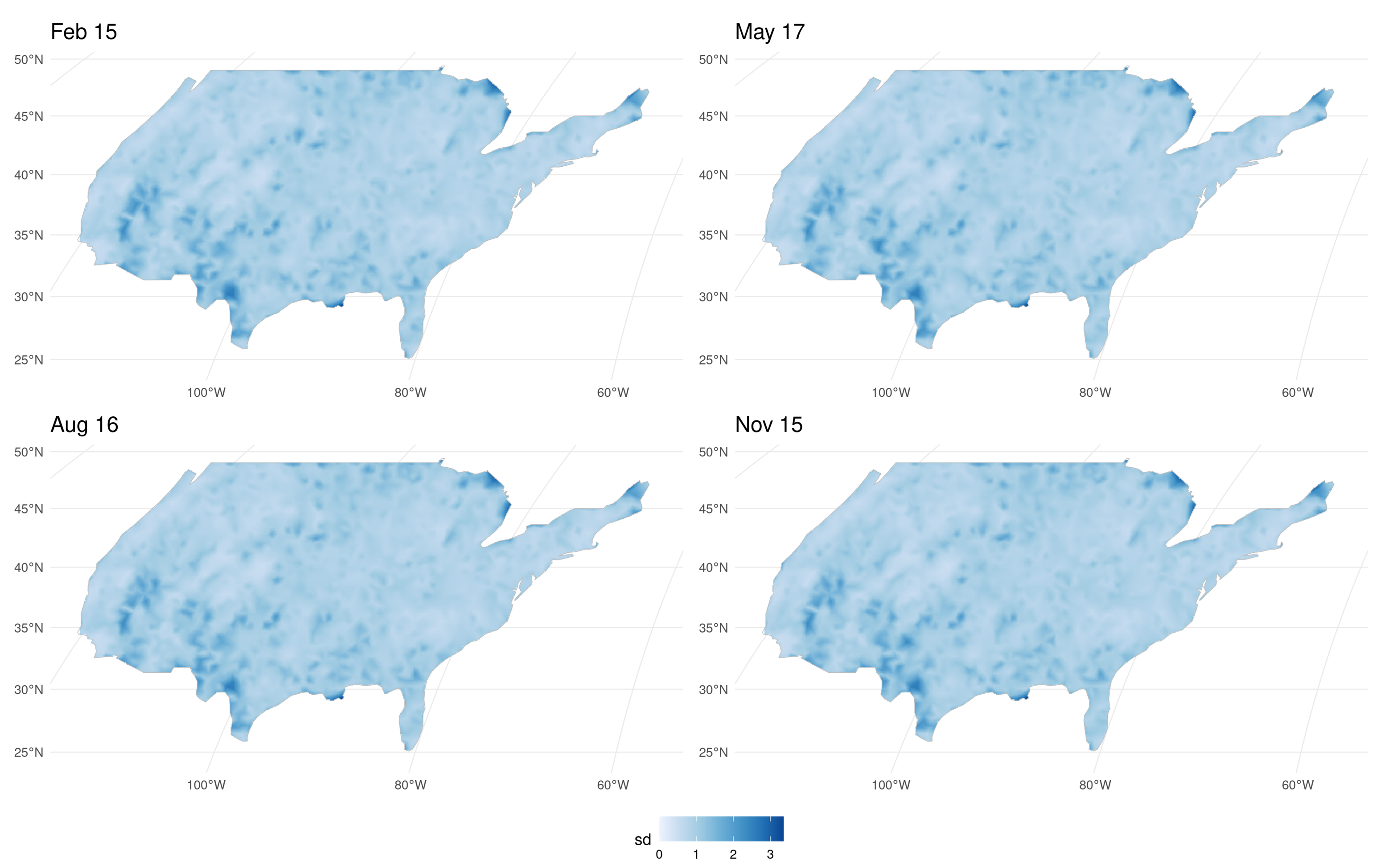
High Performance Computing Meets Approximate Bayesian Inference

Lisa Gaedke-Merzhäuser[1], Steven Rennich[2], Elias Krainski[3], Håvard Rue[3], Olaf Schenk[1]

[*] G.-M. L, Krainski E, Janalik R, Rue H, Schenk O. **Integrated Nested Laplace Approximations for Large-Scale Spatial-Temporal Bayesian Modeling**. arXiv preprint arXiv:2303.15254. 2023 Mar 27.

Application

Atmospheric Temperature Modeling



Std. Dev. spatial-temporal random field

✦ Non-separable spatial-temporal random field
✦ 365 days (year 2021)
➡ >1m latent parameters
✦ Fixed effects for e.g. elevation, latitude, temporal spline

Integrated Nested Laplace Approximations (INLA)

❖ Deterministic methodology for performing Bayesian inference
❖ Applicable to latent Gaussian models
❖ Relies on a nested approximation strategy that employs sparse Gaussian Markov random fields in the latent parameter space

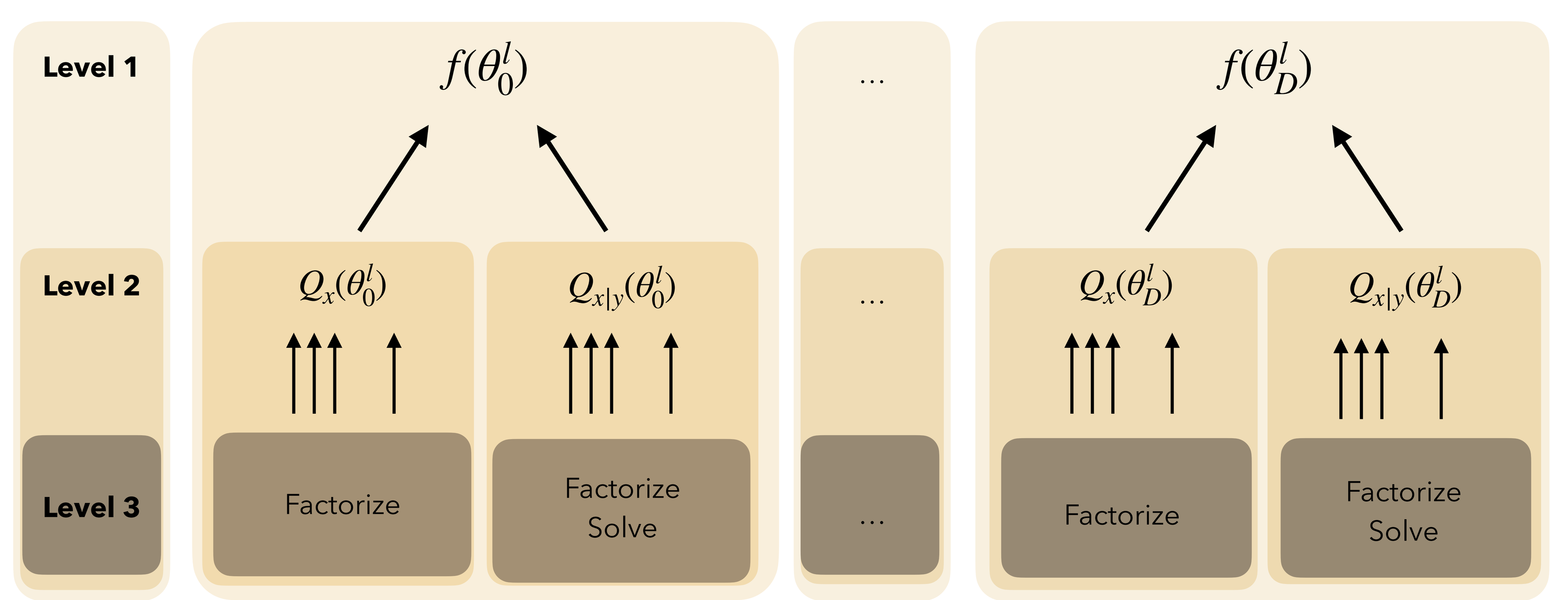
Stochastic Partial Differential Equations (SPDE)

❖ Recasts the spatial-temporal component of the model as the solution to an SPDE
❖ Can be discretised and efficiently solved using Finite Element method
❖ Based on Matérn covariance function

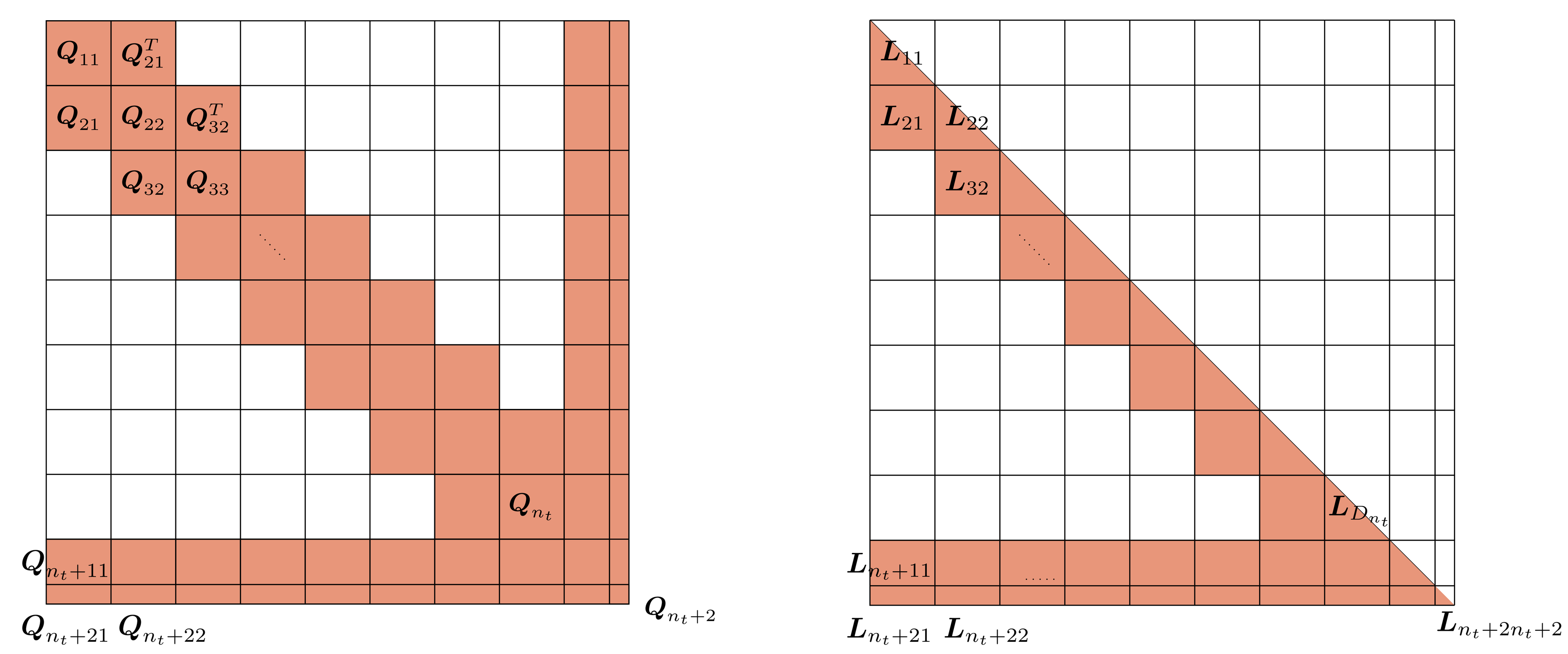
Spatial-temporal Bayesian Modeling

Despite the increasing interest in Bayesian computing, large-scale inference tasks continue to pose a computational challenge that often requires a trade-off between accuracy and computation time. We present a highly scalable approach for performing spatial-temporal Bayesian modelling based on the methodology of integrated nested Laplace approximations (INLA), combining solution strategies from the field of high-performance computing with state-of-the-art statistical learning techniques. We leverage different parallelization strategies to fully utilize the power of today's distributed compute architectures and introduce highly optimized sparse linear algebra routines to handle the computational kernel operations [*].

Distributed Shared Memory Approach



Level 3 - Parallel Block Cholesky Factorization

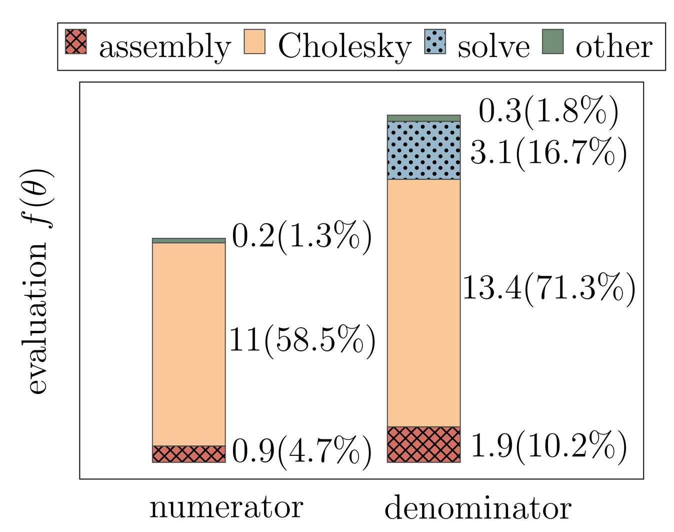


We developed a selected block inversion routine that efficiently handles the computational kernel operations. It is tailored to the arising precision matrix structure of block tridiagonal arrowhead matrices. It can efficiently compute their Cholesky decompositions and find the diagonal elements of their inverse. We put forward a GPU implementation using MAGMA and CuBLAS.

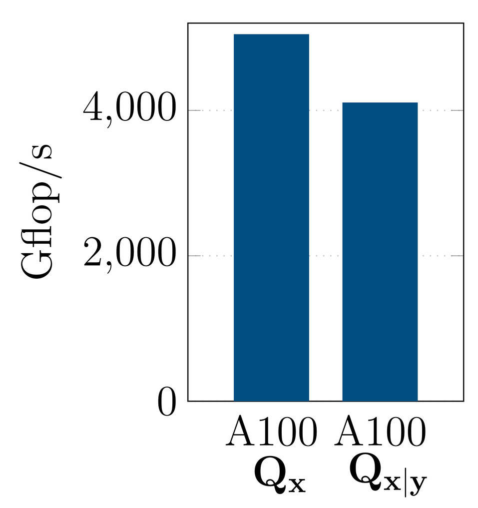
	Stream 1	Stream 2	Stream 3
	CopyHostToDev($Q_{:1}$)	$L_{D_1} \leftarrow$ chol(D_1) $L_{D_1}^{invT} \leftarrow L_{D_1}^{-T}$	
for $i = 2,3,\ldots n_t - 1$	CopyHostToDev($Q_{:i}$) CopyDevToHost($L_{:i-1}$)	$L_{E_{i-1}} \leftarrow E_{i-1} \cdot L_{D_{i-1}}^{invT}$ $D_i \leftarrow D_i - L_{E_{i-1}} \cdot L_{E_{i-1}}^T$ $L_{D_i} \leftarrow$ chol(D_i) $L_{D_i}^{invT} \leftarrow L_{D_i}^{-T}$	$L_{F_{i-1}} \leftarrow F_{i-1} \cdot L_{D_{i-1}}^{invT}$ $L_{F_i} \leftarrow L_{F_i} - L_{F_{i-1}} \cdot L_{E_{i-1}}^T$ $D_{n_t+1} \leftarrow D_{n_t+1} - L_{F_{i-1}} \cdot L_{F_{i-1}}^T$
end for			
	CopyHostToDev($Q_{:n_t}$) CopyDevToHost($L_{:n_t-1}$) CopyHostToDev($D_{:n_t+1}$) CopyDevToHost($L_{:n_t}$) CopyDevToHost($L_{:n_{t+1}}$)	$D_{n_t} \leftarrow D_{n_t} - L_{F_{n_t-1}} \cdot L_{F_{n_t-1}}^T$ $L_{D_{n_t}} \leftarrow$ chol(D_{n_t}) $L_{D_{n_t}}^{invT} \leftarrow L_{D_{n_t}}^{-T}$ $D_{n_t+1} \leftarrow D_{n_t+1} - L_{F_{n_t}} \cdot L_{F_{n_t}}^T$ $L_{D_{n_t+1}} \leftarrow$ chol(D_{n_t+1})	$L_{F_{n_t}} \leftarrow F_{n_t} \cdot L_{D_{n_t}}^{invT}$

Benchmarks

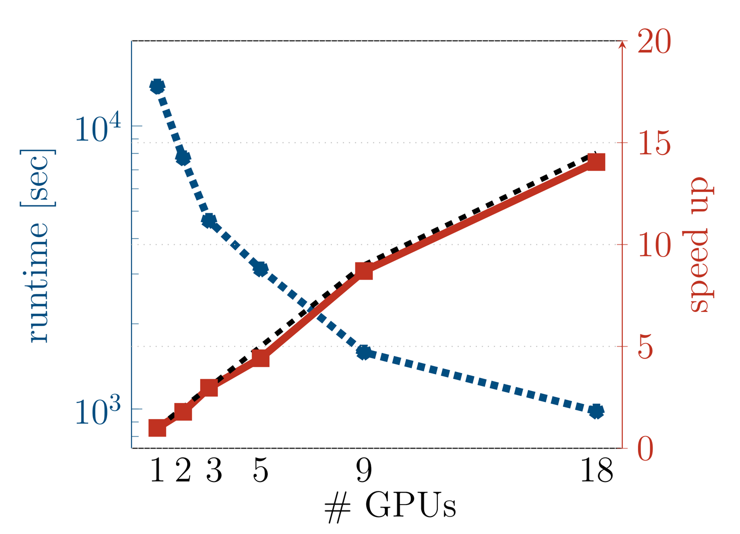
Synthetic dataset with >1m latent parameters, >2m i.i.d. observations



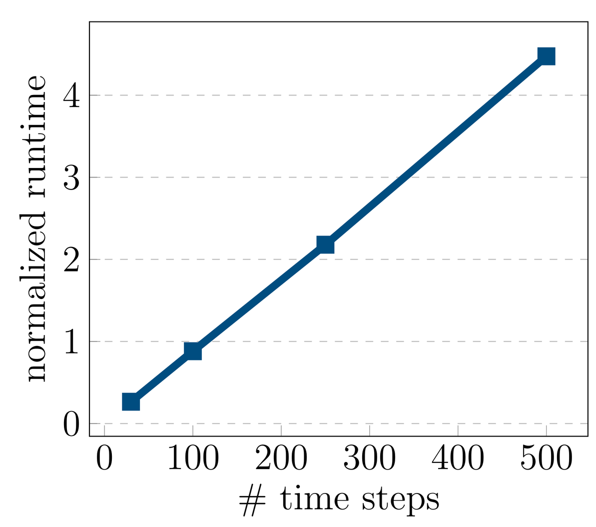
Kernel Operations



Performance



Strong Scaling



Temporal Scaling

Università della Svizzera italiana

[1]Institute of Computing, Università della Svizzera italiana,
[2]Nvidia Corp., Santa Clara, California, U.S.A.
[3]CEMSE Division, King Abdullah University of Science and Technology

KAUST NVIDIA NHR FAU