SCALABLE FILTERING METHODS FOR HIGH-DIMENSIONAL SPATIO-TEMPORAL DATA

A Dissertation

by

MARCIN PIOTR JUREK

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Matthias Katzfuss
Committee Members,	Huiyan Sang
	Ignacio Rodriguez-Iturbe
	Suhasini Subba Rao
Head of Department,	Daren Cline

August 2020

Major Subject: Statistics

Copyright 2020 Marcin Piotr Jurek

ABSTRACT

We propose a family of filtering methods for deriving the filtering distribution in the context of a high-dimensional state-space model. In the first chapter, we develop and describe in detail the basic method, which can be used in a linear case with Gaussian data. In the second chapter, we show how this method can be extended to incorporate non-Gaussian observations and non-linear temporal evolution models. We discuss how two algorithms, the multi-resolution decomposition and the incomplete Cholesky decomposition, can be used to quickly update the filtering distribution at each time step of the filtering procedures.

ACKNOWLEDGMENTS

I would like to thank Jacob Bien, Wenlong Gong, Joseph Guinness, Ephrahim Hanks, Peter Kuchement, Mohsen Pourahmadi, Ramalingam Saravanan, Florian Schäfer, Michael Stein, Jonathan Stroud, Istvan Szunyogh, Christopher Wikle, and Catherine Yan for helpful comments and discussion. Special thanks goes to Jonathan Stroud for providing code and data for the Lake Michigan example. I also appreciate the the implementation of the Lorenz model provided by Edward Ott and Seung-Jong Baek, which provided insights for my code. I am also grateful for the support and inspiration Dorit Hammerling offered during the process of creating a Python version of one of the algorithms.

Most of all I would like to thank my advisor, Dr. Matthias Katzfuss, for his constant availability, clear guidance, plentiful advice, and patient correction of my frequent mistakes.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was primarily supported by my advisor Dr. Matthias Katzfuss. All work conducted for the thesis was completed by me under his close supervision. Some parts of the implementation of Algorithm 2.3.4.2 were developed during a SIParCS internship in 2017 at the National Center for Atmospheric Research under the mentorship of Dorit Hammerling.

Funding Sources

During the course of my studies I was partially supported by National Science Foundation (NSF) Grants DMS–1521676 and DMS–1654083 and a Texas A&M Triads for Transformation grant on “Improving Unmanned Aerial System (UAS) Estimates of Crop Height by Spatio-Temporal Statistics.”

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

Spatio-temporal datasets are rapidly growing in size. For example, environmental variables are measured at ever-finer resolutions by increasing numbers of automated sensors mounted on satellites and aircraft. Such data, which are typically noisy and incomplete, often become available sequentially over time, as observations are made over consecutive hours or days. Under such circumstances, one is typically interested in two inference tasks. The first is to obtain complete maps of the spatio-temporal process, together with uncertainty quantification. Second, the process of interest is frequently represented by a parametric model, and thus another task is to find the distribution of these parameters conditional on the data.

While the phenomenon being studied is often continuous, for practical purposes it is typically modeled on a spatial grid and at equidistant time points. This naturally gives rise to so-called state-space models, where a vector represents the values of the process (state) at each grid point at a certain moment in time, and an evolution function captures its temporal dynamics. If one assumes an additive normal error, the observations are a linear function of the state with a normal measurement error, and the temporal dynamics can be represented as a linear Markov process, the standard tool used to accomplish the first inference task is the Kalman filter, while a Rao-Blackwellized filter can be used for the second. In the presence of non-Gaussian data or non-linear temporal evolution, other techniques can be utilized, such as particle filters, extended Kalman filters or ensemble Kalman filters.

Unfortunately, all of these methods tend to scale poorly as the dimension of the state and observation vectors increase. Particle filters are applicable only for moderate grid sizes, while the other approaches generally require inverting the covariance matrix of the state vector, which becomes prohibitively expensive for larger grids.

To address these challenges we develop a family of filters based on the multi-resolution approximation (MRA Katzfuss, 2017) and the general Vecchia approximation (Katzfuss and Guinness, 2019). Our approach assumes that the latent state vector follows a normal distribution and that con-

ditional on the state vector, observations follow a distribution belonging to an exponential family. We then utilize the approximation methods developed for Gaussian process models to accelerate inference by assuming certain conditional independence relationships. We show how they lead to sparsity of the Cholesky factors of the posterior and covariance matrices of the state vector and present a simple factorization algorithm that takes advantage of this sparsity and enables fast computations.

The assumption that observations follow an exponential family-type distribution allows us to apply our method to a multitude of environmental data sets, many of which contain highly non-Gaussian data. Our proposed filters are able to both reconstruct the spatial field based on incomplete and noisy data, as well as infer the conditional distribution of parameters. They accomplish both of these tasks in a sequential manner, and update their estimates as new data become available over time.

To facilitate the adoption of our method, we also substantially extend an R package, called GPvecchia, which implements the MRA and the general Vecchia approximation, in a way that makes it a convenient building block in filtering algorithms.

2. MULTI-RESOLUTION FILTERS FOR MASSIVE SPATIO-TEMPORAL DATA

2.1 Introduction

Massive spatio-temporal data have become ubiquitous in the environmental sciences, which is largely due to Earth-observing satellites providing high-resolution measurements of environmental variables on a continental or even global scale. Accounting for spatial and temporal dependence is very important for satellite data, as atmospheric variables vary over space and time, and measurements from different orbits are often complementary in their coverage.

When time and space are discretized, spatio-temporal data are typically modeled using a dynamical state-space model (SSM), which describes how the state (i.e., the spatial field evaluated at a spatial grid) evolves over time and how the state is related to the observations. Dynamical SSMs can include information from other sources and sophisticated temporal dynamics in terms of partial differential equations; for example, the effect of wind on atmospheric variables can be captured by an advection term. Such informative, physical evolution models are crucial for producing meaningful forecasts.

We focus here on real-time or on-line filtering inference in linear Gaussian SSMs, which means that at each time point t, we are interested in the posterior distribution of the spatial field at time t given all data obtained up to time t. The filtering distributions in this setting are Gaussian and can in principle be determined exactly by the Kalman filter (Kalman, 1960), but this technique is not computationally feasible for large grids. Particle filter methods such as sequential importance (re-)sampling (e.g., Gordon et al., 1993) are asymptotically exact as the number of particles increases, but suffer from particle collapse for finite particle size in high dimensions (e.g., Snyder et al., 2008).

In the geosciences, filtering inference in SSMs is referred to as data assimilation (see, e.g., Nychka and Anderson, 2010, for a review), especially when the evolution is described by a complex computer model. Data assimilation is typically carried out via variational methods (e.g., Talagrand

and Courtier, 1987) or the ensemble Kalman filter (EnKF; e.g., Evensen, 1994, 2007; Katzfuss et al., 2016; Houtekamer and Zhang, 2016). The EnKF represents distributions by an ensemble, which is propagated using the temporal evolution model and updated via a linear shift based on new observations. In practice, only small ensemble sizes are computationally feasible, resulting in a low-dimensional representation and substantial sampling error.

In the statistics literature, computationally feasible filtering approaches for dynamical spatio-temporal SSMs often rely on low-rank assumptions (e.g., Verlaan and Heemink, 1995; Pham et al., 1998; Wikle and Cressie, 1999; Katzfuss and Cressie, 2011), but such approaches cannot fully resolve fine-scale variation (Stein, 2014). Therefore, recent methods for large spatial-only data have instead achieved fast computation through sparsity assumptions (e.g., Lindgren et al., 2011; Nychka et al., 2015; Datta et al., 2016a; Katzfuss and Guinness, 2019), and idea that can also be used in the context of retrospective, "off-line" spatio-temporal analysis, in which time is essentially treated as an additional spatial dimension and the resulting spatio-temporal covariance function is modeled or approximated (e.g., Zhang et al., 2015; Datta et al., 2016b). However, most sparsity-based methods cannot be easily extended to the filtering perspective of interest here, because the sparsity structure is lost when propagating forward in time.

Here, we propose a novel multi-resolution filter (MRF) for big streaming spatio-temporal data based on linear Gaussian SSMs. The MRF is a highly scalable, fully probabilistic procedure that results in joint posterior predictive distributions for the spatio-temporal field of interest. In contrast to the EnKF, MRF computations are deterministic and do not suffer from sampling variability. In contrast to low-rank approaches, the MRF does not rely on dimension reduction. Similar to wavelet-based filtering methods (e.g. Chui, 1992; Cristi and Tummala, 2000; Renaud et al., 2005; Beezley et al., 2011; Hickmann and Godinez, 2015), the MRF can be viewed as employing a large number of basis functions at multiple levels of spatial resolution, which can capture spatial structure from very fine to very large scales. However, as opposed to wavelets, the MRF basis functions automatically adapt to approximate the covariance structure implied by the assumed SSM. These features allowed the MRF to strongly outperform existing approaches in our numerical

comparisons.

The MRF relies on a new approximate covariance-matrix decomposition, for which the resulting matrix factors exhibit a particular block-sparse multi-resolution structure. This decomposition is based on the multi-resolution approximation (Katzfuss, 2017; Katzfuss and Gong, 2019) of spatial processes, which performed very well in a recent comparison of different methods for large spatial-only data (Heaton et al., 2019). Using advanced concepts from graph theory, we prove the perhaps surprising property that the block-sparse structure of the MRF matrices can be maintained under filtering operations through time, which in turn is crucial for allowing us to show that the MRF exhibits linear computational complexity for fixed tuning parameters. Note that this is in contrast to other sparse-matrix approximations, as even matrices with simple sparsity patterns (e.g., tridiagonal matrices) do not preserve sparsity under inversion. In fact, we suspect that the multi-resolution decomposition and its special cases are unique in terms of preserving matrix sparsity.

We also establish a close connection between our multi-resolution decomposition and hierarchical matrices. Despite being relatively unknown in statistics, hierarchical matrices (e.g., Hackbusch, 2015) are a highly popular and widely studied class of matrix approximations in numerical mathematics. We introduce this matrix class into the statistical literature, and describe how hierarchical matrices can be applied to SSMs based on second-order partial differential equations, including those describing advection and diffusion. This marks a major step forward with respect to multiple previous hierarchical-matrix approaches for fast high-dimensional Kalman filtering (e.g. Li et al., 2014; Saibaba et al., 2015; Ambikasaran et al., 2016), which were only applicable in the simple case of a random walk.

Finally, we discuss extensions for inference on time-varying parameters that are not part of the spatial field, using a Rao-Blackwellized particle filter, in which the integrated likelihood is approximated using the MRF.

The remainder of this article is organized as follows. Section 2.2 describes the linear Gaussian state-space model and reviews the Kalman filter. In Section 2.3, we present the MRF. Section 2.4

details key properties of the MRF, and Section 2.5 discusses connections to existing approaches. Section 2.6 shows how the MRF can be extended when the model includes unknown parameters. In Section 2.7, we present a numerical comparison of the MRF to existing methods. Section 2.8 demonstrates a practical application of the MRF to inferring sediment concentration in Lake Michigan based on satellite data. We conclude in Section 2.9. Proofs are given in Appendix A.1.

A separate Supplementary Material document contains Sections C.1–C.8 with further properties, details, and proofs. At http://spatial.stat.tamu.edu, we provide additional illustrations. All code will be provided upon publication.

2.2 Spatio-temporal state-space models (SSMs) and filtering inference

2.2.1 Spatio-temporal state-space model

Let $\mathbf{x}_t$ be the $n_{\mathcal{S}}$-dimensional latent state vector of interest, representing a (mean-corrected) spatio-temporal process $x_t(\cdot)$ at time t evaluated at a fine grid $\mathcal{S} = \{\mathbf{g}_1, \ldots, \mathbf{g}_{n_{\mathcal{S}}}\}$ on a spatial domain or region $\mathcal{D}$. Further, let $\mathbf{y}_t$ denote the observed n_t-dimensional data vector at time t. We assume a linear Gaussian spatio-temporal state-space model given by an observation equation and an evolution equation,

$$\mathbf{y}_t = \mathbf{H}_t\mathbf{x}_t + \mathbf{v}_t, \qquad \mathbf{v}_t \sim \mathcal{N}_{n_t}(\mathbf{0}, \mathbf{R}_t), \tag{2.1}$$

$$\mathbf{x}_t = \mathbf{E}_t\mathbf{x}_{t-1} + \mathbf{w}_t, \qquad \mathbf{w}_t \sim \mathcal{N}_{n_{\mathcal{S}}}(\mathbf{0}, \mathbf{Q}_t), \tag{2.2}$$

respectively, for time $t = 1, 2, \ldots$. The initial state also follows a Gaussian distribution: $\mathbf{x}_0 \sim \mathcal{N}_{n_{\mathcal{S}}}(\boldsymbol{\mu}_{0|0}, \boldsymbol{\Sigma}_{0|0})$. The noise covariance matrix $\mathbf{R}_t$ will be assumed to be diagonal or block-diagonal here for simplicity (see Assumption 1 in Section 2.4.2.1). No computationally convenient structure is assumed for the innovation covariance matrix $\mathbf{Q}_t$. The observation noise $\mathbf{v}_t$ and the innovation $\mathbf{w}_t$ are mutually and serially independent, and independent of the state $\mathbf{x}_{t-1}$. We assume that all matrices in (2.1)–(2.2) (and $\boldsymbol{\mu}_{0|0}$ and $\boldsymbol{\Sigma}_{0|0}$) are known. The case of unknown parameters is discussed in Section 2.6.

The observation matrix $\mathbf{H}_t$ relates the state to the observations. This enables combining obser-

vations from different instruments or modeling areal observations given by averaging over certain elements of the state vector. Here, we usually assume point-level measurements for simplicity, although a block-diagonal form for $\mathbf{H}_t$ is possible (see Assumption 1).

The evolution matrix $\mathbf{E}_t$ determines how the process evolves over time. It can be specified in terms of a system of partial differential equations (PDEs), may depend on other variables, or — in the absence of further information — could simply be a scaled identity operator indicating a random walk over time. We assume that the evolution is local and $\mathbf{E}_t$ is sparse (Assumption 2 in Section 2.4.2.2).

Note that the SSM in (2.1)–(2.2), which is a latent Markov model of order 1, is very general and can describe a broad class of systems. Higher-order Markov models can also be written in the form (2.1)–(2.2) by expanding the state space. Non-Gaussian observations can often be transformed to be approximately Gaussian. Other extensions are also straightforward, such as letting the grid $\mathcal{S}$ vary over time.

2.2.2 Filtering inference using the Kalman filter (KF)

We are interested in filtering inference on the state $\mathbf{x}_t$. That is, at each time t, the goal is to find the conditional distribution of $\mathbf{x}_t$ given all observations up to and including time t, denoted by $\mathbf{x}_t|\mathbf{y}_{1:t}$, where $\mathbf{y}_{1:t} = (\mathbf{y}_1', \ldots, \mathbf{y}_t')'$.

For the linear Gaussian SSM in (2.1)–(2.2), the filtering distributions are Gaussian. These filtering distributions can be obtained recursively for $t = 1, 2, \ldots$ using the Kalman filter (Kalman, 1960), which consists of a forecast step and an update step at each time point. Denote the filtering distribution at time $t-1$ by $\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1} \sim \mathcal{N}_n(\boldsymbol{\mu}_{t-1|t-1}, \boldsymbol{\Sigma}_{t-1|t-1})$. The forecast step obtains the forecast or prior distribution of $\mathbf{x}_t$ based on the previous filtering distribution and the evolution model (2.2) as

$$\mathbf{x}_t|\mathbf{y}_{1:t-1} \sim \mathcal{N}_{n_\mathcal{S}}(\boldsymbol{\mu}_{t|t-1}, \boldsymbol{\Sigma}_{t|t-1}), \qquad \boldsymbol{\mu}_{t|t-1} := \mathbf{E}_t\boldsymbol{\mu}_{t-1|t-1}, \quad \boldsymbol{\Sigma}_{t|t-1} := \mathbf{E}_t\boldsymbol{\Sigma}_{t-1|t-1}\mathbf{E}_t' + \mathbf{Q}_t.$$

Then, the update step modifies this forecast distribution based on the observation vector $\mathbf{y}_t$ and the

observation equation (2.1), in order to obtain the filtering distribution of $\mathbf{x}_t$:

$$\mathbf{x}_t|\mathbf{y}_{1:t} \sim \mathcal{N}_{n_\mathcal{S}}(\boldsymbol{\mu}_{t|t}, \boldsymbol{\Sigma}_{t|t}), \qquad \boldsymbol{\mu}_{t|t} := \boldsymbol{\mu}_{t|t-1} + \mathbf{K}_t(\mathbf{y}_t - \mathbf{H}_t\boldsymbol{\mu}_{t|t-1}), \quad \boldsymbol{\Sigma}_{t|t} := (\mathbf{I}_{n_\mathcal{S}} - \mathbf{K}_t\mathbf{H}_t)\boldsymbol{\Sigma}_{t|t-1}, \tag{2.3}$$

where $\mathbf{K}_t := \boldsymbol{\Sigma}_{t|t-1}\mathbf{H}_t'(\mathbf{H}_t\boldsymbol{\Sigma}_{t|t-1}\mathbf{H}_t' + \mathbf{R}_t)^{-1}$ is the $n_\mathcal{S} \times n_t$ Kalman gain matrix.

While the Kalman filter provides the exact solution to our filtering problem, it requires computing and propagating the $n_\mathcal{S} \times n_\mathcal{S}$ covariance matrix $\boldsymbol{\Sigma}_{t|t}$ and decomposing the $n_t \times n_t$ matrix $(\mathbf{H}_t\boldsymbol{\Sigma}_{t|t-1}\mathbf{H}_t'+\mathbf{R}_t)$ in $\mathbf{K}_t$, and is thus computationally infeasible for large $n_\mathcal{S}$ or large n_t. Therefore, approximations are required for large spatio-temporal data.

2.3 The multi-resolution filter (MRF)

2.3.1 Overview

We now propose a multi-resolution filter (MRF) for spatio-temporal SSMs of the form (2.1)–(2.2) when the grid size $n_\mathcal{S}$ or the data sizes n_t are large, roughly between 10^4 and 10^9. The MRF can be viewed as an approximation of the Kalman filter in Section 2.2.2.

The most important ingredient of the MRF is a novel multi-resolution decomposition (MRD). Given a spatial covariance matrix $\boldsymbol{\Sigma}$, the MRD computes $\mathbf{B} = \mathrm{MRD}(\boldsymbol{\Sigma})$ such that $\boldsymbol{\Sigma} \approx \mathbf{B}\mathbf{B}'$. We will describe the MRD in detail in Section 2.3.4. For now, we merely note that the MRD algorithm is fast, and the resulting multi-resolution factor $\mathbf{B}$ is of the same dimensions as $\boldsymbol{\Sigma}$ but exhibits a particular block-sparse structure (see Figure 2.2a).

The MRF algorithm proceeds as follows:

Algorithm 1: Multi-resolution filter (MRF)

At the initial time $t = 0$, compute $\mathbf{B}_{0|0} = \mathrm{MRD}(\boldsymbol{\Sigma}_{0|0})$. Then, for each $t = 1, 2, \ldots$, do:

1. Forecast Step: Apply the evolution matrix $\mathbf{E}_t$ to obtain $\boldsymbol{\mu}_{t|t-1} = \mathbf{E}_t\boldsymbol{\mu}_{t-1|t-1}$ and $\mathbf{B}^F_{t|t-1} = \mathbf{E}_t\mathbf{B}_{t-1|t-1}$. Carry out a multi-resolution decomposition $\mathbf{B}_{t|t-1} = \mathrm{MRD}(\boldsymbol{\Sigma}^F_{t|t-1})$, where $\boldsymbol{\Sigma}^F_{t|t-1} = \mathbf{B}^F_{t|t-1}(\mathbf{B}^F_{t|t-1})' + \mathbf{Q}_t$, to obtain $\mathbf{x}_t|\mathbf{y}_{1:t-1} \sim \mathcal{N}_{n_\mathcal{S}}(\boldsymbol{\mu}_{t|t-1}, \boldsymbol{\Sigma}_{t|t-1})$ with $\boldsymbol{\Sigma}_{t|t-1} =$

$\mathbf{B}_{t|t-1}\mathbf{B}'_{t|t-1}$.

2. Update Step: Compute $\mathbf{B}_{t|t} := \mathbf{B}_{t|t-1}(\mathbf{L}_t^{-1})'$, where $\mathbf{L}_t$ is the lower Cholesky triangle of $\mathbf{\Lambda}_t := \mathbf{I}_{n_\mathcal{S}} + \mathbf{B}'_{t|t-1}\mathbf{H}'_t\mathbf{R}_t^{-1}\mathbf{H}_t\mathbf{B}_{t|t-1}$, to obtain $\mathbf{x}_t|\mathbf{y}_{1:t} \sim \mathcal{N}_{n_\mathcal{S}}(\boldsymbol{\mu}_{t|t}, \mathbf{\Sigma}_{t|t})$ with $\mathbf{\Sigma}_{t|t} = \mathbf{B}_{t|t}\mathbf{B}'_{t|t}$ and $\boldsymbol{\mu}_{t|t} = \boldsymbol{\mu}_{t|t-1} + \mathbf{B}_{t|t}\mathbf{B}'_{t|t}\mathbf{H}'_t\mathbf{R}_t^{-1}(\mathbf{y}_t - \mathbf{H}_t\boldsymbol{\mu}_{t|t-1})$.

The key to the scalability of this algorithm is that while $\mathbf{\Sigma}_{t|t-1}$ and $\mathbf{\Sigma}_{t|t}$ are large and dense matrices, they are never explicitly calculated and instead represented by the block-sparse matrices $\mathbf{B}_{t|t-1}$ and $\mathbf{B}_{t|t}$, respectively. Also, as shown in Section 2.4.2, $\mathbf{B}_{t|t}$ has the same sparsity structure as $\mathbf{B}_{t|t-1}$, which allows the cycle to start over for the next time point $t+1$. The forecast step and update step will be discussed in more detail in Sections 2.3.2 and 2.3.3, respectively.

2.3.2 Details of the MRF forecast step

Assume that we have obtained the filtering distribution $\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1} \sim \mathcal{N}_n(\boldsymbol{\mu}_{t-1|t-1}, \mathbf{\Sigma}_{t-1|t-1})$, where $\mathbf{\Sigma}_{t-1|t-1} = \mathbf{B}_{t-1|t-1}\mathbf{B}'_{t-1|t-1}$ and $\mathbf{B}_{t-1|t-1}$ is a block-sparse matrix. Following the forecast step of the standard Kalman filter, we want to obtain the prior distribution at time t, $\mathbf{x}_t|\mathbf{y}_{1:t-1} \sim \mathcal{N}_n(\boldsymbol{\mu}_{t|t-1}, \mathbf{\Sigma}_{t|t-1})$.

Because of the sparsity of $\mathbf{E}_t$ (see Assumption 2 in Section 2.4.2.2), computing the forecast mean $\boldsymbol{\mu}_{t|t-1} = \mathbf{E}_t\boldsymbol{\mu}_{t-1|t-1}$ and the forecast basis matrix $\mathbf{B}^F_{t|t-1} = \mathbf{E}_t\mathbf{B}_{t-1|t-1}$ is fast. Then, rather than calculating the dense $n_\mathcal{S} \times n_\mathcal{S}$ forecast covariance matrix $\mathbf{\Sigma}^F_{t|t-1} = \mathbf{B}^F_{t|t-1}(\mathbf{B}^F_{t|t-1})' + \mathbf{Q}_t$ explicitly, we obtain its multi-resolution decomposition $\mathbf{B}_{t|t-1} = \mathrm{MRD}(\mathbf{\Sigma}^F_{t|t-1})$ as described in Section 2.3.4. This implies an approximation to the prior covariance matrix as $\mathbf{\Sigma}_{t|t-1} = \mathbf{B}_{t|t-1}\mathbf{B}'_{t|t-1}$. Again, $\mathbf{\Sigma}_{t|t-1}$ does not need to be computed explicitly, because only $\mathbf{B}_{t|t-1}$ is used in the update step below.

2.3.3 Details of the MRF update step

The objective of the update step is to compute the posterior distribution $\mathbf{x}_t|\mathbf{y}_{1:t} \sim \mathcal{N}_{n_\mathcal{S}}(\boldsymbol{\mu}_{t|t}, \mathbf{\Sigma}_{t|t})$ given the prior quantities $\boldsymbol{\mu}_{t|t-1}$ and $\mathbf{B}_{t|t-1}$ (such that $\mathbf{\Sigma}_{t|t-1} = \mathbf{B}_{t|t-1}\mathbf{B}'_{t|t-1}$) obtained in the forecast step.

Following the Kalman filter update in (2.3), we have

$$\begin{aligned}
\boldsymbol{\Sigma}_{t|t} &= (\mathbf{I}_{n_{\mathcal{S}}} - \mathbf{K}_t\mathbf{H}_t)\boldsymbol{\Sigma}_{t|t-1} \\
&= \mathbf{B}_{t|t-1}\big(\mathbf{I}_{n_{\mathcal{S}}} - \mathbf{B}'_{t|t-1}\mathbf{H}'_t(\mathbf{H}_t\mathbf{B}_{t|t-1}\mathbf{B}'_{t|t-1}\mathbf{H}'_t + \mathbf{R}_t)^{-1}\mathbf{H}_t\mathbf{B}_{t|t-1}\big)\mathbf{B}'_{t|t-1} \\
&= \mathbf{B}_{t|t-1}\big(\mathbf{I}_{n_{\mathcal{S}}} + \mathbf{B}'_{t|t-1}\mathbf{H}'_t\mathbf{R}_t^{-1}\mathbf{H}_t\mathbf{B}_{t|t-1}\big)^{-1}\mathbf{B}'_{t|t-1} \\
&= \mathbf{B}_{t|t-1}\boldsymbol{\Lambda}_t^{-1}\mathbf{B}'_{t|t-1} = \mathbf{B}_{t|t}\mathbf{B}'_{t|t},
\end{aligned}$$

where $\mathbf{B}_{t|t} := \mathbf{B}_{t|t-1}(\mathbf{L}_t^{-1})'$, $\mathbf{L}_t$ is the lower Cholesky triangle of $\boldsymbol{\Lambda}_t := \mathbf{I}_{n_{\mathcal{S}}} + \mathbf{B}'_{t|t-1}\mathbf{H}'_t\mathbf{R}_t^{-1}\mathbf{H}_t\mathbf{B}_{t|t-1}$, and we have applied the Sherman-Morrison-Woodbury formula (e.g., Henderson and Searle, 1981) to $\boldsymbol{\Lambda}_t$.

To obtain the filtering mean, we use the Searle set of identities (Searle, 1982, p. 151), to write the Kalman gain as

$$\begin{aligned}
\mathbf{K}_t &= \boldsymbol{\Sigma}_{t|t-1}\mathbf{H}'_t(\mathbf{H}_t\boldsymbol{\Sigma}_{t|t-1}\mathbf{H}'_t + \mathbf{R}_t)^{-1} \\
&= \mathbf{B}_{t|t-1}\mathbf{B}'_{t|t-1}\mathbf{H}'_t(\mathbf{H}_t\mathbf{B}_{t|t-1}\mathbf{B}'_{t|t-1}\mathbf{H}'_t + \mathbf{R}_t)^{-1} \\
&= \mathbf{B}_{t|t-1}(\mathbf{I}_{n_{\mathcal{S}}} + \mathbf{B}'_{t|t-1}\mathbf{H}'_t\mathbf{R}_t^{-1}\mathbf{H}_t\mathbf{B}_{t|t-1})^{-1}\mathbf{B}'_{t|t-1}\mathbf{H}'_t\mathbf{R}_t^{-1} \\
&= \mathbf{B}_{t|t-1}\boldsymbol{\Lambda}_t^{-1}\mathbf{B}'_{t|t-1}\mathbf{H}'_t\mathbf{R}^{-1} = \mathbf{B}_{t|t}\mathbf{B}'_{t|t}\mathbf{H}'_t\mathbf{R}_t^{-1},
\end{aligned}$$

and so we have

$$\begin{aligned}
\boldsymbol{\mu}_{t|t} &= \boldsymbol{\mu}_{t|t-1} + \mathbf{K}_t(\mathbf{y}_t - \mathbf{H}_t\boldsymbol{\mu}_{t|t-1}) \\
&= \boldsymbol{\mu}_{t|t-1} + \mathbf{B}_{t|t}\mathbf{B}'_{t|t}\mathbf{H}'_t\mathbf{R}_t^{-1}(\mathbf{y}_t - \mathbf{H}_t\boldsymbol{\mu}_{t|t-1}).
\end{aligned}$$

Thus, the MRF update step in Algorithm 1 is exact for given $\boldsymbol{\mu}_{t|t-1}$ and $\boldsymbol{\Sigma}_{t|t-1} = \mathbf{B}_{t|t-1}\mathbf{B}'_{t|t-1}$. Crucially, we will show in Proposition 3 that $\mathbf{B}_{t|t}$ has the same sparsity structure as $\mathbf{B}_{t|t-1}$, and hence it satisfies the block-sparsity assumption at the beginning of Section 2.3.2.

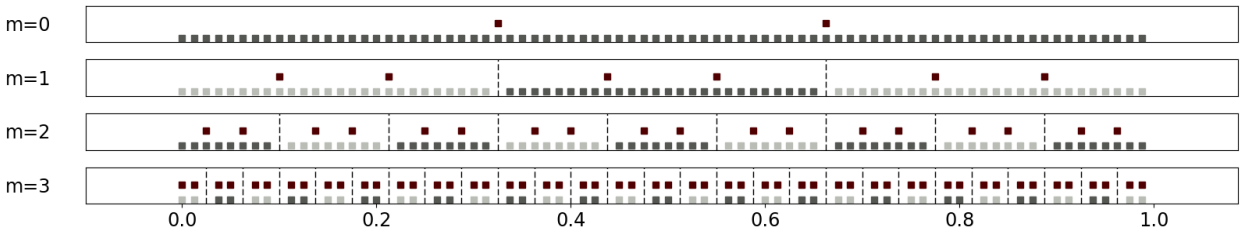


Figure 2.1: Illustration of knot placement for a regular grid of $n_{\mathcal{S}} = 80$ points on a one-dimensional domain $\mathcal{D}$ (x-axis). Recursively for $m = 0, 1, \ldots, M$ (with $M = 3$ here), each region is split into $J = 3$ subregions (dashed lines), with $r_m = 2$ knots per region (maroon dots).

2.3.4 The multi-resolution decomposition

We now propose an approximate multi-resolution decomposition (MRD) of a generic spatial covariance matrix $\mathbf{\Sigma}$, which is used in the forecast step of the MRF in Algorithm 2.3.4.2. Specifically, we consider a vector $\mathbf{x} = \big(x(\mathbf{g}_1), \ldots, x(\mathbf{g}_{n_{\mathcal{S}}})\big)' \sim \mathcal{N}_{n_{\mathcal{S}}}(\mathbf{0}, \mathbf{\Sigma})$, evaluated at a grid $\mathcal{S} = \{\mathbf{g}_1, \ldots, \mathbf{g}_{n_{\mathcal{S}}}\}$ over the spatial domain $\mathcal{D}$. The MRD is based on a multi-resolution approximation of Gaussian processes (Katzfuss, 2017) — see Section 2.5.2 for more details.

2.3.4.1 Partitioning and knots

The MRD requires a domain partitioning and selection of knots at M resolutions. Consider a recursive partitioning of $\mathcal{D}$ into J regions, $\mathcal{D}_1, \ldots, \mathcal{D}_J$, each of which is again divided into J smaller subregions (e.g., $\mathcal{D}_2$ is split into subregions $\mathcal{D}_{21}, \ldots, \mathcal{D}_{2J}$), and so forth, up to resolution M. We write this as

$$\mathcal{D}_{j_1,\ldots,j_m} = \dot{\cup}_{j_{m+1}=1}^{J} \mathcal{D}_{j_1,\ldots,j_{m+1}}, \qquad (j_1, \ldots, j_m) \in \{1, \ldots, J\}^m, \quad m = 0, \ldots, M-1.$$

Let $\mathcal{S}_{j_1,\ldots,j_m} = \mathcal{S} \cap \mathcal{D}_{j_1,\ldots,j_m}$ be the grid points that lie in region $\mathcal{D}_{j_1,\ldots,j_m}$, and let $\mathcal{I}_{j_1,\ldots,j_m} = \{i : \mathbf{g}_i \in \mathcal{D}_{j_1,\ldots,j_m}\}$ be the corresponding indices, and so $\mathcal{I} = \{1, \ldots, n_{\mathcal{S}}\}$.

Further, we require a hierarchy of "knot" indices, such that $\mathcal{K}_{j_1,\ldots,j_m}$ is a small set of r_m indices chosen from $\mathcal{I}_{j_1,\ldots,j_m}$. It is assumed that for each resolution m, the number of knots is roughly the same in each subregion (i.e., $|\mathcal{K}_{j_1,\ldots,j_m}| = r_m$), while it may change across resolutions. We use

$\mathcal{K}^m = \bigcup_{j_1,\ldots,j_m} \mathcal{K}_{j_1,\ldots,j_m}$ to denote the set of all knots at resolution m, and define $\mathcal{K}^{0:m} = \bigcup_{l=0}^{m} \mathcal{K}^l$ as the set of all knots at resolutions 0 through m. To ensure that $\{\mathcal{K}_{j_1,\ldots,j_m} : (j_1,\ldots,j_m) \in \{1,\ldots,J\}^m; m = 0,1,\ldots,M\}$ is a partition of $\{1,\ldots,n_{\mathcal{S}}\}$, we sequentially choose $\mathcal{K}_{j_1,\ldots,j_m} \subset (\mathcal{I}_{j_1,\ldots,j_m} \setminus \mathcal{K}^{0:m-1})$ for $m = 0,1,\ldots,M$.

In practice, we often choose $J = 2$ or $J = 4$. Each r_{m-1} should be sufficiently large to capture the dependence between the $\mathcal{D}_{j_1,\ldots,j_m}$ that is not already captured at lower resolutions, which often means that r_m can decrease as a function of m. Each set of knots $\mathcal{K}_{j_1,\ldots,j_m}$ could be chosen as a roughly uniform grid over the subregion $\mathcal{D}_{j_1,\ldots,j_m}$. The partitioning and knot selection is illustrated in a toy example in Figure 2.1.

Note that because $\mathcal{S}$ is assumed constant over time here, we only need to do this partitioning and selection of knots once (not at each time point). We also assume that the elements in $\mathbf{x}_t$ are ordered such that if $(j_1,\ldots,j_M) \succ_L (i_1,\ldots i_M)$, where $\succ_L$ stands for lexicographic ordering, then $\min(\mathcal{I}_{j_1,\ldots,j_M}) > \max(\mathcal{I}_{i_1,\ldots,i_M})$.

2.3.4.2 The MRD algorithm

For index sets $\mathcal{J}_1$ and $\mathcal{J}_2$, denote by $\mathbf{\Sigma}[\mathcal{J}_1,\mathcal{J}_2]$ the submatrix of $\mathbf{\Sigma}$ obtained by selecting the $\mathcal{J}_1$ rows and $\mathcal{J}_2$ columns, and $\mathbf{\Sigma}[\mathcal{J}_1,:]$ is the submatrix of the $\mathcal{J}_1$ rows and all columns. Based on grid indices $\{\mathcal{I}_{j_1,\ldots,j_m}\}$ and knot indices $\{\mathcal{K}_{j_1,\ldots,j_m}\}$ selected as described in Section 2.3.4.1, the multi-resolution decomposition of a spatial covariance matrix $\mathbf{\Sigma}$ proceeds as follows:

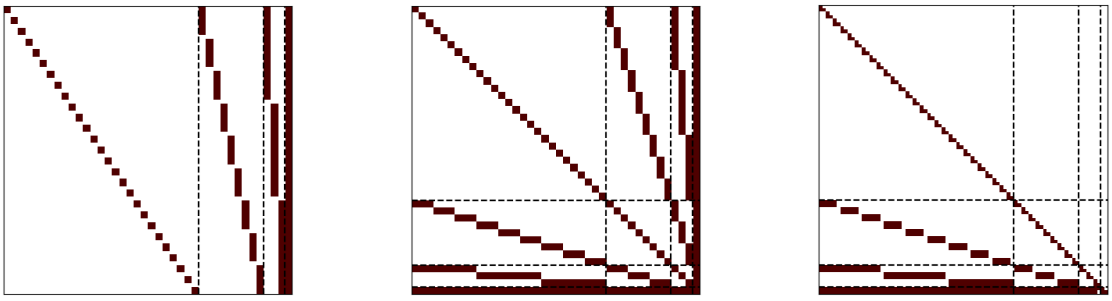
Algorithm 2: Multi-resolution decomposition of $\mathbf{\Sigma}$

For $m = 0,1,\ldots,M$ and $(j_1,\ldots,j_m) \in \{1,\ldots,J\}^m$:

- For $\ell = 0,\ldots,m$, compute

$$\begin{aligned}\mathbf{W}^{\ell}_{j_1,\ldots,j_m} &= \mathbf{\Sigma}[\mathcal{I}_{j_1,\ldots,j_m}, \mathcal{K}_{j_1,\ldots,j_\ell}] - \textstyle\sum_{k=0}^{\ell-1} \mathbf{W}^{k}_{j_1,\ldots,j_m} (\mathbf{V}^{k}_{j_1,\ldots,j_k})^{-1} (\mathbf{V}^{k}_{j_1,\ldots,j_\ell})' \\ \mathbf{V}^{\ell}_{j_1,\ldots,j_m} &= \mathbf{\Sigma}[\mathcal{K}_{j_1,\ldots,j_m}, \mathcal{K}_{j_1,\ldots,j_\ell}] - \textstyle\sum_{k=0}^{\ell-1} \mathbf{V}^{k}_{j_1,\ldots,j_m} (\mathbf{V}^{k}_{j_1,\ldots,j_k})^{-1} (\mathbf{V}^{k}_{j_1,\ldots,j_\ell})'.\end{aligned} \quad (2.4)$$

- Set $\mathbf{B}_{j_1,\ldots,j_m} = \mathbf{W}^{m}_{j_1,\ldots,j_m} (\mathbf{V}^{m}_{j_1,\ldots,j_m})^{-1/2}$.



(a) $\mathbf{B} = \mathrm{MRD}(\boldsymbol{\Sigma})$ (b) $\mathbf{B}'\mathbf{B}$ (c) $\mathbf{L} = \mathrm{chol}(\mathbf{B}'\mathbf{B})$ and $\mathbf{L}^{-1}$

Figure 2.2: Sparsity patterns for $n_{\mathcal{S}} = 80$, $M = 3$, $J = 3$, and $r_m = 2$ for $m = 0, \ldots, 3$. Rows and columns correspond to hierarchically arranged elements of the grid $\mathcal{G}$ in Figure 2.1 from resolution $m = 3$ down to $m = 0$. Blocks corresponding to different resolutions are separated by dashed lines.

Return $\mathbf{B} = \mathrm{MRD}(\boldsymbol{\Sigma})$, where $\mathbf{B} = \left(\mathbf{B}^M, \mathbf{B}^{M-1}, \ldots, \mathbf{B}^0\right)$ with $\mathbf{B}^m = \mathrm{blockdiag}(\{\mathbf{B}_{j_1,\ldots,j_m} : (j_1, \ldots, j_m) \in \{1, \ldots, J\}^m\})$.

The resulting matrix $\mathbf{B}$ is of the same size as $\boldsymbol{\Sigma}$ but has a block-sparse structure, which is illustrated in Figure 2.2a.

2.4 Properties of the multi-resolution filter

2.4.1 Approximation accuracy

The only difference between the MRF (Algorithm 1 and the exact Kalman filter (Section 2.2.2) is the MRD approximation of the forecast covariance matrix at each time point; that is, instead of taking $\boldsymbol{\Sigma}_{t|t-1} = \boldsymbol{\Sigma}^F_{t|t-1}$, the MRF assumes $\boldsymbol{\Sigma}_{t|t-1} = \mathbf{B}_{t|t-1}\mathbf{B}'_{t|t-1}$ with $\mathbf{B}_{t|t-1} = \mathrm{MRD}(\boldsymbol{\Sigma}^F_{t|t-1})$. Hence, the MRF is exact when the MRD at each time point is exact.

However, the MRD is only exact in certain special cases. One trivial example is given by $M = 0$ and $r_0 = n_{\mathcal{S}}$ (see Section C.1). Thus, the MRF converges to the exact Kalman filter as $r_0 \to n_{\mathcal{S}}$, but computational feasibility for large $n_{\mathcal{S}}$ relies on $r_0 \ll n_{\mathcal{S}}$. Another instance of exactness is when $\boldsymbol{\Sigma}^F_{t|t-1}$ is based on an exponential covariance function on a one-dimensional domain, $\mathcal{D} \subset \mathbb{R}$, and we place a total of $r_m = J - 1$ knots, one at each subregion boundary (Katzfuss and Gong, 2019, Prop. 6). Figure 2.1 provides an example of such knot placement.

Finally, approximation error can also be avoided when $\mathbf{E}_t = c_t \mathbf{I}_{n_\mathcal{S}}$ with $c_t \in \mathbb{R}^+$ and $\mathbf{Q}_t = \mathbf{0}$. In this case we can set $\mathbf{B}_{t|t-1} := \sqrt{c_t}\, \mathbf{B}_{t-1|t-1}$, rather than employing the MRD in the forecast step. In data assimilation, the assumption $\mathbf{Q}_t = \mathbf{0}$ is quite common, when model error is incorporated through multiplicative inflation of the forecast covariance matrix.

Aside from these special cases, the MRD and hence the MRF are approximate. However, the MRA, which is the technique underlying the MRD (see Section 2.5.2), can provide excellent accuracy, as has been shown, for example, by Katzfuss (2017), Katzfuss and Gong (2019) and in a recent comparison of different methods for large spatial data (Heaton et al., 2019). In applications where accuracy is crucial, one could successively increase the number of knots r_m used at low resolutions until the inference "converges." We demonstrate the MRF's accuracy numerically in Sections 2.7 and 2.8. In practice, the SSM in (2.1)–(2.2) will usually be an approximation to the true system, and we expect the MRD approximation error to often be negligible relative to the error due to model misspecification.

2.4.2 Computational complexity

We now determine the memory and time complexity of the MRF algorithm under the assumption that $n = \mathcal{O}(n_\mathcal{S}) = \mathcal{O}(n_t)$ for all $t = 1, 2, \ldots$. We also define $N := \sum_m r_m$.

2.4.2.1 Sparsity and memory requirements

As can be seen in Algorithm 2.3.4.2, a multi-resolution factor is composed of block-diagonal submatrices by construction. The following proposition quantifies the number of its nonzero elements.

PROPOSITION 1. *For a covariance matrix $\mathbf{\Sigma}$, each row of $\mathbf{B} = \mathrm{MRD}(\mathbf{\Sigma})$ has N nonzero elements.*

Thus, if $r_m = r$ for $m = 0, \ldots, M$, then each row of $\mathbf{B}$ has exactly $(M+1)r$ nonzero elements. Figure 2.2a illustrates this case for $M = 3, J = 3$ and $r = 2$. The MRD results in a convenient structure of the inner product of the multi-resolution factor. The following statement describes the sparsity pattern of this inner product (see Figure 2.2b), while Proposition 3 shows its usefulness in applications to filtering problems.

PROPOSITION 2. *Let* $\mathbf{B} = \mathrm{MRD}(\mathbf{\Sigma})$ *for some covariance matrix* $\mathbf{\Sigma}$*. Then* $\mathbf{B}'\mathbf{B}$ *is a block matrix with* $M+1$ *row blocks and* $M+1$ *column blocks. For* $k, l = 0, \ldots, M$ *with* $k \geq l$*, the* $(M+1-k, M+1-l)$*-th block is of dimension* $|\mathcal{K}^k| \times |\mathcal{K}^l|$ *and is itself block-diagonal with blocks that are* r_l *columns wide.*

The following technical assumption ensures that both $\mathbf{H}_t$ and $\mathbf{R}_t$ are block-diagonal with blocks corresponding to indices $\mathcal{I}_{j_1,\ldots,j_M}$ within each of the finest subregions:

ASSUMPTION 1. *Let* $i \in \mathcal{I}_{i_1,\ldots,i_M}$ *and* $j \in \mathcal{I}_{j_1,\ldots,j_M}$*. Assume* $\mathbf{R}_t[i,j] = 0$*, unless* $(i_1, \ldots, i_M) = (j_1, \ldots, j_M)$*. Further, if* $\mathbf{H}_t[i,j] \neq 0$*, then* $\mathbf{H}_t[i,k] = 0$ *for all* $k \notin \mathcal{I}_{j_1,\ldots,j_M}$*. Finally, if* $i_1, i_2 \in \mathcal{I}_{j_1,\ldots,j_M}$ *and* $i_1 < i_2$*, then for all* i_3 *with* $i_1 < i_3 < i_2$*, we have* $i_3 \in \mathcal{I}_{j_1,\ldots,j_M}$*.*

This assumption guarantees the key property of the MRF: The sparsity pattern of the multi-resolution factor is preserved in the update step; that is, $\mathbf{B}_{t|t} \in \mathcal{S}(\mathbf{B}_{t-1|t-1})$, where $\mathcal{S}(\mathbf{G})$ denotes the set of matrices whose set of structural zeros is the same or a superset of the structural zeros in some matrix $\mathbf{G}$. We also use $\mathbf{G}^L$ to denote the lower triangle of $\mathbf{G}$, meaning that $\mathbf{G}^L[i,j] = \mathbf{G}[i,j]$ if $i \geq j$, and $\mathbf{G}^L[i,j] = 0$ otherwise.

PROPOSITION 3. *Let* $\mathbf{B}_{t|t-1}, \mathbf{B}_{t|t}, \prec_t, \mathbf{L}_t$ *be defined as in Algorithm 1. Under Assumption 1, we have:*

1. $\prec_t \in \mathcal{S}(\mathbf{B}'_{t|t-1}\mathbf{B}_{t|t-1})$*;*
2. $\mathbf{L}_t \in \mathcal{S}(\prec_t^L)$ *and* $\mathbf{L}_t^{-1} \in \mathcal{S}(\prec_t^L)$*;*
3. $\mathbf{B}_{t|t} \in \mathcal{S}(\mathbf{B}_{t|t-1})$.

We state one more proposition that proves useful in determining the computational complexity:

PROPOSITION 4. *If* $\mathbf{L}_t$ *is the lower Cholesky factor of* $\prec_t$*, then each column of* $\mathbf{L}_t$ *has at most* $\mathcal{O}(N)$ *nonzero elements.*

Figure 2.2c illustrates the structure of $\mathbf{L}$.

The results above show that all matrices computed in the MRF Algorithm 1 are very sparse, with only $\mathcal{O}(nN)$ nonzero entries. The update step preserves the sparsity, so that $\mathbf{B}_{t|t} \in \mathcal{S}(\mathbf{B}_{t|t-1})$.

Due to the Markov structure of order 1 implied by our state-space model, there is no need to store matrices from previous time points, and so the memory complexity of the entire MRF algorithm is $\mathcal{O}(nN)$.

2.4.2.2 Computation time

For determining the time complexity of the MRF, we assume that the number of knots within each subregion is constant across resolutions (i.e. $r_m = r$ for $m = 0, \ldots, M$) and so $N = (M + 1)r$. While the efficacy of our method does not depend on this assumption, it greatly simplifies the complexity calculations and helps to develop an intuition regarding its computational benefits.

PROPOSITION 5. *Given a covariance matrix $\mathbf{\Sigma}$, $\mathbf{B} = \mathrm{MRD}(\mathbf{\Sigma})$ can be computed in $\mathcal{O}(nN^2)$ time using Algorithm 2.3.4.2.*

We further assume that the evolution is local, in the sense that the nonzero elements in any given row of $\mathbf{E}_t$ only correspond to grid points in a small number of regions at the finest resolution of the domain partitioning:

ASSUMPTION 2. *Assume that the evolution matrix $\mathbf{E}_t$ is sparse with at most $\mathcal{O}(r)$ nonzero elements per row, which must only correspond to a small, constant number of subregions,*

$$|\{\mathcal{I}_{j_1,\ldots,j_M} : \exists j \in \mathcal{I}_{j_1,\ldots,j_M} \textit{ such that } \mathbf{E}_t[i,j] \neq 0\}| \leq c, \quad i = 1, \ldots, n.$$

For example, for local evolution in two-dimensional space, we have $c \leq 4$.

PROPOSITION 6. *Under Assumptions 1 and 2, the MRF in Algorithm 1 requires $\mathcal{O}(nN^2)$ operations at each time step t.*

In practice, $N = (M + 1)r$ is chosen by the user depending on the required approximation accuracy and the available computational resources. For fixed N, the time and memory complexity of Algorithm 1 are linear in n. If M increases as $M = \mathcal{O}(\log n)$ for increasing n (e.g., Katzfuss, 2017) and r is held constant, the resulting complexity is quasilinear.

2.4.3 Distributed computation

For truly massive dimensions (i.e., $n_{\mathcal{S}} = \mathcal{O}(10^7)$ or more), memory limitations will typically require distributing the analysis across several computational nodes. The MRF is well suited for such a distributed environment, as information pertaining to different subregions of the domain can be stored and processed in separate nodes, with limited communication overhead required between nodes. We plan to leverage these properties of the MRF by designing an implementation of Algorithm 1 that can be deployed in a high-performance-computation environment. We include further details in Section C.2.

2.4.4 Forecasting and smoothing

Forecasting is straightforward using the MRF. Given the filtering distribution $\mathbf{x}_T|\mathbf{y}_{1:T}$ as obtained by Algorithm 1, we can compute the k-step-ahead forecast $\mathbf{x}_{T+k}|\mathbf{y}_{1:T}$ by simply carrying out the forecast step in Algorithm 1 k times, while skipping the update step. More precisely, we carry out Algorithm 1 for $t = T+1, \ldots, T+k$, but at each time point t, we replace Step 2 by simply setting $\boldsymbol{\mu}_{t|t} = \boldsymbol{\mu}_{t|t-1}$ and $\mathbf{B}_{t|t} = \mathbf{B}_{t|t-1}$. The accuracy of such forecasts will depend heavily on the quality of the evolution matrices $\mathbf{A}_t$, and so a physics-informed evolution can result in much better forecasts than simple models such as random walks.

In some applications, one might also be interested in obtaining retrospective smoothing distributions $\mathbf{x}_t|\mathbf{y}_{1:T}$ for $t < T$. These can be computed exactly by carrying out the Kalman filter up to time T, and then carrying out recursive backward smoothing (e.g., Rauch et al., 1965), but this is not feasible for large grids. It is challenging to extend the MRF by deterministic backward-smoothing operations that preserve sparsity, but it may be possible to devise a scalable MRF-based forward-filter-backward-sampler algorithm. We intend to investigate this modification in future work.

2.5 Connections to existing methods

In this section, we discuss in some detail the connections between our MRF and hierarchical matrix decompositions and basis-function approximations. Further, in Section C.3, we discuss

connections to multi-resolution autoregressive models, which demonstrate that the MRF can also be interpreted as a nested Kalman filter that proceeds over resolutions within each outer filtering step over time.

2.5.1 MRD as hierarchical low-rank decomposition

Hierarchical off-diagonal low-rank (HODLR) matrices are a popular tool in numerical analysis, and they have recently also been applied to Gaussian processes (e.g. Ambikasaran and Darve, 2013; Ambikasaran et al., 2016). In HODLR matrices, the off-diagonal blocks are recursively specified or approximated as low-rank matrices. In this section, we show the connection between the HODLR format and the MRD when $J = 2$.

DEFINITION 1. (Ambikasaran et al., 2016) *A matrix $\mathbf{K} \in \mathbb{R}^{N \times N}$ is termed a 1–level hierarchical off-diagonal low-rank (HODLR) matrix of rank p, if it can be written as*

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_1^{(1)} & \mathbf{U}_1^{(1)}(\mathbf{V}_1^{(1)})' \\ \mathbf{U}_2^{(1)}(\mathbf{V}_2^{(1)})' & \mathbf{K}_2^{(1)} \end{bmatrix},$$

where $\mathbf{K}_i^{(1)} \in \mathbb{R}^{N/2 \times N/2}$, and $\mathbf{U}_i^{(1)}, \mathbf{V}_i^{(1)} \in \mathbb{R}^{N/2 \times p}$. We call $\mathbf{K}$ an m–level HODLR matrix of rank p if both diagonal blocks are $(m-1)$–level HODLR matrices of rank p.

If we use H_m^p to denote the set of all m-level HODLR matrices of rank p, then it follows that $H_m^p \subset H_{m-1}^p$. The optimal low-rank representation is obtained by specifying the matrices $\mathbf{U}_i^{(j)}$ and $\mathbf{V}_i^{(j)}$ as the first p singular vectors of the corresponding off-diagonal submatrix (Hogben, 2006, Item 5.6.13), but this is prohibitively expensive. Ambikasaran et al. (2016) discuss multiple ways of approximating this low-rank representation.

We now show that the outer product of an MRD factor is a HODLR matrix, specifically one in which the low-rank approximations are obtained as skeleton factorizations.

PROPOSITION 7. *Let $\mathbf{B} = \mathrm{MRD}(\mathbf{\Sigma})$, where the decomposition is based on a partitioning scheme with $J = 2$ and $r_m = r$ for $m = 0, \ldots, M$. Then, $\mathbf{B}\mathbf{B}' \in H_M^r$.*

The proof is given in Appendix A.1. It can easily be extended to r_m varying by resolution.

Thus, the MRF approximation of the prior covariance matrix, $\mathbf{\Sigma}_{t|t-1} = \mathbf{B}_{t|t-1}\mathbf{B}'_{t|t-1}$, is a HODLR matrix (Ambikasaran et al., 2016). In contrast to previous approaches using HODLR matrices for spatio-temporal models (e.g. Li et al., 2014; Saibaba et al., 2015), the block-sparse MRD matrices allow the MRF to handle non-diagonal evolution matrices $\mathbf{E}_t$ and full-rank model-error matrices $\mathbf{Q}_t$.

2.5.2 MRD as basis-function approximation

The MRD is related to the multi-resolution approximation (MRA; Katzfuss, 2017) of a Gaussian process as a weighted sum of increasingly compactly supported basis functions at M resolutions. While the MRD adapts the MRA to an approximate decomposition of a covariance matrix evaluated at a spatial grid, $\mathbf{\Sigma} = \mathbf{B}\mathbf{B}'$, we can similarly interpret each column of $\mathbf{B}$ as a basis vector over the grid. In other words, the spatial field $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$ is approximated as $\mathbf{x} \approx \mathbf{B}\boldsymbol{\eta}$, where $\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is the vector of independent standard normal weights. By interpolating the values of the basis vectors between grid points, we can visualize the columns of $\mathbf{B}$ as basis functions, which is illustrated in Figure 2.3.

The basis functions obtained in this way exhibit interesting properties. Their support is increasingly compact as the resolution increases, and basis functions at low resolution capture the large-scale structure. There are strong connections between the MRD and stochastic wavelets, with the major difference that the shape of the basis functions in the MRD adapts to the covariance structure in $\mathbf{\Sigma}$. This adaptation is especially useful in the spatio-temporal context here, which requires approximation of the forecast covariance matrix $\mathbf{\Sigma}^F_{t|t-1}$ that depends on the data at previous time points and is hence highly nonstationary. The compact support stems from the assumption of a block-sparse structure at each resolution in the MRD, which is equivalent to assuming that the remainder at each resolution is conditionally independent between subregions at that resolution, given the terms at lower resolutions. In general, this assumption is not satisfied and thus produces an approximation error, although the MRD is exact in some special cases (see Section 2.4.1).

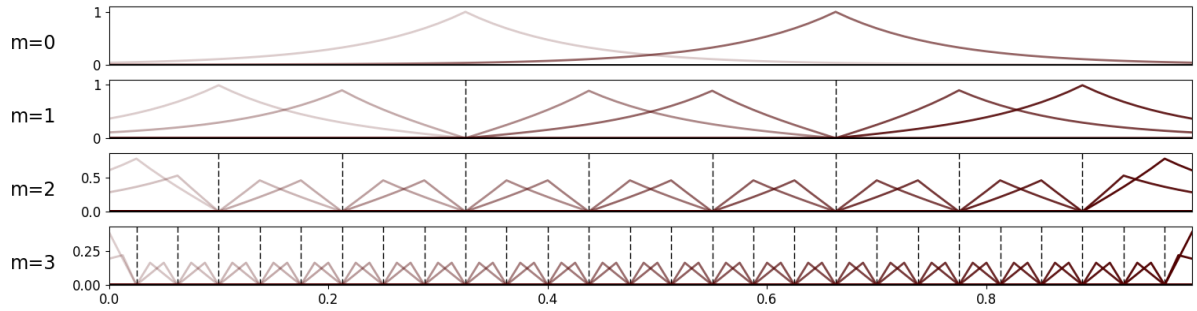


Figure 2.3: Basis functions obtained by interpolating the entries in each column of $\mathbf{B} = \mathrm{MRD}(\mathbf{\Sigma})$ in Figure 2.2a using the grid from Figure 2.1, with $\mathbf{\Sigma}$ based on an exponential covariance with range 0.3. Each basis function's support is restricted to one of the subregions (dashed lines) at each resolution.

2.6 Parameter inference

If there are random, time-varying parameters $\boldsymbol{\theta}_t$ in any of the matrices in (2.1)–(2.2), that are distinct from the Gaussian state $\mathbf{x}_t$, we can make inference on these parameters using an approximate Rao-Blackwellized particle filter (Doucet et al., 2000), in which we use the MRF algorithm to approximately integrate out the high-dimensional state $\mathbf{x}_t$ at each time point. An alternative approach, based on including the unknown parameters in the state vector, otherwise known as data augmentation, tends to work poorly for certain parameters and thus is less general (e.g., DelSole and Yang, 2010; Katzfuss et al., 2019).

To derive our filter, note that we have

$$p(\mathbf{y}_{1:T}|\boldsymbol{\theta}_{1:T}) = \prod_{t=1}^{T} p(\mathbf{y}_t|\mathbf{y}_{1:t-1}, \boldsymbol{\theta}_{1:t}) =: \prod_{t=1}^{T} \mathcal{L}_t(\boldsymbol{\theta}_{1:t}),$$

where, after integrating out $\mathbf{x}_t$, we have $\mathbf{y}_t|\mathbf{y}_{1:t-1}, \boldsymbol{\theta}_{1:t} \sim \mathcal{N}_{n_t}(\mathbf{H}_t\boldsymbol{\mu}_{t|t-1}, \mathbf{H}_t\mathbf{\Sigma}_{t|t-1}\mathbf{H}_t' + \mathbf{R}_t)$ with $\mathbf{\Sigma}_{t|t-1} = \mathbf{B}_{t|t-1}\mathbf{B}_{t|t-1}'$. By applying a matrix determinant lemma (e.g., Harville, 1997, Thm. 18.1.1) to the determinant and the Sherman-Morrison-Woodbury formula to the quadratic form in the multivariate normal density, it is straightforward to show that the integrated filtering likelihood at time t can be written, up to a constant, as

$$-2\log\mathcal{L}_t(\boldsymbol{\theta}_{1:t}) = 2\log|\mathbf{L}_t| + \log|\mathbf{R}_t| + (\mathbf{y}_t - \mathbf{H}_t\boldsymbol{\mu}_{t|t-1})'\mathbf{R}_t^{-1}(\mathbf{y}_t - \mathbf{H}_t\boldsymbol{\mu}_{t|t-1}) - \tilde{\mathbf{y}}_t'\tilde{\mathbf{y}}_t, \quad (2.5)$$

where $\tilde{\mathbf{y}}_t := \mathbf{B}'_{t|t}\mathbf{H}'_t\mathbf{R}_t^{-1}(\mathbf{y}_t - \mathbf{H}_t\boldsymbol{\mu}_{t|t-1})$, and we have omitted dependence on the parameters $\boldsymbol{\theta}_{1:t}$ for the terms on the right-hand side.

Assuming that the priors for the $\boldsymbol{\theta}_t$ are given by $p_0(\boldsymbol{\theta}_0)$ for $t = 0$, and recursively by $p_t(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{t-1})$ for $t = 1, 2, \ldots$, the particle MRF proceeds as follows:

Algorithm 3: Particle MRF

At time $t = 0$, for $i = 1, \ldots, N_p$, draw $\boldsymbol{\theta}_0^{(i)} \sim p_0(\boldsymbol{\theta}_0)$ with equal weights $w_0^{(i)} = 1/N_p$, and compute $\boldsymbol{\mu}_{0|0}^{(i)} = \boldsymbol{\mu}_{0|0}(\boldsymbol{\theta}^{(i)})$ and $\mathbf{B}_{0|0}^{(i)} = \mathrm{MRD}(\boldsymbol{\Sigma}_{0|0}(\boldsymbol{\theta}_0^{(i)}))$. Then, for each $t = 1, 2, \ldots$, do:

- For $i = 1, \ldots, N_p$:
  - Sample $\boldsymbol{\theta}_t^{(i)}$ from a proposal distribution $q_t(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{t-1}^{(i)})$.
  - Forecast step: Compute $\boldsymbol{\mu}_{t|t-1}^{(i)} = \mathbf{E}_t(\boldsymbol{\theta}_t^{(i)})\boldsymbol{\mu}_{t-1|t-1}^{(i)}$, $\mathbf{B}_{t|t-1}^{F}{}^{(i)} = \mathbf{E}_t(\boldsymbol{\theta}_t^{(i)})\mathbf{B}_{t-1|t-1}^{(i)}$, and $\mathbf{B}_{t|t-1}^{(i)} = \mathrm{MRD}(\boldsymbol{\Sigma}_{t|t-1}^{(i)})$, where $\mathbf{B}_{t|t-1}^{F}{}^{(i)}(\mathbf{B}_{t|t-1}^{F}{}^{(i)})' + \mathbf{Q}_t(\boldsymbol{\theta}_t^{(i)}))$.
  - Update step: Compute $\boldsymbol{\Lambda}_t^{(i)} = \mathbf{I}_{n_\mathcal{S}} + \mathbf{B}_{t|t-1}^{(i)\prime}\mathbf{H}_t(\boldsymbol{\theta}_t^{(i)})'\mathbf{R}_t(\boldsymbol{\theta}_t^{(i)})^{-1}\mathbf{H}_t(\boldsymbol{\theta}_t^{(i)})\mathbf{B}_{t|t-1}^{(i)}$, $\mathbf{L}_t^{(i)}$ as the lower Cholesky triangle of $\boldsymbol{\Lambda}_t^{(i)}$, $\mathbf{B}_{t|t}^{(i)} = \mathbf{B}_{t|t-1}^{(i)}((\mathbf{L}_t^{(i)})^{-1})'$, and $\boldsymbol{\mu}_{t|t}^{(i)} = \boldsymbol{\mu}_{t|t-1}^{(i)} + \mathbf{B}_{t|t}^{(i)}\mathbf{B}_{t|t}^{(i)\prime}\mathbf{H}_t(\boldsymbol{\theta}_t^{(i)})'\mathbf{R}_t(\boldsymbol{\theta}_t^{(i)})^{-1}(\mathbf{y}_t - \mathbf{H}_t(\boldsymbol{\theta}_t^{(i)})\boldsymbol{\mu}_{t|t-1}^{(i)})$.
  - Using the quantities just computed for $\boldsymbol{\theta}_t^{(i)}$, calculate $\mathcal{L}_t(\boldsymbol{\theta}_{1:t}^{(i)})$ as in (2.5), and update the particle weight $w_t^{(i)} \propto w_{t-1}^{(i)}\mathcal{L}_t(\boldsymbol{\theta}_{1:t}^{(i)})p_t(\boldsymbol{\theta}_t^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)})/q_t(\boldsymbol{\theta}_t^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)})$.
- The filtering distribution is $p(\boldsymbol{\theta}_t, \mathbf{x}_t|\mathbf{y}_{1:t}) = \sum_{i=1}^{N_p} w_t^{(i)}\delta_{\boldsymbol{\theta}_t^{(i)}}(\boldsymbol{\theta}_t)\,\mathcal{N}_{n_\mathcal{S}}(\mathbf{x}_t|\boldsymbol{\mu}_{t|t}^{(i)}, \mathbf{B}_{t|t}^{(i)}\mathbf{B}_{t|t}^{(i)\prime})$.
- If desired, resample the triplets $\{(\boldsymbol{\theta}_t^{(i)}, \boldsymbol{\mu}_{t|t}^{(i)}, \mathbf{B}_{t|t}^{(i)}) : i = 1, \ldots, N_p\}$ with weights $w_t^{(1)}, \ldots, w_t^{(N_p)}$, respectively, to obtain an equally weighted sample (see, e.g., Douc et al., 2005, for a comparison of resampling schemes).

Section C.5 presents numerical experiments demonstrating the accuracy of Algorithm 2.6 and its advantage over a low-rank particle filter.

2.7 Simulation study

We used simulated data to compare the performance of the MRF with several filtering methods:

KF: The Kalman filter (see Section 2.2.2) provides the exact filtering distributions, but has $\mathcal{O}(n^3)$ time complexity at each time point.

MRF: The multi-resolution filter proposed here in Section 2.3, with $\mathcal{O}(nN^2)$ time complexity, where $N = \sum_{m=0}^{M} r_m$.

EnKF: An ensemble Kalman filter with stochastic updates (e.g., Katzfuss et al., 2016, Sect. 3.1). We set the ensemble size to N and use Kanter's function (Kanter, 1997) for tapering such that the resulting matrix has N nonzero entries per row. This results roughly in $\mathcal{O}(nN^2)$ time complexity (e.g., Tippett et al., 2003).

LRF: A low-rank-plus-diagonal filter that can be viewed as a spatio-temporal extension of the modified predictive process (Finley et al., 2009) and as a special case of a fixed-rank filter (Cressie et al., 2010). Moreover, it can be viewed as a special case of the MRF (hence allowing for ease of comparison) with $M = 1$ resolutions and N knots at resolution 0, where each grid point is in its own partition at resolution 1, resulting in a time complexity of $\mathcal{O}(nN^2)$.

MRA: The MRA (Katzfuss, 2017) is a spatial-only method. It can essentially be viewed as a special case of the MRF, for which the filtering distribution at each time t is obtained by assuming that only $\mathbf{y}_t$ and no data at previous time points are available. It has the same $\mathcal{O}(nN^2)$ complexity as the MRF.

While the KF provides the exact filtering distributions, it is only computationally feasible due to the deliberately small problem size chosen here. All other methods attempt to approximate the exact KF solution, but have the advantage of being scalable to much larger grid sizes. For a fair comparison, all approximate methods used the same N, which trades off approximation accuracy and computational complexity. Further, we acknowledge that the EnKF was designed for nonlinear evolution in operational data assimilation, and it is thus more widely applicable than the other

	$n_t/n_{\mathcal{S}}$	ν	λ	σ_w^2	σ_v^2
baseline	0.3	0.5	0.1	0.5	0.05
smooth	0.3	**1.5**	0.1	0.5	0.05
dense obs.	**0.8**	0.5	0.1	0.5	0.05
low noise	0.3	0.5	0.1	0.5	**0.01**

Table 2.1: Settings used in the 1D simulation. Bold values indicate changes with respect to the baseline.

methods.

We used two criteria to compare the performance of the approximate filters: the Kullback-Leibler (KL) divergence between the true and approximated filtering distribution of the state vector (i.e., the joint distribution for the entire spatial field), and the ratio of the root mean squared prediction error (RMSPE) achieved by each approximate method relative to the RMSPE of the KF. Detailed definitions of the criteria can be found in Section C.4.1. Lower is better for both criteria, with optimal values of 0 for the KL divergence and 1 for the RMSPE ratio. In addition, Section C.4.4 examines the performance of all methods in terms of the confidence-interval coverage. All quantities were averaged over 50 simulated datasets.

2.7.1 One-dimensional circular domain

In our first simulation scenario, we considered a diffusion-advection model on a one-dimensional domain consisting of a circle with a unit circumference. After discretizing both the spatial and the temporal dimensions using $n_{\mathcal{S}} = 80$ and $T = 20$ regularly spaced points, respectively, we obtained a linear model as in (2.1)–(2.2), where $\mathbf{E}_t$ was a tri-diagonal matrix and $\mathbf{Q}_t = \sigma_w^2 \left[\mathcal{M}_{\nu,\lambda}(s_i, s_j)\right]_{i,j=1,\dots n_{\mathcal{S}}}$ was based on a Matérn correlation function $\mathcal{M}_{\nu,\lambda}(\cdot,\cdot)$ with smoothness ν and range λ. At each time point, we randomly selected n_t observed locations, so that $\mathbf{H}_t$ is a subset of the identity, and we set $\mathbf{R}_t = \sigma_v^2 \mathbf{I}_{n_t}$. A detailed description of the simulation, including examples of process realizations, is given in Section C.4.2.

Because of the many possible choices of parameters, we first established baseline settings that we considered relevant for practical applications, and then examined the effects of changing them one by one. The resulting simulation scenarios are detailed in Table 2.1. For the MRD, we set

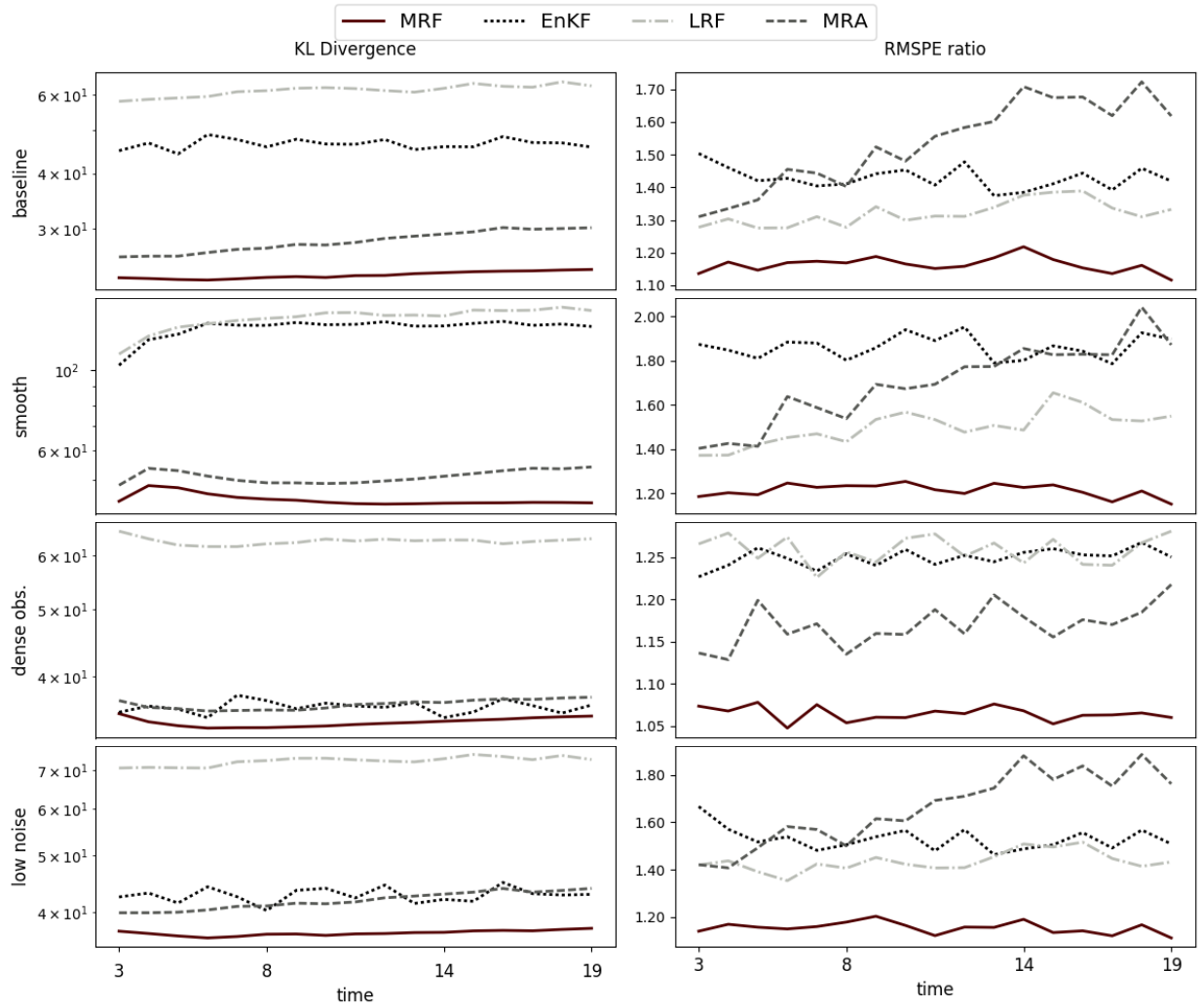


Figure 2.5: Filter scores for different parameter settings; one-dimensional domain. Note that we used different scales on the vertical axis for each plot, with a logarithmic scale for the KL divergence.

$M = 3$, $J = 3$, and $r_m = 2$ for all m, and so we used $N = (3 + 1)2 = 8$ for EnKF, LRF, and MRA.

As shown in Figure 2.5, the MRF performed best in all four scenarios, both in terms of the KL divergence and the RMSPE ratio.

2.7.2 Two-dimensional domain

We also considered a diffusion-advection model on a unit square, and we discretized it on a regular grid of size $n_S = 34 \times 34 = 1{,}156$. As before, we used $T = 20$ evenly spaced time

	$n_t/n_{\mathcal{S}}$	ν	λ	σ_w^2	σ_v^2
baseline	0.1	0.5	0.15	0.5	0.25
smooth	0.1	**1.5**	0.15	0.5	0.25
dense obs.	**0.3**	0.5	0.15	0.5	0.25
low noise	0.1	0.5	0.15	0.5	**0.1**

Table 2.2: Settings used in the 2D simulation. Bold values indicate changes with respect to the baseline.

points. Writing the model in the linear form (2.1)-(2.2), $\mathbf{E}_t$ was a sparse matrix with nonzero entries corresponding to interactions between neighboring grid points to the right, left, top and bottom. A detailed description of the simulation, including examples of process realizations, is given in Section C.4.3.

Similar to the 1D case, we first considered baseline parameter settings and then we changed some of them, one at a time. The multi-resolution decomposition used $M = 4$ and, similar to Katzfuss (2017) we changed J_m across resolutions m: $(J_1, \ldots, J_4) = (2, 4, 4, 4)$. We also varied the numbers of knots r_m used at each resolution: $(r_0, \ldots, r_4) = (16, 8, 6, 6, 6)$. Thus, to achieve a fair comparison, we used $N = 42$ for EnKF, LRF, and MRA. As shown in Figure 2.6, MRF again performed best in all four scenarios.

2.8 Sediment movements in Lake Michigan

We also considered filtering inference on sediment concentration in Lake Michigan over a period of one month, March 1998, based on satellite data. Such inference can be used by hydrologists to increase their understanding of sediment transport mechanisms and fine-tune existing domain-specific models. We closely followed an earlier study of this problem done by Stroud et al. (2010) in the context of spatio-temporal smoothing. Unless specified otherwise, we used the same model and parameter estimates. We briefly summarize the general framework below and indicate the few modifications we introduced.

The lake area was divided into $n_{\mathcal{S}} = 14{,}558$ grid cells of size 2km $\times$ 2km each. We use $\mathbf{x}_t$ to denote the sediment concentrations at the $n_{\mathcal{S}}$ cells at time t. The time dimension was discretized into 409 intervals. The sediment transport model was assumed to be $\mathbf{x}_t = \mathbf{E}_t\mathbf{x}_{t-1} + \boldsymbol{\rho}_t + \mathbf{w}_t$,

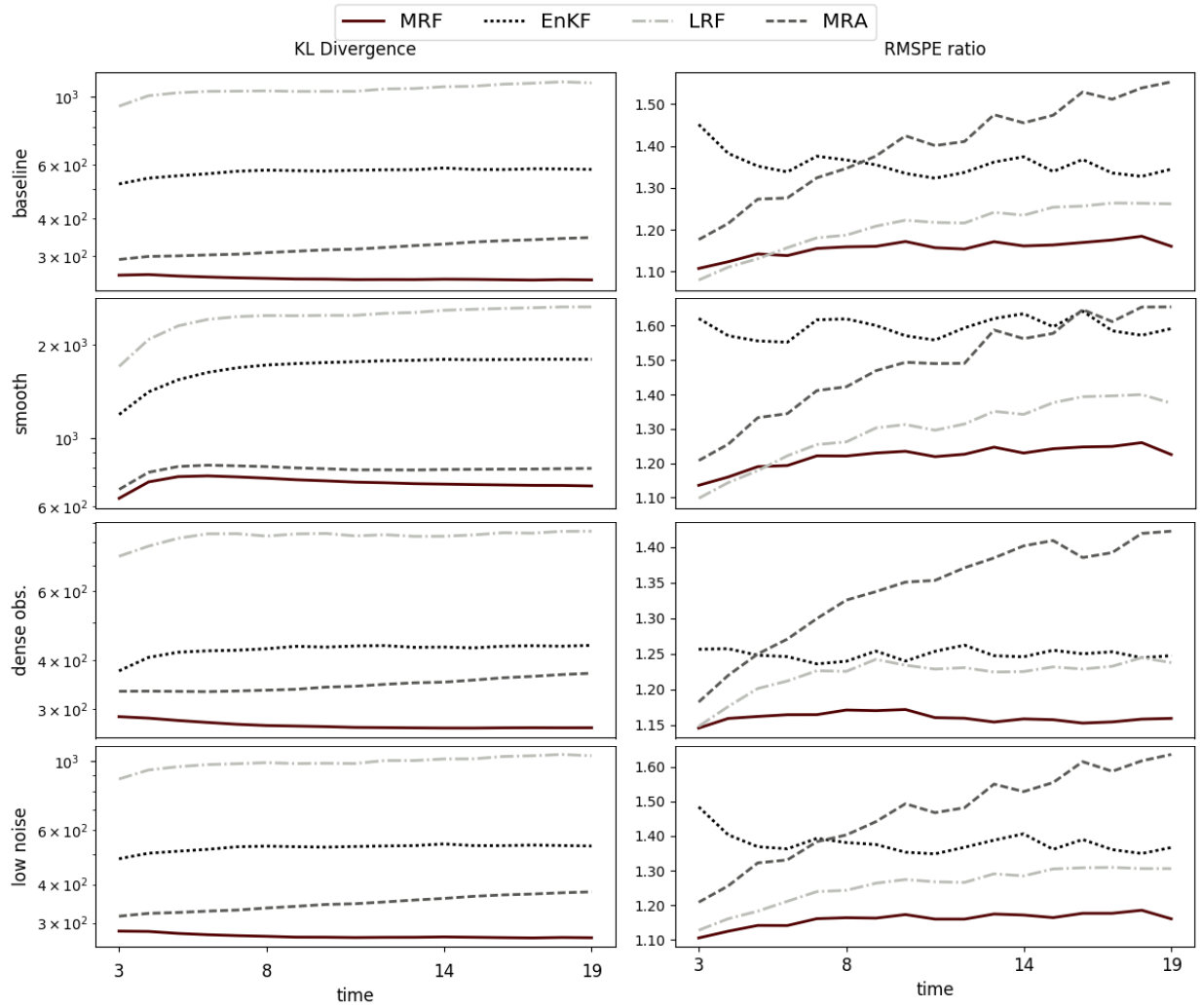


Figure 2.6: Filter scores for different parameter settings; two-dimensional domain. Note that we used different scales on the vertical axis for each plot, with a logarithmic scale for the KL divergence.

where $\mathbf{E}_t$ describes the temporal evolution based on a hydrological PDE model, $\boldsymbol{\rho}_t$ is a vector with external inputs representing the influence of water velocity and bottom sheer stress, and the model error $\mathbf{w}_t$ is assumed to follow a $\mathcal{N}(\mathbf{0}, \mathbf{Q}_t)$ distribution with covariance matrix $\mathbf{Q}_t = (\sigma_\omega^2 \boldsymbol{\Omega}_t \boldsymbol{\Omega}_t') \circ \mathbf{T}$, where $\circ$ denotes element-wise multiplication. All matrices $\boldsymbol{\Omega}_t$ have dimensions $n_\mathcal{S} \times 5$ and reflect the spatial structure of the error in the original study, while $\mathbf{T}$ is taken to be a tapering matrix based on a Kanter covariance function with a tapering radius that leaves about 200 nonzero elements in each row.

The data comprise 10 satellite measurements of remote-sensing reflectance (RSR) at the fre-

	MRF	EnKF	LRF	MRA
RASD	0.08	0.22	0.42	0.72

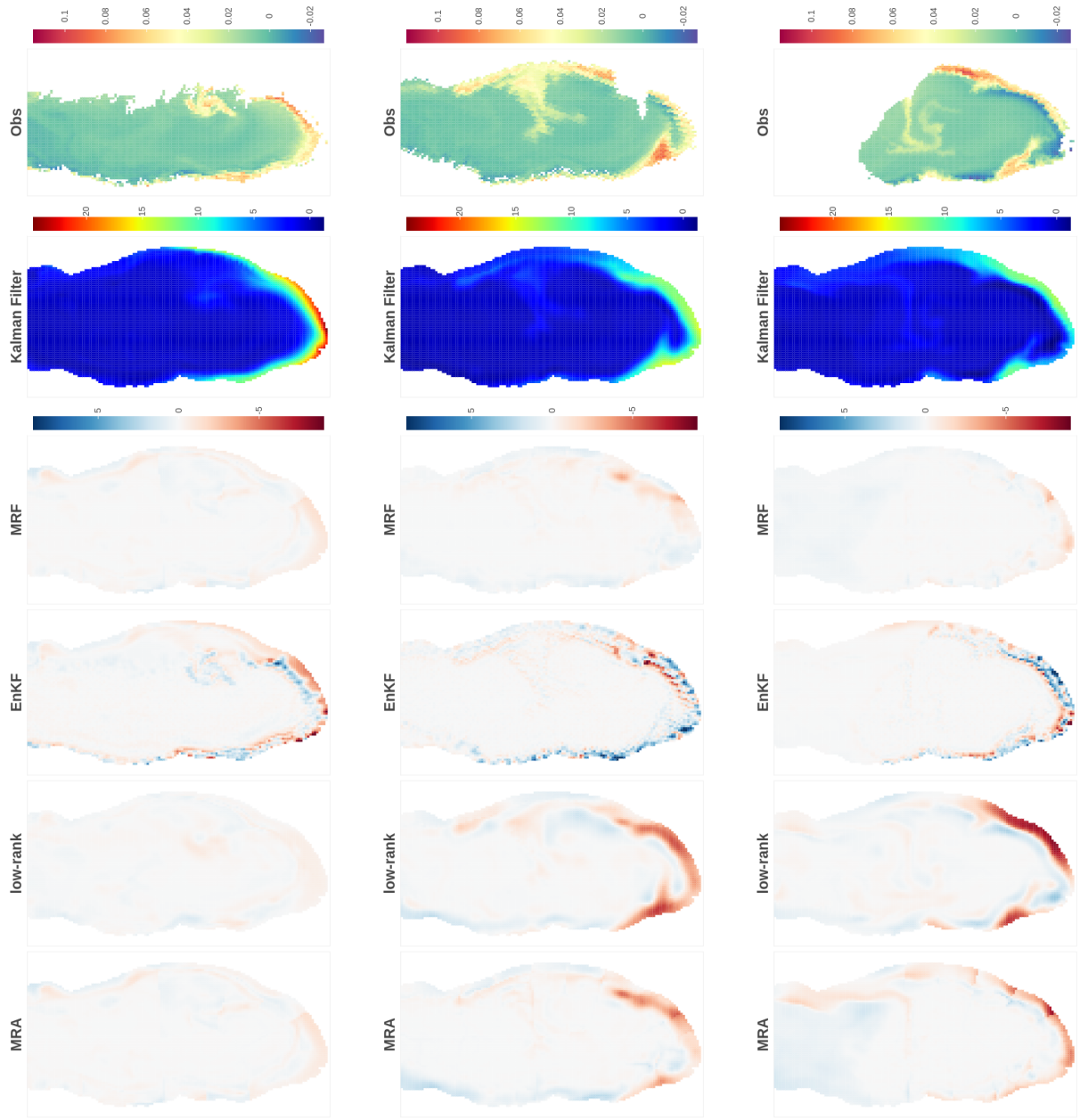
Table 2.3: Root average squared difference (RASD) between approximate and exact filtering means for sediment concentration.

quency of 555 nm taken over the southern basin of Lake Michigan, modified in a way that accounts for the effects of the cloud cover. The observed value at each grid point was assumed to be the first-order Taylor expansion of $h(c) = \theta_0 + \theta_1 \log(1 + \theta_2(c + \theta_3))$ taken around the initial mean of the sediment concentration at time $t = 0$. Using $\mathbf{y}_t$ to denote the vector of observations at time t after removing a time-varying instrument bias and accounting for constant terms in the Taylor expansion, we assumed $\mathbf{y}_t = \mathbf{H}_t\mathbf{x}_t + \mathbf{v}_t$ as in (2.1), where $\mathbf{H}_t$ had only one nonzero element in each row, $\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_t)$, and $\mathbf{R}_t$ was diagonal.

Because of the moderate size of the spatial grid, we were able to compute the exact Kalman filter solution. We set $M = 5$, $J = 4$, and $(r_0, \ldots, r_5) = (16, 8, 8, 8, 4, 4)$ for the MRF, which implied that $N = \sum_m r_m = 48$ for the other approximation methods in Section 2.7. The tapering range used in EnKF was selected such that the tapering matrix had only 5 nonzero elements per row, which corresponds to the setting used by Stroud et al. (2010). While this is inconsistent with the comparison principles outlined in Section 2.7, it made the EnKF perform better in this case.

As the true concentrations were unknown, we compared the approximate filtering means to the exact means obtained by the Kalman filter, using the root average squared difference $\big(\sum_t \sum_i (\hat{\boldsymbol{\mu}}_t[i] - \boldsymbol{\mu}_t^{KF}[i])^2\big)^{1/2}$ between the approximate filtering means $\hat{\boldsymbol{\mu}}_t[i]$ and the KF means $\boldsymbol{\mu}_t^{KF}[i]$, averaged over all times t and grid points i. The results, reported in Table 2.3, show the MRF outperforming all other approximate methods. To visually verify these results, we also present satellite data and sediment concentration estimates for three selected time points in Figure 2.7. A video with all time points can be found at http://spatial.stat.tamu.edu.

For a grid of the size $n_{\mathcal{S}}$ considered here, a single step of the MRF took roughly 5% of the time required by the exact Kalman filter (on a laptop with 8GB of memory and Intel(R) Core(TM) i7-3630QM CPU @ 2.40GHz). More importantly, the MRF scales well to even larger grid sizes



(a) t = 2 (b) t= 267 (c) t=409

Figure 2.7: Satellite data (in log RSR) and exact Kalman filtering means of sediment concentrations (in mg/L), along with differences of approximate filtering means to the Kalman filter. We display the results for the southern basin of the lake only, where differences between the methods are most pronounced.

(see Section 2.4.2), while exact calculations will quickly become infeasible due to memory con-straints. Exact computation times and memory limitations will, of course, depend heavily on the computational environment.

2.9 Conclusions and future work

We introduced the multi-resolution filter (MRF), a new filtering method for linear Gaussian spatio-temporal state-space models, which relies on a block-sparse multi-resolution matrix decomposition. We proved that the sparsity can be preserved under filtering through time, ensuring scalability of the MRF to very large spatial grids. In our comparisons, the MRF substantially outperformed existing methods that can be used to approximate the Kalman filter. We also successfully applied the MRF to inferring sediment concentration in Lake Michigan.

Spatio-temporal processes typically exhibit highly complicated structures that make exact inference intractable, especially in high dimensions. We believe that it is often better to conduct approximate inference for a realistic, intractable model, rather than carrying out "exact" inference for a crude simplification (e.g., a low-rank version) of the model. While it might be challenging to precisely quantify the approximation accuracy in the former case (e.g., for the MRF), approximate inference can give better results than exact inference in a simplified model, which often completely ignores the error incurred by simplifying the model.

While we have focused on spatio-temporal data here, our methods are also applicable to general SSMs of the form (2.1)–(2.2) that do not correspond to physical space and time, as long as some distance between the elements of each state vector can be specified.

Potential future work includes extensions to non-Gaussian data, nonlinear evolution, and smooth-ing inference. We are also developing a user-friendly implementation of the MRF with sensible default settings for the number of knots and domain partitioning.

3. HIERARCHICAL SPARSE CHOLESKY DECOMPOSITION WITH APPLICATIONS TO HIGH-DIMENSIONAL SPATIO-TEMPORAL FILTERING

3.1 Introduction

Symmetric positive-definite matrices arise in spatial statistics, Gaussian-process inference, and spatio-temporal filtering, with a wealth of application areas, including geoscience (e.g., Cressie, 1993; Banerjee et al., 2004), machine learning (e.g., Rasmussen and Williams, 2006), data assimilation (e.g., Nychka and Anderson, 2010; Katzfuss et al., 2016), and the analysis of computer experiments (e.g., Sacks et al., 1989; Kennedy and O'Hagan, 2001). Inference in these areas typically relies on Cholesky decomposition of the positive-definite matrices. However, this operation scales cubically in the dimension of the matrix, and it is thus computationally infeasible for many modern problems and applications, which are increasingly high-dimensional.

Countless approaches have been proposed to address these computational challenges. Heaton et al. (2019) provide a recent review from a spatial-statistics perspective, and Liu et al. (2018) review approaches in machine learning. In high-dimensional filtering, proposed solutions include low-dimensional approximations (e.g., Verlaan and Heemink, 1995; Pham et al., 1998; Wikle and Cressie, 1999; Katzfuss and Cressie, 2011), spectral methods (e.g. Wikle and Cressie, 1999; Sigrist et al., 2015), and hierarchical approaches (e.g., Johannesson et al., 2003; Li et al., 2014; Saibaba et al., 2015; Jurek and Katzfuss, 2018). Operational data assimilation often relies on ensemble Kalman filters (e.g., Evensen, 1994; Burgers et al., 1998; Anderson, 2001; Evensen, 2007; Katzfuss et al., 2016, 2019), which represent distributions by samples or ensembles.

Maybe the most promising approximations for spatial data and Gaussian processes implicitly or explicitly rely on sparse Cholesky factors. The assumption of ordered conditional independence in the popular Vecchia approximation (Vecchia, 1988) and its extensions (e.g., Stein et al., 2004; Datta et al., 2016a; Guinness, 2018; Katzfuss and Guinness, 2019; Katzfuss et al., 2018, 2020; Schäfer et al., 2020) implies sparsity in the Cholesky factor of the precision matrix. Schäfer

et al. (2017) uses an incomplete Cholesky decomposition (IC0) to quickly construct a sparse approximate Cholesky factor of the covariance matrix. However, these methods are not generally applicable to spatio-temporal filtering, because the assumed sparsity is not preserved under filtering operations.

Here, we relate the sparsity of the Cholesky factors of the covariance matrix and the precision matrix to specific assumptions regarding ordered conditional independence. We show that these assumptions are simultaneously satisfied for a particular Gaussian-process approximation that we call hierarchical Vecchia (HV), which is a special case of the general Vecchia approximation (Katzfuss and Guinness, 2019) based on hierarchical domain partitioning (e.g., Katzfuss, 2017; Katzfuss and Gong, 2019). We show that the HV approximation can be computed using a simple and fast incomplete Cholesky decomposition (IC0).

Due to its remarkable property of implying a sparse Cholesky factor whose inverse has equivalent sparsity structure, HV is well suited for extensions to spatio-temporal filtering; this is in contrast to other Vecchia approximations and other spatial approximations relying on sparsity. We provide a scalable HV-based filter for linear Gaussian spatio-temporal state-space models, which is related to the multi-resolution filter of Jurek and Katzfuss (2018). Further, by combining HV with a Laplace approximation (cf. Zilber and Katzfuss, 2019), our method can be used for the analysis of non-Gaussian data. Finally, by combining the methods with the extended Kalman filter (e.g., Grewal and Andrews, 1993, Ch. 5), we obtain fast filters for high-dimensional, non-linear, and non-Gaussian spatio-temporal models. For a given formulation of HV, the computational cost of all of our algorithms scales linearly in the state dimension, assuming sufficiently sparse temporal evolution.

The remainder of this document is organized as follows. In Section 3.2, we specify the relationship between ordered conditional independence and sparse (inverse) Cholesky factors. Then, we build up increasingly complex and general methods, culminating in non-linear and non-Gaussian spatio-temporal filters: in Section 3.3, we introduce hierarchical Vecchia for a linear Gaussian spatial field at a single time point; in Section 3.4, we extend this to non-Gaussian data; and in

Section 3.5, we consider the general spatio-temporal filtering case, including nonlinear evolution and parameter inference on unknown parameters in the model. Section 3.6 contains numerical comparisons to existing approaches. Section 3.7 concludes. Appendices B.1–B.2 contain proofs and further details. Code used for simulations is available at `https://github.com/katzfuss-group/vecchiaFilter`.

3.2 Sparsity of Cholesky factors

We begin by specifying the connections between ordered conditional independence and sparsity of the Cholesky factor of the covariance and precision matrix.

CLAIM 1. *Let* $\mathbf{w}$ *be a normal random vector with variance-covariance matrix* $\mathbf{K}$.

1. *Let* $\mathbf{L} = \mathrm{chol}(\mathbf{K})$ *be the lower-triangular Cholesky factor of the covariance matrix* $\mathbf{K}$. *For* $i > j$*:*

$$\mathbf{L}_{i,j} = 0 \iff w_i \perp w_j \,|\, \mathbf{w}_{1:j-1}$$

2. *Let* $\mathbf{U} = \mathrm{rchol}(\mathbf{K}^{-1}) = \mathbf{P}\,\mathrm{chol}(\mathbf{P}\mathbf{K}^{-1}\mathbf{P})\,\mathbf{P}$ *be the Cholesky factor of the precision matrix under reverse ordering, where* $\mathbf{P}$ *is the reverse-ordering permutation matrix. Then* $\mathbf{U}$ *is upper-triangular, and for* $i > j$*:*

$$\mathbf{U}_{j,i} = 0 \iff w_i \perp w_j \,|\, \mathbf{w}_{1:j-1}, \mathbf{w}_{j+1:i-1}$$

The connection between ordered conditional independence and the Cholesky factor of the precision matrix is well known (e.g., Rue and Held, 2010); Part 2 of our claim states this connection under reverse ordering (e.g., Katzfuss and Guinness, 2019, Prop. 3.3). In Part 1, we consider the lesser-known relationship between ordered conditional independence and sparsity of the Cholesky factor of the covariance matrix, which was recently discussed in Schäfer et al. (2017, Sect. 1.4.2). For completeness, we provide a proof of Claim 1 in Appendix B.2.

Claim 1 is crucial for our later developments and proofs. We will specify a hierarchical Vecchia approximation of Gaussian processes that satisfies both types of conditional independence in Claim

1; the resulting sparsity of the Cholesky factor and its inverse allows extensions to spatio-temporal filtering in Section 3.5.

3.3 Hierarchical Vecchia for large Gaussian spatial data

Consider a latent realization $\mathbf{x} = (x_1, \ldots, x_n)^\top$ of a Gaussian process on a spatial grid $\mathcal{S} = \{\mathbf{s}_1, \ldots, \mathbf{s}_n\}$ in a spatial domain $\mathcal{D} \in \mathbb{R}^d$, in the sense that $x_i = x(\mathbf{s}_i)$ and $\mathbf{s}_i \in \mathcal{D}$, $i = 1, \ldots, n$. We assume the following model:

$$y_i \,|\, \mathbf{x} \overset{ind}{\sim} \mathcal{N}(x_i, \tau_i^2), \qquad i \in \mathcal{I} \tag{3.1}$$

$$\mathbf{x} \sim \mathcal{N}_n(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \tag{3.2}$$

where $\mathbf{y}$ is the data vector consisting of observations $\{y_i : i \in \mathcal{I}\}$, and $\mathcal{I} \subset \{1, \ldots, n\}$ contains the observation indices. Note that we can equivalently express (3.1) using matrix notation as $\mathbf{y} \,|\, \mathbf{x} \sim \mathcal{N}(\mathbf{H}\mathbf{x}, \mathbf{R})$, where $\mathbf{H}$ is obtained by selecting only the rows with indices $i \in \mathcal{I}$ from an identity matrix, and $\mathbf{R}$ is a diagonal matrix with entries $\{\tau_i^2 : i \in \mathcal{I}\}$.

Our interest is in computing the posterior distribution of $\mathbf{x}$ given $\mathbf{y}$, which requires inverting or decomposing an $n \times n$ matrix at a cost of $\mathcal{O}(n^3)$ if $|\mathcal{I}| = \mathcal{O}(n)$. This is computationally infeasible for large n.

3.3.1 The hierarchical Vecchia approximation

We now describe a hierarchical Vecchia approximation with unique sparsity and computational properties, which enable fast computation for spatial models as in (3.1)–(3.2) and also allow extensions to spatio-temporal filtering as explained later.

Assume that the elements of the vector $\mathbf{x}$ are hierarchically partitioned into a set $\mathcal{X}^{0:M} = \bigcup_{m=0}^{M} \mathcal{X}^m$, where $\mathcal{X}^m = \bigcup_{k=1}^{m} \bigcup_{j_k=1}^{J_k} \mathcal{X}_{j_1,\ldots,j_m}$, and $\mathcal{X}_{j_1,\ldots,j_m}$ is a set consisting of $|\mathcal{X}_{j_1,\ldots,j_m}|$ elements of $\mathbf{x}$, such that there is no overlap between any two sets $\mathcal{X}_{j_1,\ldots,j_m} \cap \mathcal{X}_{i_1,\ldots,i_l} = \emptyset$. We assume that $\mathbf{x}$ is ordered according to $\mathcal{X}^{0:M}$, in the sense that if $i > j$, then $x_i \in \mathcal{X}^{m_1}$ and $x_j \in \mathcal{X}^{m_2}$ with $m_1 \geq m_2$. As a toy example with $n = 6$, the vector $\mathbf{x} = (x_1, \ldots, x_6)$ might be partitioned

with $M = 1$, $J_1 = 2$ as $\mathcal{X}^{0:1} = \mathcal{X}^0 \cup \mathcal{X}^1$, $\mathcal{X}^0 = \mathcal{X} = \{x_1, x_2\}$, and $\mathcal{X}^1 = \mathcal{X}_{1,1} \cup \mathcal{X}_{1,2}$, where $\mathcal{X}_{1,1} = \{x_3, x_4\}$, and $\mathcal{X}_{1,2} = \{x_5, x_6\}$. Another toy example is illustrated in Figure 3.1.

The exact distribution of $\mathbf{x} \sim \mathcal{N}_n(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ can be written as

$$p(\mathbf{x}) = \prod_{m=0}^{M} \prod_{j_1,\ldots,j_m} p(\mathcal{X}_{j_1,\ldots,j_m} | \mathcal{X}^{0:m-1}, \mathcal{X}_{j_1,\ldots,j_{m-1},1:j_m-1}),$$

where the conditioning set of $\mathcal{X}_{j_1,\ldots,j_m}$ consists of all sets $\mathcal{X}^{0:m-1}$ at lower resolution, plus those at the same resolution that are previous in lexicographic ordering. The idea of Vecchia (1988) was to remove many of these variables in the conditioning set, which for geostatistical applications often incurs only small approximation error due to the so-called screening effect (e.g., Stein, 2002).

Here we consider a hierarchical Vecchia approximation of the form

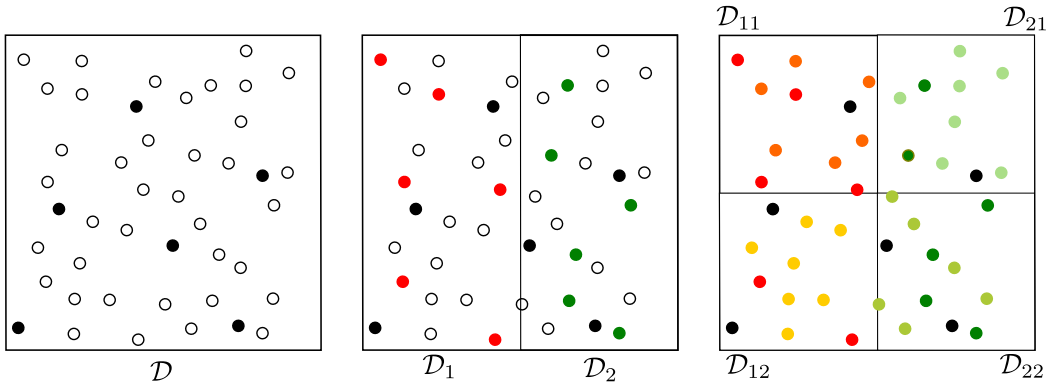
$$\hat{p}(\mathbf{x}) = \prod_{m=0}^{M} \prod_{j_1,\ldots,j_m} p(\mathcal{X}_{j_1,\ldots,j_m} | \mathcal{A}_{j_1,\ldots,j_m}), \tag{3.3}$$

where $\mathcal{A}_{j_1,\ldots,j_m} = \mathcal{X} \cup \mathcal{X}_{j_1} \cup \ldots \cup \mathcal{X}_{j_1,\ldots,j_{m-1}}$ is the set of ancestors of $\mathcal{X}_{j_1,\ldots,j_m}$. For example, the set of ancestors of $\mathcal{X}_{2,1,2}$ is $\mathcal{A}_{2,1,2} = \mathcal{X} \cup \mathcal{X}_2 \cup \mathcal{X}_{2,1}$. Thus, $\mathcal{A}_{j_1,\ldots,j_m} = \mathcal{A}_{j_1,\ldots,j_{m-1}} \cup \mathcal{X}_{j_1,\ldots,j_{m-1}}$, and the ancestor sets are nested: $\mathcal{A}_{j_1,\ldots,j_{m-1}} \subset \mathcal{A}_{j_1,\ldots,j_m}$. We can equivalently write (3.3) in terms of individual variables as

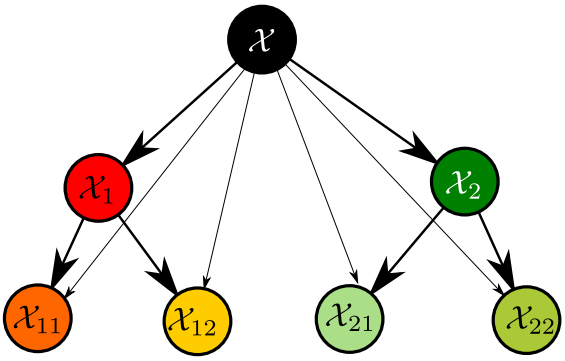
$$\hat{p}(\mathbf{x}) = \prod_{i=1}^{n} p(x_i | \mathcal{C}_i), \tag{3.4}$$

where $\mathcal{C}_i = \mathcal{A}_{j_1,\ldots,j_m} \cup \{x_k \in \mathcal{X}_{j_1,\ldots,j_m} : k < i\}$ for $x_i \in \mathcal{X}_{j_1,\ldots,j_m}$.

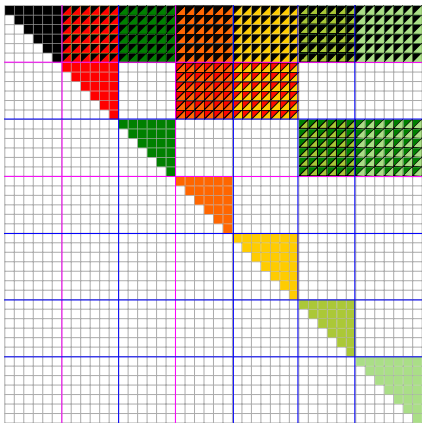
Vecchia approximations and their conditional-independence assumptions are closely connected to directed acyclic graphs (DAGs; Datta et al., 2016a; Katzfuss and Guinness, 2019). Summarizing briefly, as illustrated in Figure 3.1b, we associate a vertex with each set $\mathcal{X}_{j_1,\ldots,j_m}$, and we draw an arrow from the vertex corresponding to $\mathcal{X}_{i_1,\ldots,i_l}$ to the vertex corresponding to $\mathcal{X}_{j_1,\ldots,j_m}$ if and only if $\mathcal{X}_{i_1,\ldots,i_l}$ is in the conditioning set of $\mathcal{X}_{j_1,\ldots,j_m}$ (i.e., $\mathcal{X}_{i_1,\ldots,i_l} \subset \mathcal{A}_{j_1,\ldots,j_m}$). DAGs corresponding to hierarchical Vecchia approximations always have a tree structure, due to the nested ancestor sets. Necessary terminology and notation from graph theory is reviewed in Appendix B.1.



(a) Iterative domain partitioning for $n = 35$ locations



(b) Directed acyclic graph (DAG)



(c) Sparsity pattern of the $\mathbf{U}$ matrix

Figure 3.1: Toy example with $n = 35$ of the hierarchical Vecchia approximation in (3.3) with $M = 2$ and $J_1 = J_2 = 2$; the color of each set $\mathcal{X}_{j_1,\ldots,j_m}$ is consistent across (a)–(c). (a) For resolution $m = 0, 1, 2$, locations of $\mathcal{X}^{0:m}$ (•) and locations of points at higher resolution (○). (b) DAG illustrating the conditional-dependence structure, with thicker lines for connections between vertices at neighboring levels of the hierarchy, to emphasize the tree structure. (c) Corresponding sparsity pattern of $\mathbf{U}$ (see Proposition 8), with groups of columns/rows corresponding to different resolutions separated by pink lines, and groups of columns/rows corresponding to different $\mathcal{X}_{j_1,\ldots,j_m}$ at the same resolution separated by blue lines.

In practice, as illustrated in Figure 3.1a, we partition the spatial field $\mathbf{x}$ into the hierarchical set $\mathcal{X}^{0:M}$ based on a recursive partitioning of the spatial domain $\mathcal{D}$ into J_1 regions $\mathcal{D}_1, \ldots, \mathcal{D}_{J_1}$, each of which is again split into J_2 regions, and so forth, up to resolution M (Katzfuss, 2017): $\mathcal{D}_{j_1,\ldots,j_{m-1}} = \bigcup_{j_m=1}^{J_m} \mathcal{D}_{j_1,\ldots,j_m}$, $m = 1, \ldots, M$. We then set each $\mathcal{X}_{j_1,\ldots,j_m}$ to be a subset of the variables in $\mathbf{x}$ whose location is in $\mathcal{D}_{j_1,\ldots,j_m}$: $\mathcal{X}_{j_1,\ldots,j_m} \subset \{x_i : \mathbf{s}_i \in \mathcal{D}_{j_1,\ldots,j_m}\}$. This implies that the ancestors $\mathcal{A}_{j_1,\ldots,j_m}$ of each set $\mathcal{X}_{j_1,\ldots,j_m}$ consist of the variables associated with regions at lower resolutions $m = 0, \ldots, m-1$ that contain $\mathcal{D}_{j_1,\ldots,j_m}$.

The hierarchical Vecchia approximation (3.3) is closely related to the multi-resolution approximation (Katzfuss, 2017; Katzfuss and Gong, 2019), as noted in Katzfuss and Guinness (2019, Sec. 2.5), which in turn is closely related to hierarchical off-diagonal low-rank (HODLR) matrices (e.g. Hackbusch, 2015; Ambikasaran et al., 2016; Saibaba et al., 2015; Geoga et al., 2018), as noted in Jurek and Katzfuss (2018). However, the definition, exposition, and details provided here facilitate our later proofs, simple incomplete-Cholesky-based computation, and extensions to non-Gaussian data and to nonlinear space-time filtering.

3.3.2 Sparsity of the hierarchical Vecchia approximation

For all Vecchia approximations, the assumed conditional independence implies a sparse Cholesky factor of the precision matrix (e.g., Datta et al., 2016a; Katzfuss and Guinness, 2019, Prop. 3.3). The conditional-independence assumption made in our hierarchical Vecchia approximation also implies a sparse Cholesky factor of the covariance matrix, which is in contrast to many other formulations of the Vecchia approximation:

PROPOSITION 8. *For the hierarchical Vecchia approximation in* (3.3)*, we have* $\hat{p}(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \hat{\boldsymbol{\Sigma}})$. *Define* $\mathbf{L} = \mathrm{chol}(\hat{\boldsymbol{\Sigma}})$ *and* $\mathbf{U} = \mathrm{rchol}(\hat{\boldsymbol{\Sigma}}^{-1})$.

1. *For* $i \neq j$*:*
   - *(a)* $\mathbf{L}_{i,j} = 0$ *unless* $x_j \in \mathcal{C}_i$
   - *(b)* $\mathbf{U}_{j,i} = 0$ *unless* $x_j \in \mathcal{C}_i$
2. $\mathbf{U} = \mathbf{L}^{-\top}$

The proof relies on Claim 1. All proofs can be found in Appendix B.2. Proposition 8 says that the Cholesky factors of the covariance and precision matrix implied by a hierarchical Vecchia approximation are both sparse, and $\mathbf{U}$ has the same sparsity pattern as $\mathbf{L}^\top$. An example of this pattern is shown in Figure 3.1c. Furthermore, because $\mathbf{L} = \mathbf{U}^{-\top}$, we can quickly compute one of these factors given the other, as described in Section 3.3.3 below.

For other Vecchia approximations, the sparsity of the prior Cholesky factor $\mathbf{U}$ does not necessarily imply the same sparsity for the Cholesky factor of the posterior precision matrix, and in fact there can be substantial in-fill (Katzfuss and Guinness, 2019). However, this is not the case for the particular case of hierarchical Vecchia, for which the posterior sparsity is exactly the same as the prior sparsity:

PROPOSITION 9. *Assume that* $\mathbf{x}$ *has the distribution* $\hat{p}(\mathbf{x})$ *given by the hierarchical Vecchia approximation in* (3.3). *Let* $\widetilde{\boldsymbol{\Sigma}} = \mathrm{Var}(\mathbf{x}|\mathbf{y})$ *be the posterior covariance matrix of* $\mathbf{x}$ *given data* $y_i \,|\, \mathbf{x} \overset{ind}{\sim} \mathcal{N}(x_i, \tau_i^2)$, $i \in \mathcal{I} \subset \{1, \ldots, n\}$ *as in* (3.1). *Then:*

1. $\widetilde{\mathbf{U}} = \mathrm{rchol}(\widetilde{\boldsymbol{\Sigma}}^{-1})$ *has the same sparsity pattern as* $\mathbf{U} = \mathrm{rchol}(\hat{\boldsymbol{\Sigma}}^{-1})$.

2. $\widetilde{\mathbf{L}} = \mathrm{chol}(\widetilde{\boldsymbol{\Sigma}})$ *has the same sparsity pattern as* $\mathbf{L} = \mathrm{chol}(\hat{\boldsymbol{\Sigma}})$.

3.3.3 Fast computation using incomplete Cholesky factorization

For notational and computational convenience, we assume now that each conditioning set $\mathcal{C}_i$ consists of at most N elements of $\mathbf{x}$. For example, this can be achieved by setting $|\mathcal{X}_{j_1,\ldots,j_m}| \leq r$ with $r = N/(M+1)$. The matrix $\mathbf{U}$ can be computed using general expressions for the Vecchia approximation in $\mathcal{O}(nN^3)$ time (e.g., Katzfuss and Guinness, 2019). Alternatively, inference can be carried out using multi-resolution decompositions (Katzfuss, 2017; Katzfuss and Gong, 2019; Jurek and Katzfuss, 2018) in $\mathcal{O}(nN^2)$, but these algorithms are fairly involved.

Instead, we show here how hierarchical Vecchia inference can be carried out in $\mathcal{O}(nN^2)$ time using standard sparse-matrix algorithms, including the incomplete Cholesky factorization, based on at most nN entries of $\boldsymbol{\Sigma}$. Our algorithm, which is based on ideas in Schäfer et al. (2017), is much simpler than multi-resolution decompositions.

The incomplete Cholesky factorization (e.g., Golub and Van Loan, 2012), denoted by $\mathrm{ichol}(\mathbf{A}, \mathbf{S})$ and given in Algorithm 1, is identical to the standard Cholesky factorization of the matrix $\mathbf{A}$, except that we skip all operations that involve elements that are not in the sparsity pattern represented by the zero-one matrix $\mathbf{S}$. It is important to note that to compute $\mathbf{L} = \mathrm{ichol}(\mathbf{A}, \mathbf{S})$ for a large dense

Algorithm 1: Incomplete Cholesky decomposition: $\text{ichol}(\mathbf{A}, \mathbf{S})$

Input: positive-definite matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$, sparsity matrix $\mathbf{S} \in \{0,1\}^{n \times n}$
Result: lower-triangular $n \times n$ matrix $\mathbf{L}$

for $i = 1$ to n **do**
  for $j = 1$ to $i - 1$ **do**
    if $\mathbf{S}_{i,j} = 1$ **then**
      $\mathbf{L}_{i,j} = \frac{1}{\mathbf{L}_{j,j}} \left(\mathbf{A}_{i,j} - \sum_{k=1}^{j-1} \mathbf{L}_{i,k} \mathbf{L}_{j,k} \right)$
    end if
  end for
  $\mathbf{L}_{i,i} = \sqrt{\mathbf{A}_{i,i} - \sum_{k=1}^{i-1} \mathbf{L}_{k,k}}$
end for

matrix $\mathbf{A}$, we do not actually need to form the entire $\mathbf{A}$; instead, to reduce memory usage and computational cost, we simply compute $\mathbf{L} = \text{ichol}(\mathbf{A} \circ \mathbf{S}, \mathbf{S})$ based on the sparse matrix $\mathbf{A} \circ \mathbf{S}$, where $\circ$ denotes element-wise multiplication. Thus, while we write expressions like $\mathbf{L} = \text{ichol}(\mathbf{A}, \mathbf{S})$ for notational simplicity below, this should always be read as $\mathbf{L} = \text{ichol}(\mathbf{A} \circ \mathbf{S}, \mathbf{S})$.

For our hierarchical Vecchia approximation in (3.3), we set $\mathbf{S}$ to be a sparse lower-triangular matrix with $\mathbf{S}_{i,j} = 1$ if $x_j \in \mathcal{C}_i$, and 0 otherwise. Thus, the sparsity pattern of $\mathbf{S}$ is the same as that of $\mathbf{L}$, and its transpose is that of $\mathbf{U}$ shown in Figure 3.1c.

PROPOSITION 10. *Assuming* (3.3)*, denote* $\text{Var}(\mathbf{x}) = \hat{\mathbf{\Sigma}}$ *and* $\mathbf{L} = \text{chol}(\hat{\mathbf{\Sigma}})$*. Then,* $\mathbf{L} = \textit{ichol}(\mathbf{\Sigma}, \mathbf{S})$.

Hence, the Cholesky factor of the covariance matrix implied by the hierarchical Vecchia approximation can be computed using the incomplete Cholesky algorithm based on the (at most) nN entries of the exact covariance $\mathbf{\Sigma}$ indicated by $\mathbf{S}$. Using this result, we propose Algorithm 2 for posterior inference on $\mathbf{x}$ given $\mathbf{y}$.

The combination of the incomplete Cholesky factorization and the results in Propositions 8 and 9 that all Cholesky factors are sparse, allow us to perform posterior inference very quickly.

PROPOSITION 11. *Algorithm 2 can be carried out in* $\mathcal{O}(nN^2)$ *time and* $\mathcal{O}(nN)$ *space, assuming that* $|\mathcal{C}_i| \leq N$ *for all* $i = 1, \ldots, n$.

Algorithm 2: Posterior inference using hierarchical Vecchia: $\texttt{HV}(\mathbf{y}, \mathbf{S}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{H}, \mathbf{R})$

Input: data $\mathbf{y}$; sparsity $\mathbf{S}$; $\boldsymbol{\mu}, \boldsymbol{\Sigma}$ s.t. $\mathbf{x} \sim \mathcal{N}_n(\boldsymbol{\mu}, \boldsymbol{\Sigma})$; obs. matrix $\mathbf{H}$, noise variances $\mathbf{R}$
Result: $\widetilde{\boldsymbol{\mu}}$ and $\widetilde{\mathbf{L}}$ such that $\hat{p}(\mathbf{x}|\mathbf{y}) = \mathcal{N}_n(\mathbf{x}|\widetilde{\boldsymbol{\mu}}, \widetilde{\mathbf{L}}\widetilde{\mathbf{L}}^\top)$

1: $\mathbf{L} = \text{ichol}(\boldsymbol{\Sigma}, \mathbf{S})$, using Algorithm 1
2: $\mathbf{U} = \mathbf{L}^{-\top}$
3: $\boldsymbol{\Lambda} = \mathbf{U}\mathbf{U}^\top + \mathbf{H}^\top\mathbf{R}^{-1}\mathbf{H}$
4: $\widetilde{\mathbf{U}} = \mathbf{P}\left(\text{chol}(\mathbf{P}\boldsymbol{\Lambda}\mathbf{P})\right)\mathbf{P}$, where $\mathbf{P}$ is the order-reversing permutation matrix
5: $\widetilde{\mathbf{L}} = \widetilde{\mathbf{U}}^{-\top}$
6: $\widetilde{\boldsymbol{\mu}} = \boldsymbol{\mu} + \widetilde{\mathbf{L}}\widetilde{\mathbf{L}}^\top\mathbf{H}^\top\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{H}\boldsymbol{\mu}\right)$

3.4 Extensions to non-Gaussian spatial data using the Laplace approximation

Now consider the model

$$y_i \,|\, \mathbf{x} \overset{ind}{\sim} g_i(y_i|x_i), \qquad i \in \mathcal{I}, \tag{3.5}$$

$$\mathbf{x} \sim \mathcal{N}_n(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \tag{3.6}$$

where g_i is a distribution from an exponential family. Using the hierarchical Vecchia approximation in (3.3)–(3.4) for $\mathbf{x}$, the implied posterior can be written as:

$$\hat{p}(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x})\hat{p}(\mathbf{x})}{\int p(\mathbf{y}|\mathbf{x})\hat{p}(\mathbf{x})d\mathbf{x}} = \frac{(\prod_{i\in\mathcal{I}} g_i(y_i|x_i))\hat{p}(\mathbf{x})}{\int (\prod_{i\in\mathcal{I}} g_i(y_i|x_i))\hat{p}(\mathbf{x})d\mathbf{x}}. \tag{3.7}$$

Unlike in the Gaussian case as in (3.1), the integral in the denominator cannot generally be evaluated in closed form, and Markov Chain Monte Carlo methods are often used to numerically approximate the posterior. Instead, Zilber and Katzfuss (2019) proposed a much faster method that combines a general Vecchia approximation with the Laplace approximation (e.g. Tierney and Kadane, 1986; Rasmussen and Williams, 2006, Sect. 3.4). The Laplace approximation is equivalent to a Gaussian approximation of the posterior, obtained by carrying out a second-order Taylor expansion of the posterior log-density around its mode. Although the mode cannot generally be obtained in closed form, it can be computed straightforwardly using a Newton-Raphson procedure,

because $\log \hat{p}(\mathbf{x}|\mathbf{y}) = \log p(\mathbf{y}|\mathbf{x}) + \log \hat{p}(\mathbf{x}) + c$ is a sum of two concave functions and hence also concave (as a function of $\mathbf{x}$, under appropriate parametrization of the g_i).

While each Newton-Raphson update requires the computation and decomposition of the $n \times n$ Hessian matrix, the update can be carried out quickly by making use of the sparsity implied by the Vecchia approximation. To do so, we follow Zilber and Katzfuss (2019) in exploiting the fact that the Newton-Raphson update is equivalent to computing the conditional mean of $\mathbf{x}$ given pseudo-data. Specifically, at the ℓ-th iteration of the algorithm, given the current state value $\mathbf{x}^{(\ell)}$, let us define

$$\mathbf{u}^{(\ell)} = \left[u_i^{(\ell)}\right]_{i \in \mathcal{I}}, \qquad \text{where} \quad u_i^{(\ell)} = \tfrac{\partial}{\partial x} \log g_i(y_i|x)\big|_{x=x_i^{(\ell)}}, \tag{3.8}$$

and

$$\mathbf{D}^{(\ell)} = \operatorname{diag}\left(\{d_i^{(\ell)} : i \in \mathcal{I}\}\right), \qquad \text{where} \quad d_i^{(\ell)} = -\left(\tfrac{\partial^2}{\partial x^2} \log g_i(y_i|x)\right)^{-1}\big|_{x=x_i^{(\ell)}}. \tag{3.9}$$

Then, we compute the next iteration's state value $\mathbf{x}^{(\ell+1)} = \mathbb{E}(\mathbf{x}|\mathbf{t}^{(\ell)})$ as the conditional mean of $\mathbf{x}$ given pseudo-data $\mathbf{t}^{(\ell)} = \mathbf{x}^{(\ell)} + \mathbf{D}^{(\ell)}\mathbf{u}^{(\ell)}$ assuming Gaussian noise, $t_i^{(\ell)}|\mathbf{x} \overset{ind.}{\sim} \mathcal{N}(x_i, d_i^{(\ell)})$, $i \in \mathcal{I}$. Zilber and Katzfuss (2019) recommend computing the conditional mean $\mathbb{E}(\mathbf{x}|\mathbf{t}^{(\ell)})$ based on a general-Vecchia-prediction approach proposed in Katzfuss et al. (2018). Here, we instead compute the posterior mean using Algorithm 2 based on the hierarchical Vecchia method described in Section 3.3, due to its sparsity-preserving properties. In contrast to the approach recommended in Zilber and Katzfuss (2019), our algorithm is guaranteed to converge, because it is equivalent to Newton-Raphson optimization of the log of the posterior density in (3.7), which is concave. Once the algorithm converges to the posterior mode $\widetilde{\boldsymbol{\mu}}$, the HV-Laplace approximation of the posterior is given by

$$\hat{p}_L(\mathbf{x}|\mathbf{y}) = \mathcal{N}_n(\mathbf{x}|\widetilde{\boldsymbol{\mu}}, \widetilde{\mathbf{L}}\widetilde{\mathbf{L}}^\top),$$

where $\widetilde{\mathbf{L}}$ is the Cholesky factor of the negative Hessian of the log-posterior at $\widetilde{\boldsymbol{\mu}}$. Our approach is described in Algorithm 3. The main computational expense for each iteration of the for loop is carrying out Algorithm 2, and so each iteration requires only $\mathcal{O}(nN^2)$ time.

Algorithm 3: Hierarchical-Vecchia-Laplace inference: $\texttt{HVL}(\mathbf{y}, \mathbf{S}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \{g_i\})$

Input: data $\mathbf{y}$; sparsity $\mathbf{S}$; $\boldsymbol{\mu}, \boldsymbol{\Sigma}$ such that $\mathbf{x} \sim \mathcal{N}_n(\boldsymbol{\mu}, \boldsymbol{\Sigma})$; likelihoods $\{g_i : i \in \mathcal{I}\}$
Result: $\widetilde{\boldsymbol{\mu}}$ and $\widetilde{\mathbf{L}}$ such that $\hat{p}_L(\mathbf{x}|\mathbf{y}) = \mathcal{N}_n(\mathbf{x}|\widetilde{\boldsymbol{\mu}}, \widetilde{\mathbf{L}}\widetilde{\mathbf{L}}^\top)$
Initialize $\mathbf{x}^{(0)} = \boldsymbol{\mu}$
Set $\mathbf{H} = \mathbf{I}_{\mathcal{I},:}$ as the rows $\mathcal{I}$ of the $n \times n$ identity matrix $\mathbf{I}$
for $\ell = 0, 1, 2, \ldots$ **do**
  Calculate $\mathbf{u}^{(\ell)}$ as in (3.8), $\mathbf{D}^{(\ell)}$ as in (3.9), and $\mathbf{t}^{(\ell)} = \mathbf{x}^{(\ell)} + \mathbf{D}^{(\ell)}\mathbf{u}^{(\ell)}$
  Calculate $[\mathbf{x}^{(\ell+1)}, \widetilde{\mathbf{L}}] = \texttt{HV}(\mathbf{t}^{(\ell)}, \mathbf{S}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{H}, \mathbf{D}^{(\ell)})$ using Algorithm 2
  if $\|\mathbf{x}^{(\ell+1)} - \mathbf{x}^{(\ell)}\|/\|\mathbf{x}^{(\ell)}\| < \epsilon$ **then**
    break
  end if
end for
return $\widetilde{\boldsymbol{\mu}} = \mathbf{x}^{(\ell+1)}$ and $\widetilde{\mathbf{L}}$

It can be shown using straightforward calculations that in the Gaussian case, when $g_i(y_i|x_i) = \mathcal{N}(y_i|a_i x_i, \tau_i^2)$ for some $a_i \in \mathbb{R}$, the pseudo-data $t_i = y_i/a_i$ and pseudo-variances $d_i = \tau_i^2$ do not depend on $\mathbf{x}$, and so Algorithm 3 converges in a single iteration (cf. Zilber and Katzfuss, 2019). If $a_i = 1$, (3.5) becomes equivalent to (3.1), and Algorithm 3 simplifies to Algorithm 2.

3.5 Fast filters for spatio-temporal models

3.5.1 Linear evolution

We now turn to a spatio-temporal state-space model, which adds a temporal evolution model to the spatial model (3.5) considered in Section 3.4. For now, assume that the evolution is linear. Starting with an initial distribution $\mathbf{x}_0 \sim \mathcal{N}_n(\boldsymbol{\mu}_{0|0}, \boldsymbol{\Sigma}_{0|0})$, we consider the following SSM for discrete time $t = 1, 2, \ldots$:

$$y_{ti} \,|\, \mathbf{x}_t \overset{ind}{\sim} g_{ti}(y_{ti}|x_{ti}), \qquad i \in \mathcal{I}_t \tag{3.10}$$

$$\mathbf{x}_t \,|\, \mathbf{x}_{t-1} \sim \mathcal{N}_n(\mathbf{E}_t \mathbf{x}_{t-1}, \mathbf{Q}_t), \tag{3.11}$$

where $\mathbf{y}_t$ is the data vector consisting of observations $\{y_{ti} : i \in \mathcal{I}_t\}$, $\mathcal{I}_t \subset \{1, \ldots, n\}$ contains the $n_t \leq n$ observation indices at time t, g_{ti} is a distribution from the exponential family, $\mathbf{x}_t = (x_1, \ldots, x_n)^\top$ is the latent spatial field of interest at time t observed at a spatial grid $\mathcal{S}$, and $\mathbf{E}_t$ is a

sparse $n \times n$ evolution matrix.

At time t, our goal is to obtain or approximate the filtering distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ of $\mathbf{x}_t$ given data $\mathbf{y}_{1:t}$ up to the current time t. This task, also referred to as data assimilation or on-line inference, is a very common task across many fields of science, to quantify uncertainty in the state and to obtain forecasts into the future. If the observation equations g_{ti} are all Gaussian, the filtering distribution can be derived using the Kalman filter (Kalman, 1960) for small to moderate n. At each time t, the Kalman filter consist of a forecast step that computes $p(\mathbf{x}_t|\mathbf{y}_{1:t-1})$, and an update step which then obtains $p(\mathbf{x}_t|\mathbf{y}_{1:t})$. For linear Gaussian SSMs, both of these distributions are multivariate normal.

Our Kalman-Vecchia-Laplace (KVL) filter extends the Kalman filter to high-dimensional SSMs (i.e., large n) with non-Gaussian data, as in (3.10)–(3.11). Its update step is very similar to the inference problem in Section 3.4, and hence it essentially consists of the HVL in Algorithm 3. We complement this update with a forecast step, in which the estimates of the mean and variance are propagated forward using the temporal evolution model. This results in our KVL filter given in Algorithm 4.

Algorithm 4: Kalman-Vecchia-Laplace (KVL) filter

Input: $\mathbf{S}$, $\boldsymbol{\mu}_{0|0}$, $\boldsymbol{\Sigma}_{0|0}$, $\{(\mathbf{y}_t, \mathbf{E}_t, \mathbf{Q}_t, \{g_{t,i}\}) : t = 1, 2, \ldots\}$
Result: $\boldsymbol{\mu}_{t|t}, \mathbf{L}_{t|t}$, such that $\hat{p}(\mathbf{x}_t|\mathbf{y}_{1:t}) = \mathcal{N}_n(\mathbf{x}_t|\boldsymbol{\mu}_{t|t}, \mathbf{L}_{t|t}\mathbf{L}_{t|t}^\top)$
1: Compute $\mathbf{U}_{0|0} = \text{ichol}(\boldsymbol{\Sigma}_{0|0}, \mathbf{S})$ and $\mathbf{L}_{0|0} = \mathbf{U}_{0|0}^{-\top}$
2: **for** $t = 1, 2, \ldots$ **do**
3: Forecast: $\boldsymbol{\mu}_{t|t-1} = \mathbf{E}_t\boldsymbol{\mu}_{t-1|t-1}$ and $\mathbf{L}_{t|t-1} = \mathbf{E}_t\mathbf{L}_{t-1|t-1}$
4: For all (i, j) with $\mathbf{S}_{i,j} = 1$: $\boldsymbol{\Sigma}_{t|t-1;i,j} = \mathbf{L}_{t|t-1;i,:}\mathbf{L}_{t|t-1;j,:}^\top + \mathbf{Q}_{t;i,j}$
5: Update: $[\boldsymbol{\mu}_t, \mathbf{L}_{t|t}] = \texttt{HVL}(\mathbf{y}_t, \mathbf{S}, \boldsymbol{\mu}_{t|t-1}, \boldsymbol{\Sigma}_{t|t-1}, \{g_{t,i}\})$ using Algorithm 3
6: **return** $\boldsymbol{\mu}_{t|t}, \mathbf{L}_{t|t}$
7: **end for**

In Line 4, $\mathbf{L}_{t|t-1;i,:}$ denotes the ith row of $\mathbf{L}_{t|t-1}$. The KVL filter scales well with the state dimension n. The evolution matrix $\mathbf{E}_t$, which is often derived using a forward-finite-difference

scheme and thus has only a few nonzero elements in each row, can be quickly multiplied with $\mathbf{L}_{t-1|t-1}$ in Line 3, as the latter is sparse (see Section 3.3.3). The $\mathcal{O}(nN)$ necessary entries of $\mathbf{\Sigma}_{t|t-1}$ in Line 4 can also be calculated quickly due to the sparsity of $\mathbf{L}_{t|t-1;i,:}$. The low computational cost of the `HVL` algorithm has already been discussed in Section 3.4. Thus, assuming sufficiently sparse $\mathbf{E}_t$, the KVL filter scales approximately as $\mathcal{O}(nN^2)$ per iteration. In the case of Gaussian data (i.e., all g_{ti} in (3.10) are Gaussian), our KVL filter will produce essentially equivalent filtering distributions as the multi-resolution filter of Jurek and Katzfuss (2018).

3.5.2 An extended MRF for nonlinear evolution

Finally, we consider a nonlinear and non-Gaussian model, which extends (3.10)–(3.11) by allowing nonlinear evolution operators, $\mathcal{E}_t : \mathbb{R}^n \to \mathbb{R}^n$. This results in the model

$$y_{ti} \,|\, \mathbf{x}_t \overset{ind}{\sim} g_{ti}(y_{ti}|x_{ti}), \qquad i \in \mathcal{I}_t \tag{3.12}$$

$$\mathbf{x}_t \,|\, \mathbf{x}_{t-1} \sim \mathcal{N}_n(\mathcal{E}_t(\mathbf{x}_{t-1}), \mathbf{Q}_t). \tag{3.13}$$

Due to the nonlinearity of the evolution operator $\mathcal{E}_t$, the KVL filter in Algorithm 4 is not directly applicable anymore. However, similar inference is still possible as long as the evolution is not too far from linear. Approximating the evolution as linear is generally reasonable if the time steps are short, or if the measurements are highly informative. In this case, we propose the extended Kalman-Vecchia-Laplace filter (EKVL) in Algorithm 5, which approximates the extended Kalman filter (e.g., Grewal and Andrews, 1993, Ch. 5) and extends it to non-Gaussian data using the Vecchia-Laplace aproach. For the forecast step, EKVL computes the forecast mean as $\boldsymbol{\mu}_{t|t-1} = \mathcal{E}_t(\boldsymbol{\mu}_{t-1|t-1})$. The forecast covariance matrix $\mathbf{\Sigma}_{t|t-1}$ is obtained as before, after approximating the evolution using the Jacobian as $\mathbf{E}_t = \frac{\partial \mathcal{E}_t(\mathbf{y}_{t-1})}{\partial \mathbf{y}_{t-1}}\big|_{\mathbf{y}_{t-1}=\boldsymbol{\mu}_{t-1|t-1}}$. Errors in the forecast covariance matrix due to this linear approximation can be captured in the innovation covariance, $\mathbf{Q}_t$. If the Jacobian matrix cannot be computed, it is sometimes possible to build a statistical emulator (e.g. Kaufman et al., 2011) instead, which approximates the true evolution operator.

Once $\boldsymbol{\mu}_{t|t-1}$ and $\mathbf{\Sigma}_{t|t-1}$ have been obtained, the update step of the EKVL proceeds exactly as

in the KVL filter by approximating the forecast distribution as Gaussian.

Algorithm 5: Extended Kalman-Vecchia-Laplace (EKVL) filter

Input: $\mathbf{S}$, $\boldsymbol{\mu}_{0|0}$, $\boldsymbol{\Sigma}_{0|0}$, $\{(\mathbf{y}_t, \mathcal{E}_t, \mathbf{Q}_t, \{g_{t,i}\}) : t = 1, 2, \ldots\}$
Result: $\boldsymbol{\mu}_{t|t}, \mathbf{L}_{t|t}$, such that $\hat{p}(\mathbf{x}_t|\mathbf{y}_{1:t}) = \mathcal{N}_n(\mathbf{x}_t|\boldsymbol{\mu}_{t|t}, \mathbf{L}_{t|t}\mathbf{L}_{t|t}^\top)$
1: Compute $\mathbf{U}_{0|0} = \text{ichol}(\boldsymbol{\Sigma}_{0|0}, \mathbf{S})$ and $\mathbf{L}_{0|0} = \mathbf{U}_{0|0}^{-\top}$
2: **for** $t = 1, 2, \ldots$ **do**
3: Calculate $\mathbf{E}_t = \frac{\partial \mathcal{E}_t(\mathbf{x}_{t-1})}{\partial \mathbf{x}_{t-1}}\big|_{\mathbf{x}_{t-1}=\boldsymbol{\mu}_{t-1|t-1}}$
4: Forecast: $\boldsymbol{\mu}_{t|t-1} = \mathcal{E}_t(\boldsymbol{\mu}_{t-1|t-1})$ and $\mathbf{L}_{t|t-1} = \mathbf{E}_t\mathbf{L}_{t-1|t-1}$
5: For all (i, j) with $\mathbf{S}_{i,j} = 1$: $\boldsymbol{\Sigma}_{t|t-1;i,j} = \mathbf{L}_{t|t-1;i,:}\mathbf{L}_{t|t-1;j,:}^\top + \mathbf{Q}_{t;i,j}$
6: Update: $[\boldsymbol{\mu}_t, \mathbf{L}_{t|t}] = \texttt{HVL}(\mathbf{y}_t, \mathbf{S}, \boldsymbol{\mu}_{t|t-1}, \boldsymbol{\Sigma}_{t|t-1}, \{g_{t,i}\})$ using Algorithm 3
7: **return** $\boldsymbol{\mu}_{t|t}, \mathbf{L}_{t|t}$
8: **end for**

Similarly to Algorithm 4, EKVL scales very well with the dimension of $\mathbf{x}$, the only difference being the additional operation of calculating the Jacobian in Line 3, whose cost is problem dependent. Only those entries of $\mathbf{E}_t$ need to be calculated that are multiplied with non-zero entries of $\mathbf{L}_{t-1|t-1}$, whose sparsity structure is known ahead of time.

3.5.3 A particle-EVKL filter in case of unknown parameters

The distributions and matrices in model (3.12)–(3.13) may depend on parameters $\boldsymbol{\theta}_t$ at each time t, which we have implicitly assumed to be known thus far. We now discuss the case of a (small) number of unknown parameters $\boldsymbol{\theta}_t$. Specifically, $\boldsymbol{\mu}_{0|0}$ and $\boldsymbol{\Sigma}_{0|0}$ may depend on $\boldsymbol{\theta}_0$, and the quantities $\{g_{t,i}\}$, $\mathcal{E}_t$, and $\mathbf{Q}_t$ at each time t may depend on $\boldsymbol{\theta}_t$. There are two main approaches to simultaneous filtering for the state $\mathbf{x}_t$ and the parameters $\boldsymbol{\theta}_t$: state augmentation and Rao-Blackwellized filters (Doucet and Johansen, 2009). The main idea behind the former is to include $\boldsymbol{\theta}_t$ in the state vector $\mathbf{x}_t$ and to modify the evolution and the model error matrices accordingly, but this approach is known to work poorly in certain cases (e.g., DelSole and Yang, 2010; Katzfuss et al., 2019). Thus, following Jurek and Katzfuss (2018), we now present a Rao-Blackwellized filter in which integration over $\mathbf{x}_t$ is performed based on our HVL approximation.

Writing $\boldsymbol{\theta}_{0:t} = (\boldsymbol{\theta}_0, \ldots, \boldsymbol{\theta}_t)$, the integrated likelihood at time t is given by

$$p(\mathbf{y}_{1:t}|\boldsymbol{\theta}_{0:t}) = p(\mathbf{y}_1|\boldsymbol{\theta}_{0:1}) \prod_{k=2}^{t} p(\mathbf{y}_k|\mathbf{y}_{1:k-1}, \boldsymbol{\theta}_{0:k}).$$

It is well known that

$$p(\mathbf{y}_t|\mathbf{y}_{1:t-1}, \boldsymbol{\theta}_{0:t}) = \frac{p(\mathbf{y}_t, \mathbf{x}_t|\mathbf{y}_{1:t-1}, \boldsymbol{\theta}_{0:t})}{p(\mathbf{x}_t|\mathbf{y}_{1:t}, \boldsymbol{\theta}_{0:t})} = \frac{p(\mathbf{y}_t|\mathbf{x}_t, \boldsymbol{\theta}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1}, \boldsymbol{\theta}_{0:t})}{p(\mathbf{x}_t|\mathbf{y}_{1:t}, \boldsymbol{\theta}_{0:t})},$$

where $p(\mathbf{y}_t|\mathbf{x}_t, \boldsymbol{\theta}_t)$ is available in closed form from (3.12), and the forecast and filtering distributions can be approximated using the EKVL, to obtain

$$\mathcal{L}_t(\boldsymbol{\theta}_{0:t}) := \hat{p}(\mathbf{y}_t|\mathbf{y}_{1:t-1}, \boldsymbol{\theta}_{0:t}) = \frac{p(\mathbf{y}_t|\mathbf{x}_t, \boldsymbol{\theta}_t)\mathcal{N}(\boldsymbol{\mu}_{t|t-1}, \boldsymbol{\Sigma}_{t|t-1})}{\mathcal{N}(\boldsymbol{\mu}_{t|t}, \boldsymbol{\Sigma}_{t|t})}. \tag{3.14}$$

The normal densities can be quickly evaluated for given parameter values $\boldsymbol{\theta}_{0:t}$, because Algorithm 5 calculates sparse Cholesky factors of their precision matrices. For $t = 1$, the term $\mathcal{L}_1(\boldsymbol{\theta}_{0:1}) := \hat{p}(\mathbf{y}_1|\boldsymbol{\theta}_{0:1})$ can be approximated in a similar way using $\boldsymbol{\mu}_{0|0}$ and $\boldsymbol{\Sigma}_{0|0}$.

The particle-EVKL filter is given by Algorithm 6, assuming that the parameter priors are given by $f_0(\boldsymbol{\theta}_0)$ and then recursively by $f_t(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{t-1})$.

3.6 Numerical comparison

3.6.1 Methods and criteria

We considered and compared the following methods:

Hierarchical Vecchia (HV): Our methods as described in this paper.

Low rank (LR): A special case of out methods with $M = 1$, in which the diagonal and the first N columns of $\mathbf{S}$ are nonzero, and all other entries are zero. This results in a matrix approximation $\hat{\boldsymbol{\Sigma}}$ that is of rank N plus diagonal, which is known as the modified predictive process (Banerjee et al., 2008; Finley et al., 2009) in spatial statistics. LR has the same computational complexity as HV.

Algorithm 6: Particle-EKVL filter

Input: $\mathbf{S}$, $\boldsymbol{\mu}_{0|0}$, $\boldsymbol{\Sigma}_{0|0}$, $\{(\mathbf{y}_t, \mathcal{E}_t, \mathbf{Q}_t, \{g_{t,i}\}) : t = 1, 2, \ldots\}$, priors $\{f_t\}$, proposal distributions $\{q_t\}$, desired number of particles N_p

Result: $\{(\boldsymbol{\theta}_t^{(l)}, w_t^{(l)}, \boldsymbol{\mu}_{t|t}^{(l)}, \mathbf{L}_{t|t}^{(l)}) : l = 1, \ldots, N_p\}$, such that $\hat{p}(\boldsymbol{\theta}_t, \mathbf{x}_t|\mathbf{y}_{1:t}) = \sum_{l=1}^{N_p} w_t^{(l)} \delta_{\boldsymbol{\theta}_t^{(l)}}(\boldsymbol{\theta}_t) \mathcal{N}_n(\mathbf{x}_t|\boldsymbol{\mu}_{t|t}^{(l)}, \mathbf{L}_{t|t}^{(l)} \mathbf{L}_{t|t}^{(l)\top})$

for $l = 1, 2, \ldots, N_p$ **do**
  Draw $\boldsymbol{\theta}_0^{(l)} \sim f_0(\boldsymbol{\theta}_0)$ and set weight $w_0^{(l)} = 1/N_p$
  Compute $\mathbf{L}_{0|0}(\boldsymbol{\theta}_0^{(l)}) = \text{ichol}(\boldsymbol{\Sigma}_{0|0}(\boldsymbol{\theta}_0^{(l)}), \mathbf{S})$
  Compute $\boldsymbol{\mu}_{0|0}^{(l)}(\boldsymbol{\theta}_0^{(l)})$ and $\mathbf{U}_{0|0}(\boldsymbol{\theta}_0^{(l)}) = \mathbf{L}_{0|0}^{-\top}(\boldsymbol{\theta}_0^{(l)})$
end for
for $t = 1, 2, \ldots$ **do**
  for $l = 1, 2, \ldots, N_p$ **do**
    Draw $\boldsymbol{\theta}_t^{(l)} \sim q_t(\boldsymbol{\theta}_t^{(l)}|\boldsymbol{\theta}_{t-1}^{(l)})$
    Calculate $\mathbf{E}_t^{(l)} = \frac{\partial \mathcal{E}_t(\mathbf{y}_{t-1}, \boldsymbol{\theta}_t^{(l)})}{\partial \mathbf{y}_{t-1}}\Big|_{\mathbf{y}_{t-1} = \boldsymbol{\mu}_{t-1|t-1}(\boldsymbol{\theta}_{t-1}^{(l)})}$
    Forecast: $\boldsymbol{\mu}_{t|t-1}^{(l)} = \mathcal{E}_t(\boldsymbol{\mu}_{t-1|t-1}, \boldsymbol{\theta}_{t-1}^{(l)})$ and $\mathbf{L}_{t|t-1}^{(l)} = \mathbf{E}_t^{(l)} \mathbf{L}_{t-1|t-1}^{(l)}$
    For (i, j) s.t. $\mathbf{S}_{i,j} = 1$: $\boldsymbol{\Sigma}_{t|t-1;i,j}^{(l)} = \mathbf{L}_{t|t-1;i,:}^{(l)} (\mathbf{L}_{t|t-1;j,:}^{(l)})^\top + \mathbf{Q}_{t;i,j}(\boldsymbol{\theta}_t^{(l)})$
    Update: $[\boldsymbol{\mu}_t^{(l)}, \mathbf{L}_{t|t}^{(l)}] = \texttt{HVL}(\mathbf{y}_t, \mathbf{S}, \boldsymbol{\mu}_{t|t-1}^{(l)}, \boldsymbol{\Sigma}_{t|t-1}^{(l)}, \{g_{t,i}(\boldsymbol{\theta}_t^{(l)})\})$
    Calculate $\mathcal{L}_t(\boldsymbol{\theta}_{0:t}^{(l)})$ as in (3.14)
    Update particle weight $w_t^{(l)} \propto w_{t-1}^{(l)} \mathcal{L}_t(\boldsymbol{\theta}_{0:t}^{(l)}) f_t(\boldsymbol{\theta}_t^{(l)}|\boldsymbol{\theta}_{t-1}^{(l)}) / q_t(\boldsymbol{\theta}_t^{(l)}|\boldsymbol{\theta}_{t-1}^{(l)})$
    return $\boldsymbol{\mu}_{t|t}^{(l)}, \mathbf{L}_{t|t}^{(l)}, \boldsymbol{\theta}_t^{(l)}, w_t^{(l)}$
  end for
  Resample $\{(\boldsymbol{\theta}_t^{(l)}, \boldsymbol{\mu}_{t|t}^{(l)}, \mathbf{L}_{t|t}^{(l)})\}_{l=1}^{N_p}$ with weights $\{w_t^{(l)}\}_{l=1}^{N_p}$ to obtain equally weighted particles (e.g., Douc et al., 2005)
end for

Dense Laplace (DL): A further special case of HV with $M = 0$, in which $\mathbf{S}$ is a fully dense matrix of ones. Thus, there is no error due to the Vecchia approximation, and so in the non-Gaussian spatial-only setting, this is equivalent to a Laplace approximation. DL will generally be more accurate than HV and low-rank, but it scales as $\mathcal{O}(n^3)$ and is thus not feasible for high dimension n.

For each scenario below, we simulated observations using (3.12), taking $g_{t,i}$ to be each of four exponential-family distributions (Gaussian, logistic, Poisson and gamma), assuming a shape parameter $\alpha = 2$ for the gamma case. For most scenarios, we assumed a moderate state dimension

n, so that DL remained feasible; a large n was considered in Section 3.6.4.

The main metric to compare HV and LR was the difference in KL divergence between their posterior or filtering distributions and those generated by DL; as the exact distributions were not known here, we approximated this metric by the average difference in log scores (dLS; e.g., Gneiting and Katzfuss, 2014) over several simulations. We also calculated the relative root mean square prediction error (RRMSPE), defined as the root mean square prediction error of HV and LR, respectively, divided by the root mean square prediction error of DL. For each of the two criteria, lower values are better.

3.6.2 Spatial-only data

In our first scenario, we considered spatial-only data according to (3.5)–(3.6) on a grid $\mathcal{S}$ of size $n = 34 \times 34 = 1{,}156$ on the unit square, $\mathcal{D} = [0,1]^2$. We set $\boldsymbol{\mu} = \mathbf{0}$ and $\boldsymbol{\Sigma}_{i,j} = \exp(-\|\mathbf{s}_i - \mathbf{s}_j\|/0.15)$. For the Gaussian likelihood, we assumed variance $\tau^2 = 0.2$.

The comparison scores averaged over 100 simulations for the posteriors obtained using Algorithm 3 are shown as a function of N in Figure 3.2. HV was much more accurate than LR for each value of N.

3.6.3 Linear evolution

Next, we considered a linear spatio-temporal advection-diffusion process with diffusion parameter $\alpha = 4 \times 10^{-5}$ and advection parameter $\beta = 10^{-2}$ as in Jurek and Katzfuss (2018). The spatial domain $\mathcal{D} = [0,1]^2$ was discretized on a grid of size $n = 34 \times 34 = 1{,}156$ using the centered finite difference method, and we considered discrete time points $t = 1, \ldots, T$ with $T = 20$. After this discretization, our model was of the form (3.10)–(3.11), where $\boldsymbol{\Sigma}_{0|0} = \mathbf{Q}_1 = \ldots = \mathbf{Q}_T$ with (i,j)th entry $\exp(-\|\mathbf{s}_i - \mathbf{s}_j\|/0.15)$, and $\mathbf{E}_t$ was a sparse matrix with nonzero entries corresponding to interactions between neighboring grid points to the right, left, top and bottom. See the supplementary material of Jurek and Katzfuss (2018) for details.

At each time t, we generate $n_t = 0.1n$ observations with indices $\mathcal{I}_t$ sampled randomly from $\{1, \ldots, n\}$. For the Gaussian case, we assume variance $\tau^2 = 0.25$. We used conditioning sets of

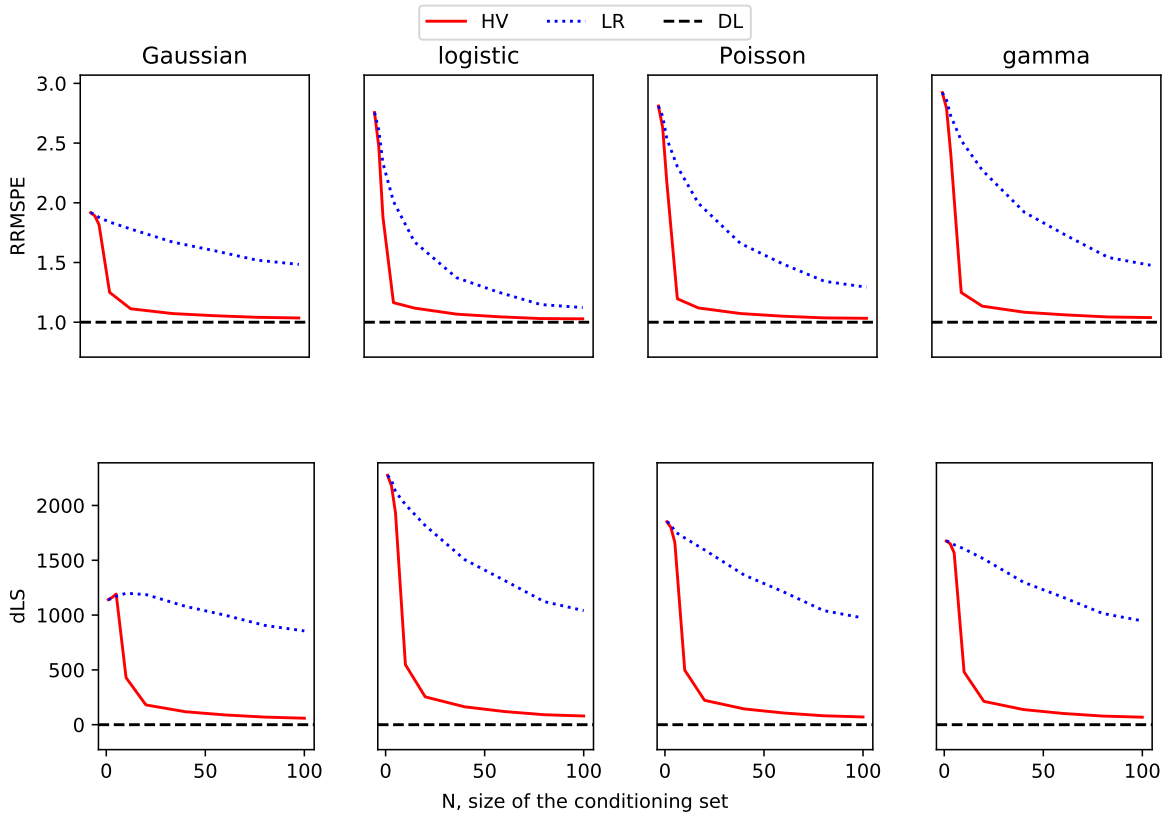


Figure 3.2: Approximation accuracy for the posterior distribution $\mathbf{x}|\mathbf{y}$ for spatial data (see Section 3.6.2)

size at most $N = 41$ for both HV and LR; specifically, for HV, we used $J = 2$ partitions at $M = 7$ resolutions, with set sizes $|\mathcal{X}_{j_1,\ldots,j_m}|$ of $5, 5, 5, 5, 6, 6, 6$ respectively for $m = 0, 1, \ldots, M-1$ and $|\mathcal{X}_{j_1,\ldots,j_m}| < 4$.

Figure 3.3 compares the scores for the filtering distributions $\mathbf{x}_t|\mathbf{y}_{1:t}$ obtained using Algorithm 4, averaged over 80 simulations. Again, HV was much more accurate than LR. Importantly, while the accuracy of HV was relatively stable over time, LR became less accurate over time, with the approximation error accumulating.

3.6.4 Simulations using a very large n

We repeated the advection-diffusion experiment from Section 3.6.3 on a very fine grid of size $n = 300 \times 300 = 90{,}000$ and we assumed that we have $n_t = 9{,}000$ corresponding to 10% of the grid points. In order to avoid numerical artifacts related to the finite differencing scheme, we

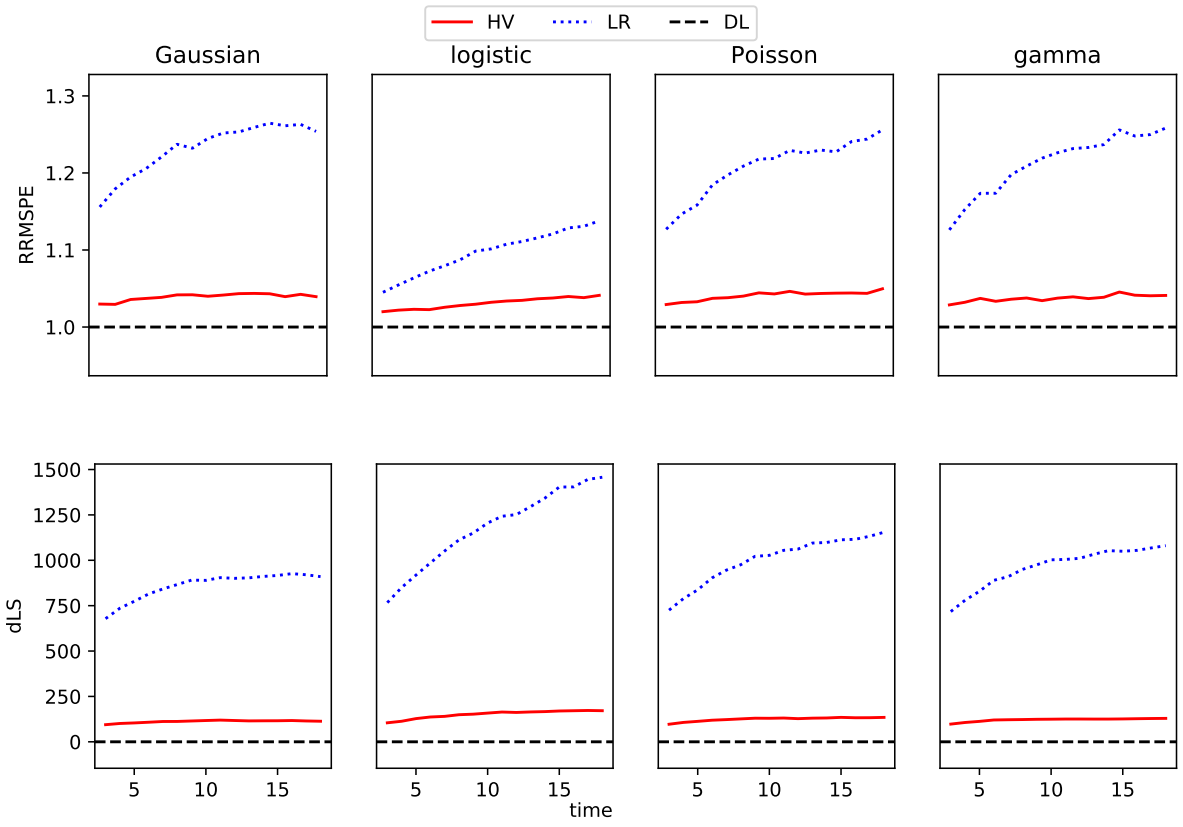


Figure 3.3: Accuracy of filtering distributions $\mathbf{x}_t|\mathbf{y}_{1:t}$ for the advection-diffusion model in Section 3.6.3

reduced the advection and diffusion coefficients to $\alpha = 10^{-7}$ and $\beta = 10^{-3}$, respectively. We set $N = 44$, $M = 14$, $J = 2$, and $|\mathcal{X}_{j_1,\ldots,j_m}| = 3$ for all $m = 0, 1, \ldots, M-1$, and $|\mathcal{X}_{j_1,\ldots,j_M}| \leq 3$. DL was too computationally expensive due to the high dimension n, and so we simply compared HV and LR based on the root mean square prediction error (RMSPE) between the true state and their respective filtering means, averaged over 10 iterarions.

As shown in Figure 3.4, HV was again much more accurate than LR. Comparing to Figure 3.3, we see that the relative improvement of HV to LR increased even further; taking the Gaussian case as an example, the ratio of the RMSPE for HV and LR was around 1.2 in the small-n setting, and greater than 2 in the large-n setting.

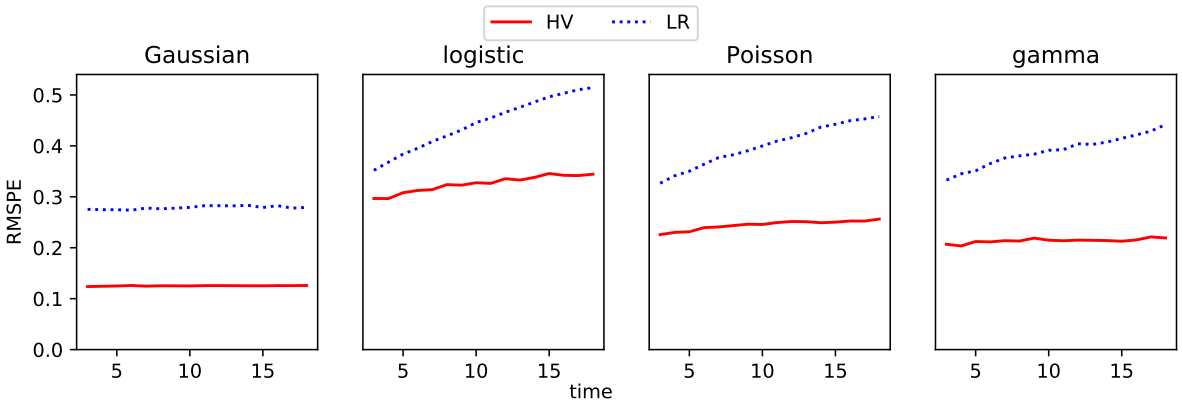


Figure 3.4: Root mean square prediction error (RMSPE) for the filtering mean in the high-dimensional advection-diffusion model with $n = 90{,}000$ in Section 3.6.4

3.6.5 Nonlinear evolution with non-Gaussian data

In the final set of our simulations, we discuss the most general form of (3.12)-(3.13) when $\mathcal{E}_t$ is a nonlinear function. Following the model introduced by Lorenz (2005), Section 3, we consider a stochastic process x defined on $n = 960$ points $\{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n\}$ evenly distributed on a circle with circumference equal to 1. If we define $X_i := x(\mathbf{s}_i)/b$ to be the value of the process at the n-th point scaled by b, its dynamics are described by the following equation:

$$\frac{\partial X_i}{\partial t} = [X]_{K,i} - X_i + F$$

where K is an even number and

$$[X]_{K,i} = \frac{1}{K^2} \sum_{l=-\frac{K}{2}}^{\frac{K}{2}} \sum_{j=-\frac{K}{2}}^{\frac{K}{2}} -X_{i-2K-l}X_{i-K-j} + X_{i-K+j-l}X_{i+K+j}$$

For our simulation we take $K = 32$, $b = 0.2$. As explained in detail in Lorenz (2005), the model is able to replicate some features of atmospheric variables along a given latitude band. Aiming at representing the model in the form (3.13) we solve it using a 4-th order Runge-Kutta scheme and a time step of $dt = 0.005$ to derive the evolution $\mathcal{E}_t$ operator at each t. We also calculate the analytic

expression for its derivative $\nabla \mathcal{E}_t$ which is necessary for Algorithm 5.

Next we run the model for 20000 time steps starting from a standard normal distribution. We then use the state of the model at time $t = 20000$ and run it for another $1,000,000$ time steps. We take every 40-th time step and calculate the sample mean and covariance which we use as moments of the initial distribution $\mathbf{x}_0 \sim \mathcal{N}(\boldsymbol{\mu}_{0|0}, \boldsymbol{\Sigma}_{0|0})$.

We then simulate $\mathbf{x}_t$ using (3.13) and $T = 100$ time steps and $\mathbf{Q}_t$ derived from an exponential covariance function with range 0.15 and marginal variance 0.2. Next, for each t we generate a vector of observations $\mathbf{y}_t = (y_{ti})_{i \in \mathcal{I}_t}$ as in (3.12). For each $i \in \mathcal{I}_t$ we take $g_{ti}(y_{ti}|x_{ti}) = h_{ti}(y_{ti}|x_{ti})$ where h_{ti} is the density of one of the four exponential family distributions listed in Section 3.6.1. We set $\mathcal{I}_t$ to be a random subset of locations $\mathbf{s}_1, \ldots, \mathbf{s}_n$ such that $\#\mathcal{I}_t = 0.1n = 96$.

For each data set obtained in this way we apply the three filtering methods described in Section 3.6.1. We use N=39 and for the HV filter we set $J = 2$, $M = 7$ and $|\mathcal{X}_{j_1,\ldots,j_m}|$ equal to 5, 5, 5, 5, 6, 6, 6 respectively for $m = 0, 1, \ldots, M-1$ and $|\mathcal{X}_{j_1,\ldots,j_M}| < 2$. We repeat the whole process 40 times and report in Figure 3.5 the average scores at the 20 time points when data was available.

The results show that our method (HV) compares favorably to the low-rank filter and provides excellent approximation accuracy as evidenced by very low RRMSPE and dLS scores.

3.7 Conclusions

After specifying the relationship between ordered conditional independence and sparse (inverse) Cholesky factors, we described a hierarchical Vecchia approximation, which exhibits sparse Cholesky factors on the covariance and on the precision scale. Due to this remarkable sparsity property, the approximation is suitable for high-dimensional spatio-temporal filtering. The hierarchical Vecchia approximation can be computed using a simple and fast sparse Cholesky decomposition (IC0). Further, by combining the approach with a Laplace approximation and the extended Kalman filter, we obtained scalable filters for non-Gaussian and non-linear spatio-temporal state-space models.

Our methods can be directly applied to spatio-temporal point patterns modeled using log-Gaussian Cox processes, which can be viewed as Poisson data after discretization of the spatial

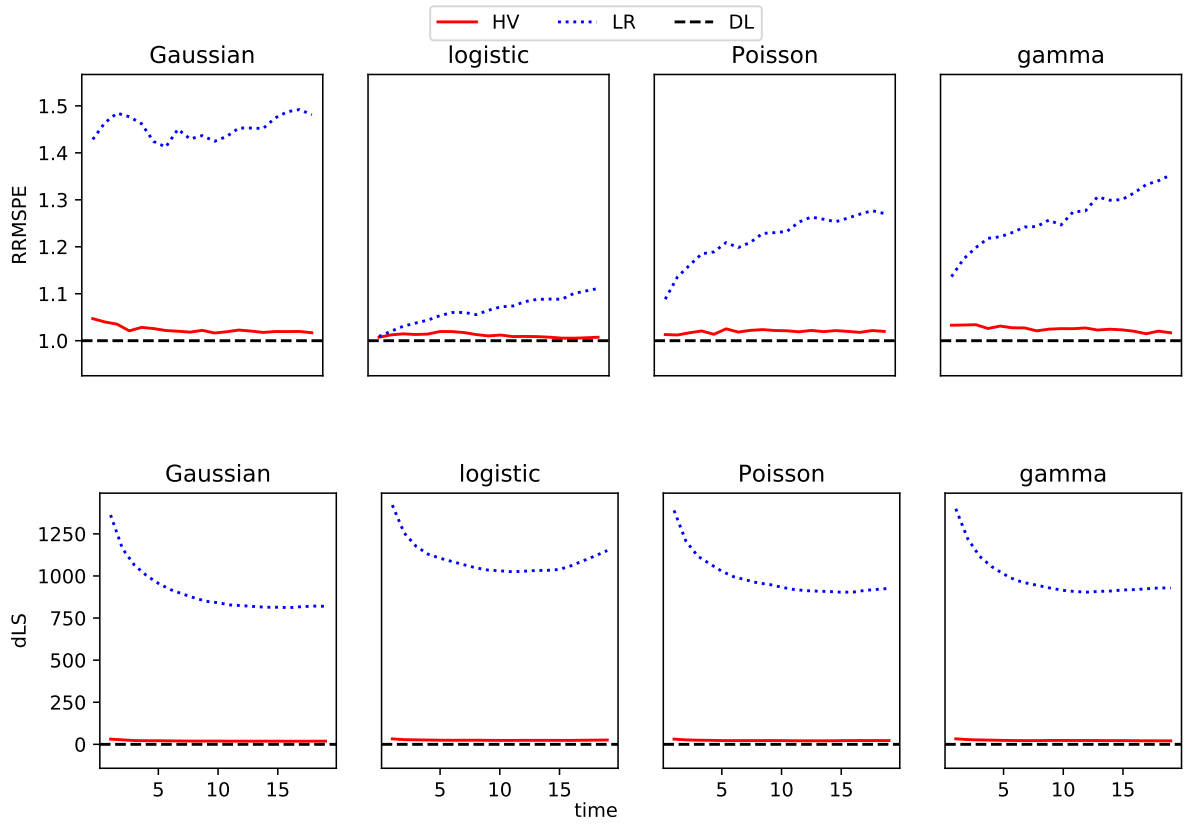


Figure 3.5: Accuracy of filtering distribution $\mathbf{x}_t|\mathbf{y}_{1:t}$ for the Lorenz model in Section 3.6.5

domain, resulting in accurate Vecchia-Laplace-type approximations (Zilber and Katzfuss, 2019). We plan on investigating an extension of our methods to retrospective smoothing over a fixed time period. Another interesting extension would be to combine our methodology with the unscented Kalman filter (Julier and Uhlmann, 1997) for strongly nonlinear evolution. Finally, while we focused our attention on spatio-temporal data, our work can be extended to other applications, as long as a sensible hierarchical partitioning of the state vector can be obtained as in Section 3.3.1.

4. CONCLUSION

In conclusion, this thesis presents a family of new filtering methods based on the Vecchia approximation for latent Gaussian state-space models. Thanks to the versatility of the approximation, our algorithm can be used in the presence of non-Gaussian data. We also show how to address the case of non-linear temporal evolution. Our approach scales well to high dimensions while maintaining accuracy and being able to resolve fine-scale features of the filtering distribution. As indicated in the preceding chapters, we envision extending our algorithm to enable smoothing inference. We also plan to improve the accuracy in highly non-linear models using an approach based on the unscented Kalman filter. Finally, we also provide an implementation of our method, which can be found at `https://github.com/katzfuss-group/vecchiaFilter`.

REFERENCES

Ambikasaran, S. and Darve, E. (2013). An O(n log n) fast direct solver for partial hierarchically semi-separable matrices. *Journal of Scientific Computing*, 57(3):477–501.

Ambikasaran, S., Foreman-Mackey, D., Greengard, L., Hogg, D. W., and O'Neil, M. (2016). Fast direct methods for Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2):252–265.

Anderson, J. L. (2001). An ensemble adjustment Kalman filter for data assimilation. *Monthly Weather Review*, 129(12):2884–2903.

Banerjee, S., Carlin, B. P., and Gelfand, A. E. (2004). *Hierarchical Modeling and Analysis for Spatial Data*. Chapman & Hall.

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

Beezley, J. D., Mandel, J., and Cobb, L. (2011). Wavelet ensemble Kalman filters. In *Proceedings of the 6th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems*, volume 2, pages 514–517.

Burgers, G., Jan van Leeuwen, P., and Evensen, G. (1998). Analysis scheme in the ensemble Kalman filter. *Monthly Weather Review*, 126(6):1719–1724.

Choi, M. J., Chandrasekaran, V., and Willsky, A. S. (2010). Gaussian multiresolution models: Exploiting sparse markov and covariance structure. *IEEE Transactions on Signal Processing*, 58(3):1012–1024.

Chou, K. C., Willsky, A. S., and Benveniste, A. (1994a). Multiscale recursive estimation, data fusion, and regularization. *IEEE Transactions on Automatic Control*, 39(3):464–478.

Chou, K. C., Willsky, A. S., and Nikoukhah, R. (1994b). Multiscale systems, Kalman filters, and Riccati equations. *IEEE Transactions on Automatic Control*, 39(3):479–492.

Chui, C. (1992). *An Introduction to Wavelets*. Academic Press.

Cressie, N. (1993). *Statistics for Spatial Data, revised edition*. John Wiley & Sons, New York, NY.

Cressie, N., Shi, T., and Kang, E. L. (2010). Fixed rank filtering for spatio-temporal data. *Journal of Computational and Graphical Statistics*, 19(3):724–745.

Cristi, R. and Tummala, M. (2000). Multirate, multiresolution, recursive Kalman filter. *Signal Processing*, 80:1945–1958.

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

Datta, A., Banerjee, S., Finley, A. O., Hamm, N. A. S., and Schaap, M. (2016b). Non-separable dynamic nearest-neighbor Gaussian process models for large spatio-temporal data with an application to particulate matter analysis. *Annals of Applied Statistics*, 10(3):1286–1316.

DelSole, T. and Yang, X. (2010). State and parameter estimation in stochastic dynamical models. *Physica D: Nonlinear Phenomena*, 239(18):1781–1788.

Douc, R., Cappé, O., and Moulines, E. (2005). Comparison of resampling schemes for particle filtering. *ISPA 2005. Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis, 2005.*, pages 64–69.

Doucet, A., de Freitas, N., Murphy, K., and Russell, S. (2000). Rao-Blackwellised particle filtering for dynamic Bayesian networks. *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 176–183.

Doucet, A. and Johansen, A. M. (2009). A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of nonlinear filtering*, 12(656-704):3.

Evensen, G. (1994). Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research*, 99(C5):1014310162.

Evensen, G. (2007). *Data Assimilation: The Ensemble Kalman Filter*. Springer.

Ferreira, M. A. and Lee, H. K. (2007). *Multiscale Modeling: A Bayesian Perspective*. Springer.

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

Frakt, A. B. and Willsky, A. S. (2001). Computationally efficient stochastic realization for internal multiscale autoregressive models. *Multidimensional Systems and Signal Processing*, 12(2):109–142.

Geoga, C. J., Anitescu, M., and Stein, M. L. (2018). Scalable Gaussian Process Computations Using Hierarchical Matrices. *arXiv:1808.03215*.

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

Golub, G. H. and Van Loan, C. F. (2012). *Matrix Computations*. JHU Press, 4th edition.

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

Grewal, M. S. and Andrews, A. P. (1993). *Kalman Filtering: Theory and Applications*. Prentice Hall.

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

Hackbusch, W. (1999). A sparse matrix arithmetic based on H-matrices. Part I: Introduction to H-matrices. *Computing*, 62(2):89–108.

Hackbusch, W. (2015). *Hierarchical Matrices: Algorithms and Analysis*, volume 49. Springer.

Harville, D. A. (1997). *Matrix Algebra From a Statistician's Perspective*. Springer, New York, NY.

Heaton, M. J., Datta, A., Finley, A. O., Furrer, R., Guinness, J., Guhaniyogi, R., Gerber, F., Gramacy, R. B., Hammerling, D., Katzfuss, M., Lindgren, F., Nychka, D. W., Sun, F., and Zammit-Mangion, A. (2019). A case study competition among methods for analyzing large spatial data. *Journal of Agricultural, Biological, and Environmental Statistics*, 24(3):398–425.

Henderson, H. and Searle, S. (1981). On deriving the inverse of a sum of matrices. *SIAM Review*,

23(1):5360.

Hickmann, K. S. and Godinez, H. C. (2015). A multiresolution ensemble Kalman filter using wavelet decomposition. *arXiv:1511.01935*.

Hogben, L. (2006). *Handbook of Linear Algebra*. Discrete Mathematics and Its Applications. CRC Press.

Houtekamer, P. L. and Zhang, F. (2016). Review of the ensemble Kalman filter for atmospheric data assimilation. *Monthly Weather Review*, 144(12):4489–4532.

Huang, H.-C., Cressie, N., and Gabrosek, J. (2002). Fast, resolution-consistent spatial prediction of global processes from satellite data. *Journal of Computational and Graphical Statistics*, 11(1):6388.

Johannesson, G., Cressie, N., and Huang, H.-C. (2003). Dynamic multi-resolution spatial models. In Higuchi, T., Iba, Y., and Ishiguro, M., editors, *Proceedings of AIC2003: Science of Modeling*, volume 14, pages 167–174, Tokyo. Institute of Statistical Mathematics.

Julier, S. J. and Uhlmann, J. K. (1997). New extension of the Kalman filter to nonlinear systems. In *Int. Symp. Aerospace/Defense Sensing, Simul. and Controls*, pages 182–193.

Jurek, M. and Katzfuss, M. (2018). Multi-resolution filters for massive spatio-temporal data. *arXiv:1810.04200*.

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

Kanter, M. (1997). Unimodal spectral windows. *Statistics & Probability Letters*, 34(4):403411.

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

Katzfuss, M. and Cressie, N. (2011). Spatio-temporal smoothing and EM estimation for massive remote-sensing data sets. *Journal of Time Series Analysis*, 32(4):430–446.

Katzfuss, M. and Gong, W. (2019). A class of multi-resolution approximations for large spatial datasets. *Statistica Sinica*, accepted.

Katzfuss, M. and Guinness, J. (2019). A general framework for Vecchia approximations of Gaus-

sian processes. *Statistical Science*, accepted.

Katzfuss, M., Guinness, J., Gong, W., and Zilber, D. (2018). Vecchia approximations of Gaussian-process predictions. *arXiv:1805.03309*.

Katzfuss, M., Guinness, J., and Lawrence, E. (2020). Scaled Vecchia approximation for fast computer-model emulation. *arXiv:2005.00386*.

Katzfuss, M., Hammerling, D., and Smith, R. L. (2017). A Bayesian hierarchical model for climate-change detection and attribution. *Geophysical Research Letters*, 44(11):5720–5728.

Katzfuss, M., Stroud, J. R., and Wikle, C. K. (2016). Understanding the ensemble Kalman filter. *The American Statistician*, 70(4):350–357.

Katzfuss, M., Stroud, J. R., and Wikle, C. K. (2019). Ensemble Kalman methods for high-dimensional hierarchical dynamic space-time models. *Journal of the American Statistical Association*, accepted.

Kaufman, C. G., Bingham, D., Habib, S., Heitmann, K., and Frieman, J. A. (2011). Efficient emulators of computer experiments using compactly supported correlation functions, with an application to cosmology. *The Annals of Applied Statistics*, 5(4):2470–2492.

Kennedy, M. C. and O'Hagan, A. (2001). Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B*, 63(3):425–464.

Khare, K. and Rajaratnam, B. (2012). Sparse matrix decompositions and graph characterizations. *Linear Algebra and its Applications*, 437(3):932–947.

Kincaid, D. and Cheney, E. (2002). *Numerical Analysis: Mathematics of Scientific Computing*. Pure and applied undergraduate texts. American Mathematical Society.

Lauritzen, S. (1996). *Graphical Models*. Oxford Statistical Science Series. Clarendon Press.

Li, J. Y., Ambikasaran, S., Darve, E. F., and Kitanidis, P. K. (2014). A Kalman filter powered by H-matrices for quasi-continuous data assimilation problems. *Water Resources Research*, 50(5):3734–3749.

Lindgren, F., Rue, H., and Lindström, J. (2011). An explicit link between Gaussian fields and Gaussian Markov random fields: the stochastic partial differential equation approach. *Journal*

of the Royal Statistical Society, Series B, 73(4):423–498.

Liu, H., Ong, Y.-S., Shen, X., and Cai, J. (2018). When Gaussian process meets big data: A review of scalable GPs. *arXiv:1807.01065*.

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

Luettgen, M. R. and Willsky, A. S. (1995). Likelihood calculation for a class of multiscale stochastic models, with application to texture discrimination. *IEEE transactions on image processing*, 4(2):194–207.

Nychka, D. W. and Anderson, J. L. (2010). Data assimilation. In Gelfand, A. E., Diggle, P. J., Fuentes, M., and Guttorp, P., editors, *Handbook of Spatial Statistics*, chapter 27, pages 477–494. CRC Press.

Nychka, D. W., Bandyopadhyay, S., Hammerling, D., Lindgren, F., and Sain, S. R. (2015). A multi-resolution Gaussian process model for the analysis of large spatial data sets. *Journal of Computational and Graphical Statistics*, 24(2):579–599.

Pham, D. T., Verron, J., and Christine Roubaud, M. (1998). A singular evolutive extended Kalman filter for data assimilation in oceanography. *Journal of Marine Systems*, 16(3-4):323–340.

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. MIT Press.

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

Renaud, O., Starck, J. L., and Murtagh, F. (2005). Wavelet-based combined signal filtering and prediction. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 35(6):1241–1251.

Rue, H. and Held, L. (2010). Discrete spatial variation. In *Handbook of Spatial Statistics*, chapter 12, pages 171–200. CRC Press.

Sacks, J., Welch, W., Mitchell, T., and Wynn, H. (1989). Design and analysis of computer experiments. *Statistical Science*, 4(4):409–435.

Saibaba, A. K., Miller, E. L., and Kitanidis, P. K. (2015). Fast Kalman filter using hierarchical matrices and a low-rank perturbative approach. *Inverse Problems*, 31(1):015009.

Schäfer, F., Katzfuss, M., and Owhadi, H. (2020). Sparse Cholesky factorization by Kullback-Leibler minimization. *arXiv:2004.14455*.

Schäfer, F., Sullivan, T. J., and Owhadi, H. (2017). Compression, inversion, and approximate PCA of dense kernel matrices at near-linear computational complexity. *arXiv:1706.02205*.

Searle, S. (1982). *Matrix Algebra Useful for Statistics*. John Wiley & Sons.

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

Snyder, C., Bengtsson, T., Bickel, P. J., and Anderson, J. L. (2008). Obstacles to high-dimensional particle filtering. *Monthly Weather Review*, 136(12):4629–4640.

Stein, M. L. (2002). The screening effect in kriging. *Annals of Statistics*, 30(1):298–323.

Stein, M. L. (2014). Limitations on low rank approximations for covariance matrices of spatial data. *Spatial Statistics*, 8:1–19.

Stein, M. L., Chi, Z., and Welty, L. (2004). Approximating likelihoods for large spatial data sets. *Journal of the Royal Statistical Society: Series B*, 66(2):275296.

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

Talagrand, O. and Courtier, P. (1987). Variational assimilation of meteorological observations with the adjoint vorticity equation. I: Theory. *Quarterly Journal of the Royal Meteorological Society*, 113(478):13111328.

Tierney, L. and Kadane, J. B. (1986). Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association*, 81(393):82–86.

Tippett, M. K., Anderson, J. L., Bishop, C. H., Hamill, T. M., and Whitaker, J. S. (2003). Ensemble square-root filters. *Monthly Weather Review*, 131:1485–1490.

Toledo, S. (2007). Lecture Notes on Combinatorial Preconditioners, Chapter 3. http://www.tau.ac.il/˜stoledo/Support/chapter-direct.pdf.

Tzeng, S., Huang, H.-C., and Cressie, N. (2005). A fast, optimal spatial-prediction method for massive datasets. *Journal of the American Statistical Association*, 100(472):1343–1357.

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

Verlaan, M. and Heemink, A. W. (1995). Reduced rank square root filters for large scale data assimilation problems. In *Proceedings of the 2nd International Symposium on Assimilation in Meteorology and Oceanography*, pages 247–252. World Meteorological Organization.

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

Willsky, A. S. (2002). Multiresolution markov models for signal and image processing. *Proceedings of the IEEE*, 90(8):1396–1458.

Xu, K. and Wikle, C. K. (2007). Estimation of parameterized spatio-temporal dynamic models. *Journal of Statistical Planning and Inference*, 137(2):567–588.

Zhang, B., Sang, H., and Huang, J. Z. (2015). Full-scale approximations of spatio-temporal covariance models for large datasets. *Statistica Sinica*, 25:99–114.

Zilber, D. and Katzfuss, M. (2019). Vecchia-Laplace approximations of generalized Gaussian processes for big non-Gaussian spatial data. *arXiv:1906.07828*.

APPENDIX A

PROOFS OF RESULTS FROM CHAPTER 2

A.1 Proofs

We now provide proofs for the propositions stated throughout the article. We simplify notation by dropping most time subscripts; to avoid confusion, we denote $\mathbf{B}_{t|t-1}$ by $\mathbf{B}$, and $\mathbf{B}_{t|t}$ by $\tilde{\mathbf{B}}$. In Section C.8, we provide lemmas with proofs that are used in the proof of Proposition 3 here. Sections C.6-C.7 contain additional technical concepts used in the lemmas, including a review of basic ideas from graph theory, hierarchical-matrix theory, and some illustrative figures. Finally, throughout this appendix, if $\mathbf{G}$ is a square matrix, we use $\mathbf{G}^L$ and $\mathbf{G}^U$ to denote its lower and upper triangles, respectively.

Proof of Proposition 1. Recall that $\mathbf{B} = \big(\mathbf{B}^M, \mathbf{B}^{M-1}, \ldots, \mathbf{B}^0\big)$. This lets us write the i-th row of $\mathbf{B}$ as $\mathbf{B}[i,:] = \big(\mathbf{B}^M[i,:], \mathbf{B}^{M-1}[i,:], \ldots, \mathbf{B}^0[i,:]\big)$. By construction, each block $\mathbf{B}^m$ is block-diagonal and such that for $m \leq M$, each segment $\mathbf{B}^m[i,:]$ has only r_m nonzero elements. Because each row of $\mathbf{B}$ is composed of $M+1$ blocks $\mathbf{B}^m[i,:]$ for $m = 0, \ldots, M$, this ends the proof. □

Proof of Proposition 2. Direct calculation shows that $\mathbf{B}'\mathbf{B}$ is a block matrix consisting of $(M+1) \times (M+1)$ blocks with $(M-k+1, M-l+1)$-th block $(\mathbf{B}^k)'\mathbf{B}^l$. Since for each j the matrix $\mathbf{B}^j$ has dimensions $n_{\mathcal{S}} \times |\mathcal{K}^j|$ it follows that $(\mathbf{B}^k)'\mathbf{B}^l$ is of size $|\mathcal{K}^k| \times |\mathcal{K}^l|$. Note that $\mathbf{B}^k$ and $\mathbf{B}^l$ are block-diagonal with blocks of size $|\mathcal{I}_{j_1,\ldots,j_k}| \times r_k$ and $|\mathcal{I}_{j_1,\ldots j_l}| \times r_l$, respectively. Assuming without loss of generality that $k \leq l$, we have that

$$\mathcal{I}_{j_1,\ldots,j_k} = \bigcup_{j_{k+1}=1}^{J} \cdots \bigcup_{j_l=1}^{J} \mathcal{I}_{j_1,\ldots,j_\ell}, \qquad \Longrightarrow \qquad |\mathcal{I}_{j_1,\ldots,j_k}| = \sum_{j_{k+1}=1}^{J} \cdots \sum_{j_l=1}^{J} |\mathcal{I}_{j_1,\ldots,j_\ell}|.$$

Thus $\mathbf{B}^l$ can be viewed as a block-diagonal matrix with blocks of height $|\mathcal{I}_{j_1,\ldots,j_k}|$. We can also determine their width to be $w_{j_1,\ldots,j_k} = \sum_{j_{k+1}=1}^{J} \cdots \sum_{j_l=1}^{J} |\mathcal{K}_{j_1,\ldots,j_k,j_{k+1},\ldots j_l}|$. This means $(\mathbf{B}^k)'\mathbf{B}^l$ is

the product of two block-diagonal matrices with matching block sizes. Therefore the product will be also block-diagonal with blocks of dimensions $w_{j_1,\ldots,j_k} \times r_k$. □

Proof of Proposition 3.

1. Observe that under Assumption 1, $\mathbf{R}^{-1}$ and $\mathbf{H}$ are block-diagonal with blocks of matching dimensions. Since $\mathbf{R}^{-1}$ has square blocks, we conclude that $\mathbf{H}'\mathbf{R}^{-1} \in \mathcal{S}(\mathbf{H}')$. Thus, if $\tilde{\mathbf{R}}^{-1} := \mathbf{H}'\mathbf{R}^{-1}\mathbf{H}$, then $\tilde{\mathbf{R}}^{-1} \in \mathcal{S}(\mathbf{H}'\mathbf{H})$. The latter is a block-diagonal matrix with square blocks of size $|\mathcal{I}_{j_1,\ldots,j_M}|$.

   Next, we demonstrate that $\mathbf{B}'\tilde{\mathbf{R}}^{-1}\mathbf{B} \in \mathcal{S}(\mathbf{B}'\mathbf{B})$. First, as $\tilde{\mathbf{R}}^{-1}$ is block-diagonal, the $(M + 1 - k, M + 1 - l)$-th block of $\mathbf{B}'\tilde{\mathbf{R}}^{-1}\mathbf{B}$ is given by $(\mathbf{B}^k)'\tilde{\mathbf{R}}^{-1}\mathbf{B}^l$.

   Now, for each $0 \leq k \leq M$, $\mathbf{B}^k$ is a block-diagonal matrix with blocks of size $|\mathcal{I}_{j_1,\ldots,j_k}| \times r_k$, but $\tilde{\mathbf{R}}^{-1}$ has blocks of size $|\mathcal{I}_{j_1,\ldots,j_M}| \times |\mathcal{I}_{j_1,\ldots,j_M}|$. However, recalling (A.1), blocks of $\tilde{\mathbf{R}}^{-1}$ can also be viewed as having dimensions $|\mathcal{I}_{j_1,\ldots,j_k}| \times r_k$. Because this implies that $(\mathbf{B}^k)'\tilde{\mathbf{R}}^{-1} \in \mathcal{S}(\mathbf{B}^k)'$, we have $(\mathbf{B}^k)'\tilde{\mathbf{R}}^{-1}\mathbf{B}^l \in \mathcal{S}((\mathbf{B}^k)'\mathbf{B}^l)$ and hence $(\mathbf{B})'\tilde{\mathbf{R}}^{-1}\mathbf{B} \in \mathcal{S}(\mathbf{B}'\mathbf{B})$. Finally, we conclude that $\prec \in \mathcal{S}(\mathbf{B}'\mathbf{B})$, because $\prec = \mathbf{I}_{n_S} + \mathbf{B}'\tilde{\mathbf{R}}^{-1}\mathbf{B}$ and all diagonal elements of $\mathbf{B}'\mathbf{B}$ are nonzero.

2. According to Khare and Rajaratnam (2012, Thm. 1), for any positive definite matrix $\mathbf{S}$, the sparsity pattern in the Cholesky factor and its inverse are the same as that of the lower triangle of $\mathbf{S}$, if (a) the pattern of zeros in $\mathbf{S}$ corresponds to a homogeneous graph, and (b) the order of the vertices of the graph implied by the order of the rows is a Hasse-tree-based elimination scheme. Lemmas 1 and 2 in Section C.8 show that these two conditions are met for $\mathbf{B}'\mathbf{B}$. These lemmas, together with Part 1 above, imply that $\mathbf{L} \in \mathcal{S}(\prec^L)$ and $\mathbf{L}^{-1} \in \mathcal{S}(\prec^L)$.

3. Observe that $\prec^{-1} = (\mathbf{L}\mathbf{L}')^{-1} = (\mathbf{L}^{-1})'\mathbf{L}^{-1}$. Thus $(\mathbf{L}^{-1})'$ is the Cholesky factor of $\prec^{-1}$.

Moreover, by Part 2, $(\mathbf{L}^{-1})' \in \mathcal{S}((\mathbf{B}'\mathbf{B})^U)$. This allows us to define blocks $\tilde{\mathbf{L}}^{m,k}$ such that

$$(\mathbf{L}^{-1})' = \begin{bmatrix} \tilde{\mathbf{L}}^{M,M} & \dots & \tilde{\mathbf{L}}^{M,1} & \tilde{\mathbf{L}}^{M,0} \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \dots & \tilde{\mathbf{L}}^{1,1} & \tilde{\mathbf{L}}^{1,0} \\ \mathbf{0} & \dots & \mathbf{0} & \tilde{\mathbf{L}}^{0,0} \end{bmatrix} = \left[\tilde{\mathbf{L}}^{m,k}\right]_{m,k=M,\dots,0},$$

where each $\tilde{\mathbf{L}}^{m,k} \in \mathcal{S}((\mathbf{B}^m)'\mathbf{B}^k)$ for $m \geq k$ and is zero when $m < k$. This means that for each m, k with $m \geq k$, we can consider the sparsity of $\mathbf{B}^m(\mathbf{B}^m)'\mathbf{B}^k$ instead of $\mathbf{B}^m\tilde{\mathbf{L}}^{m,k}$.

Recall that $\mathbf{B}^k$ is block-diagonal with blocks of size $|\mathcal{I}_{j_1,\dots,j_k}| \times r_k$. Similarly, $\mathbf{B}^m$ has blocks that are $|\mathcal{I}_{j_1,\dots,j_m}| \times r_m$. However, since $k \leq m$, using (A.1)

we can also see $\mathbf{B}^m$ as a block-diagonal matrix whose blocks have dimensions $|\mathcal{I}_{j_1,\dots,j_k}| \times r_k$ (cf. proof of Proposition 2). This implies that $\mathbf{B}^m(\mathbf{B}^m)' \in \mathcal{S}(\mathbf{B}^k(\mathbf{B}^k)')$, which means that $\mathbf{B}^m(\mathbf{B}^m)'\mathbf{B}^k \in \mathcal{S}(\mathbf{B}^k)$ and hence $\mathbf{B}^m\tilde{\mathbf{L}}^{m,k} \in \mathcal{S}(\mathbf{B}^k)$.

Finally, we observe that

$$\mathbf{B} \cdot (\mathbf{L}^{-1})' = \begin{bmatrix} \mathbf{B}^M & \mathbf{B}^{M-1} & \dots & \mathbf{B}^0 \end{bmatrix} \cdot \begin{bmatrix} \tilde{\mathbf{L}}^{M,M} & \dots & \tilde{\mathbf{L}}^{M,1} & \tilde{\mathbf{L}}^{M,0} \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \dots & \tilde{\mathbf{L}}^{1,1} & \tilde{\mathbf{L}}^{1,0} \\ \mathbf{0} & \dots & \mathbf{0} & \tilde{\mathbf{L}}^{0,0} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{B}}^M & \tilde{\mathbf{B}}^{M-1} & \dots & \tilde{\mathbf{B}}^0 \end{bmatrix},$$

where $\tilde{\mathbf{B}}^k = \sum_{m=k}^{M} \mathbf{B}^m\tilde{\mathbf{L}}^{m,k}$. Since we showed that $\mathbf{B}^m\tilde{\mathbf{L}}^{m,k} \in \mathcal{S}(\mathbf{B}^k)$, this means that $\tilde{\mathbf{B}}^k \in \mathcal{S}(\mathbf{B}^k)$.

□

Proof of Proposition 4. By Proposition 3, Parts 1 and 2, $\mathbf{L} \in \mathcal{S}((\mathbf{B}'\mathbf{B})^L)$. Therefore, it suffices to show that $\mathbf{c}_j = (\mathbf{B}'\mathbf{B})^L[:, j]$, the j-th column of $(\mathbf{B}'\mathbf{B})^L$, has $\mathcal{O}(N)$ nonzero elements for each j. Notice that $\mathbf{c}_j = (0, \dots, 0, \mathbf{c}_j^{k,k}, \dots \mathbf{c}_j^{M,k})'$ where $\mathbf{c}_j^{k,l} = ((\mathbf{B}^k)'\mathbf{B}^l)[:, j]$, the j-th column of

$(\mathbf{B}^k)'\mathbf{B}^l$. Because $l \geq k$, each of the $\mathbf{B}^k(\mathbf{B}^l)'$ matrices is block-diagonal with blocks of height $|\mathcal{K}_{j_1,\ldots,j_k}|$. The vector $\mathbf{c}_j^{k,l}$ intersects exactly one of such diagonal blocks, and so the total number of nonzero elements in $\mathbf{c}$ is at most $N = \sum_m r_m$. □

Proof of Proposition 5. Observe that it is enough to consider only the complexity of operations in (2.4) because $\mathbf{V}^l_{j_1,\ldots,j_m}$ can be obtained by selecting appropriate rows from $\mathbf{W}^l_{j_1,\ldots,j_m}$. Given matrix $\mathbf{\Sigma}$, we only need to calculate the second term in (2.4). First, note that calculating $\mathbf{W}^l_{j_1,\ldots,j_m}$ for all $(j_1,\ldots,j_m)$ is the same as computing $\mathbf{W}^l_{j_1,\ldots,j_\ell}$ for all l, and then, for each $(j_1,\ldots,j_m)$, selecting the rows corresponding to $\mathcal{I}_{j_1,\ldots,j_m}$. Thus we show the complexity of calculating all $\mathbf{W}^l_{j_1,\ldots,j_\ell}$.

Assume that all $\mathbf{W}^k_{j_1,\ldots,j_\ell}$ for $k < l$ are already given and consider the summation term. Each of its components takes $\mathcal{O}(|\mathcal{I}_{j_1,\ldots,j_\ell}|r^2 + r^3 + r^3 + r^3) = \mathcal{O}(|\mathcal{I}_{j_1,\ldots,j_\ell}|r^2)$ to compute. Because for any given l, there are at most M terms under the summation, their joint computation time is $\mathcal{O}(M \cdot |\mathcal{I}_{j_1,\ldots,j_\ell}|r^2)$. For a given l, these calculations have to be performed for each set of indices $\mathcal{I}_{j_1,\ldots,j_\ell}$. Thus, obtaining all $\mathbf{W}^l_{j_1,\ldots,j_\ell}$ requires $\mathcal{O}(M \cdot \sum_{j_1,\ldots,j_\ell} |\mathcal{I}_{j_1,\ldots,j_\ell}|r^2) = \mathcal{O}(M \cdot nr^2)$ time. Now notice that $\mathcal{I}_{j_1,\ldots,j_m} \subset \mathcal{I}_{j_1,\ldots,j_\ell}$. Therefore, once we have $\mathbf{W}^l_{j_1,\ldots,j_\ell}$, we obtain $\mathbf{W}^l_{j_1,\ldots,j_m}$ by selecting appropriate rows from $\mathbf{W}^l_{j_1,\ldots,j_\ell}$. Finally, iterating over $l = 0,\ldots,M$ means that the total cost of Algorithm 2 is $\mathcal{O}(M^2nr^2) = \mathcal{O}(nN^2)$. □

Proof of Proposition 6. The forecast step requires calculating $\boldsymbol{\mu}_{t|t-1} = \mathbf{E}_t\boldsymbol{\mu}_{t-1|t-1}$ and $\mathbf{B}^F_{t|t-1} = \mathbf{E}_t\mathbf{B}_{t-1|t-1}$, which can be obtained in $\mathcal{O}(nr)$ and $\mathcal{O}(nrN)$ time, respectively, due to the sparsity structures of $\mathbf{B}_{t-1|t-1}$ (see Proposition 1) and $\mathbf{E}_t$ (Assumption 2).

By Proposition 5, the MRD of a given covariance matrix $\mathbf{\Sigma}$ requires $\mathcal{O}(nN^2)$ operations. Here, $\mathbf{\Sigma} = \mathbf{\Sigma}_{t|t-1}$ is not given, but each (i,j) element must be computed as

$$\mathbf{\Sigma}_{t|t-1}[i,j] = (\mathbf{B}^F_{t|t-1}[i,:])(\mathbf{B}^F_{t|t-1}[j,:])' + \mathbf{Q}_t[i,j].$$

This does not increase the complexity of the MRD, because the MRD requires only $\mathcal{O}(nN)$ elements of $\mathbf{\Sigma}_{t|t-1}$, each of which can be computed in $\mathcal{O}(N)$ time due to the sparsity structure of $\mathbf{B}^F_{t|t-1}$. Thus, the entire forecast step can be performed in $\mathcal{O}(nN^2)$ time.

In the update step, we must compute $\widetilde{\mathbf{\Lambda}}$, $\mathbf{L}^{-1} = \widetilde{\mathbf{\Lambda}}^{-1/2}$, and $\mathbf{B}_{t|t} = \mathbf{B}_{t|t-1}(\mathbf{L}^{-1})'$. Under Assumption 1, $\mathbf{H}$ and $\mathbf{R}$ are block-diagonal matrices with at most J^M blocks of size $\mathcal{O}(r \times r)$ each. Thus, calculating $\tilde{\mathbf{R}} := \mathbf{H}'\mathbf{R}^{-1}\mathbf{H}$ requires $\mathcal{O}(J^M r^3) = \mathcal{O}(nr^2)$ operations. The resulting matrix is block-diagonal with blocks of size $\mathcal{O}(r \times r)$, conformable with the blocks of $\mathbf{B}_{t|t-1}$. Given $\tilde{\mathbf{R}}$, the cost of calculating $\widetilde{\mathbf{\Lambda}}$ is dominated by multiplying $\mathbf{B}_{t|t-1}$ by $\tilde{\mathbf{R}}$. By Proposition 1, each row of $\mathbf{B}_{t|t-1}$ has N nonzero elements, so in view of the structure of $\tilde{\mathbf{R}}$ determined above, it takes $\mathcal{O}(nN^2)$ operations to obtain the product $\mathbf{B}_{t|t-1}\tilde{\mathbf{R}}$ and, consequently, to calculate $\widetilde{\mathbf{\Lambda}}$.

The complexity of computing a Cholesky factor is on the order of the sum of the squared number of nonzero elements per column (e.g., Toledo, 2007, Thm. 2.2). Thus, computing $\mathbf{L}$ requires $\mathcal{O}(nN^2)$ time, because $\mathbf{L}$ has $\mathcal{O}(N)$ elements in each of its n columns (Proposition 3). Computing $\mathbf{L}^{-1}$ can be accomplished by solving a triangular system of equations for each column of $\mathbf{L}^{-1}$. Using Proposition 4, we conclude that each of these systems will have only $\mathcal{O}(N)$ equations and thus can be solved in $\mathcal{O}(N^2)$ time (Kincaid and Cheney, 2002, Ch. 4.2). As we need to compute n columns, the total effort required for obtaining $\mathbf{L}^{-1}$ is $\mathcal{O}(nN^2)$.

Finally, recall that both $\mathbf{B}_{t|t-1}$ and $\mathbf{L}^{-1}$ have $\mathcal{O}(N)$ elements in each row and that, by Proposition 3, their product, $\mathbf{B}_{t|t}$, has only $\mathcal{O}(nN)$ nonzero elements. Because each of these elements can be computed in $\mathcal{O}(N)$ time, the total computation cost of this step is $\mathcal{O}(nN^2)$.

To summarize, all three matrices necessary in the update step can be obtained in $\mathcal{O}(nN^2)$ time. Thus, we showed that both steps of Algorithm 1 require $\mathcal{O}(nN^2)$ time, which completes the proof. □

Proof of Proposition 7. For $m = 1, \ldots, M$, define $\mathbf{B}^{0:m} = (\mathbf{B}^m, \ldots, \mathbf{B}^0)$ as the submatrix of $\mathbf{B}$ consisting of the column blocks corresponding to resolutions $0, \ldots, m$. To show that $\mathbf{B}\mathbf{B}' \in H_M^r$, we prove by induction over $m = 1, \ldots, M$ that $(\mathbf{B}^{0:m}\mathbf{B}^{0:m})' \in H_m^r$. For $m = 1$, we have $\mathbf{B}^{0:1} =$

$\begin{bmatrix} \mathbf{B}_1 & \mathbf{0} & \mathbf{B}_{01} \\ \mathbf{0} & \mathbf{B}_2 & \mathbf{B}_{02} \end{bmatrix}$, where $\mathbf{B}_{01}$ and $\mathbf{B}_{02}$ are each r columns wide. Thus,

$$\mathbf{B}^{0:1}(\mathbf{B}^{0:1})' = \begin{bmatrix} \mathbf{B}_{01}\mathbf{B}_{01}' + \mathbf{B}_1\mathbf{B}_1' & \mathbf{B}_{01}\mathbf{B}_{02}' \\ \mathbf{B}_{02}\mathbf{B}_{01}' & \mathbf{B}_{02}\mathbf{B}_{02}' + \mathbf{B}_2\mathbf{B}_2' \end{bmatrix}.$$

and so $\mathbf{B}^{0:1}(\mathbf{B}^{0:1})' \in H_1^r$.

Now, assume that $\mathbf{B}^{0:m-1}(\mathbf{B}^{0:m-1})' \in H_{m-1}^r$. We have

$$\mathbf{B}^{0:m}(\mathbf{B}^{0:m})' = \sum_{j=0}^{m} \mathbf{B}^j(\mathbf{B}^j)' = \sum_{j=0}^{m-1} \mathbf{B}^j(\mathbf{B}^j)' + \mathbf{B}^m(\mathbf{B}^m)' = \mathbf{B}^{0:m-1}(\mathbf{B}^{0:m-1})' + \mathbf{B}^m(\mathbf{B}^m)'.$$

Next observe that for any k, the matrix $\mathbf{B}^k$ is block-diagonal, which means that $\mathbf{B}^k(\mathbf{B}^k)'$ is also block-diagonal with dense blocks $\mathbf{B}^k(\mathbf{B}^k)'[\mathcal{I}_{j_1,\ldots,j_k}, \mathcal{I}_{j_1,\ldots,j_k}]$. However, recursive partitioning of the domain means that $\mathcal{I}_{j_1,\ldots,j_{k-1}} \supset \mathcal{I}_{j_1,\ldots,j_k}$. Therefore, if $k > j$, then blocks of $\mathbf{B}^k(\mathbf{B}^k)'$ are nested within the blocks of $\mathbf{B}^j(\mathbf{B}^j)'$. Since this holds also for $k = m - 1$ and $j = m$, it means that $\mathbf{B}^{1:m}(\mathbf{B}^{1:m})' \in H_m^r$. □

APPENDIX B

PROOFS OF RESULTS FROM CHAPTER 3

B.1 Glossary of graph theory terms

We briefly review here some graph terminology necessary for our exposition and proofs, following Lauritzen (1996).

If $a \to b$ then we say that a is a *parent* of b and, conversely, that b is a *child* of a. Moreover, if there is a sequence of distinct vertices $h_1, \ldots, h_k$ such that $h_i \to h_{i+1}$ or $h_i \leftarrow h_{i+1}$ for all $i < k$, then we say that $h_1, \ldots, h_k$ is a *path*. If all arrows point to the right, we say that h_k is a *descendant* of h_i for $i < k$, while each h_i is an *ancestor* of h_k.

A *moral graph* is an undirected graph obtained from a DAG by first finding the pairs of parents of a common child that are not connected, adding an edge between them, and then by removing the directionality of all edges. If no edges need to be added to a DAG to make it moral, we call it a *perfect graph*.

Let $G = (V, E)$ be a directed graph with vertices V and edges E. If V_1 is a subset of V, then the *ancestral set* of V_1, denoted $\text{An}(V_1)$, is the smallest subset of V that contains V_1 and such that for each $v \in \text{An}(V_1)$ all ancestors of v are also in $\text{An}(V_1)$.

Finally, consider three disjoint sets of vertices A, B, C in an undirected graph. We say that C *separates* A and B if for every pair of vertices $a \in A$ and $b \in B$ the every path connecting a and b passes through C.

B.2 Proofs

Proof of Claim 1.

1. This proof is based on ideas in Schäfer et al. (2017). Split $\mathbf{w} = (w_1, \ldots, w_n)^\top$ into two

vectors, $\mathbf{u} = \mathbf{w}_{1:j-1}$ and $\mathbf{v} = \mathbf{w}_{j:n}$. Then,

$$\begin{aligned}
\mathbf{L} &= \mathbf{K}^{\frac{1}{2}} = \text{chol}\begin{pmatrix} \mathbf{K}_{uu} & \mathbf{K}_{uv} \\ \mathbf{K}_{vu} & \mathbf{K}_{vv} \end{pmatrix} \\
&= \text{chol}\left(\begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{K}_{vu}\mathbf{K}_{uu}^{-1} & \mathbf{I} \end{pmatrix}\begin{pmatrix} \mathbf{K}_{uu} & \mathbf{0} \\ \mathbf{0} & \mathbf{K}_{vv} - \mathbf{K}_{vu}\mathbf{K}_{uu}^{-1}\mathbf{K}_{uv} \end{pmatrix}\begin{pmatrix} \mathbf{I} & \mathbf{K}_{uu}^{-1}\mathbf{K}_{uv} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}\right) \\
&= \begin{pmatrix} \mathbf{K}_{uu}^{\frac{1}{2}} & \mathbf{0} \\ \mathbf{K}_{vu}\mathbf{K}_{uu}^{-\frac{1}{2}} & (\mathbf{K}_{vv} - \mathbf{K}_{vu}\mathbf{K}_{uu}^{-1}\mathbf{K}_{vu})^{\frac{1}{2}} \end{pmatrix}.
\end{aligned}$$

Note that $\mathbf{L}_{i,j}$ is the $(i-j+1)$-th element in the first column of $(\mathbf{K}_{vv} - \mathbf{K}_{vu}\mathbf{K}_{uu}^{-1}\mathbf{K}_{vu})^{\frac{1}{2}}$, which is the Cholesky factor of $\text{Var}(\mathbf{v}|\mathbf{u}) = \mathbf{K}_{vv} - \mathbf{K}_{vu}\mathbf{K}_{uu}^{-1}\mathbf{K}_{vu}$. Careful examination of the Cholesky factorization of a generic matrix $\mathbf{A}$, which is described in Algorithm 1 when setting $s_{i,j} = 0$ for all (i,j), shows that the computations applied to the first column of $\mathbf{A}$ are fairly simple. In particular, this implies that

$$\mathbf{L}_{i,j} = \big(\text{chol}(\text{Var}(\mathbf{v}|\mathbf{u}))\big)_{i-j+1,1} = \frac{\text{Cov}(w_i, w_j|\mathbf{w}_{1:j-1})}{\sqrt{\text{Var}(w_j|\mathbf{w}_{1:j-1})}},$$

because $w_j = \mathbf{v}_1$ and $w_i = \mathbf{v}_{i-j+1}$. Thus, $\mathbf{L}_{i,j} = 0 \iff \text{Cov}(w_i, w_j|\mathbf{w}_{1:j-1}) = 0 \iff w_i \perp w_j \,|\, \mathbf{w}_{1:j-1}$ because $\mathbf{w}$ was assumed to be jointly normal.

2. Thm. 12.5 in Rue and Held (2010) implies that for a Cholesky factor $\breve{\mathbf{U}}$ of a precision matrix $\mathbf{P}\mathbf{K}^{-1}\mathbf{P}$ of a normal random vector $\breve{\mathbf{w}} = \mathbf{P}\mathbf{w}$, we have $\breve{\mathbf{U}}_{i,j} = 0 \iff \breve{w}_i \perp \breve{w}_j \,|\, \{\breve{\mathbf{w}}_{j+1:i-1}, \breve{\mathbf{w}}_{i+1:n}\}$. Equivalently, because $\mathbf{U} = \mathbf{P}\breve{\mathbf{U}}\mathbf{P}$, we conclude that $\mathbf{U}_{j,i} = 0 \iff w_i \perp w_j \,|\, \{\mathbf{w}_{1:j-1}, \mathbf{w}_{j+1:i}\}$.

□

Proof of Proposition 8. The fact that $\hat{p}(\mathbf{x})$ is jointly normal holds for any Vecchia approximation (e.g., Datta et al., 2016a; Katzfuss et al., 2017, Prop. 1).

1. First, note that $\mathbf{L}$ and $\mathbf{U}$ are lower- and upper-triangular matrices, respectively. Hence, we assume in the following that $j < i$ but $x_j \notin \mathcal{C}_i$, and then show the appropriate conditional-independence results.

   (a) By Claim 1, we only need to show that $x_i \perp x_j | \mathbf{x}_{1:j-1}$. Let G be the graph corresponding to factorization (3.3) and denote by $G^m_{\text{An}(A)}$ the moral graph of the ancestral set of A. By Corollary 3.23 in Lauritzen (1996), it is enough to show that $\{x_1, \ldots, x_{j-1}\}$ separates x_i and x_j in $G^m_{An(\{x_1,\ldots,x_j,x_i\})}$. In the rest of the proof we label each vertex by its index, to simplify notation.

   We make three observations which can be easily verified. [1] $\text{An}\left(\{1, \ldots, j, i\}\right) \subset \{1, \ldots, i\}$; [2] Given the ordering of variables described in Section 3.3.1, if $k \to l$ then $k < l$; [3] G is a perfect graph, so $G^m_{An(\{1,\ldots,j,i\})}$ is a subgraph of G after all edges are turned into undirected ones.

   We now prove Proposition 8.1.a by contradiction. Assume that $\{x_1, \ldots, x_{j-1}\}$ does not separate x_i and x_j, which means that there exists a path $(h_1, \ldots, h_k)$ in $\{x_1, \ldots, x_i\}$ connecting x_i and x_j such that $h_k \in \text{An}(\{1, \ldots, j, i\})$ and $j + 1 \leq h_k \leq i - 1$.

   There are four cases we need to consider and we show that each one of them leads to a contradiction. First, assume that the last edge in the path is $h_k \to j$. This violates observation [2]. Second, assume that the first edge is $i \leftarrow h_1$. But because of [1] we know that $h_1 < i$, and by [2] we get a contradiction again. Third, let the path be of the form $i \to h_1 \leftarrow \cdots \leftarrow h_k \leftarrow j$ (i.e., all edges are of the form $h_r \leftarrow h_{r+1}$). However, this would mean that $\mathcal{X}_{j_1,\ldots,j_\ell} \subset \mathcal{A}_{i_1,\ldots,i_m}$, for $x_i \in \mathcal{X}_{i_1,\ldots,i_m}$ and $x_j \in \mathcal{X}_{j_1,\ldots,j_\ell}$. This implies that $j \in \mathcal{C}_i$, which in turn contradicts the assumption of the proposition. Finally, the only possibility we have not excluded yet is a path such that $i \leftarrow h_1 \ldots h_k \leftarrow j$ with some edges of the form $h_r \to h_{r+1}$. Consider the largest r for which this is true. Then by [3] there has to exist an edge $h_r \leftarrow h_p$ where $h_p \in \{h_{r+2}, \ldots, h_k, j\}$. But this means that j is an ancestor of h_r so the path can be reduced to $i \to h_1, \ldots h_r \to j$. We continue in this way for each edge "$\leftarrow$" which

reduces this path to case 3 and leads to a contradiction.

Thus we showed that all paths in $G^m_{An(\{1,\dots,j,i\})}$ connecting i and j necessarily have be contained in $\{1,\dots,j-1\}$, which proves Proposition 8.1.a.

(b) Like in part (a), we note that by Claim 1 it it enough to show that $x_i \perp x_j | \mathbf{x}_{1:j-1,j+1:i-1}$. Therefore, proceeding in a way similar to the previous case, we need to show that $\{1,\dots,j-1,j+1,\dots,i\}$ separates i and j in $G^m_{An(\{1,\dots,i\})}$. However, notice that it can be easily verified that $\text{An}(\{1,\dots,i\}) \subset \{1,\dots,i\})$, which means that $\text{An}(\{1,\dots,i\}) = \{1,\dots,i\})$. Moreover, observe that the subgraph of G generated by $\text{An}(\{1,\dots,i\})$ is already moral, which means that if two vertices did not have a connecting edge in the original DAG, they also do not share an edge in $G^m_{1,\dots,i}$. Thus i and j are separated by $\{1,\dots,j-1,j+1,\dots,i-1\}$ in $G^m_{1,\dots,i}$, which by Corollary 3.23 in (Lauritzen, 1996) proves part (b).

2. Let $\mathbf{P}$ be the reverse-ordering permutation matrix. Let $\mathbf{B} = \text{chol}(\mathbf{P}\hat{\mathbf{\Sigma}}^{-1}\mathbf{P})$. Then $\mathbf{U} = \mathbf{PBP}$. By the definition of $\mathbf{B}$, we know that $\mathbf{BB}^\top = \mathbf{P}\hat{\mathbf{\Sigma}}^{-1}\mathbf{P}$, and consequently $\mathbf{PBB}^\top\mathbf{P} = \hat{\mathbf{\Sigma}}^{-1}$. Therefore, $\hat{\mathbf{\Sigma}} = (\mathbf{PBB}'\mathbf{P})^{-1}$. However, we have $\mathbf{PP} = \mathbf{I}$ and $\mathbf{P} = \mathbf{P}^\top$, and hence $\hat{\mathbf{\Sigma}} = \big((\mathbf{PBP})(\mathbf{PB}^\top\mathbf{P})\big)^{-1}$. So we conclude that $\hat{\mathbf{\Sigma}} = (\mathbf{UU}^\top)^{-1} = (\mathbf{U}^\top)^{-1}\mathbf{U}^{-1} = (\mathbf{U}^{-1})^\top\mathbf{U}^{-1}$ and $(\mathbf{U}^{-1})^\top = \mathbf{L}$, or alternatively $\mathbf{L}^{-\top} = \mathbf{U}$.

□

Proof of Proposition 9.

1. We observe that hierarchical Vecchia satisfies the sparse general Vecchia requirement specified in Katzfuss and Guinness (2019, Sect. 4), because the nested ancestor sets imply that $\mathcal{C}_j \subset \mathcal{C}_i$ for all $j < i$ with $i, j \in \mathcal{C}_k$. Hence, reasoning presented in Katzfuss and Guinness (2019, proof of Prop. 6) allow us to conclude that $\widetilde{\mathbf{U}}_{j,i} = 0$ if $\mathbf{U}_{j,i} = 0$.

2. As the observations in $\mathbf{y}$ are conditionally independent given $\mathbf{x}$, we have

$$\hat{p}(\mathbf{x}|\mathbf{y}) \propto \hat{p}(\mathbf{x})p(\mathbf{y}|\mathbf{x}) = \big(\prod_{i=1}^{n} p(x_i|\mathcal{C}_i)\big)\big(\prod_{i\in\mathcal{I}} p(y_i|x_i)\big), \tag{B.1}$$

Let G be the graph representing factorization (3.3), and let $\tilde{G}$ be the DAG corresponding to (B.1). We order vertices in $\tilde{G}$ such that vertices corresponding to $\mathbf{y}$ have numbers $n+1, n+2, \ldots, n+|\mathcal{I}|$. For easier notation we also define $\tilde{\mathcal{I}} = \{i+n : i \in \mathcal{I}\}$. Similar to the proof of Proposition 8.1, we suppress the names of variables and use the numbers of vertices instead (i.e., we refer to the vertex x_k as k and y_j as $n+j$). Using this notation, and following the proof of Proposition 1, it is enough to show that $\{1, \ldots, j-1\} \cup \tilde{\mathcal{I}}$ separate i and j in $\tilde{\mathcal{G}}^m$, where $\tilde{\mathcal{G}} := G_{\mathrm{An}(\{1,\ldots,j,i\}\cup\tilde{\mathcal{I}})}$.

We first show that $1, \ldots, j-1$ separate i and j in G. Assume the opposite, that there exists a path $(h_1, \ldots, h_k)$ in $\{j+1, \ldots, i-1, i+1, \ldots, n\}$ connecting x_i and x_j. Let us start with two observations. First, note that the last arrow has to go toward j (i.e., $h_k \to j$), because $h_k > j$. Second, let $p_0 = \max\{p < k : h_p \to h_{p+1}\}$, the index of the last vertex with an arrow pointing toward j that is not h_k. If p_0 exists, then $(h_1, \ldots, h_{p_0})$ is also a path connecting i and j. This is because h_{p_0} and j are parents of h_{p_0+1}, and so $h_{p_0} \to j$, because G is perfect and $h_p > j$.

Now notice that a path (h_1) (i.e., one consisting of a single vertex) cannot exist, because we would either have $i \to h_1 \to j$ or $i \leftarrow h_1 \leftarrow j$. The first case implies that $i \to j$, because in G a node is a direct parent of all its descendants. Similarly in the second case, because G is perfect and $j < i$, we also have that $i \leftarrow j$. In either case the assumption $j \notin \mathcal{C}_i$ is violated.

Now consider the general case of a path $(h_1, \ldots, h_k)$ and recall that by observation 1 also $h_k \leftarrow j$. But then the path (h_1) also exists because by 3.3 all descendants are also direct children of their ancestors. As shown in the previous paragraph, we thus have a contradiction.

Finally, consider the remaining case such that $p_0 = \max\{p : h_p \to h_{p+1}\}$ exists. But

then $(h_1, \ldots, h_{p_0})$ is also a path connecting i and j. If all arrows in this reduced paths are to the left, we already showed it produces a contradiction. If not, we can again find $\max\{p : h_p \to h_{p+1}\}$ and continue the reduction until all arrows are in the same direction, which leads to a contradiction again.

Thus we show that i and j are separated by $\{1, \ldots, j-1\}$ in G. This implies that they are separated by this set in every subgraph of G that contains vertices $\{1, \ldots, j, i\}$ and in particular in $\mathcal{G} := G_{An(\{1,\ldots,j-1\}\cup\{j\}\cup\{i\}\cup\tilde{\mathcal{I}})}$. Recall that we showed in Proposition 1 that G is perfect, which means that $\mathcal{G} = \mathcal{G}^m$.

Next, for a directed graph $\mathcal{F} = (V, E)$ define the operation of *adding a child* as extending $\mathcal{F}$ to $\tilde{\mathcal{F}} = (V \cup \{w\}, E \cup \{v \to w\})$ where $v \in V$. In other words, we add one vertex and one edge such that one of the old vertices is a parent and the new vertex is a child. Note that a perfect graph with an added child is still perfect. Moreover, because the new vertex is connected to only a single existing one, adding a child does not create any new connections between the old vertices. It follows that if C separates A and B in $\mathcal{F}$, then $C \cup \{w\}$ does so in $\bar{\mathcal{F}}$ as well.

Finally, notice that $\tilde{\mathcal{G}}$, the graph we are ultimately interested in, can be obtained from $\mathcal{G}$ using a series of child additions. Because these operations preserve separation even after adding the child to the separating set, we conclude that i and j are separated by $\{1, \ldots, j-1\} \cup \tilde{\mathcal{I}}$ in $\tilde{\mathcal{G}}$. Moreover, because $\mathcal{G}$ was perfect and because graph perfection is preserved under child addition, we have that $\tilde{\mathcal{G}} = \tilde{\mathcal{G}}^m$.

□

CLAIM 2. *Assuming the joint distribution $\hat{p}(\mathbf{x})$ as in* (3.4)*, we have $\hat{p}(x_i, x_j) = p(x_i, x_j)$ if $x_j \in \mathcal{C}_i$; that is, the marginal bivariate distribution of a pair of variables is exact if one of the variables is in the conditioning set of the other.*

Proof. First, consider the case where $x_i, x_j \in \mathcal{X}_{j_1,\ldots,j_m}$. Then note that $\hat{p}(\mathcal{X}_{j_1,\ldots,j_m}) = \int \hat{p}(\mathbf{x}) d\mathbf{x}_{-\mathcal{X}_{j_1,\ldots,j_m}}$. Furthermore, notice that given the decomposition (3.3) and combining appropriate terms we can

write

$$\hat{p}(\mathbf{x}) = p\left(\mathcal{X}_{j_1,\ldots,j_m}|\mathcal{A}_{j_1,\ldots,j_m}\right) p\left(\mathcal{A}_{j_1,\ldots,j_m}\right) p\left(\mathbf{x}_{-\left\{\mathcal{X}_{j_1,\ldots,j_m}\cup\mathcal{A}_{j_1,\ldots,j_m}\right\}}\right).$$

Using these two observations, we conclude that

$$\hat{p}(\mathcal{X}_{j_1,\ldots,j_m}) = \int \hat{p}(\mathbf{x}) d\mathbf{x}_{-\mathcal{X}_{j_1,\ldots,j_m}} = \int \prod_{k=0}^{m} p(\mathcal{X}_{j_1,\ldots,j_m}|\mathcal{A}_{j_1,\ldots,j_m}) d(\mathcal{X}, \mathcal{X}_{j_1}, \ldots, \mathcal{X}_{j_1,\ldots,j_{m-1}}) = p(\mathcal{X}_{j_1,\ldots,j_m}),$$

which proves that $\hat{p}(x_i, x_j) = p(x_i, x_j)$ if $x_i, x_j \in \mathcal{X}_{j_1,\ldots,j_m}$.

Now let $x_i \in \mathcal{X}_{j_1,\ldots,j_m}$ and $x_j \in \mathcal{X}_{j_1,\ldots,j_\ell}$ with $\ell < m$, because $x_j \in \mathcal{C}_i$ implies that $j < i$. Then,

$$\begin{aligned}
\hat{p}(\mathcal{X}_{j_1,\ldots,j_m}, \mathcal{X}_{j_1,\ldots,j_\ell}) &= \int \hat{p}(\mathbf{x}) d\mathbf{x}_{-\{\mathcal{X}_{j_1,\ldots,j_m}\cup\mathcal{X}_{j_1,\ldots,j_\ell}\}} = \\
&= \int \prod_{k=0}^{M} \prod_{j_1,\ldots,j_k} p(\mathcal{X}_{j_1,\ldots,j_k}|\mathcal{A}_{j_1,\ldots,j_k}) d\mathbf{x}_{-\{\mathcal{X}_{j_1,\ldots,j_m}\cup\mathcal{X}_{j_1,\ldots,j_\ell}\}} \\
&= \int \prod_{k=0}^{m} p(\mathcal{X}_{j_1,\ldots,j_k}|\mathcal{A}_{j_1,\ldots,j_k}) d(A_{j_1,\ldots,j_\ell} \cup \mathcal{X}_{j_1,\ldots,j_{\ell+1}} \cup \cdots \cup \mathcal{X}_{j_1,\ldots,j_{m-1}}).
\end{aligned}$$

The second equality uses (3.3), the definition of $\hat{p}$; the last equation is obtained by integrating out $\mathcal{X}^{0:M} \setminus \bigcup_{k=0}^{m} \mathcal{X}_{j_1,\ldots,j_k}$. Note that $\mathcal{A}_{j_1,\ldots,j_m} = \mathcal{A}_{j_1,\ldots,j_\ell} \cup \bigcup_{k=\ell}^{m-1} \mathcal{X}_{j_1,\ldots,j_k}$. Therefore, by Bayes law, for any $k > \ell$:

$$\begin{aligned}
p(\mathcal{X}_{j_1,\ldots,j_k}|\mathcal{A}_{j_1,\ldots,j_k}) &= p\left(\mathcal{X}_{j_1,\ldots,j_k}|\mathcal{A}_{j_1,\ldots,j_\ell} \cup (\mathcal{A}_{j_1,\ldots,j_k} \setminus \mathcal{A}_{j_1,\ldots,j_\ell})\right) \\
&= \frac{p\left(\mathcal{A}_{j_1,\ldots,j_\ell}|\mathcal{X}_{j_1,\ldots,j_k}\cup(\mathcal{A}_{j_1,\ldots,j_k}\setminus\mathcal{A}_{j_1,\ldots,j_\ell})\right) p(\mathcal{X}_{j_1,\ldots,j_k}|\mathcal{A}_{j_1,\ldots,j_k}\setminus\mathcal{A}_{j_1,\ldots,j_\ell})}{p(\mathcal{A}_{j_1,\ldots,j_\ell}|\mathcal{A}_{j_1,\ldots,j_k}\setminus\mathcal{A}_{j_1,\ldots,j_\ell})} \\
&= \frac{p(\mathcal{A}_{j_1,\ldots,j_\ell}|\mathcal{A}_{j_1,\ldots,j_{k+1}}\setminus\mathcal{A}_{j_1,\ldots,j_\ell}) p(\mathcal{X}_{j_1,\ldots,j_k}|\mathcal{A}_{j_1,\ldots,j_k}\setminus\mathcal{A}_{j_1,\ldots,j_\ell})}{p(\mathcal{A}_{j_1,\ldots,j_\ell}|\mathcal{A}_{j_1,\ldots,j_k}\setminus\mathcal{A}_{j_1,\ldots,j_\ell})} = (*)
\end{aligned}$$

The last equality holds because $\mathcal{X}_{j_1,\ldots,j_m} \cup \mathcal{A}_{j_1,\ldots,j_m} = \mathcal{A}_{j_1,\ldots,j_{m+1}}$. As a consequence

$$\begin{aligned}
\prod_{k=0}^{m} p(\mathcal{X}_{j_1,\ldots,j_k}|\mathcal{A}_{j_1,\ldots,j_k}) &= \prod_{k=0}^{m} p(\mathcal{X}_{j_1,\ldots,j_k}|\mathcal{A}_{j_1,\ldots,j_k} \setminus \mathcal{A}_{j_1,\ldots,j_\ell}) p(\mathcal{A}_{j_1,\ldots,j_\ell}) \\
&= \prod_{k=0}^{m} p(\mathcal{X}_{j_1,\ldots,j_k}|\bigcup_{s=k-1}^{\ell} \mathcal{X}_{j_1,\ldots,j_s}) p(\mathcal{A}_{j_1,\ldots,j_\ell})
\end{aligned}$$

and

$$
\begin{aligned}
(*) &= \int \textstyle\prod_{k=0}^{m} p(\mathcal{X}_{j_1,\ldots,j_m} | \mathcal{A}_{j_1,\ldots,j_m}) p(\mathcal{A}_{j_1,\ldots,j_\ell}) d(\mathcal{A}_{j_1,\ldots,j_\ell} \cup \mathcal{X}_{j_1,\ldots,j_{\ell+1}} \cup \ldots, \cup \mathcal{X}_{j_1,\ldots,j_{m-1}}) \\
&= \int p(\mathcal{X}_{j_1,\ldots,j_m}, \mathcal{X}_{j_1,\ldots,j_{m-1}}, \ldots, \mathcal{X}_{j_1,\ldots,j_\ell}, \mathcal{A}_{j_1,\ldots,j_\ell}) d(\mathcal{A}_{j_1,\ldots,j_\ell} \cup \mathcal{X}_{j_1,\ldots,j_{\ell+1}} \cup \ldots, \cup \mathcal{X}_{j_1,\ldots,j_{m-1}}) \\
&= p(\mathcal{X}_{j_1,\ldots,j_m}, \mathcal{X}_{j_1,\ldots,j_\ell})
\end{aligned}
$$

This means that $\hat{p}(\mathcal{X}_{j_1,\ldots,j_m}, \mathcal{X}_{j_1,\ldots,j_\ell}) = p(\mathcal{X}_{j_1,\ldots,j_m}, \mathcal{X}_{j_1,\ldots,j_\ell})$, or that the marginal distribution of $\mathcal{X}_{j_1,\ldots,j_m}$ and $\mathcal{X}_{j_1,\ldots,j_\ell}$ in (3.3) is the same as in the true distribution p. Because p is Gaussian, it follows that $\hat{p}(x_i, x_j) = p(x_i, x_j)$. This ends the proof.

□

Proof of Proposition 10. We use $l^{\text{inc}}_{i,j}$, $l_{i,j}$, $\sigma_{i,j}$, $\hat{\sigma}_{i,j}$ to denote the (i,j)-th elements of $\mathbf{L}^{\text{inc}} = \text{ichol}(\boldsymbol{\Sigma}, \mathbf{S})$, $\mathbf{L} = \text{chol}(\hat{\boldsymbol{\Sigma}})$, $\boldsymbol{\Sigma}$, $\hat{\boldsymbol{\Sigma}}$, respectively. It can be seen easily in Algorithm 1 that $\text{chol}(\boldsymbol{\Sigma}) = \text{ichol}(\boldsymbol{\Sigma}, \mathbf{S^1})$, where $\mathbf{S}^{\mathbf{1}}_{i,j} = 1$ for $i \geq j$ and 0 otherwise.

We prove that $l^{\text{inc}}_{i,j} = l_{i,j}$ by induction over the elements of the Cholesky factor, following the order in which they are computed. First, we observe that $l^{\text{inc}}_{1,1} = l_{1,1}$. Next, consider the computation of the (i,j)-th entry, assuming that we have $l^{\text{inc}}_{k,q} = l_{k,q}$ for all previously computed entries. According to Algorithm 1, we have

$$
l_{i,j} = \tfrac{1}{l_{j,j}}\big(\hat{\sigma}_{i,j} - \textstyle\sum_{k=1}^{j-1} l_{i,k} l_{j,k}\big), \qquad l^{inc}_{i,j} = \tfrac{s_{i,j}}{l_{j,j}}\big(\sigma_{i,j} - \textstyle\sum_{k=1}^{j-1} l_{i,k} l_{j,k}\big).
$$

Now, if $s_{i,j} = 1 \iff x_j \in \mathcal{C}_i$, then Claim 2 tells us that $\sigma_{i,j} = \hat{\sigma}_{i,j}$, and hence $l_{i,j} = l^{\text{inc}}_{i,j}$. If $s_{i,j} = 0 \iff x_j \notin \mathcal{C}_i$, then $l^{\text{inc}}_{i,j} = 0$, and also $l_{i,j} = 0$ by Proposition 8.1(a). This completes the proof via induction. □

CLAIM 3. *Let* $\mathbf{x}$ *have density* $\hat{p}(\mathbf{x})$ *as in* (3.3)*, and let each conditioning set* $\mathcal{C}_i$ *have size at most* N*. Then* $\boldsymbol{\Lambda}$*, the precision matrix of* $\mathbf{x}$*, has* $\mathcal{O}(nN)$ *nonzero elements. Moreover, the columns of* $\mathbf{U} = \text{rchol}(\boldsymbol{\Lambda})$ *and the rows of* $\mathbf{L} = \text{chol}(\boldsymbol{\Lambda}^{-1})$ *each have at most* N *nonzero elements.*

Proof. Because the precision matrix is symmetric, it is enough to show that there are only $\mathcal{O}(nN)$

nonzero elements $\mathbf{\Lambda}_{i,j}$ in the upper triangle (i.e., with $i < j$). Let $x_i \notin \mathcal{C}_j$. This means that $x_i \not\to x_j$ and because by (3.3) all edges go from a lower index to a higher index, x_i and x_j are not connected. Moreover because $\mathcal{A}_{j_1,\ldots,j_m} \supset \mathcal{A}_{j_1,\ldots,j_{m-1}}$ there is also no x_k with $k > \max(i,j)$ such that $x_i \to x_k$ and $x_j \to x_k$. Thus, using Proposition 3.2 in Katzfuss and Guinness (2019) we conclude that $\mathbf{\Lambda}_{i,j} = 0$ for $x_i \notin \mathcal{C}_j$. This means that each row i has at most $|\mathcal{C}_i| \leq N$ nonzero elements, and the entire lower triangle has $\mathcal{O}(nN)$ nonzero values.

Proposition 8 implies that the i-th column of $\mathbf{U}$ has at most as many nonzero entries as there elements in the conditioning set $\mathcal{C}_i$, which we assumed to be of size at most N. Similarly, the i-th row of $\mathbf{L}$ has at most as many nonzero entries as the number of elements in the conditioning set $\mathcal{C}_i$. □

CLAIM 4. *Let $\mathbf{A}$ be an $n \times n$ lower triangular matrix with at most $N < n$ nonzero elements in each row at known locations. Letting a_{ij} and $\tilde{a}_{ij}$ be the (i,j)-th element of $\mathbf{A}$ and $\mathbf{A}^{-1}$, respectively, assume that $\tilde{a}_{i,j} = 0$ if $a_{i,j} = 0$. Then, the cost of calculating $\mathbf{A}^{-1}$ from $\mathbf{A}$ is $\mathcal{O}(nN^2)$.*

Proof. Notice that calculating $\tilde{\mathbf{a}}_k$, the kth column of $\mathbf{A}^{-1}$, is equivalent to solving a linear system of the form

$$\begin{bmatrix} a_{11} & 0 & 0 & \ldots & 0 \\ a_{21} & a_{22} & 0 & \ldots & 0 \\ a_{31} & a_{32} & a_{33} & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \ldots & a_{nn} \end{bmatrix} \begin{bmatrix} \tilde{a}_{1k} \\ \tilde{a}_{2k} \\ \tilde{a}_{3k} \\ \vdots \\ \tilde{a}_{nk} \end{bmatrix} = e_k,$$

where $e_{ik} = 1$ if $k = i$ and 0 otherwise. Using forward substitution, the i-th element of $\tilde{\mathbf{a}}_k$ can be calculated as $\tilde{a}_{ik} = \frac{1}{a_{kk}} \left(e_{ik} - \sum_{j=1}^{i-1} a_{ij} \tilde{a}_{jk} \right)$. This requires $\mathcal{O}(N)$ time, because our assumptions imply that there are at most N nonzero terms under the summation. Moreover, we also assumed that $\tilde{\mathbf{a}}_k$ has at most N nonzero elements at known locations, and so we only need to calculate those. Thus computing $\tilde{\mathbf{a}}_k$ has $\mathcal{O}(N^2)$ time complexity. As there are n columns to calculate, this ends the proof. □

Proof of Proposition 11. Starting with the ichol() procedure in Line 1, obtaining each nonzero element $l_{i,j}$ requires calculating the outer product of previously computed segments of rows i and j. Because Claim 3 implies that each row of $\mathbf{L}$ has at most N nonzero elements, obtaining $l_{i,j}$ is $\mathcal{O}(N)$. Claim 3 also shows that for each i, there are at most N nonzero elements $s_{i,j} = 1$, which implies that each row of the incomplete Cholesky factor can be calculated in $\mathcal{O}(N^2)$. Finally, because the matrix to be decomposed has n rows, the overall cost of the algorithm is $\mathcal{O}(nN^2)$. In Line 2, because $\mathbf{L}^{-1} = \mathbf{U}^\top$, Proposition 8 tells us exactly which elements of $\mathbf{L}^{-1}$ need to be calculated (i.e., are non-zero), and that there are only N of them (Claim 3). Using Claim 4, this means that computing $\mathbf{L}^{-1}$ can be accomplished in $\mathcal{O}(nN^2)$ time. Analogous reasoning and Proposition 2 allow us to conclude that computing $\widetilde{\mathbf{L}}$ in Line 5 has the same complexity. The cost of Line 3 is dominated by taking the outer product of $\mathbf{U}$, because $\mathbf{H}$ and $\mathbf{R}$ are assumed to have only one non-zero element in each row. However, $\mathbf{U}\mathbf{U}^\top$ is by definition equal to the precision matrix of $\mathbf{x}$ under (3.3). Therefore, by Claim 3 there are at most $\mathcal{O}(nN)$ elements to calculate and each requires multiplication of two rows with at most N nonzero elements. This means that this step can be accomplished in $\mathcal{O}(nN^2)$ time. The most expensive operation in Line 4 is taking the Cholesky factor. However, its cost proportional to the square of the number of nonzero elements in each column (e.g., Toledo, 2007, Thm. 2.2), which by Claim 3 we know to be N. As there are n columns, this step requires $\mathcal{O}(nN^2)$ time. Finally, the most expensive operation in Line 6 is the multiplication of a vector by matrix $\widetilde{\mathbf{L}}$. By Proposition 9, $\widetilde{\mathbf{L}}$ has the same number of nonzero elements per row as $\mathbf{L}$, which is at most N by Claim 3. Thus, multiplication of $\widetilde{\mathbf{L}}$ and any dense vector can be performed in $\mathcal{O}(nN)$ time. To conclude, each line of Algorithm 2 can be computed in at most $\mathcal{O}(nN^2)$ time, and so the total time complexity of the algorithm is also $\mathcal{O}(nN^2)$.

Regarding memory complexity, notice that by Claims 3 and 8, matrices $\mathbf{L}$, $\mathbf{U}$, $\widetilde{\mathbf{L}}$, $\widetilde{\mathbf{U}}$, and $\mathbf{\Lambda}$ have $\mathcal{O}(nN)$ nonzero elements, and (3.1) implies that matrices $\mathbf{H}$ and $\mathbf{R}$ have at most n entries. Further, the incomplete Cholesky decomposition in Line 1 requires only those elements of $\mathbf{\Sigma}$ that correspond to the nonzero elements of $\mathbf{S}$. Because $\mathbf{S}$ has at most N non-zero elements in each row by construction, each of the matrices that are decomposed can be stored using $\mathcal{O}(nN)$ memory,

and so the memory requirement for Algorithm 2 is $\mathcal{O}(nN)$. □

APPENDIX C

SUPPLEMENTARY MATERIAL

C.1 Proof of the exactness of MRF when $r_0 = n_{\mathcal{S}}$

In this section, we discuss one of the settings mentioned in Section 2.4.1 under which Algorithm 2.3.4.2 (MRD) is exact. Because the only approximation made in Algorithm 1 is

$$\boldsymbol{\Sigma}_{t|t-1} \approx \mathbf{B}\mathbf{B}' \tag{C.1}$$

where $\mathbf{B} = \mathrm{MRD}(\boldsymbol{\Sigma}^F_{t|t-1})$, the MRF is exact whenever (C.1) holds with equality. As stated in Section 2.4.1, this is the case when $M = 0$ and $r_0 = n_{\mathcal{S}}$. Then Algorithm 2.3.4.2 reduces to computing $\mathbf{W} = \boldsymbol{\Sigma}^F_{t|t-1}$ and $\mathbf{V} = \boldsymbol{\Sigma}^F_{t|t-1}$. Hence, we have

$$\mathbf{B} = \mathbf{B}^0 = \mathbf{W}\mathbf{V}^{-\frac{1}{2}}$$

and

$$\boldsymbol{\Sigma}_{t|t-1} = \mathbf{B}\mathbf{B}' = \mathbf{W}\mathbf{V}^{-\frac{1}{2}}\mathbf{V}^{-\frac{1}{2}}\mathbf{W}' = \boldsymbol{\Sigma}^F_{t|t-1}.$$

C.2 More on distributed computation

As discussed briefly in Section 2.4.3, the MRF algorithm is well suited to distributed computations. This can be achieved by extending the approach in Katzfuss (2017, Sect. 3.5), as outlined here. Let us consider a set up in which node $\mathcal{N}_{j_1,\ldots,j_m}$ only stores the submatrix $\mathbf{B}_{t|t}[\mathcal{I}_{j_1,\ldots,j_m}, \mathcal{K}_{j_1,\ldots,j_m}]$ and the mean vector $\boldsymbol{\mu}_{t|t}[\mathcal{I}_{j_1,\ldots,j_m}]$. In order to execute the forecast step, we need to multiply $\mathbf{B}_{t-1|t-1}$ and $\boldsymbol{\mu}_{t-1|t-1}$ by $\mathbf{E}_t[:, \mathcal{I}_{j_1,\ldots,j_m}]$. In general, information from other computational nodes might be needed for this operation, but in our case only a small amount of data is required, because we assumed $\mathbf{E}_t$ to be local (see Assumption 2). Similarly, the MRD decomposition can largely be performed locally as its main computational burden is in calculating

the $\mathbf{W}^l_{j_1,\ldots,j_m}$ matrices.

In the update step, computing $\boldsymbol{\mu}_{t|t}$ is accomplished by a series of matrix-vector multiplications. Under Assumption 1, calculating $\mathbf{H}_t'\mathbf{R}_t^{-1}(\mathbf{y}_t - \mathbf{H}_t\boldsymbol{\mu}_{t|t-1})$ can be executed separately on each node, provided the corresponding blocks of the $\mathbf{H}_t$ and $\mathbf{R}_t$ matrices are sent to same node. In order to obtain the $\mathbf{B}_{t|t}$ matrix, we need the Cholesky factor of $\prec$ and its product with $\mathbf{B}_{t|t-1}$. Computation of elements $\mathbf{L}_t[\mathcal{K}_{j_1,\ldots,j_m}, \mathcal{K}_{j_1,\ldots,j_k}]$, $m < k$, can be mostly handled by node $\mathcal{N}_{j_1,\ldots,j_m}$. Moreover, since it already holds the $\mathbf{B}_{t|t-1}[\mathcal{I}_{j_1,\ldots,j_m}, \mathcal{K}_{j_1,\ldots,j_m}]$ block, which it acquired in the forecast step, it can also be tasked with calculating the $\mathbf{B}_{t|t}[\mathcal{I}_{j_1,\ldots,j_m}, \mathcal{K}_{j_1,\ldots,j_m}]$ block, since that requires little communication with other nodes.

C.3 Connections to multi-resolution autoregressive models

The idea of the multi-resolution analysis of spatial stochastic processes has been explored in great depth (e.g., Chou et al., 1994a,b; Willsky, 2002; Frakt and Willsky, 2001; Ferreira and Lee, 2007; Choi et al., 2010), and gave rise to many fast algorithms in the field of signal processing (e.g., Chou et al., 1994a; Luettgen and Willsky, 1995). The general focus of this strand of literature was on scale-recursive state-space models on trees and using Kalman-filter-like inference to derive the distribution of variables corresponding to the nodes of the tree. Building on these developments for purely spatial domains, several authors (e.g., Huang et al., 2002; Johannesson et al., 2003; Tzeng et al., 2005) applied the multi-resolution paradigm in modelling spatio-temporal stochastic processes. These approaches can be expressed using a random-effects process with spatial basis functions that are either constant or step-wise.

In this section we show how the MRA, which powers the multi-resolution filter (see Section 2.5.2), can be described in these terms.

DEFINITION 2. (Frakt and Willsky, 2001) *Let $\mathcal{S}(\mathcal{V}, E)$ be a directed acyclic graph. For a given node v, let $\gamma(v)$ denote the parent node of v and let $x(v)$ be a random vector associated with v.*

Then $x(\cdot)$ is a zero-mean multi-resolution autoregressive process *(MAR) if*

$$x(v) = A(v)x(\gamma(v)) + w(v),$$

where $w(v)$ is white, uncorrelated with $x(w)$ for any $w \in \mathcal{V}$ and has autocovariance $Q(v)$.

To show that the MRA is a MAR, we start with the observation that if $\mathbf{\Sigma} \approx \mathbf{B}\mathbf{B}'$ and we take $\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, then for $\mathbf{x} := \mathbf{B}\boldsymbol{\eta}$ we have $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{B}\mathbf{B}')$. Now let us partition the vector $\mathbf{x}$ into segments whose sizes correspond to those of blocks $\mathbf{B}^1, \ldots \mathbf{B}^M$. We can then write

$$\mathbf{x} = \mathbf{B}\boldsymbol{\eta} = \mathbf{B}^0\boldsymbol{\eta}^0 + \mathbf{B}^1\boldsymbol{\eta}^1 + \cdots + \mathbf{B}^M\boldsymbol{\eta}^M$$

and define the level-m approximation as

$$\mathbf{x}^m = \mathbf{B}^0\boldsymbol{\eta}^0 + \cdots + \mathbf{B}^m\boldsymbol{\eta}^m.$$

In order to represent $\mathbf{x}$ in the MAR form, we also need to define the graph $\mathcal{F} = (\mathcal{V}, \mathcal{E})$ that will provide indexing for the elements of $\mathbf{x}$. Let

$$\mathcal{V} = \{v_0\} \cup \{v_1, \ldots, v_J\} \cup \{v_{11}, v_{12}, \ldots, v_{JJ}\} \cup \ldots$$

be the set of vertices. We associate each vertex with some subdomain in the hierarchy described in Section 2.3.4.1. For example, v_0 and $v_{j_1,\ldots,j_m}$ would correspond to $\mathcal{D}$ and $\mathcal{D}_{j_1,\ldots,j_m}$, respectively. Next, we define the set of edges $\mathcal{E}$ in a way that represents the domain partitioning hierarchy:

$$\mathcal{E} = \{v_0 \to v_j | j = 1, \ldots J\} \cup \bigcup_{j=1}^{J} \{v_j \to v_{jk} | k = 1, \ldots, J\} \cup \ldots$$

Figure C.1 illustrates what this graph might look like.

We can now associate vertex $v_{j_1,\ldots,j_m}$ with the m-th level approximation of the elements of

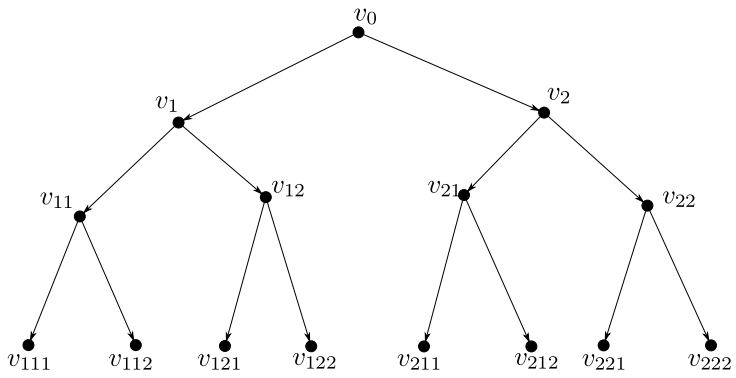


Figure C.1: Graph $\mathcal{F}$ for $M = 3$, $J = 2$

vector $\mathbf{x}$ corresponding to grid points in $\mathcal{D}_{j_1,\ldots,j_m}$. More precisely, using the notation of Definition 2, we can express it as

$$x(v_{j_1,\ldots,j_m}) = \mathbf{B}^0\boldsymbol{\eta}^0 + \mathbf{B}^1[\mathcal{I}_{j_1,\ldots,j_m}, \mathcal{K}_{j_1}]\,\boldsymbol{\eta}^1 + \mathbf{B}^1[\mathcal{I}_{j_1,\ldots,j_m}, \mathcal{K}_{j_1,j_2}]\,\boldsymbol{\eta}^2 + \cdots + \mathbf{B}^m[\mathcal{I}_{j_1,\ldots,j_m}, \mathcal{K}_{j_1,\ldots,j_m}]\,\boldsymbol{\eta}^m.$$

This implies that the scale-recursive equations analogous to (2) can then be written as

$$x(v_{j_1,\ldots,j_m}) = \mathbf{I} \cdot x(v_{j_1,\ldots,j_{m-1}}) + w(v_{j_1,\ldots,j_m}),$$

where $w(v_{j_1,\ldots,j_m}) = \mathbf{B}^m[\mathcal{I}_{j_1,\ldots,j_m}, \mathcal{K}_{j_1,\ldots,j_m}]\boldsymbol{\eta}^m$. From the definition of $\boldsymbol{\eta}$, it is clear that w are uncorrelated (and thus independent) between scales.

To conclude, we showed that the MRA can be expressed as a multi-resolution autoregressive model, which allows to use a Kalman smoother for inference (Ferreira and Lee, 2007, Ch. 8). Therefore, one can view the MRF as a scale-recursive Kalman smoother nested within each step of an outer Kalman filter, the latter proceeding along the time dimension.

C.4 More details on the numerical simulations in Section 2.7

C.4.1 Definition of scores

Kullback-Leibler (KL) divergence

KL divergence is often used as a way of quantifying how different two distributions are. In the context of the simulation study in Section 2.7, KL divergence measures how much the approximate filtering distribution differs from the exact distribution calculated by the Kalman filter. Formally, if f is the exact k-variate normal distribution with mean $\boldsymbol{\mu}_f$ and covariance matrix $\boldsymbol{\Sigma}_f$, and g_j is the normal distribution obtained using method j with mean $\boldsymbol{\mu}_j$ and covariance $\boldsymbol{\Sigma}_j$, then the KL divergence can be expressed as

$$\mathrm{KL}(f||g_j) = \frac{1}{2}\left(\mathrm{tr}\left(\boldsymbol{\Sigma}_j^{-1}\boldsymbol{\Sigma}_f\right) + (\boldsymbol{\mu}_j - \boldsymbol{\mu}_f)\,\boldsymbol{\Sigma}_j^{-1}\,(\boldsymbol{\mu}_j - \boldsymbol{\mu}_f) - k + \ln\left(\frac{\det \boldsymbol{\Sigma}_j}{\det \boldsymbol{\Sigma}_f}\right)\right).$$

In Section 2.7, we have $k = n_{\mathcal{S}} = 1{,}156$. For the KL divergence to be small, we need both the mean and the covariance matrix to be accurate (i.e., $\boldsymbol{\mu}_j \approx \boldsymbol{\mu}_f$ and $\boldsymbol{\Sigma}_j \approx \boldsymbol{\Sigma}_f$).

Root mean squared prediction error (RMSPE)

MSPE captures how much, on average, the predicted values differ from the true values. In our case, if $\boldsymbol{\mu}_j[i]$ denotes the value predicted by method j (i.e., the filtering mean) of process $\mathbf{x}$ at grid point i, and if $\mathbf{x}[i]$ is the true value at this point, RMSPE is calculated as follows:

$$\mathrm{RMSPE} = \sqrt{\frac{1}{n_{\mathcal{S}}}\sum_{i=1}^{n_{\mathcal{S}}}\left(\boldsymbol{\mu}_j[i] - \mathbf{x}[i]\right)^2},$$

where $n_{\mathcal{S}}$ is the number of grid points.

C.4.2 Circular domain

Consider a diffusion-advection model with an initial state drawn from a Gaussian process:

$$\begin{cases} \frac{\partial}{\partial t}x(s,t) = \alpha\frac{\partial}{\partial s}x(s,t) + \beta\frac{\partial^2}{\partial s^2}x(s,t) + \zeta(s,t) \\ x(s,0) \sim GP\left(0, K_{1d}(\cdot,\cdot)\right) \end{cases} . \tag{C.2}$$

We assume that $x : S \times [0,T] \to \mathbb{R}$ is the quantity of interest between $t = 0$ and $t = T$, over a one-dimensional sphere with unit circumference, ζ is a zero-mean stationary Gaussian process with an isotropic spatial covariance function $\sigma_w^2 C(\cdot,\cdot)$ and independent increments over time, and K_{1d} is the spatial covariance function of the process x at time $t = 0$, defined as $K_{1d}(s_1, s_2) = \mathcal{M}_{\nu,\lambda}\left(|s_1 - s_2| \mod 1\right)$.

We discretize both the spatial and the temporal domains using $n_{\mathcal{S}} = 80$ and $T = 20$ regularly spaced points, respectively. Applying first-order forward differences in time and centered differences in space as in Xu and Wikle (2007), we can approximate the derivatives in (C.2) with

$$\begin{aligned} \tfrac{\partial}{\partial t}x(s,t) &\approx \left(x(s,t+\Delta t) - x(s,t)\right)\tfrac{1}{\Delta t}, \\ \tfrac{\partial}{\partial s}x(s,t) &\approx \left(x(s+\Delta s,t) - x(s-\Delta s,t)\right)\tfrac{1}{2\Delta s} \\ \tfrac{\partial^2}{\partial s^2}x(s,t) &\approx \left(x(s+\Delta s,t) - 2x(s,t) + x(s-\Delta s,t)\right)\tfrac{1}{\Delta^2 s}. \end{aligned}$$

Then, taking $\Delta t = 1$ and $\Delta s = \frac{1}{n_{\mathcal{S}}}$, the first equation in (C.2) can be expressed as

$$x(s,t) = c_1 \cdot x(s,t-1) + c_2 \cdot x(s+\tfrac{1}{n_{\mathcal{S}}},t-1) + c_3 \cdot x(s-\tfrac{1}{n_{\mathcal{S}}},t-1)$$

with $c_1 = 1 - 2\beta n_{\mathcal{S}}^2$, $c_2 = 0.5\alpha n_{\mathcal{S}} + \beta n_{\mathcal{S}}^2$ and $c_3 = -0.5\alpha n_{\mathcal{S}} + \beta n_{\mathcal{S}}^2$. Following Xu and Wikle (2007), we use $\alpha = 0.5/n_{\mathcal{S}}, \beta = 0.35/n_{\mathcal{S}}^2$, which correspond to $c_1 = 0.3, c_2 = 0.6, c_3 = 0.1$. These parameters also ensure the stability of the scheme.

The discretization allows us to express the original model (C.2) as

$$\begin{cases} \mathbf{x}_t = \mathbf{E}_t \mathbf{x}_{t-1} + \mathbf{w}_t, \\ \mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}), \end{cases}$$

where $\mathbf{x}_t$ is a vector with values of $x(\cdot, t)$ at the spatial grid points, the evolution matrix

$$\mathbf{E}_t = \begin{bmatrix} c_1 & c_2 & & & & & & & c_3 \\ c_3 & c_1 & c_2 & & & & & & \\ & c_3 & c_1 & c_2 & & & & & \\ & & & & \ddots & & & & \\ & & & & & c_3 & c_1 & c_2 & \\ & & & & & & c_3 & c_1 & c_2 \\ c_2 & & & & & & & c_3 & c_1 \end{bmatrix}$$

is tri-diagonal with two non-zero elements in the bottom-left and top-right corners, the covariance matrix

$$\mathbf{\Sigma} = \left[\mathcal{M}_{\nu,\lambda}\left(|s_i - s_j| \mod 2\pi\right)\right]_{i,j=1,\dots n_{\mathcal{S}}}$$

is obtained by evaluating K_{1d} at all spatial grid points s_i, and $\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, \sigma_w^2 \mathbf{Q})$ with $\mathbf{Q} = \left[C(s_i, s_j)\right]_{i,j=1,\dots n_{\mathcal{S}}}$. We see that (C.4.2) has the same form as (3.13) in Section 2.2.1.

We also assume that at every time point t, we have $n_t < n_{\mathcal{S}}$ noisy observations in the vector $\mathbf{y}_t$, each corresponding to a grid point in $\mathcal{S}$. We represent this assumption using (2.1), where the matrix $\mathbf{H}_t$ is built by removing those rows from $\mathbf{I}_{n_{\mathcal{S}}}$ that correspond to grid points for which no data are available. Thus $\mathbf{H}_t$ is $n_t \times n_{\mathcal{S}}$. Examples of realizations are shown in Figure C.2.

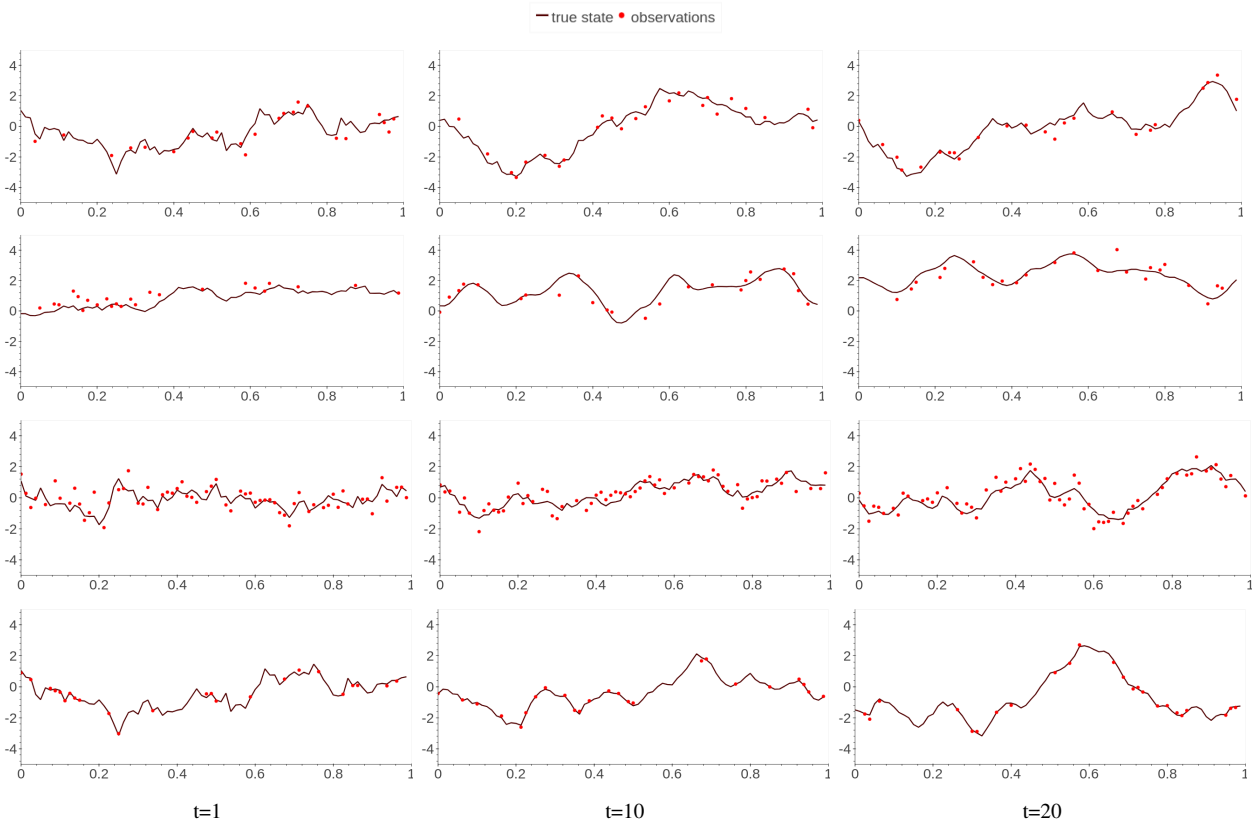


Figure C.2: Sample realizations simulated from the model described in Sections 2.7.1 and C.4.2. The rows correspond to the scenarios in Table 2.1, from top to bottom: baseline, smooth, dense obs., and low noise.

C.4.3 Square domain

In the second set of simulations, we consider a rectangular domain, which is common for spatial data sets. We generalize the model (C.2) as

$$\begin{cases} \frac{\partial}{\partial t}x(\mathbf{s},t) = \alpha_1 \frac{\partial}{\partial s_1}x(\mathbf{s},t) + \alpha_2 \frac{\partial}{\partial s_2}x(\mathbf{s},t) + \beta\frac{\partial^2}{\partial s_1^2}x(\mathbf{s},t) + \frac{\partial^2}{\partial s_2^2}x(\mathbf{s},t)) + \zeta(\mathbf{s},t) \\ x(\mathbf{s},0) = GP\left(0, K_{2d}(\cdot,\cdot)\right) \end{cases} . \tag{C.3}$$

We assume that $x(\mathbf{s},t) : [0,1]^2 \times [0,T] \to \mathbb{R}$, that $\zeta(\mathbf{s},t)$ is a two dimensional zero-mean stationary Gaussian process with an isotropic spatial covariance function $\sigma_w^2 C(\cdot,\cdot)$ and independent over time, and we define $K_{2d}(\mathbf{s}_i,\mathbf{s}_j) = \mathcal{M}_{\nu,\lambda}(\|\mathbf{s}_i - \mathbf{s}_j\|_2)$. We set the diffusion coefficient $\beta = 0.0004$ and we use $\alpha_1 = \alpha_2 = 0.01$. Similar to the 1D case, we discretize (C.3) using a regular spatial

grid with $n_x = n_y = 34$ points along each dimension and 20 equidistant time points. We approximate the time derivative as in Section C.4.2 and use the following approximations for the spatial derivatives:

$$\begin{aligned}
\tfrac{\partial}{\partial \mathbf{s}_1} x(\mathbf{s},t) &\approx \left(x(\mathbf{s},t) - x(\mathbf{s} - [\Delta s_1, 0]^T, t) \right) \tfrac{1}{\Delta s_1}, \\
\tfrac{\partial}{\partial \mathbf{s}_2} x(\mathbf{s},t) &\approx \left(x(\mathbf{s},t) - x(\mathbf{s} - [0, \Delta s_2,]^T, t) \right) \tfrac{1}{\Delta s_2}, \\
\tfrac{\partial^2}{\partial \mathbf{s}_1^2} x(\mathbf{s},t) &\approx \left(x(\mathbf{s} - [\Delta s_1, 0]^T, t) - 2x(\mathbf{s}) + x(\mathbf{s} + [\Delta s_1, 0]^T, t) \right) \tfrac{1}{\Delta^2 s_1}, \\
\tfrac{\partial^2}{\partial \mathbf{s}_2^2} x(\mathbf{s},t) &\approx \left(x(\mathbf{s} - [0, \Delta s_2]^T, t) - 2x(\mathbf{s}) + x(\mathbf{s} + [\Delta s_2, 0]^T, t) \right) \tfrac{1}{\Delta^2 s_2}.
\end{aligned}$$

We set $\Delta s_1 = \Delta s_2 = 1/(n_x + 1) = 0.029$ and $\Delta t = 1$. Similar to the 1D case, we then represent the model (C.3) as a linear state-space model:

$$\begin{cases} \mathbf{x}_t = \mathbf{E}_t \mathbf{x}_{t-1} + \mathbf{w}_t, \\ \mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}), \end{cases}$$

with the evolution matrix having 5 diagonal nonzero bands. If we use c_k and c_{-k} to denote the values on the k-th diagonal above and below the main diagonal, respectively, then all nonzero entries of $\mathbf{E}_t$ are given by:

$$\begin{aligned}
c_{-n_x} &= -\beta \Delta^2 s_2 + \alpha_2 \cdot \Delta s_2 \\
c_{-1} &= -\beta \Delta^2 s_1 + \alpha_1 \cdot \Delta s_1 \\
c_0 &= 1 + 2\beta(\Delta^2 s_1 + \Delta^2 s_2) - \alpha_1 \Delta s_1 - \alpha_2 \Delta s_2 \\
c_1 &= -\beta \Delta^2 s_1 \\
c_{n_x} &= -\beta \Delta^2 s_2.
\end{aligned}$$

We also take $\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, \sigma_w^2 \mathbf{Q})$ with $\mathbf{Q} = [C(\mathbf{s}_i, \mathbf{s}_j]_{i,j=1,\ldots n_{\mathcal{S}}}$ and $\mathbf{\Sigma} = [K_{2d}(\mathbf{s}_i, \mathbf{s}_j)]_{i,j=1,\ldots n_{\mathcal{S}}}$. We model the observations in the same way as described in the 1D case. Examples of realizations

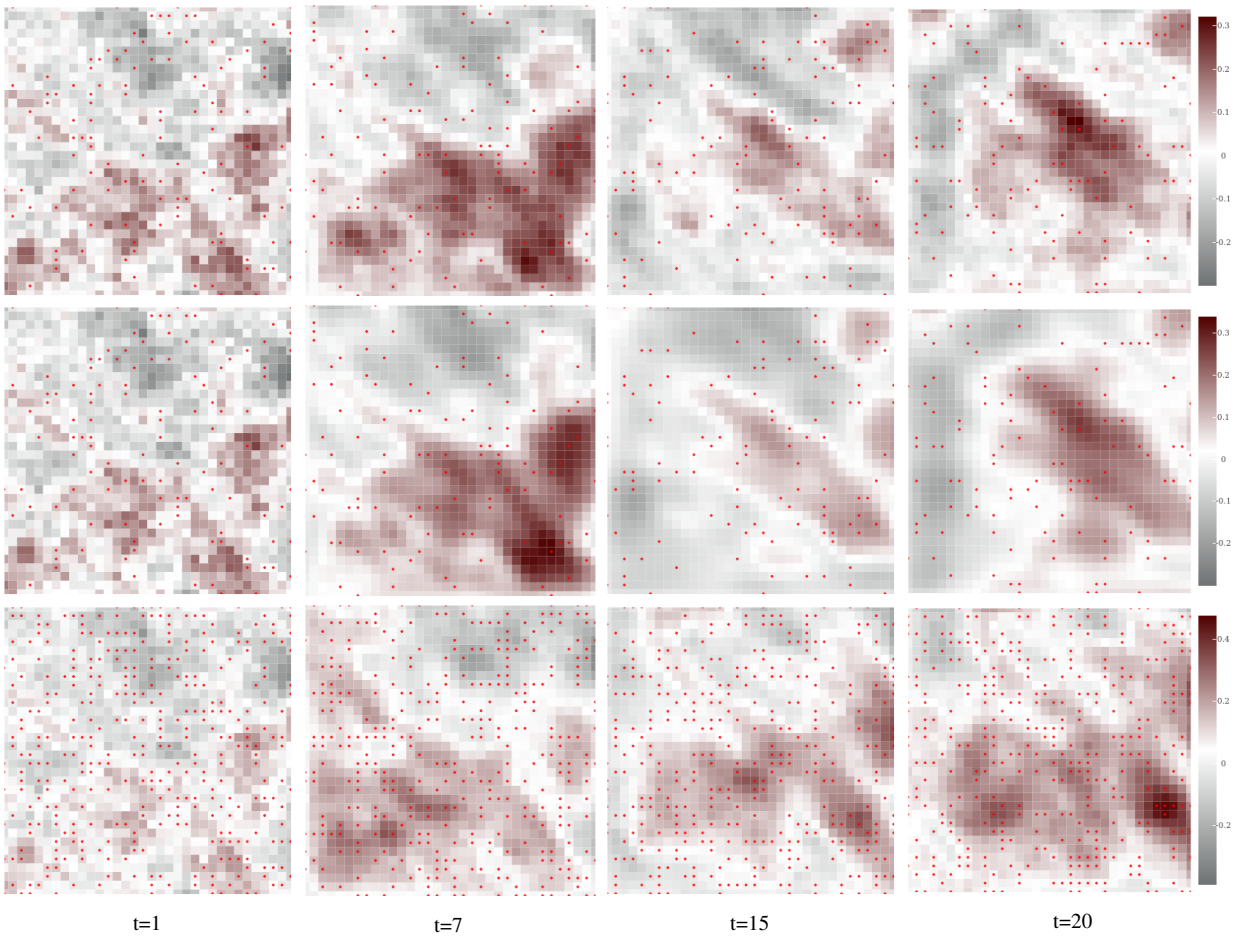


Figure C.3: Sample realizations simulated from the model described in Sections 2.7.2 and C.4.3. The rows correspond to the first three scenarios in Table 2.2, from top to bottom: baseline, smooth, and dense obs. Red dots denote observation locations.

are given in Figure C.3.

C.4.4 Interval coverage

Comparing the frequentist coverage of intervals to their nominal level is a quick way of assessing the calibration of predictive distributions.

Figure C.5 presents interval coverage for filtering methods described in Section 2.7 and using parameter settings discussed there. At each time t, we calculate 95% filtering intervals (i.e., with endpoints consisting of the 2.5 and 97.5 percentiles of the filtering distribution) for each grid point and report the proportion of intervals that cover the true value. On average, 95% of exact confi-

dence intervals (i.e., those generated using the Kalman filter) should cover the corresponding true values.

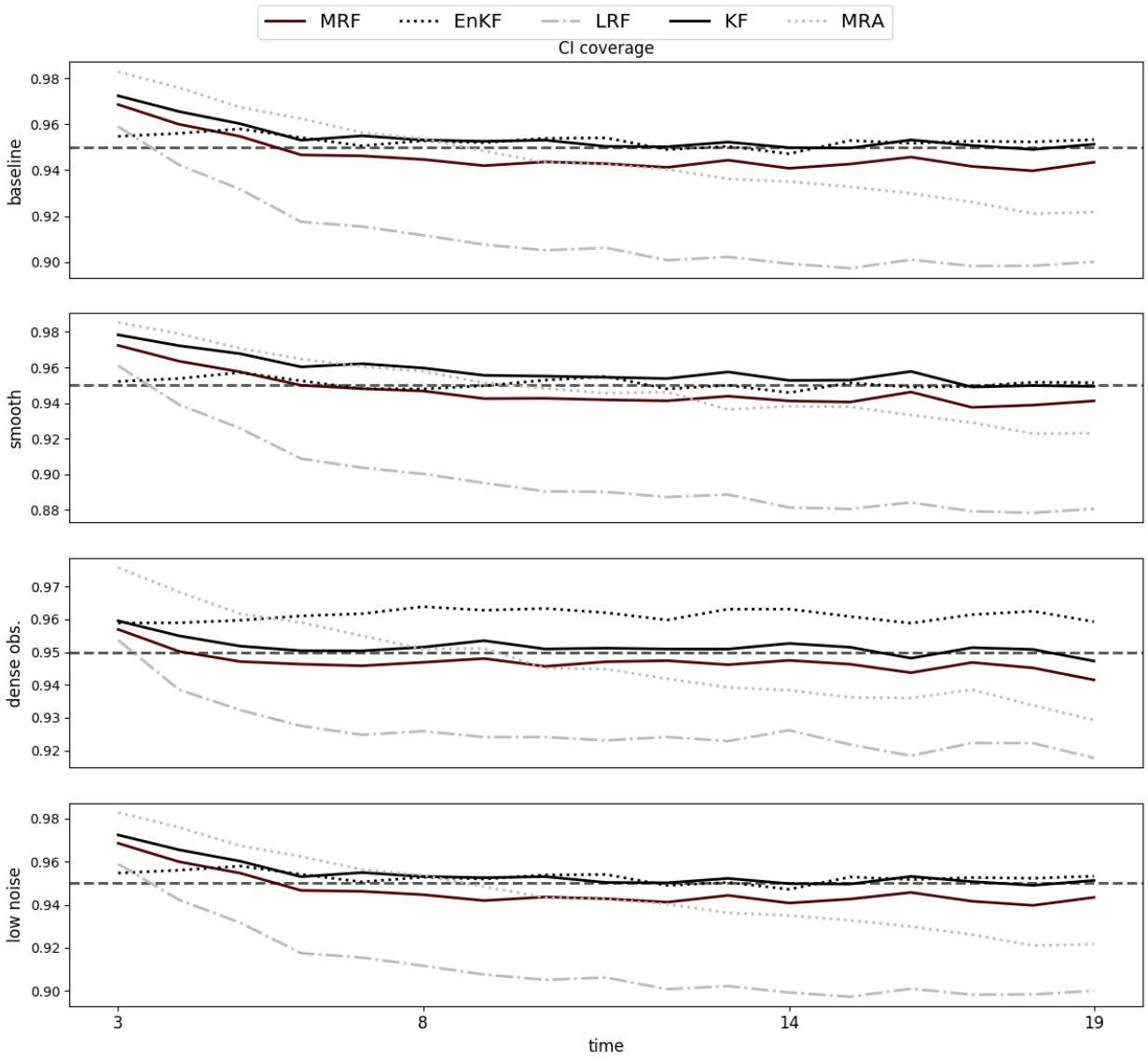


Figure C.5: Interval coverage for different filtering methods. Straight dashed horizontal line indicates the nominal coverage (95%).

The interval coverage is similar to the nominal coverage under all parameter settings for Kalman filter, EnKF, and MRF. The filtering distributions of the remaining two methods seem to be poorly calibrated: low-rank filter generates intervals that are too narrow, while the ones

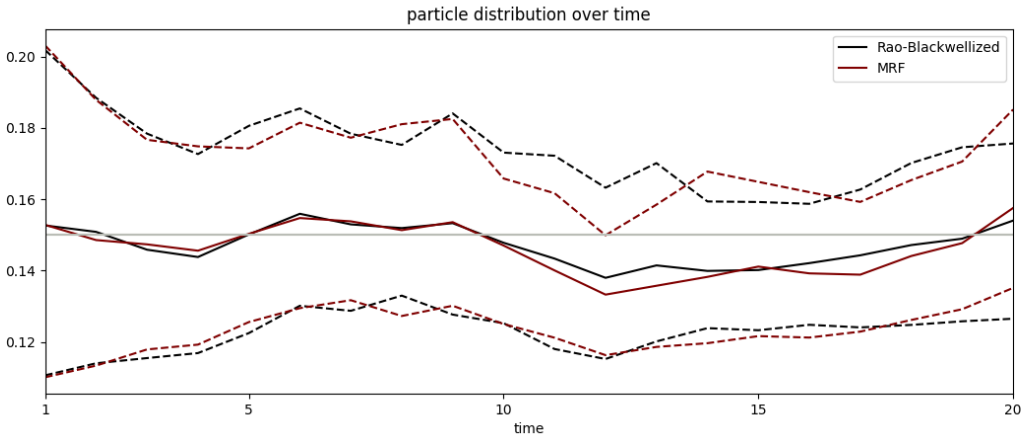


Figure C.6: Summary of particle distributions over time. The solid line is the mean and the dashed lines are the 10th and 90th percentiles, respectively, of the filtering distributions. The solid grey line indicates the true value of the parameter.

produced by MRA start as too wide and become too narrow as time progresses.

C.5 Numerical experiments using the particle MRF

In order to illustrate the performance of the particle MRF described in Section 2.6, we simulated data from the two-dimensional model with baseline parameter settings (see Sections 2.7 and C.4.3), and used Algorithm 2.6 to infer the filtering distribution of a λ, the range parameter of the innovation covariance matrix. We assumed the prior distribution to be such that $\log \lambda_t \sim N(\log(0.15), 0.25^2)$. The proposal distribution was taken to be $\log \lambda_t | \lambda_{t-1} \sim N(\log \lambda_{t-1}, 0.5^2)$, while the initial values were drawn from $\log \lambda_0 \sim N(\log(0.15), 0.25^2)$. Figure C.6 shows the performance of the MRF and the exact Rao-Blackwellized (Doucet et al., 2000) particle filters, both using 1,000 particles. The particle MRF produced similar distributions as the exact filter.

We also compared the accuracy of parameter inference using the MRF to that using the LRF, which can be viewed as a special case of the MRF as described in Section 2.7. We simulated 50 datasets at a single time point $t = 1$, assuming the 2D baseline settings, except that x_0 and $\mathbf{\Sigma}_{0|0}$ were initialized as zero. For each dataset and each method, we numerically found the λ value that maximized the integrated likelihood $\mathcal{L}_t$ from (2.5) at time $t = 1$. The distributions of the resulting 50 parameter estimates are summarized in Figure C.7, while Table C.1 contains the root mean

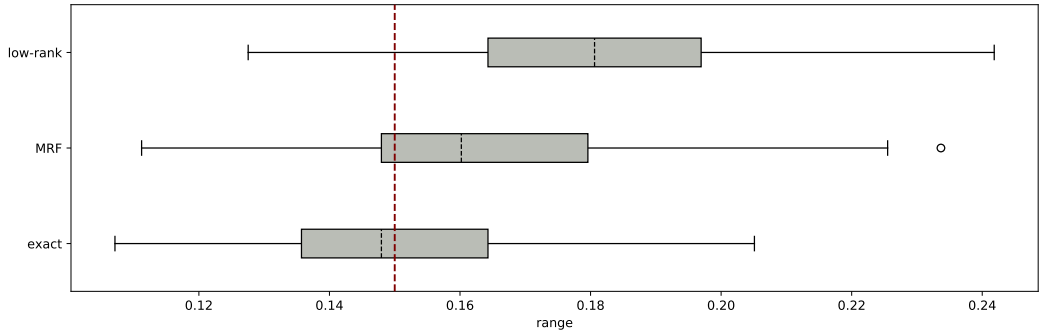


Figure C.7: Boxplots of MLEs of the range parameter λ, as implied by three different integrated likelihoods. The true value $\lambda = 0.15$ is indicated by the vertical dashed line.

	exact	MRF	LRF
RMSE	0.057	0.066	0.076

Table C.1: Root mean squared error of MLEs of the range parameter λ, as implied by three different integrated likelihoods

squared error of the MLEs for each of the methods.

While both approximate methods produced acceptable results, the estimates obtained by the MRF were considerably more accurate than those produced by the LRF, both at the same computational complexity.

C.6 Review of some graph theoretical results

In this section we define several terms commonly used in graph theory that will be needed in Section C.8. These definitions follow the terminology used in Lauritzen (1996) and Khare and Rajaratnam (2012). We denote by $G(V, E)$ a graph with vertices V and edges E.

DEFINITION 3. *We call G' a* subgraph *induced by $A \subseteq V$ if $G' = (A, E \cap (A \times A))$.*

In other words, a subgraph is a graph formed by taking a subset A of vertices and all the edges whose endpoints are in A.

DEFINITION 4. *A* path *is a sequence of vertices $v_1, \ldots, v_n \subset V$ such that for $i = 1, \ldots, n$, $(v_i, v_{i-1}) \in E$. A* cycle *is a path such that $v_n = v_1$.*

DEFINITION 5. *Consider a* cycle $(v_1, \ldots, v_n)$. *A chord is an edge* $e \in E$ *which connects two non-consecutive vertices within the cycle. Graph* G *is* chordal *if every cycle of length greater than 4 has a chord.*

Chordal graphs are also called triangulated graphs or decomposable graphs.

DEFINITION 6. *A graph is* homogeneous *if it is chordal and if it does not contain the graph* A_4*, defined as* $\overset{1}{\bullet} - \overset{2}{\bullet} - \overset{3}{\bullet} - \overset{4}{\bullet}$*, as an induced subgraph.*

Following Khare and Rajaratnam (2012), we write that $v \to w$ when

$$\{u : u = w \vee (u, w) \in E\} \subseteq \{u : u = v \vee (u, v) \in E\}.$$

In other words, when $v \to w$, then the neighborhood (all direct neighbors) of w is contained within the neighborhood of v. Within the context of of our graph G, we can make the symbol $\to$ more intuitive by thinking of if it as saying that "w is downstream from v." In principle, it is possible that two vertices have the exact same neighborhoods. To capture it formally, we could define an equivalence relation $\equiv$ as

$$u \equiv v \iff u \to v \ \wedge \ v \to u.$$

We denote by $\bar{v}$ the equivalence class under $\equiv$ containing v.

DEFINITION 7. *For a homogeneous graph* $G = (V, E)$*, an ordering* σ *is called a* Hasse tree-based elimination scheme *for* G*, if for* $u, v \in V$

$$u \to v, \bar{u} \neq \bar{v} \implies \sigma(u) > \sigma(v),$$

which we adopt after Khare and Rajaratnam (2012).

DEFINITION 8. *A* tree *is a directed graph with no cycles, such that each vertex has degree one (i.e., for a vertex* u*, there is only one edge that ends at* u*).*

DEFINITION 9. *Let* $T = (V, E)$ *be a tree and let* $(i, j) \in E$*. Then* i *is called a* parent *of* j*, while* j

is called a child *of* i.

C.7 Hierarchical matrices

This section introduces basic concepts of hierarchical matrix theory. They come in useful in the proof of Lemma 1 in Section C.8.

If t is a vertex of a tree T, let $c(t)$ denote the set of children of t. Formally, $c(t) := \{s \in T : s \text{ is a child of } t\}$. Similarly, if $s \in c(t)$ (i.e., t is a parent of s), we write $t = p(s)$.

DEFINITION 10. (Hackbusch, 1999) *Let I be an index set. A tree T is called an $\mathcal{H}$-*tree *(based on I) if the following conditions hold:*

1. *All vertices $t \in T$ are subsets of I;*
2. $I \in T$;
3. $|c(t)| \neq 1 \quad \forall t \in T$.

It can be concluded that I is the root of T, the only vertex without a parent element.

DEFINITION 11. *The* depth *of a vertex t is a function $d(\cdot)$ defined as*

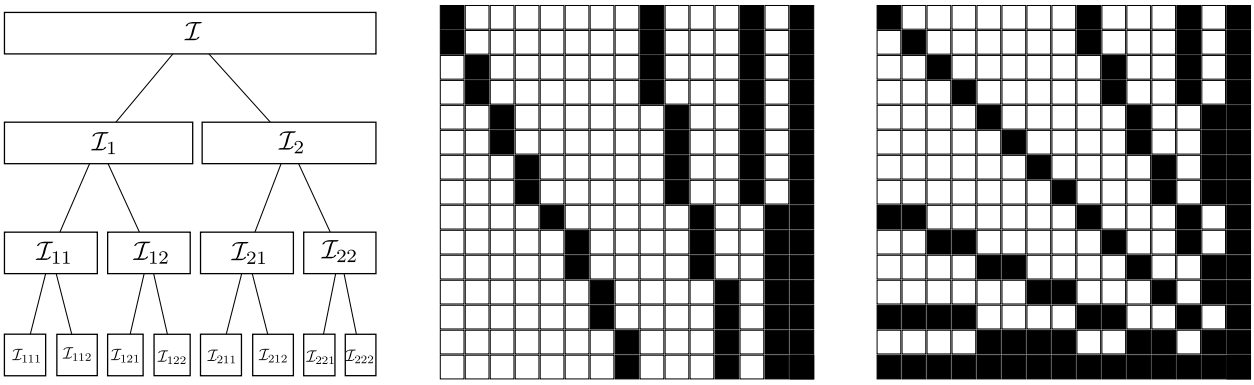
$$d(t) = \begin{cases} 0 & \textit{if } t \textit{ is the root} \\ d(p(t)+1) & \textit{otherwise} \end{cases}.$$

Our notation will often be more readable if we write $t \searrow s$ whenever $d(t) > d(s)$, and $t \nearrow s$ if $d(t) < d(s)$.

We use $\mathcal{A}(t)$ to label the set of all ancestors of the vertex t, that is

$$\mathcal{A}(s) = \begin{cases} \emptyset & \text{if t is the root} \\ \mathcal{A}(p(s)) \cup \{p(s)\} & \text{otherwise} \end{cases}.$$

Throughout Section C.8 and Appendix A.1, we assume for simplicity that $r = 1$. If this is not true, we can consider an equivalence relation $\equiv_m$ such that for two columns $\mathbf{c}_1, \mathbf{c}_2$ in $\mathbf{B}^m$



(a) An $\mathcal{H}$-tree over grid $\mathcal{I}$ (b) $\mathbf{B}$ matrix (c) $\mathbf{B}'\mathbf{B}$ matrix

Figure C.8: Mapping τ used in Section C.8 goes from b) to a), while mapping π goes from b) to c).

$\mathbf{c}_1 \equiv_m \mathbf{c}_2 \iff \mathbf{c}_1, \mathbf{c}_2$ both contain elements of the same block $\mathbf{B}_{j_1,\ldots,j_m}$. Verification that $\equiv_m$ is indeed an equivalence relation is elementary. The assumption that $r = 1$ means that each of the blocks $\mathbf{B}_{j_1,\ldots,j_m}$ has dimensions $|\mathcal{I}_{j_1,\ldots,j_m}| \times 1$.

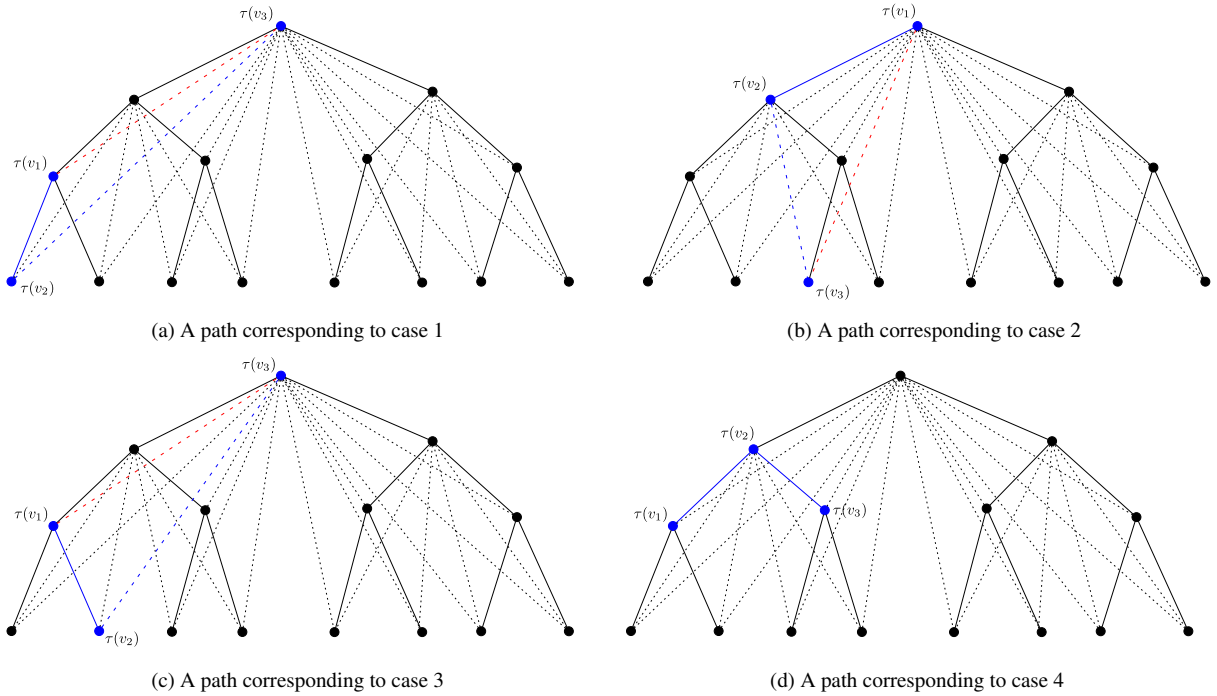
Now notice that the partitioning scheme defined in Section 2.3.4.1 is an $\mathcal{H}$-tree based on $\mathcal{I}$. Indeed: (1) all elements $\mathcal{I}_{j_1,\ldots,j_m}$ are subsets of $\mathcal{I}$, (2) $\mathcal{I}$ is the root node and (3) $|c(\mathcal{I}_{j_1,\ldots,j_m})| = J > 1$. We will refer to this $\mathcal{H}$-tree as $T_{\mathcal{I}}$. From this perspective, each $\mathcal{I}_{j_1,\ldots,j_m}$ is a vertex of $T_{\mathcal{I}}$. Figure C.8a illustrates $T_{\mathcal{I}}$ for the case $M = 3$ and $J = 2$.

C.8 Lemmas used in the proof of Proposition 3

LEMMA 1. *The pattern of zeros in* $\mathbf{B}'\mathbf{B}$ *corresponds to a homogeneous graph.*

Proof. We start by observing that there is a $1-1$ mapping $\tau : \mathcal{I} \to T_{\mathcal{I}}$ between the indices of columns of $\mathbf{B}$ and the vertices of $T_{\mathcal{I}}$. Specifically, each index $j \in \mathcal{I}$ of some column $\mathbf{c}_j$ can be identified with the vertex that has the same index as the $\mathbf{B}_{j_1,\ldots,j_m}$ block that $\mathbf{c}_j$ contains. For example, by construction, $\mathbf{c}_1$, the first column in $\mathbf{B}^M$, contains elements $\mathbf{B}_{1,\ldots,1}$, so $\tau(1) = \mathcal{I}_{1,\ldots,1}$. Also notice that if $\mathbf{c}_i$ contains $\mathbf{B}_{j_1,\ldots,j_m}$ and $\mathbf{c}_j$ contains $\mathbf{B}_{j_1,\ldots,j_{m-1}}$, then $\tau(j) \in \mathcal{A}(\tau(i))$ (see Figure C.8 for illustration).

Next, take two columns $\mathbf{c}_i, \mathbf{c}_j$ of $\mathbf{B}$ and $\tau(i), \tau(j)$, the corresponding vertices in $T_{\mathcal{I}}$. For the rest of the proof, we assume without loss of generality that $i < j$. Then there are two cases to consider.



(a) A path corresponding to case 1

(b) A path corresponding to case 2

(c) A path corresponding to case 3

(d) A path corresponding to case 4

Figure C.9: Different paths of length 3 in graph G considered in the proof of Lemma 1. For clarity, dashed lines were used to indicate edges that connect vertices whose depths differ by more than 1. Elements of the path are marked in blue. A red line indicates a chord.

1. $d(\tau(i)) = d(\tau(j))$:

   In this situation $\mathbf{c}_i'\mathbf{c}_j = 0$ because $\mathbf{c}_i, \mathbf{c}_j$ are columns of the same block diagonal matrix $\mathbf{B}^m$ and each diagonal block is only one column wide.

2. $d(\tau(i)) > d(\tau(j))$:

   We examine two subcases:

   (a) $\tau(j) \notin \mathcal{A}(\tau(i))$; this implies that $\mathbf{c}_i'\mathbf{c}_j = 0$;

   (b) $\tau(j) \in \mathcal{A}(\tau(i))$; then, in general, $\mathbf{c}_i'\mathbf{c}_j \neq 0$.

Let us now look at $\mathbf{B}'\mathbf{B}$ as an adjacency matrix of graph $G = (\mathcal{I}, E)$ where

$$E = \{(i, j) : \mathbf{B}'\mathbf{B}_{ij} \neq 0\}.$$

Define π as a mapping that assigns a vertex in G to each column of $\mathbf{B}$, i.e. $\pi(\mathbf{c}_j) = j$ for each

$j \in \mathcal{I}$. Defining $\rho := \tau \circ \pi^{-1}$ gives us a $1-1$ correspondence between the vertices of graph G and $T_{\mathcal{I}}$ (Figure C.8). We can thus define d, $\mathcal{A}$, c and p for the vertices of the undirected graph G by referring to the corresponding definitions on $T_{\mathcal{I}}$. For example, if u is some vertex of G, then $\mathcal{A}(u) := \mathcal{A}(\rho(u))$.

Using this new notation, the fact that $\mathbf{B}'\mathbf{B}_{ij} = \mathbf{c}_i'\mathbf{c}_j$, as well as the previous observations, we have

$$(i,j) \in E \iff i \in \mathcal{A}(j).$$

In particular, if $d(i) = d(j)$ then $(i,j) \notin E$.

Next, let (v_1, v_2, v_3, v_4) be some path in G. We will show that each such path contains a chord, which means that either $(v_1, v_3) \in E$ or $(v_2, v_4) \in E$. There are four cases we need to consider to demonstrate this (see Figure C.9):

1. $v_1 \nearrow v_2 \nearrow v_3$:

   This means that $v_3 \in \mathcal{A}(v_1)$, and so $(v_3, v_1) \in E$ is a chord.

2. $v_1 \searrow v_2 \searrow v_3$:

   In this case $v_1 \in \mathcal{A}(v_3)$, and so $(v_3, v_1) \in E$ is a chord.

3. $v_1 \searrow v_2 \nearrow v_3$:

   Let $d(v_3) < d(v_1)$. Then v_3 has to be a descendant of v_1, and so $(v_3, v_1) \in E$ is a chord. If $d(v_3) > d(v_1)$, then v_1 is a descendant of v_3. It is not possible that $d(v_1) = d(v_3)$ and $v_3 \neq v_1$, because every node has only one parent.

4. $v_1 \nearrow v_2 \searrow v_3$:

   We need to consider two subcases here:

   (a) $v_3 \searrow v_4$; this means that $v_2 \searrow v_3 \searrow v_4$, which reduces to case 2;

   (b) $v_3 \nearrow v_4$; this means that $v_2 \searrow v_3 \nearrow v_4$, which reduces to case 3.

The reasoning above shows that A_4 is not a subgraph of G and that G is decomposable. Therefore we conclude that G is homogeneous. □

LEMMA 2. *Let $G = (\mathcal{I}, E)$ be the graph described by the pattern of zeros in our $\mathbf{B}'\mathbf{B}$ matrix. Let σ be an ordering of the vertices of G such that*

$$\sigma(i) = i.$$

Then σ is a Hasse tree based elimination scheme for G.

Proof. Let $N(u) = \{v \in G : (v, u) \in E\} \cup \{u\}$.

We begin by showing that

$$N(v) \subsetneq N(u) \iff u \in \mathcal{A}(v).$$

First, let us assume that $u \in \mathcal{A}(v)$, and take some w such that $u \in \mathcal{A}(w)$ and $w \in \mathcal{A}(v)$. Because u is not a leaf, Definition 10 implies $|S(u)| > 1$. Therefore, there exists a $w' \neq w \in S(u)$. Since $w' \notin \mathcal{A}(v)$, by (C.8) w' is not a neighbor of v, but it is a neighbor of u. Now observe that

$$N(v) = \mathcal{A}(v) \cup \{v\} \cup \{w : v \in \mathcal{A}(w)\}.$$

Let $w \in \{v\} \cup \{w : v \in \mathcal{A}(w)\}$, then we have $v \in \mathcal{A}(w)$. Thus $w \in N(v)$. Now take $w \in \mathcal{A}(v)$. We have three cases here:

- if $d(w) < d(v)$, then $v \in \mathcal{A}(w)$ and so $w \in N(v)$;
- if $d(w) = d(v)$, then $w = v$ and so $w \in N(v)$;
- if $d(w) > d(v)$, then $w \in \mathcal{A}(v)$ and so $w \in N(v)$.

Thus we showed that $N(v) \subset N(u)$ but $N(v) \neq N(u)$.

Second, assuming $N(v) \subsetneq N(u)$ implies that $(v, u) \in \mathcal{E}$, so either $v \in \mathcal{A}(u)$ or $u \in \mathcal{A}(v)$. But if the former was to hold, it would contradict what we have shown above. Therefore, we conclude that $u \in \mathcal{A}(v)$.

The equivalence we have just demonstrated means that the lemma is proved once we show that

$$u \in \mathcal{A}(v) \implies \sigma(u) > \sigma(v).$$

But we already established in (C.8) that

$$\begin{cases} i < j \\ (i, j) \in E \end{cases} \iff i \in \mathcal{A}(j).$$

This means that if we order the vertices in V according to the order of the rows of $\mathbf{B}'\mathbf{B}$ to which they correspond, this ordering will be a Hasse tree-based elimination scheme. □