), \dots, x(t, \mathbf{s}_{n_c})\right]^\top$. Note that the grid is taken to be the same at all time points, which is common in the case of big environmental datasets, for example those collected using remote sensing. We assume that the dynamics of the process at the subsequent time points can be expressed as an autoregressive model:

$$\mathbf{x}_t = \mathbf{E}_t \mathbf{x}_{t-1} + \mathbf{w}_t, \quad \mathbf{w}_t \sim \mathcal{N}_{n_c}(\mathbf{0}, \mathbf{Q}_t), \tag{1}$$

where the evolution matrix $\mathbf{E}_t$ is assumed to be sparse. We do not make any special additional assumptions regarding the covariance matrix $\mathbf{Q}_t$. The initial state follows a normal distribution: $\mathbf{x}_0 \sim \mathcal{N}_{n_c}(\boldsymbol{\mu}_{0|0}, \boldsymbol{\Sigma}_{0|0})$.

We consider a situation in which at each time point we are given $\mathbf{y}_t$, an n_t-dimensional vector of data observed at time $t = 1, 2, \dots, T$, related to the true process through a linear function:

$$\mathbf{y}_t = \mathbf{H}_t \mathbf{x}_t + \mathbf{v}_t, \quad \mathbf{v}_t \sim \mathcal{N}_{n_t}(\mathbf{0}, \mathbf{R}_t). \tag{2}$$

We assume that observation error covariance matrix $\mathbf{R}_t$ is diagonal. (This can be extended to block-diagonal $\mathbf{R}_t$ with small blocks.) We use $\mathbf{y}_{1:t} := (\mathbf{y}_1^\top, \dots, \mathbf{y}_t^\top)^\top$ to denote a vector of observations from time 1 to time t and we define $\mathbf{x}_{1:t}$

analogously. At each time t, the locations of the observations $\mathbf{y}_t$ can be a (different) subset of size n_t of the grid $\mathcal{S}$, indicated by the $n_t \times n$ matrix $\mathbf{H}_t$. In this article, we are interested in obtaining the filtering and smoothing distributions of $\mathbf{x}_t$ for $t = 1, 2, \dots, T$, that is, $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ and $p(\mathbf{x}_t|\mathbf{y}_{1:T})$, respectively. To accomplish this goal, we start with the canonical algorithms for filtering and generating samples from the smoothing distribution.

2.1 | The filtering distribution

Under the assumptions introduced in Section 2 the filtering distribution, $[\mathbf{x}_t|\mathbf{y}_{1:t}]$ is Gaussian and can be obtained using the Kalman filter (Kalman, 1960). We use $\boldsymbol{\mu}_{t|t}$ to denote $\mathbb{E}[\mathbf{x}_t|\mathbf{y}_{1:t}]$ and set $\boldsymbol{\Sigma}_{t|t} := \mathrm{Cov}(\mathbf{x}_t|\mathbf{y}_{1:t})$. To derive the Kalman filtering procedure, we first give the one-step ahead forecasting distribution:

$$\mathbf{x}_t|\mathbf{y}_{1:t-1} \sim \mathcal{N}_{n_\mathcal{G}}(\boldsymbol{\mu}_{t|t-1}, \boldsymbol{\Sigma}_{t|t-1}),$$

where $\boldsymbol{\mu}_{t|t-1} := \mathbf{E}_t\boldsymbol{\mu}_{t-1|t-1}$ and $\boldsymbol{\Sigma}_{t|t-1} := \mathbf{E}_t\boldsymbol{\Sigma}_{t-1|t-1}\mathbf{E}_t^\top + \mathbf{Q}_t$.

Based on Bayes' theorem, it follows that $[\mathbf{x}_t|\mathbf{y}_{1:t}] \propto [\mathbf{y}_t|\mathbf{x}_t][\mathbf{x}_t|\mathbf{y}_{1:t-1}]$. Thus, we have

$$\begin{aligned}\boldsymbol{\mu}_{t|t} &:= \boldsymbol{\mu}_{t|t-1} + \mathbf{K}_t(\mathbf{y}_t - \mathbf{H}_t\boldsymbol{\mu}_{t|t-1}),\\ \boldsymbol{\Sigma}_{t|t} &:= \boldsymbol{\Sigma}_{t|t-1} - \mathbf{K}_t\mathbf{H}_t\boldsymbol{\Sigma}_{t|t-1},\end{aligned}$$

where $\mathbf{K}_t := \boldsymbol{\Sigma}_{t|t-1}\mathbf{H}_t^\top(\mathbf{H}_t\boldsymbol{\Sigma}_{t|t-1}\mathbf{H}^\top + \mathbf{R}_t)^{-1}$ is the $n_\mathcal{G} \times n_t$ Kalman gain matrix.

Algorithm 1. Kalman filter (KF)

Input: moments of the initial distribution $\boldsymbol{\mu}_{0|0}, \boldsymbol{\Sigma}_{0|0}$, evolution model $\{\mathbf{E}_t, \mathbf{Q}_t\}_{t=0}^T$, observation model $\{\mathbf{H}_t, \mathbf{R}_t\}_{t=0}^T$, data $\{\mathbf{y}_t\}_{t=0}^T$

Result: moments of the filtering distribution $\left\{\boldsymbol{\mu}_{t|t}, \boldsymbol{\Sigma}_{t|t}\right\}_{t=0}^T$

1: **for** $t = 1, 2, \dots, T$ **do**
2: Compute forecast mean $\boldsymbol{\mu}_{t|t-1} = \mathbf{E}_t\boldsymbol{\mu}_{t-1|t-1}$.
3: Compute forecast covariance $\boldsymbol{\Sigma}_{t|t-1} = \mathbf{E}_t\boldsymbol{\Sigma}_{t-1|t-1}\mathbf{E}_t^T + \mathbf{Q}_t$.
4: Calculate $\mathbf{K}_t = \boldsymbol{\Sigma}_{t|t-1}\mathbf{H}_t^\top(\mathbf{H}_t\boldsymbol{\Sigma}_{t|t-1}\mathbf{H}_t^\top + \mathbf{R}_t)^{-1}$.
5: Calculate filtering mean $\boldsymbol{\mu}_{t|t} = \boldsymbol{\mu}_{t|t-1} + \mathbf{K}_t(\mathbf{y}_t - \mathbf{H}_t\boldsymbol{\mu}_{t|t-1})$.
6: Calculate filtering covariance $\boldsymbol{\Sigma}_{t|t} = \boldsymbol{\Sigma}_{t|t-1} - \mathbf{K}_t\mathbf{H}_t\boldsymbol{\Sigma}_{t|t-1}$.
7: **end for**

2.2 | Kalman smoother

Computing the smoothing distribution can be accomplished using the Kalman smoother (Rauch et al., 1965). Let $\boldsymbol{\mu}_{t|T} := \mathbb{E}(\mathbf{x}_t|\mathbf{y}_{1:T})$ and $\boldsymbol{\Sigma}_{t|T} := \mathrm{Cov}(\mathbf{x}_t|\mathbf{y}_{1:T})$. Then the linear Gaussian state-space model of Section 2 implies that the smoothing distribution will also be Gaussian: $\mathbf{x}_t|\mathbf{y}_{1:T} \sim \mathcal{N}_{n_\mathcal{G}}(\boldsymbol{\mu}_{t|T}, \boldsymbol{\Sigma}_{t|T})$. Notice that

$$[\mathbf{x}_t|\mathbf{y}_{1:T}] = \int [\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{y}_{1:T}][\mathbf{x}_{t+1}|\mathbf{y}_{1:T}]\, d\mathbf{x}_{t+1},$$

where $[\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{y}_{1:T}] \propto [\mathbf{x}_{t+1}|\mathbf{x}_t][\mathbf{x}_t|\mathbf{y}_{1:t}]$. It follows that the conditional mean and conditional covariance in the smoothing distribution are given by

$$\begin{aligned}\boldsymbol{\mu}_{t|T} &:= \boldsymbol{\mu}_{t|t} + \mathbf{J}_t(\boldsymbol{\mu}_{t+1|T} - \boldsymbol{\mu}_{t+1|t}),\\ \boldsymbol{\Sigma}_{t|T} &:= \boldsymbol{\Sigma}_{t|t} + \mathbf{J}_t(\boldsymbol{\Sigma}_{t+1|T} - \boldsymbol{\Sigma}_{t+1|t})\mathbf{J}_t^\top,\end{aligned}$$

where $\mathbf{J}_t := \boldsymbol{\Sigma}_{t|t}\mathbf{E}_{t+1}^\top\boldsymbol{\Sigma}_{t+1|t}^{-1}$.

Algorithm 2. Kalman smoother (KS)

Input: moments of the initial distribution $\boldsymbol{\mu}_{0|0}, \boldsymbol{\Sigma}_{0|0}$, evolution model $\{\mathbf{E}_t, \mathbf{Q}_t\}_{t=0}^{T}$, observation model $\{\mathbf{H}_t, \mathbf{R}_t\}_{t=0}^{T}$, data $\{\mathbf{y}_t\}_{t=0}^{T}$

Result: moments of the smoothing distribution $\left\{\boldsymbol{\mu}_{t|T}, \boldsymbol{\Sigma}_{t|T}\right\}_{t=0}^{T}$

1: Obtain moments of the filtering distribution $\left\{\boldsymbol{\mu}_{t|t}, \boldsymbol{\Sigma}_{t|t}\right\}_{t=0}^{T}$ using KF (Algorithm 1).
2: **for** $t = T-1, T-2, \ldots, 1$ **do**
3: Compute $\mathbf{J}_t = \boldsymbol{\Sigma}_{t|t}\mathbf{E}_{t+1}^{\top}\boldsymbol{\Sigma}_{t+1|t}^{-1}$
4: Compute smoothing mean $\boldsymbol{\mu}_{t|T} = \boldsymbol{\mu}_{t|t} + \mathbf{J}_t(\boldsymbol{\mu}_{t+1|T} - \boldsymbol{\mu}_{t+1|t})$,
5: **end for**

Algorithm 3. Forward filter backward sampler (Durbin & Koopman, 2002; Jarociński, 2015)

Input: moments of the initial distribution $\boldsymbol{\mu}_{0|0}, \boldsymbol{\Sigma}_{0|0}$, evolution model $\{\mathbf{E}_t, \mathbf{Q}_t\}_{t=0}^{T}$, observation model $\{\mathbf{H}_t, \mathbf{R}_t\}_{t=0}^{T}$, data $\{\mathbf{y}_t\}_{t=0}^{T}$, desired number of samples N_{samp}

Result: sample from the smoothing distribution: $\mathbf{x}_{1:T}$

1: Generate $\hat{\mathbf{x}}_{0|0} \sim \mathcal{N}_{n_G}(\mathbf{0}, \boldsymbol{\Sigma}_{0|0})$.
2: Generate $\hat{\mathbf{x}}_{1:T}$ and $\hat{\mathbf{y}}_{1:T}$ using (1) and (2).
3: Calculate $\mathbf{y}^*_{1:T}$ where $\mathbf{y}^*_t = \mathbf{y}_t - \hat{\mathbf{y}}_t$.
4: Use KS (Algorithm 2) to obtain $\{\hat{\boldsymbol{\mu}}_{t|T}\}_{t=1}^{T}$, where $\hat{\boldsymbol{\mu}}_{t|T} = \mathbb{E}(\mathbf{x}_{1:T}|\hat{\mathbf{y}}^*_{1:t})$.
5: **for** $t = 1, \ldots, T$ **do**
6: $\mathbf{x}_t = \hat{\mathbf{x}}_t + \hat{\boldsymbol{\mu}}_{t|T}$ is a sample from $\left[\mathbf{x}_t|\mathbf{y}_{1:T}\right]$.
7: **end for**

The full Kalman smoother typically can compute also the smoothing covariance matrix $\boldsymbol{\Sigma}_{t|T} = \boldsymbol{\Sigma}_{t|t} + \mathbf{J}_t(\boldsymbol{\Sigma}_{t+1|T} - \boldsymbol{\Sigma}_{t+1|t})\mathbf{J}_t^{\top}$. We skip this calculation in our Algorithm 2, as it is not necessary for the construction of the algorithm which samples from the smoothing distribution.

2.3 | Forward filter backward sampler

In Bayesian statistics instead of calculating the full smoothing distribution, it is often enough to be able to draw samples from $[\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{y}_{1:T}]$. This is particularly true in Markov chain Monte Carlo (MCMC)-based methods. Inspired by this fact, some authors (Carter & Kohn, 1994; Durbin & Koopman, 2002; Frühwirth-Schnatter, 1994) developed algorithms which draw a sample from (1) and (2) and then linearly transform it based on actual observations from (2) to obtain a sample from the smoothing distribution. It is preferable to simulation using moments generated by the Kalman smoother, which would require, in general, factorization of all smoothing covariance matrices $\boldsymbol{\Sigma}_{t|T}$. We briefly summarize the algorithm known as FFBS (Durbin & Koopman, 2002) below, using the helpful insights from Jarociński (2015).

We note that a sample from the smoothing distribution can also be used as the approximation of the full distribution. For example, if we are interested in prediction, the sample mean and quantiles can be used as a tool for making predictions and quantifying uncertainty, respectively.

2.4 | Computational complexity

Algorithms 2 and 3 rely on calculating the correction factor $\mathbf{J}_t$, which requires computing the inverse of the forecast covariance matrix $\boldsymbol{\Sigma}_{t|t-1}$. In the case of Algorithm 1, a prerequisite for the other two, we also need to obtain the Kalman gain matrix $\mathbf{K}_t$ which is a linear function of the inverse of $\mathbf{H}_t\boldsymbol{\Sigma}_{t|t-1}\mathbf{H}^{\top} + \mathbf{R}_t$. This proves to be the computational bottleneck, since the number of operations required for matrix inversion is proportional to the cube of its dimension. As the size of the grid n_G and the number of observations at each time point n_t grow, these inversion operations take a prohibitive

amount of time. In the next section, we review the sparse Cholesky factorization method and subsequently show how it can be used to approximate Algorithms 1–3.

3 | SPARSE CHOLESKY FACTORIZATION

3.1 | HV approximation

In this section, we describe the HV approximation. It has recently been shown that this approach ensures that the sparsity of the approximate Cholesky factor of the filtering covariance matrix is the same at all time points (Jurek & Katzfuss, 2022). Moreover, following the findings of Schäfer et al. (2021) the approximation to the forecast distribution at each time point is optimal in the sense of KL-divergence, given the sparsity pattern. Here we summarize a special case of the Vecchia approximation which was shown to be near optimal (Zilber & Katzfuss, 2021) and which additionally has the property of preserving the sparsity of the Cholesky decomposition of the covariance matrix under inversion. As we show in the following sections, this characteristic is fundamental for a construction of a scalable FFBS.

We start by defining an order relation $\prec$ among the elements of the grid $\mathcal{S}$ using the maxmin ordering (Schäfer et al., 2021). From now on we assume that the elements of $\mathbf{x}_0$ are sorted according to $\prec$. We then define a directed acyclic graph over the subsets of elements of $\mathbf{x}_0$ in the following way. We begin by selecting the first r_0 elements of $\mathbf{x}_0$, which we call knots, and label them as $\mathcal{K}^0$. Next we partition the remaining $n_{\mathcal{G}} - r_0$ variables into J groups $G_1, \dots, G_J$ and for each preserve the order $\prec$ truncated to members of that group. Finally, we select r_1 knots from each group and label them as $\mathcal{K}_j$ for $j = 1, \dots, J$. Variables $\mathcal{K}^1 = \{\mathcal{K}_1, \dots, \mathcal{K}_J\}$ form the next level of the hierarchy.

The remaining elements of each group are further partitioned. For example, the $\#G_j - r_1$ remaining elements of G_j are divided into sets $G_{j,1}, \dots, G_{j,J}$. Then r_2 first elements from each of those smaller groups are put into sets $\mathcal{K}_{j,1}, \dots, \mathcal{K}_{j,J}$. In this way, we obtain the second level of the hierarchy $\mathcal{K}^2 = \left\{\mathcal{K}_{1,1}, \dots, \mathcal{K}_{1,J}, \mathcal{K}_{2,1}, \dots, \mathcal{K}_{2,J}, \dots, \mathcal{K}_{J,J}\right\}$.

This hierarchy can be visually represented in the form of a directed graph $\mathcal{G} = (V, E)$ where $V = \mathcal{K}^0 \cup \mathcal{K}^1 \cup \cdots$ and E is defined as follows. For two vertices $\mathcal{K}_{j_1,\dots j_m}$ and $\mathcal{K}_{i_1,\dots,i_l}$ we have $\mathcal{K}_{j_1,\dots j_m} \to \mathcal{K}_{i_1,\dots,i_l}$ if $\mathcal{K}_{i_1,\dots,i_l} \subset G_{j_1,\dots j_m}$ and $\mathcal{K}_{j_1,\dots j_m} \leftarrow \mathcal{K}_{i_1,\dots,i_l}$ if $\mathcal{K}_{j_1,\dots j_m} \subset G_{j_1,\dots j_\ell}$. The construction of this hierarchy is illustrated in Figure 1.

We also introduce lexicographic order $\prec_L$ on vertices $\mathcal{K}_{j_1,\dots j_m}$ with respect to their subscripts and define $\mathbf{S}$ to be an adjacency matrix of graph $\mathcal{G}$. Note that this matrix is lower triangular because for $w, v \in V$ we can have $w \to v$ only if $w \prec_L v$.

Further details of the HV construction can be found in Jurek and Katzfuss (2022).

3.2 | Sparse Cholesky decomposition based on HV

With the sparsity pattern encoding the HV approximation we now modify the standard Cholesky factorization algorithm in the following way. If an i,jth element of the sparsity pattern matrix $\mathbf{S}$ equals 1, we calculate the corresponding element

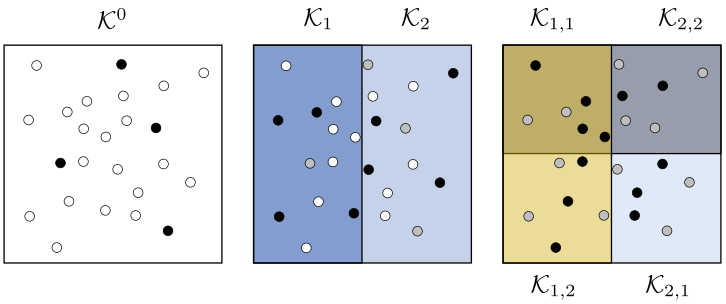

FIGURE 1 Construction of the hierarchical Vecchia approximation. The first panel shows the entire domain with each dot representing an element of $\mathbf{x}_0$; note that they do not need to be regularly spaced. The black dots, in accordance with the label above, represent the elements selected as $\mathcal{K}^0$. The second panel shows the domain split in two. The elements of $\mathbf{x}_0$ corresponding to locations in the left half are assigned to G_1, while the remaining elements are assigned to G_2. Black dots within each group denote elements of $\mathcal{K}^1$, the gray dots stand for elements already assigned to $\mathcal{K}^0$ and the remaining dots are white. The right panel shows another level of the hierarchy with each quadrant, from the top-left and counter-clockwise, the area covering $G_{1,1}, G_{1,2}, G_{2,1}$, and $G_{2,2}$. Similar to the middle panel, the gray dots represent elements of $\mathcal{K}^0 \cup \mathcal{K}^1$ and black dots represent elements of $\mathcal{K}^2$ split into four sets $K_{i,j}$ such that $K_{i,j} \subset G_{i,j}$ for $i, j \in \{1, 2\}$.

Algorithm 4. Hierarchical Cholesky factorization (HCF)

Input: Sparsity pattern matrix $\mathbf{S}$, p.d. matrix $\mathbf{A}$ of size $n \times n$
Result: Sparse Cholesky factor $\mathbf{L}$

1: **for** $i = 1, \dots, n$ **do**
2: **for** $j = 1, \dots, i$ **do**
3: $\mathbf{L}_{i,j} = \mathbf{S}_{i,j} \cdot (\mathbf{A}_{i,j} - \sum_{k=1}^{j-1} \mathbf{L}_{i,k}\mathbf{L}_{j,k})/\mathbf{L}_{j,j}$
4: **end for**
5: $\mathbf{L}_{i,i} = (\mathbf{A}_{i,i} - \sum_{k=1}^{i-1} \mathbf{L}_{i,k}^2)^{1/2}$
6: **end for**

of the Cholesky factor, using the regular formula and set it to zero otherwise. Note also that given the HV construction the diagonal elements will always be calculated. Our approach is summarized in Algorithm 4.

If we use N to denote the maximum number of nonzero elements in a row of $\mathbf{S}$, then the complexity of Algorithm 4 is $\mathcal{O}(nN^2)$. This is because line 3 requires $\mathcal{O}(N)$ operations and is executed at most N times for each of the n rows.

4 | FAST SAMPLING USING SPARSE CHOLESKY FACTORIZATION

In this section, we show how Algorithm 4 (HCF) can be used to ensure the scalability of Algorithm 3 (FFBS). Recall that the most computationally intensive steps in Algorithm 3 were those calculating the $\mathbf{K}_t$ matrix in the forward pass and inverting the forecast covariance in the backward pass. We show how HCF can be used to accelerate both.

4.1 | Approximate filtering

The application of hierarchical Cholesky factorization to filtering was described previously (Jurek & Katzfuss, 2022) and we briefly summarize it here. Unlike in Algorithm 1, we do not calculate the entire filtering and forecast covariance matrices, $\mathbf{\Sigma}_{t|t-1}$ and $\mathbf{\Sigma}_{t|t}$, respectively, but rather their hierarchical Cholesky factor. In particular, given the prescribed sparsity $\mathbf{S}$, we approximate $\mathbf{\Sigma}_{t|t-1} \approx \tilde{\mathbf{\Sigma}}_{t|t-1} = \mathbf{L}_{t|t-1}\mathbf{L}_{t|t-1}^\top$, where $\mathbf{L}_{t|t-1} = \mathrm{HCF}(\mathbf{S}, \mathbf{\Sigma}_{t|t-1})$, which is optimal in the sense of KL divergence (Schäfer et al., 2021). The computational benefits of using this approximation can be further taken advantage of Jurek & Katzfuss (2022, Section 3.3) as shown in the following

Claim 1. Assume $\mathbf{L}_{t|t-1} = \mathrm{HCF}(\mathbf{S}, \mathbf{\Sigma}_{t|t-1})$, where $\mathbf{S}$ encodes the HV approximation, $\mathbf{\Sigma}_{t|t-1}$ is a (approximate or exact) forecast covariance matrix and that $\mathbf{P}$ is an order reversing permutation matrix. We have

$$\mathbf{U}_{t|t} = \mathbf{P}\mathrm{chol}(\mathbf{P}(\mathbf{L}_{t|t-1}^{-\top}\mathbf{L}_{t|t-1}^{-1} + \mathbf{H}_t\mathbf{R}_t^{-\top}\mathbf{H}_t)\mathbf{P})\mathbf{P}$$

and

$$\tilde{\mathbf{\Sigma}}_{t|t} = \mathbf{U}_{t|t}^{-\top}\mathbf{U}_{t|t}^{-1}.$$

We can thus define $\mathbf{L}_{t|t} := \mathbf{U}_{t|t}^{-\top}$. Then, as Jurek and Katzfuss (2022) noted, given $\mathbf{L}_{t|t-1}$ with at most N nonzero elements in a row, $\mathbf{L}_{t|t}$ has the same sparsity pattern as $\mathbf{L}_{t|t-1}$ and can be calculated in $\mathcal{O}(nN^2)$ time. These properties allow us to approximate Algorithm 1 (Kalman filter) using Algorithm 5, which Jurek and Katzfuss (2022) show to have $\mathcal{O}(nN^2T)$ time complexity.

Note that the approximate filtering and forecast means are denoted with a tilde over each symbol, to differentiate them from their exact counterparts calculated in Algorithm 1.

4.2 | Approximate sampling

Following Algorithm 2, we see that the most time consuming part of the backward pass is matrix inversion in line 3. Additionally, the multiplication of dense $n_\mathcal{G} \times n_\mathcal{G}$ matrices also requires much computation time for large $n_\mathcal{G}$. These bottlenecks

Algorithm 5. Hierarchical Vecchia filter (HVF)

Input: moments of the initial distribution $\boldsymbol{\mu}_{0|0}, \boldsymbol{\Sigma}_{0|0}$, evolution model $\{\mathbf{E}_t, \mathbf{Q}_t\}_{t=0}^T$, observation model $\{\mathbf{H}_t, \mathbf{R}_t\}_{t=0}^T$, data $\{\mathbf{y}_t\}_{t=0}^T$ and sparsity matrix $\mathbf{S}$

Result: approximate representation of the filtering and forecast distributions $\left\{\tilde{\boldsymbol{\mu}}_{t|t-1}, \mathbf{L}_{t|t-1}\right\}_{t=0}^T$, $\left\{\tilde{\boldsymbol{\mu}}_{t|t}, \mathbf{L}_{t|t}\right\}_{t=0}^T$

1: Calculate the HV sparsity matrix $\mathbf{S}$
2: Calculate $\mathbf{L}_{0|0} = \text{HCF}(\mathbf{S}, \boldsymbol{\Sigma}_{0|0})$
3: **for** $t = 1, 2, \dots, T$ **do**
4: Compute $\tilde{\boldsymbol{\mu}}_{t|t-1} = \mathbf{E}_t \tilde{\boldsymbol{\mu}}_{t-1|t-1}$
5: Calculate the (i,j)th elements of $\tilde{\boldsymbol{\Sigma}}_{t|t-1} = \mathbf{E}_t \mathbf{L}_{t-1|t-1} \mathbf{L}_{t-1|t-1}^\top \mathbf{E}_t^\top + \mathbf{Q}_t$, for (i,j) such that $\mathbf{S}_{i,j} = 1$
6: Calculate the HCF of the forecast matrix $\mathbf{L}_{t|t-1} = \text{HCF}(\mathbf{S}, \tilde{\boldsymbol{\Sigma}}_{t|t-1})$
7: Calculate $\mathbf{L}_{t|t}$ using Claim 1.
8: Compute $\tilde{\boldsymbol{\mu}}_{t|t} = \tilde{\boldsymbol{\mu}}_{t|t-1} + \mathbf{L}_{t|t} \mathbf{L}_{t|t}^\top \mathbf{H}_t^\top \mathbf{R}_t^{-1} \left(\mathbf{y}_t - \mathbf{H}_t \tilde{\boldsymbol{\mu}}_{t|t-1}\right)$
9: **end for**

Algorithm 6. Hierarchical Vecchia smoother (HVS)

Input: moments of the initial distribution $\boldsymbol{\mu}_{0|0}, \boldsymbol{\Sigma}_{0|0}$, evolution model $\{\mathbf{E}_t, \mathbf{Q}_t\}_{t=0}^T$, observation model $\{\mathbf{H}_t, \mathbf{R}_t\}_{t=0}^T$, data $\{\mathbf{y}_t\}_{t=0}^T$

Result: approximate mean of the smoothing distribution $\left\{\tilde{\boldsymbol{\mu}}_{t|T}\right\}_{t=0}^T$

1: Obtain representation of the forecast and filtering distributions $\left\{\tilde{\boldsymbol{\mu}}_{t|t}, \mathbf{L}_{t|t}\right\}_{t=0}^T$, $\left\{\tilde{\boldsymbol{\mu}}_{t|t-1}, \mathbf{L}_{t|t-1}\right\}_{t=0}^T$ using HVF (Algorithm 5)
2: **for** $t = T-1, T-2, \dots, 1$ **do**
3: Compute approximate smoothing mean as $\tilde{\boldsymbol{\mu}}_{t|T} = \tilde{\boldsymbol{\mu}}_{t|t} + \mathbf{L}_{t|t} \mathbf{L}_{t|t}^\top \mathbf{E}_{t+1}^\top \mathbf{L}_{t+1|t}^{-\top} \mathbf{L}_{t+1|t}^{-1} (\tilde{\boldsymbol{\mu}}_{t+1|T} - \tilde{\boldsymbol{\mu}}_{t+1|t})$,
4: **end for**

can be eliminated if matrices $\boldsymbol{\Sigma}_{t+1|t}$ and $\boldsymbol{\Sigma}_{t|t}$ are replaced with their hierarchical Cholesky factors $\mathbf{L}_{t+1|t}$ and $\mathbf{L}_{t|t}$, respectively. This substitution also decreases the cost of matrix multiplication, since all matrices in line 3 are now sparse. This let allows us to approximate Algorithm 3 by proposing a scalable FFBS in Algorithm 7.

Similar to Algorithm 5, we used symbols with a tilde to denote the approximations of corresponding variables in Algorithm 2. Regarding complexity of Algorithm 6, $\mathbf{U}_{t|t+1}$ is sparse with a known sparsity pattern which means that line 3 and can be executed in $\mathcal{O}(nN^2T)$ time (Jurek & Katzfuss, 2022). Line 4 can be executed efficiently series of matrix-vector multiplications is performed instead. The matrices $\mathbf{L}_{t|t}$ and $\mathbf{U}_{t+1|t}$ have at most N nonzero elements in each row (Jurek & Katzfuss, 2021). Therefore, if we recall the complexity of Algorithm 5 discussed in Section 4.1 and assume $\mathbf{E}_t$ is sparse, then operations in line 4 have complexity $\mathcal{O}(nN)$. Consequently, Algorithm 6 can be executed in $\mathcal{O}(nN^2T)$ time.

4.3 | Scalable FFBS

Using the approximations described in Sections 4.1 and 4.2, we can now provide an algorithm for a scalable FFBS. Following the approach adopted earlier in Section 3, we used the tilde notation to indicate approximations. Notice that we use Vecchia approximation in order to quickly calculate the square roots $\{\mathbf{L}_t\}_{t=1}^T$ of the model error covariance matrices $\{\mathbf{Q}_t\}_{t=1}^T$. These square roots are then be used for quick generation the synthetic data.

4.4 | Computational complexity

Using the HV approximation substantially reduces the computational cost of sampling from the smoothing distribution. If $\mathbf{S}$ corresponds to a HV approximation, the first line of the algorithm can be calculated in $\mathcal{O}(nN^2)$ time. The computationally

Algorithm 7. Scalable FFBS

Input: moments of the initial distribution $\boldsymbol{\mu}_{0|0}, \boldsymbol{\Sigma}_{0|0}$, evolution model $\{\mathbf{E}_t, \mathbf{Q}_t\}_{t=0}^{T}$, observation model $\{\mathbf{H}_t, \mathbf{R}_t\}_{t=0}^{T}$, data $\{\mathbf{y}_t\}_{t=0}^{T}$, and sparsity pattern $\mathbf{S}$

Result: sample $\mathbf{x}_{1:T}$ from the approximate smoothing distribution

1: Calculate $\mathbf{L}_{0|0} = \text{HCF}(\mathbf{S}, \boldsymbol{\Sigma}_{0|0})$.
2: Generate $\boldsymbol{\epsilon}_0 \sim \mathcal{N}_{n_\mathcal{G}}(\mathbf{0}, \mathbf{I}_{n_\mathcal{G}})$, and set $\hat{\mathbf{x}}_{0|0} = \mathbf{L}_{0|0}\boldsymbol{\epsilon}_0$.
3: **for** $t = 1, \ldots, T-1$ **do**
4: Calculate $\mathbf{L}_t^Q = \text{HCF}(\mathbf{S}, \mathbf{Q}_t)$
5: Calculate $\boldsymbol{\epsilon}_t \sim \mathcal{N}_{n_\mathcal{G}}(\mathbf{0}, \mathbf{I}_{n_\mathcal{G}})$ and set $\hat{\mathbf{w}}_t = \mathbf{L}_t^Q \boldsymbol{\epsilon}_t$.
6: **end for**
7: Generate $\hat{\mathbf{x}}_{1:T}$ and $\hat{\mathbf{y}}_{1:T}$ using (1) and (2), replacing $\mathbf{w}_t$ with $\hat{\mathbf{w}}_t$.
8: Calculate $\mathbf{y}^*_{1:T}$ where $\mathbf{y}^*_t = \mathbf{y}_t - \hat{\mathbf{y}}_t$.
9: Use HVS (Algorithm 2) to obtain $\{\hat{\hat{\boldsymbol{\mu}}}_{t|T}\}_{t=1}^{T}$, where $\hat{\hat{\boldsymbol{\mu}}}_{t|T} = \mathbb{E}(\mathbf{x}_t|\hat{\mathbf{y}}^*_{1:t})$.
10: **for** t = 1, dots, T **do**
11: $\mathbf{x}_t = \hat{\mathbf{x}}_t + \hat{\hat{\boldsymbol{\mu}}}_{t|T}$ is a sample from an approximation of $\left[\mathbf{x}_t|\mathbf{y}_{1:T}\right]$.
12: **end for**

intense operation in the second line is the matrix-vector multiplication, but because $\mathbf{L}_{0|0}$ has the same sparsity pattern as $\mathbf{S}$, this product can be obtained in $\mathcal{O}(nN)$ time. Analogous arguments let us conclude that the total cost of line 3 is $\mathcal{O}(nN^2T)$. Generating synthetic data $\hat{\mathbf{x}}_{1:T}$ and $\hat{\mathbf{y}}_{1:T}$ can be done in $\mathcal{O}(nNT)$ time, because we assumed that the evolution matrix $\mathbf{E}_t$ is sparse and that $\mathbf{R}_t$ is block diagonal with small blocks. The only operation in the remaining lines is the use of HV smoother in line 9, which requires $\mathcal{O}(nN^2T)$ time.

A typical user of Algorithm 7 will typically generate $N_{\text{samp}} > 1$ samples from the approximate smoothing distribution, which means that it will take $\mathcal{O}(nN^2TN_{\text{samp}})$ time.

5 | NUMERICAL COMPARISON

5.1 | Setup

In this section, we evaluate our scalable FFBS using simulated data.

We consider an advection diffusion process $x(\mathbf{s}, t)$ defined over $\mathbb{R}^2 \times [0, \ldots, T]$, which means that its dynamics are expressed by the following partial differential equation:

$$\frac{\partial x}{\partial t} = \alpha\left(\frac{\partial^2 x}{\partial^2 s_x} + \frac{\partial^2 x}{\partial^2 s_y}\right) + \beta\left(\frac{\partial x}{\partial s_x} + \frac{\partial x}{\partial s_y}\right) + \eta, \tag{3}$$

where $\eta(\mathbf{s}, t)$ is a zero-mean stationary Gaussian process with an exponential covariance function with marginal variance $\sigma^2_{\mathbf{w}} = 0.1$ and range $\lambda = 0.15$. This setting of λ allows the process to exhibit clear variation over the chosen grid (see below) but preserves substantial dependence between neighboring locations. We set $\alpha = 4 \times 10^{-5}$ and $\beta = 10^{-2}$ which leads to a stable differencing scheme for our chosen grid (below) while producing visible advection and diffusion. We also assume that $\eta(\cdot, \cdot)$ is independent across time. We then consider a regular grid of size $n_\mathcal{G} = 34 \times 34 = 1156$ covering the square $\mathcal{D} = [0, 1] \times [0, 1]$ and discretize x over this grid using centered finite differences. This results in a vector $\mathbf{x}_t$ with each component representing the value of x at a corresponding grid point and gives a discrete version of (3) which takes the form (1). We use $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{0|0})$, where $\boldsymbol{\Sigma}_{0|0}$ corresponds to the exponential covariance function with range λ, marginal variance $\sigma^2_0 = 1$. This choice of marginal variance, 10 times greater than the marginal variance of the model error, means that most of the variation is explained by the model, but that the model error is nevertheless non-negligible. We further assume that at each time point $t \in [1, 2, \ldots, T]$, where $T = 20$, we are given a set of noisy observations $\mathbf{y}_t$ corresponding to some of the points from the grid. We take the measurement error to be Gaussian which means that $\mathbf{y}_t$ follows the data model (2) with $\mathbf{R}_t = \sigma^2_{\mathbf{v}}\mathbf{I}_{n_t}$ where we set $\sigma^2_{\mathbf{v}} = 0.05$ and the matrix $\mathbf{H}_t$ is obtained by taking a diagonal matrix $\mathbf{I}_{n_\mathcal{G}}$ and removing the rows which correspond to the grid points with no associated observations. This choice of $\sigma^2_{\mathbf{v}}$

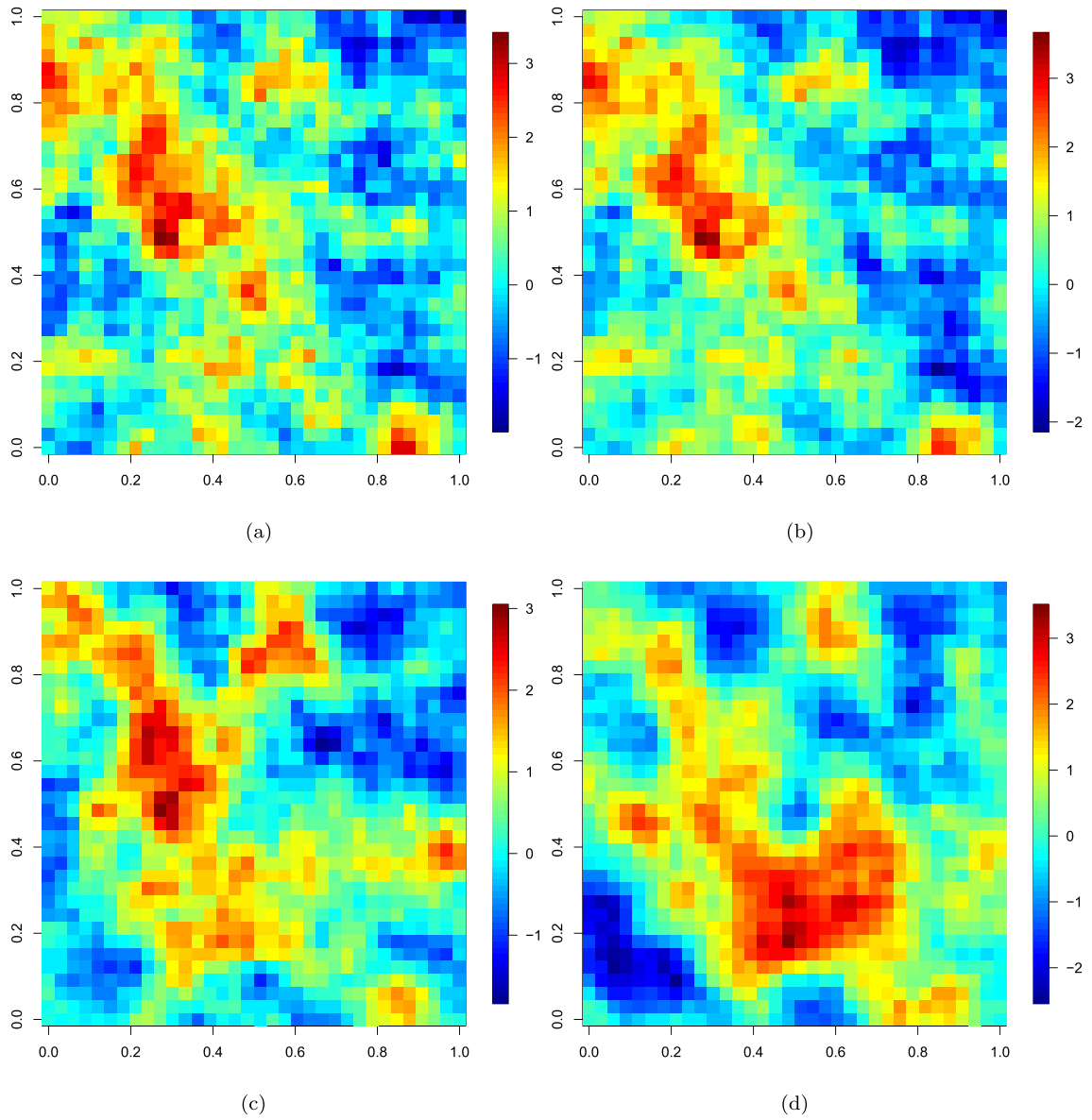

FIGURE 2 Sample realization of the 2D advection diffusion process described in Section 5.1 at select time points. The first row shows two consecutive time points while the plots in the second row, corresponding to time points further apart, illustrates the long term evolution of the process. (a) $t = 1$, (b) $t = 2$, (c) $t = 10$, and (d) $t = 20$

means that the signal to noise ratio is relatively high. A sample realization of this process at two time points is shown in Figure 2. Many other combinations of parameter values were previously considered in the case of filtering (Jurek & Katzfuss, 2021), but the relative performance of the analogs of the HV-based and low-rank filters was robust to these changes.

We then perform several numerical experiments using the following methods:

Scalable FFBS (scalable): Our method as described in Algorithm 7.

Low-rank-based FFBS (low-rank): A sampling method based on a low-rank approximation of the latent process x. Within the context of our article and for ease of comparison, we can view it as a special case of Algorithm 7 with the $\mathbf{S}$ matrix in which only the diagonal and the first N columns of $\mathbf{S}$ are nonzero. This is equivalent to using the modified predictive process approach (Banerjee et al., 2008; Finley et al., 2009) to approximate the process x and has the same computational complexity as scalable FFBS.

TABLE 1 The average time in seconds required to generate one sample from the model from Section 5.1 using each of the sampling methods

Method	Standard	Low-rank	Scalable
Average time (s)	322.2	13.2	13.3

Standard FFBS (standard): The method described in Algorithm 3. It can be viewed a special case of Algorithm 7, in which $\mathbf{S} = \mathbf{1}_{n_\mathcal{G}}\mathbf{1}_{n_\mathcal{G}}^\top$ and $N = n_\mathcal{G}$.

5.2 | Timing

We start by showing the difference in wall-clock time required to generate a single sample using the model settings and sampling methods described in Section 5.1. We run our code on a high-end laptop equipped with 16 Intel i7 CPUs each with a clock speed of 2.30 GHz and 16 GB of memory. In order to eliminate the influence of random processes executed at the same time, we use one method at a time, measure the time elapsed from the beginning until the end of Algorithm 3, repeat it 10 times and report the average. The results are shown in Table 1 and show that both approximate methods have a similar run time, which is much less than the run time of the standard FFBS. In the subsequent simulations, we show that the low-rank method, while comparable in execution time, is inferior in performance according to several criteria.

5.3 | Sampling the latent vector

In the second set of our simulations, we demonstrate the excellent accuracy of Algorithm 7 by generating a sample of size m from smoothing distribution of the latent vector $\mathbf{x}_t$. We then compare the results generated by other methods using continuous rank probability score (CRPS) for ensembles (Gneiting et al., 2008, Section 4.2). In general, if $\mathcal{Q} = \left\{q_1, \ldots, q_{N_{\text{samp}}}\right\}$ is the ensemble of size N_{samp} forecasting the vector q we can calculate this score as

$$\text{CRPS}(\mathcal{Q}, q) = \frac{1}{N_{\text{samp}}}\sum_{i=1}^{N_{\text{samp}}}\|q_i - q\|_2 - \frac{1}{2N_{\text{samp}}}\sum_{i=1}^{N_{\text{samp}}}\sum_{j=1}^{N_{\text{samp}}}\|q_i - q_j\|_2,$$

where $\|\cdot\|_2$ denotes the second (i.e., Euclidian) norm. The lower the value of the CPRS, the more accurately the ensemble predicts the true realization q. Under some mild conditions, CRPS is a strictly proper scoring rule (Gneiting & Katzfuss, 2014; Gneiting & Raftery, 2007). In order to evaluate the performance of scalable FFBS we adopt the following approach. We generate a sample of size $N_{\text{samp}} = 50$ using methods described in Section 5.1 and calculate the CRPS for each of them at each time point. For each of the approximate methods we then calculate the ratio of their respective scores and the score of the standard FFBS. We repeat this procedure $N_{\text{iter}} = 10$ times and present these average score ratios in Figure 3. We conclude that scalable version of the FFBS algorithm we propose is an excellent approximation of its standard version and that it significantly outperforms the low-rank approach.

5.4 | Gibbs sampling

One of the more common applications of the standard FFBS algorithm consists in using it as one of the steps in a Gibbs sampler. In this section, we demonstrate the performance of such a sampler which relies on the methods described in Section 5.1.

Using the model from Section 5.1 now, we assume that the $\sigma^2_{\mathbf{w}}$ parameter is unknown. Imposing an inverse gamma prior with shape parameter $a = 0.001$ and scale parameter $b = 0.001$ results in the conditional posterior distribution $p(\sigma^2_w|\mathbf{x}_{1:T}, \mathbf{y}_{1:T})$ that is inverse gamma with the shape parameter $\tilde{a} = a + \frac{n(T-1)}{2}$ and the scale parameter $\tilde{b} = b + \frac{1}{2}\sum_{t=2}^{T}(\mathbf{x}_t - \mathbf{E}_t\mathbf{x}_{t-1})^\top\Sigma_w^{-1}(\mathbf{x}_t - \mathbf{E}_t\mathbf{x}_{t-1})$. We then run the Gibbs sampler in which we sample the latent vectors $\{\mathbf{x}_t\}_{t=1}^T$ using each of the methods described at the beginning of Section 1 assuming that we have observations corresponding to

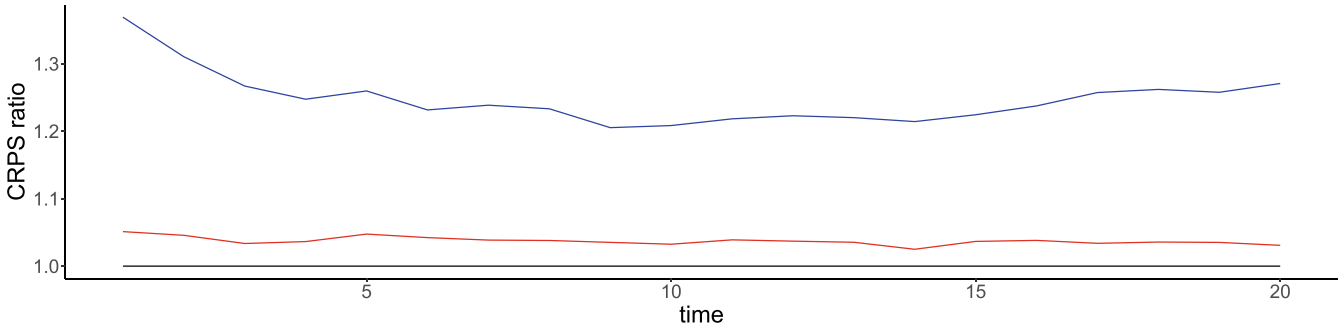

FIGURE 3 Average CRPS ratios for methods described in Section 5 $N_{\text{samp}} = 50$ samples based on different versions of the HV approximation averaged over $N_{\text{iter}} = 10$ repetitions. The red line corresponds to the scalable method, the blue line to the low-rank method and the black line to the standard sampler. All scores are calculated as relative to the score of the standard FFBS (for which its ratio with itself is therefore always equal to 1).

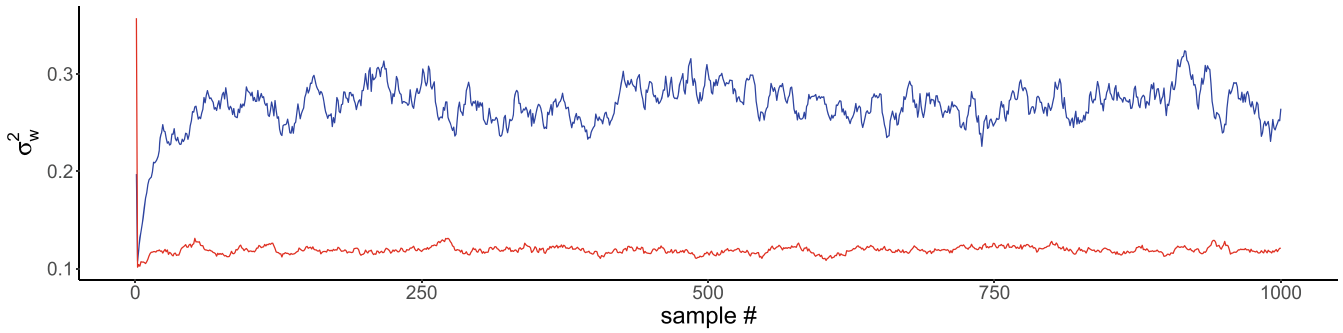

FIGURE 4 Samples from the conditional posterior distribution $p(\sigma_w^2|\mathbf{x}_{1:T}, \mathbf{y}_{1:T})$. (The true value was $\sigma_{\mathbf{w}}^2 = 0.1$.) The blue line denotes samples obtained using the low-rank method and the red line was generated using the scalable method. The standard method was not computationally feasible.

a random selection of 30% of grid points, that is, $n_t = 0.3n_G$. Figure 4 shows the samples from the conditional posterior distribution. In each case, the sampler was initialized at a random value between 0 and 0.5. We used only the approximate methods in the construction of the Gibbs sampler, because standard FFBS was not computationally feasible for a problem of this size. The results show that the sampler based on the scalable method generates draws of $\sigma_{\mathbf{w}}^2$ that are much closer to the true value (0.1) than the draws obtained using the low-rank method.

6 | ANALYSIS OF TOTAL PRECIPITABLE WATER

In this section, we apply our proposed sampler to real observations of the amount total precipitable water (TPW) in the atmosphere, defined as the mass of the water vapor in a column of air above a given area. TPW is commonly used in numerical weather prediction, forecasting extreme weather events, or assessing fire danger in drought-stricken areas. Hence, inferring complete and noise-free spatio-temporal maps of TPW is of considerable scientific value. The collection of data we work with is comprised of the total of 47,007 measurements made over a portion of the continental United States and the Gulf of Mexico at $T = 9$ points in time over a period of 40 hours in January 2011. Each data point corresponds to a cell in a $0.5° \times 0.5°$ latitude/longitude grid covering the area between $-125.18°$ W and $-107.21°$ W and 37.14° N and 50.07° N, resulting in 15,876 spatial grid cells. All of the observations were acquired using the Microwave Integrated Retrieval System (MIRS) satellite and are available from the authors upon request.

A superset of the data we use here has been analyzed previously using a low-rank filtering approach similar to the one described in Section 5.1 in Katzfuss and Hammerling (2017), and a filtering approach based on the HV approximation (Algorithm 5) in Jurek and Katzfuss (2022).

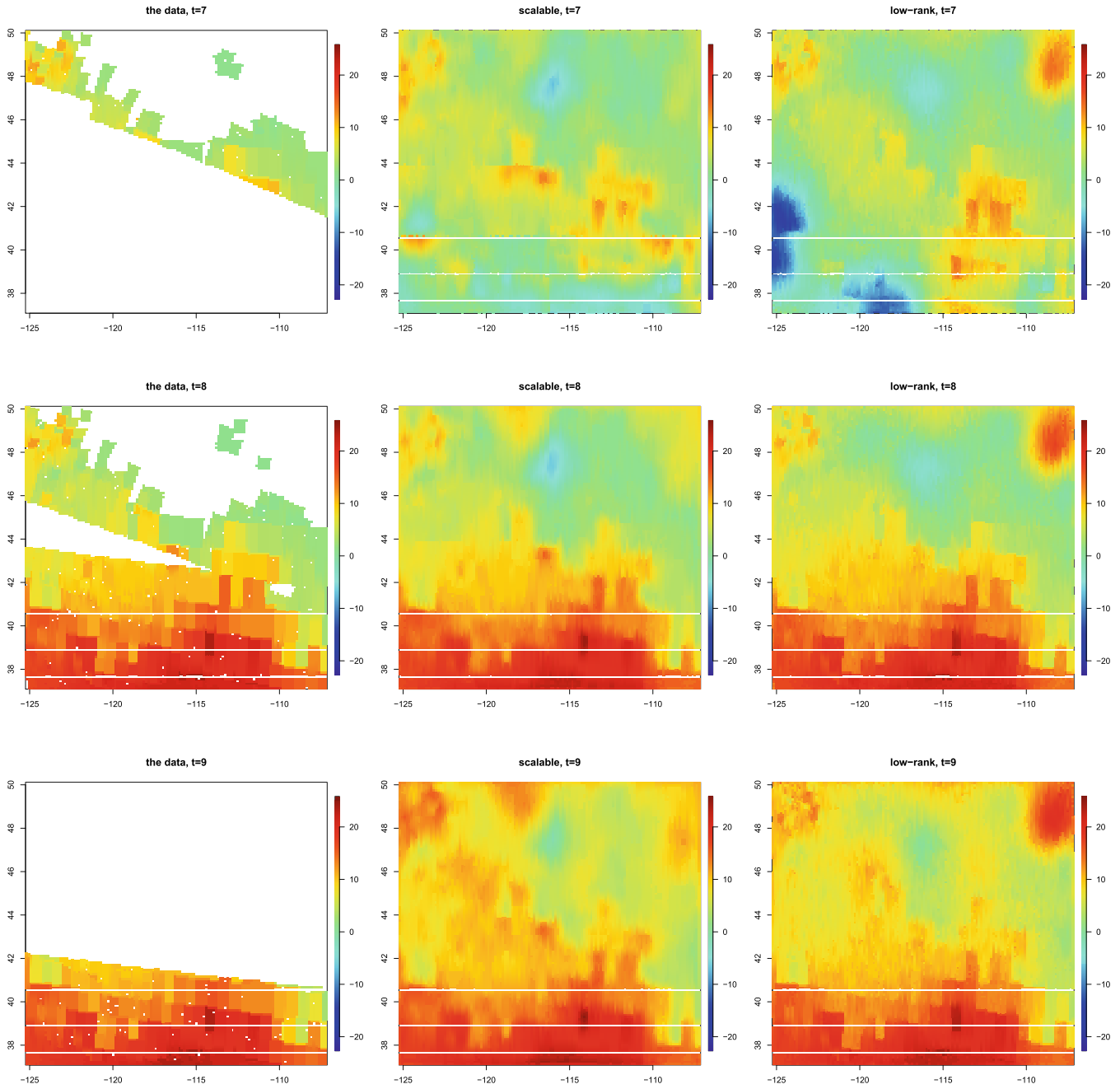

FIGURE 5 Total precipitable water (first column) and the pointwise mean of all the samples generated using the scalable method (second column) and the low-rank method (third column)

For a given time point t, we use $\mathbf{y}_t$ to denote the corresponding data, each of which is assumed to contain an independently and identically distributed normal measurement error with mean 0 and variance $\sigma^2_{\mathbf{v}}$. At each point, we calculated the mean of all measurements and subtracted it from the observations gathered at that time. The resulting value at selected time points are shown in the first column in Figure 5. At each time point, we set aside 1% of all available measurements to be later used for result verification.

We assume that the temporal evolution of TPW during the study period can be captured by an advection-diffusion equation, as described in Section 5.1. We use the diffusion coefficient to be $\alpha = 0.000003$ and the advection coefficient $\beta = 0$ (no advection). If $\tilde{\mathbf{E}}_t$ denotes the temporal evolution operator obtained using differencing, we take $\mathbf{E}_t = c\tilde{\mathbf{E}}_t$ with $c = 0.9$ to allow for more random variation at each time point.

TABLE 2 Values of parameters used in the application to TPW

λ	σ_0^2	$\sigma_{\mathbf{w}}^2$	$\sigma_{\mathbf{v}}^2$	c	α	β
1.0	74.7	8.3	1.63	0.9	0.000003	0

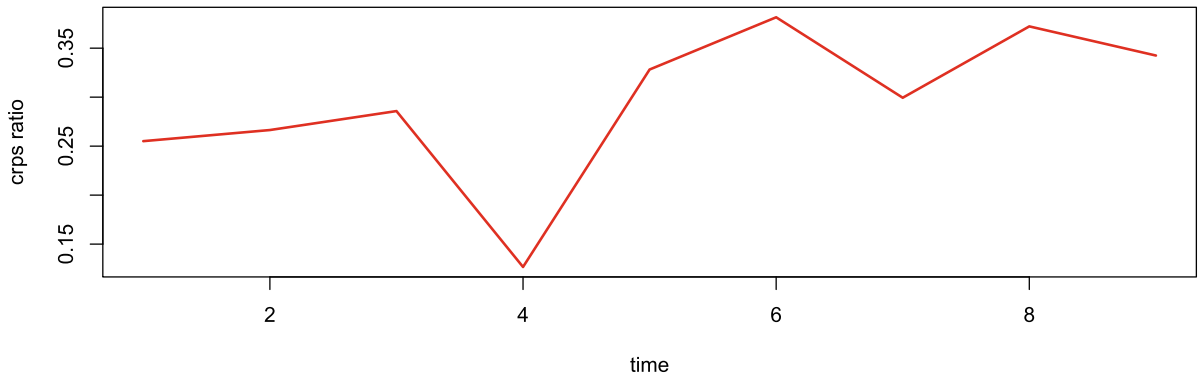

FIGURE 6 For the TPW data, CRPS for samples calculated using the scalable method relative to the CRPS for samples calculated using the low-rank method

We take the initial covariance function $\mathbf{\Sigma}_{0|0}$ to be derived from a Matérn covariance function with smoothness $\nu = 1.5$, range λ, marginal variance σ_0^2 and the $\mathbf{Q}_t$ matrix to be derived from a Matérn covariance function with the same smoothness $\nu = 1.5$ and range λ and marginal variance $\sigma_{\mathbf{w}}^2$.

In order to determine the values of parameters $\lambda, \sigma_0^2, \sigma_{\mathbf{w}}^2$ and $\sigma_{\mathbf{v}}^2$ we consider a purely spatial problem and assume that at each time $\mathbf{x}_t$ corresponds to a discretization of a mean-zero, 2D Gaussian random field with a Matérn covariance function with smoothness 1.5, range λ_t and marginal variance σ_t^2 and that $\mathbf{y}_t$ are the corresponding observations with $\mathbf{y}_t|\mathbf{x}_t = \mathcal{N}(\mathbf{x}_t, \tau_t^2)$. We use the Vecchia approximation with $N = 80$ nonzero elements in each row of $\mathbf{S}$ and use it to optimize the approximate likelihood function (see Zilber & Katzfuss, 2021). We take $\lambda = \frac{1}{T}\sum_t \lambda_t$, $\sigma_{\mathbf{v}}^2 = \frac{1}{T}\sum_t \tau_t^2$. For the marginal variance parameters, we assume that $\sigma_0^2 + \sigma_w^2 = \frac{1}{T}\sum_t \sigma_t^2$ and then set $\sigma_{\mathbf{w}}^2 = (1-c)\sigma_0^2$. Table 2 summarizes the parameter values obtained in this way.

Then using Algorithm 7, we generate $N_{\text{samp}} = 20$ samples $\{\mathbf{x}_{1:T}^i\}_{i=1}^{N_{\text{samp}}}$ from the smoothing distribution of the state vector $\mathbf{x}_t$ using the scalable method and the low rank method with the conditioning set of size $N = 52$. In Figure 5, we present the mean field $\overline{\mathbf{x}} = \frac{1}{N_{\text{samp}}}\sum_{i=1}^{N_{\text{samp}}}\mathbf{x}_t^i$ for select values of t.

In order to evaluate our method, we use CRPS as described in Section 5.3 using the observations which we set aside at the beginning. Because of the scale of the problem, the standard method was not feasible. Instead we report the ratio $\text{rCRPS} = 100\%\frac{\text{CRPS}_{\text{HV}}}{\text{CRPS}_{\text{LR}}}$, which tells us by what percentage the score is reduced, if we use the scalable method as opposed to the low-rank method. We report the rCRPS for each time point in Figure 6. The results show, that using the scalable method instead of the low-rank method leads to about 20% lower CRPS at a typical point in time.

7 | CONCLUSIONS

Our article proposes an approximate method of sampling the latent state in the context of linear Gaussian state space models. Our approach, called scalable FFBS, can be applied even to fields with tens of thousands of random variables. It also outperforms samplers based on a popular low-rank approximation according to several important metrics. The proposed algorithm can be extended in several directions. First, combining it with the Laplace approximation, similar to (Jurek & Katzfuss, 2022), it can be applied to a large class of non-Gaussian distributions. Using the correlation distance (Kang & Katzfuss, 2021), might allow to accommodate data without a clear spatial structure. We also envision extending our framework to incorporate several random fields and nonlinear temporal evolution.

ACKNOWLEDGMENTS

Matthias Katzfuss's research was partially supported by NSF Grants DMS-1654083, DMS-1953005, and CCF-1934904, and by the National Aeronautics and Space Administration (80NM0018F0527). We would like to thank Kate Calder, Mevin Hooten, and Cory Zigler for helpful comments and discussions. Special thanks to Pulong Ma, who first pointed out the possibility of using the Vecchia approximation in the context of an FFBS algorithm, and to Dorit Hammerling for helping us access the TPW data.

ORCID


Marcin Jurek https://orcid.org/0000-0002-2331-8236

REFERENCES

Banerjee, S., Gelfand, A. E., Finley, A. O., & Sang, H. (2008). Gaussian predictive process models for large spatial data sets. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *70*(4), 825–848.

Bugliaro, L., Piontek, D., Kox, S., Schmidl, M., Mayer, B., Müller, R., Vázquez-Navarro, M., Peters, D. M., Grainger, R. G., Gasteiger, J., et al. (2021). Combining radiative transfer calculations and a neural network for the remote sensing of volcanic ash using MSG/SEVIRI. *Natural Hazards and Earth System Sciences Discussions*, 1–42.

Carter, C. K., & Kohn, R. (1994). On Gibbs sampling for state space models. *Biometrika*, *81*(3), 541–553.

Carvalho, C. M., Johannes, M. S., Lopes, H. F., & Polson, N. G. (2010). Particle learning and smoothing. *Statistical Science*, *25*(1), 88–106.

Durbin, J., & Koopman, S. J. (2002). A simple and efficient simulation smoother for state space time series analysis. *Biometrika*, *89*(3), 603–615.

ECMWF (2021). *IFS documentation CY47R3 - Part II: Data assimilation*. Number 2 in IFS documentation. ECMWF.

Evensen, G. (1994). Sequential data assimilation with a nonlinear quasi-geostrophic model using monte carlo methods to forecast error statistics. *Journal of Geophysical Research: Oceans*, *99*(C5), 10143–10162.

Evensen, G., Vossepoel, F. C., and van Leeuwen, P. J. (2022). Data assimilation fundamentals: A unified formulation of the state and parameter estimation problem.

Finley, A. O., Sang, H., Banerjee, S., & Gelfand, A. E. (2009). Improving the performance of predictive process modeling for large datasets. *Computational Statistics & Data Analysis*, *53*(8), 2873–2884.

Frühwirth-Schnatter, S. (1994). Data augmentation and dynamic linear models. *Journal of Time Series Analysis*, *15*(2), 183–202.

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

Gneiting, T., & Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, *102*(477), 359–378.

Gneiting, T., Stanberry, L., Grimit, E., Held, L., & Johnson, N. (2008). Assessing probabilistic forecasts of multivariate quantities, with an application to ensemble predictions of surface winds. *TEST: An Official Journal of the Spanish Society of Statistics and Operations Research*, *17*(2), 211–235.

Grudzien, C., & Bocquet, M. (2021). A fast, single-iteration ensemble kalman smoother for sequential data assimilation. *Geoscientific Model Development Discussions, pages*, 1–62.

Jarociński, M. (2015). A note on implementing the durbin and koopman simulation smoother. *Computational Statistics and Data Analysis*, *91*, 1–3.

Jayne, S. R., Roemmich, D., Zilberman, N., Riser, S. C., Johnson, K. S., Johnson, G. C., & Piotrowicz, S. R. (2017). The argo program: Present and future. *Oceanography*, *30*(2), 18–28.

Jurek, M., & Katzfuss, M. (2021). Multi-resolution filters for massive spatio-temporal data. *Journal of Computational and Graphical Statistics*, *30*(4), 1095–1110.

Jurek, M., & Katzfuss, M. (2022). Hierarchical sparse Cholesky decomposition with applications to high-dimensional spatio-temporal filtering. *Statistics and Computing*, *32*(1), 1–19.

Kalman, R. E. (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 arXiv:2112.14591.

Katzfuss, M., & Cressie, N. (2012). Bayesian hierarchical spatio-temporal smoothing for very large datasets. *Environmetrics*, *23*(1), 94–107.

Katzfuss, M., & Hammerling, D. (2017). Parallel inference for massive distributed spatial data using low-rank models. *Statistics and Computing*, *27*(2), 363–375.

Katzfuss, M., Stroud, J. R., & Wikle, C. K. (2020). Ensemble kalman methods for high-dimensional hierarchical dynamic space-time models. *Journal of the American Statistical Association*, *115*(530), 866–885.

Rauch, H. E., Tung, F., & Striebel, C. T. (1965). Maximum likelihood estimates of linear dynamic systems. *AIAA Journal*, *3*(8), 1445–1450.

Schäfer, F., Katzfuss, M., and Owhadi, H. (2021). Sparse Cholesky Factorization by Kullback–Leibler Minimization. *SIAM Journal on scientific computing*, *43*(3), A2019-A2046.

Sigrist, F., Künsch, H. R., & Stahel, W. A. (2015). Stochastic partial differential equation based modelling of large space–Time data sets. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *77*(1), 3–33.

Stroud, J. R., Stein, M. L., Lesht, B. M., Schwab, D. J., & Beletsky, D. (2010). An ensemble Kalman filter and smoother for satellite data assimilation. *Journal of the American Statistical Association*, *105*(491), 978–990.

Sun, Y., Frankenberg, C., Wood, J. D., Schimel, D. S., Jung, M., Guanter, L., Drewry, D. T., Verma, M., Porcar-Castell, A., Griffis, T. J., Gu, L., Magney, T. S., Köhler, P., Evans, B., & Yuen, K. (2017). Oco-2 advances photosynthesis observation from space via solar-induced chlorophyll fluorescence. *Science*, *358*(6360), eaam5747.

Wikle, C. K., Zammit-Mangion, A., & Cressie, N. (2019). *Spatio-temporal Statistics with R*. Chapman & Hall/CRC Press.

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

How to cite this article: Jurek, M., & Katzfuss, M. (2022). Scalable spatio-temporal smoothing via hierarchical sparse Cholesky decomposition. *Environmetrics*, e2757. https://doi.org/10.1002/env.2757