Advancing Spatio-Temporal Statistics in Geo-Environmental Data Science through Deep Learning and High Performance Computing

Ying Sun, Professor of Statistics
ying.sun@kaust.edu.sa

Environmental Statistics Research Group
King Abdullah University of Science and Technology (KAUST)

Outline

1. Background and Motivation

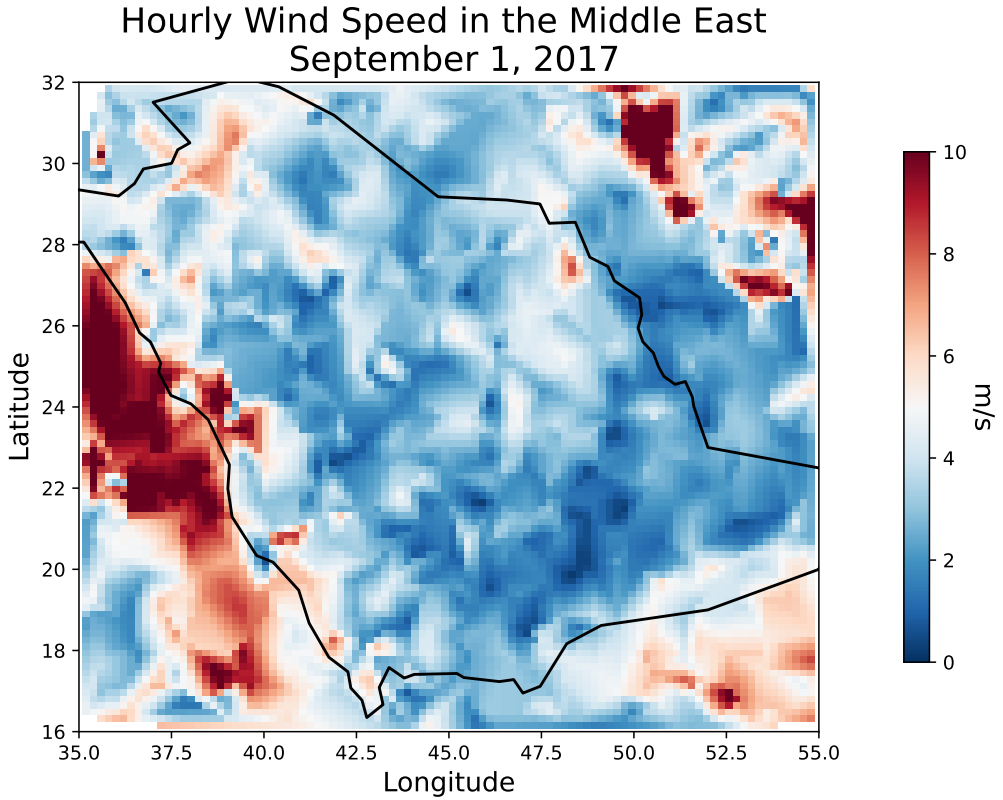
Introduction

Tackling the Complexity of Environmental Data

- As technological advancements flood us with high-dimensional data, the need for rigorous spatio-temporal analysis has never been greater
- Today, I will present my research, demonstrating how advanced statistical models and methods are reshaping our approach to complex environmental challenges

Examples of geostatistical datasets

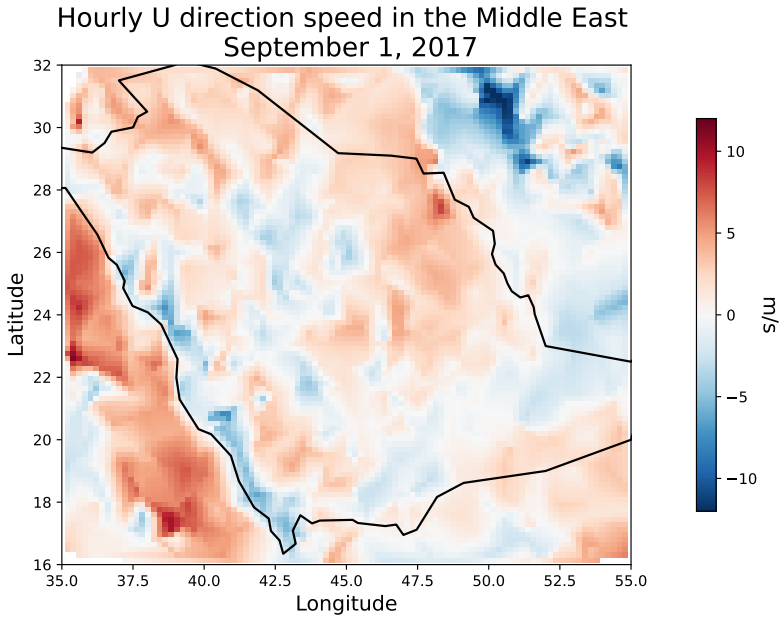
Space-time dataset: wind speed in the Middle East

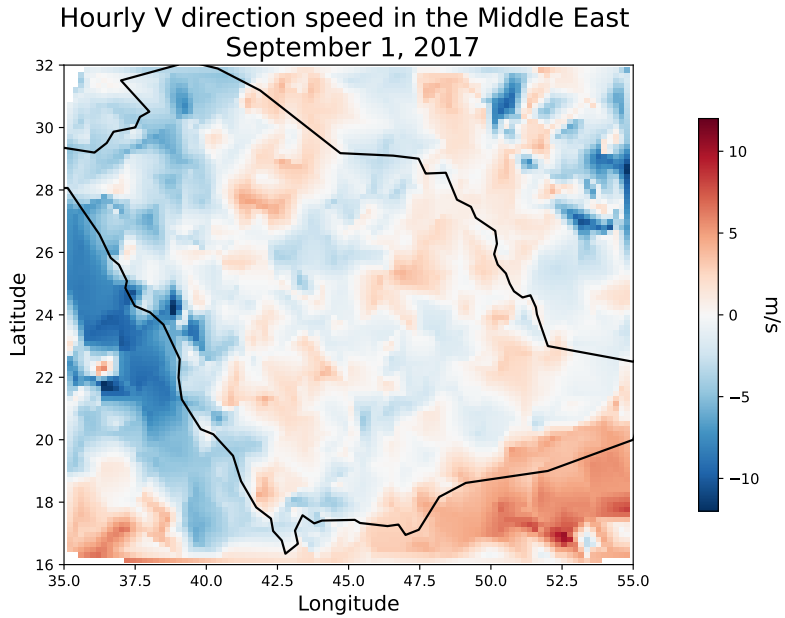


Examples of geostatistical datasets

Multivariate spatial data: wind components

- U component (parallel to the x-axis)
- V component (parallel to the y-axis)





Some Fundamental Problems in Environmental Data Science

Typical goals associated with modeling geostatistical data:

1. Statistical inference
2. Stochastic simulation
3. Prediction

For example, Gaussian process models are fundamental and often serve as building blocks for more complex models in these analyses, but they face certain limitations and challenges in modeling and computations

- Modeling: the specification of covariance functions
- Computations: Gaussian likelihood inference, kriging, random field simulations

1.1 Covariance models

Multivariate spatial data:

- Let $\mathbf{X}(\mathbf{s}) = \{X_1(\mathbf{s}), \ldots, X_p(\mathbf{s})\}^\mathsf{T}$, $\mathbf{s} \in \mathbb{R}^d$ be a zero-mean p-variate random field defined on a d-dimensional Euclidean space
- The matrix-valued multivariate covariance function for $\mathbf{X}$ is defined as:

$$\mathbf{C}(\mathbf{s}, \mathbf{s}') = \{\mathsf{C}_{ij}(\mathbf{s}, \mathbf{s}')\}_{i,j=1}^{p}$$

  where $\mathsf{C}_{ij}(\mathbf{s}, \mathbf{s}') = \mathbb{E}\{X_i(\mathbf{s})X_j(\mathbf{s}')\},\ i, j = 1, \ldots, p$
- $\mathsf{C}_{ii}(\mathbf{s}, \mathbf{s}'),\ i = 1, \ldots, p$ denotes marginal covariance function for the individual process X_i
- $\mathsf{C}_{ij}(\mathbf{s}, \mathbf{s}'),\ 1 \leq i \neq j \leq p$ denotes cross-covariance function between the two distinct process components X_i and X_j

Space-time data:

- Let $X(\mathbf{s}, t), \mathbf{s} \in \mathbb{R}^d, t \in \mathbb{R}$ be a zero-mean spatio-temporal random field, then the associated spatio-temporal covariance for X is given as:

$$\mathsf{C}(\mathbf{s}, \mathbf{s}'; t, t') = \mathbb{E}\{X(\mathbf{s}, t)X(\mathbf{s}', t')\}$$

- For $\mathbf{s} = \mathbf{s}'$, we get a purely temporal covariance function for Y as:

$$\mathsf{C}(\mathbf{s}, \mathbf{s}, t, t') = \mathbb{E}\{X(\mathbf{s}, t)X(\mathbf{s}, t')\}$$

- For $t = t'$, we get a purely spatial covariance function for Y as:

$$\mathsf{C}(\mathbf{s}, \mathbf{s}', t, t) = \mathbb{E}\{X(\mathbf{s}, t)X(\mathbf{s}', t)\}$$

Challenges

- The covariance matrix of the random vector $\{\mathbf{X}(\mathbf{s}_1)^\top, \ldots, \mathbf{X}(\mathbf{s}_n)^\top\}^\top$ or $\{X(\mathbf{s}_1, t_1), \ldots, X(\mathbf{s}_n, t_n)\}^\top$ must be nonnegative definite for any positive integer n, spatial locations $\mathbf{s}_1, \ldots, \mathbf{s}_n \in \mathbb{R}^d$ and time-points $t_1, \ldots, t_n \in \mathbb{R}$.
- The choice of functions for modeling covariances is limited to the class of nonnegative definite functions
- Existing class of models lacks flexibility in different aspects, such as:
  1. Cross-spectral features in $\mathsf{C}_{ij}(\mathbf{s}, \mathbf{s}')$
  2. Asymmetric cross-dependence in $\mathsf{C}_{ij}(\mathbf{s}, \mathbf{s}')$
  3. Nonstationarity in $\mathsf{C}_{ii}(\mathbf{s}, \mathbf{s}')$
  4. Nonstationarity in $\mathsf{C}(\mathbf{s}, \mathbf{s}'; t, t')$

1.2 Gaussian likelihood inference

- Univariate case $p = 1$: n irregularly-spaced observations $\boldsymbol{X} = \{X(\mathbf{s}_1), \ldots, X(\mathbf{s}_n)\}^\top$ from zero-mean Gaussian random field with covariance function $C(\cdot, \cdot; \boldsymbol{\theta})$
- Spatial Gaussian log-likelihood:

$$\boxed{\ell(\boldsymbol{\theta}) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\log|\boldsymbol{\Sigma}(\boldsymbol{\theta})| - \frac{1}{2}\boldsymbol{X}^\top\boldsymbol{\Sigma}(\boldsymbol{\theta})^{-1}\boldsymbol{X}}$$

- Log determinant and linear solver require a Cholesky factorization of the symmetric positive definite covariance matrix $\boldsymbol{\Sigma}(\boldsymbol{\theta})$
- Cholesky factorization requires $O(n^3)$ floating point operations and $O(n^2)$ memory
- Computations become challenging for large n

1.3 Kriging

- Kriging is spatial interpolation (Best Linear Unbiased Predictor, BLUP)
- Let

$$\begin{pmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \end{pmatrix} \sim \mathcal{N}_{n+m} \left(\begin{pmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{pmatrix}, \begin{pmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{pmatrix} \right)$$

  then:

$$(\mathbf{X}_2 | \mathbf{X}_1 = \mathbf{x}_1) \sim \mathcal{N}_m \left(\boldsymbol{\mu}_{\mathbf{X_2}|\mathbf{X_1}}, \boldsymbol{\Sigma}_{\mathbf{X_2}|\mathbf{X_1}} \right)$$

- Kriging with conditional mean $\boxed{\boldsymbol{\mu}_2 + \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}(\mathbf{x}_1 - \boldsymbol{\mu}_1)}$
- Uncertainty quantification with conditional variance $\boxed{\boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12}}$
- Computations become challenging for large n and/or m

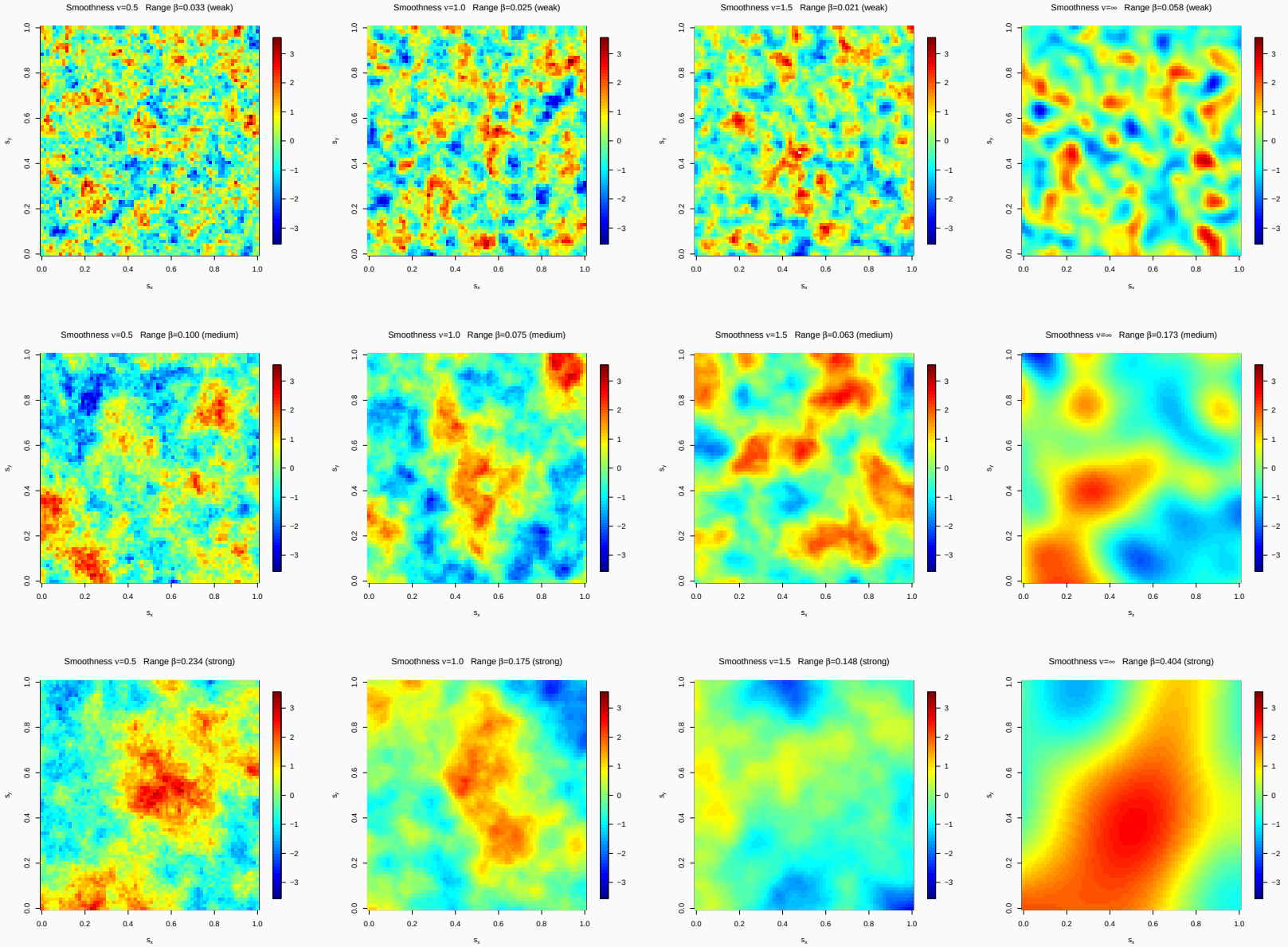
1.4 Gaussian random field simulations

Unconditional simulations:

- $\mathbf{Y} = (Y_1, \ldots, Y_n)^\top$ where Y_i are iid from $\mathcal{N}(0, 1)$
- $\boldsymbol{\Sigma}$ is an $n \times n$ covariance matrix with $(\boldsymbol{\Sigma})_{ij} = \text{cov}\{X(\mathbf{s}_i), X(\mathbf{s}_j)\}$
- $\boldsymbol{\Sigma}^{1/2}$ from spectral decomposition or Cholesky decomposition of $\boldsymbol{\Sigma}$
- Then: $\boxed{\mathbf{X} = \boldsymbol{\mu} + \boldsymbol{\Sigma}^{1/2}\mathbf{Y} \text{ is } \mathcal{N}_n(\boldsymbol{\mu}, \boldsymbol{\Sigma})}$

Computations become challenging for large n

Examples of Gaussian random field simulations



2. Innovative Software Tools

Leveraging high performance computing and deep learning

- Developed *ExaGeoStat*, *ParallelVecchiaGP*, and *ParallelBlockVecchiaGP* software to efficiently process large-scale geostatistical data using advanced computing platforms
- Developed *ParallelScaledBlockVecchiaGP* for computer model emulation
- Introduced *DeepKriging*, a novel neural network architecture providing fast, robust spatial predictions for spatio-temporal data
- These tools utilize high performance computing (HPC) and deep learning (DL) to tackle environmental statistics challenges, enhancing scalability and computational efficiency

2.1 *ExaGeoStat*

HPC can help when fitting a GP model to large datasets with n locations

- Exact computations: $O(n^3)$ floating point operations and $O(n^2)$ memory
- *ExaGeoStat* software: exascale geostatistics (exaFLOPS, 10^{18})
  `https://github.com/ecrc/exageostatcpp`
  `https://github.com/ecrc/exageostat`
  `https://github.com/ecrc/exageostatr`
  Note: $n = 1'000'000$ then $n^3 = 10^{18}$= 1 billion billions
- Various approximation methods have been proposed in the literature to ease the computation and memory burden
- 2021/2022/2023 KAUST Competitions on Spatial Statistics for Large Datasets investigate the performance of different approximation methods with large synthetic data generated by *ExaGeoStat*

The GPU-based framework for Vecchia approximation

1. Exact log-likelihood

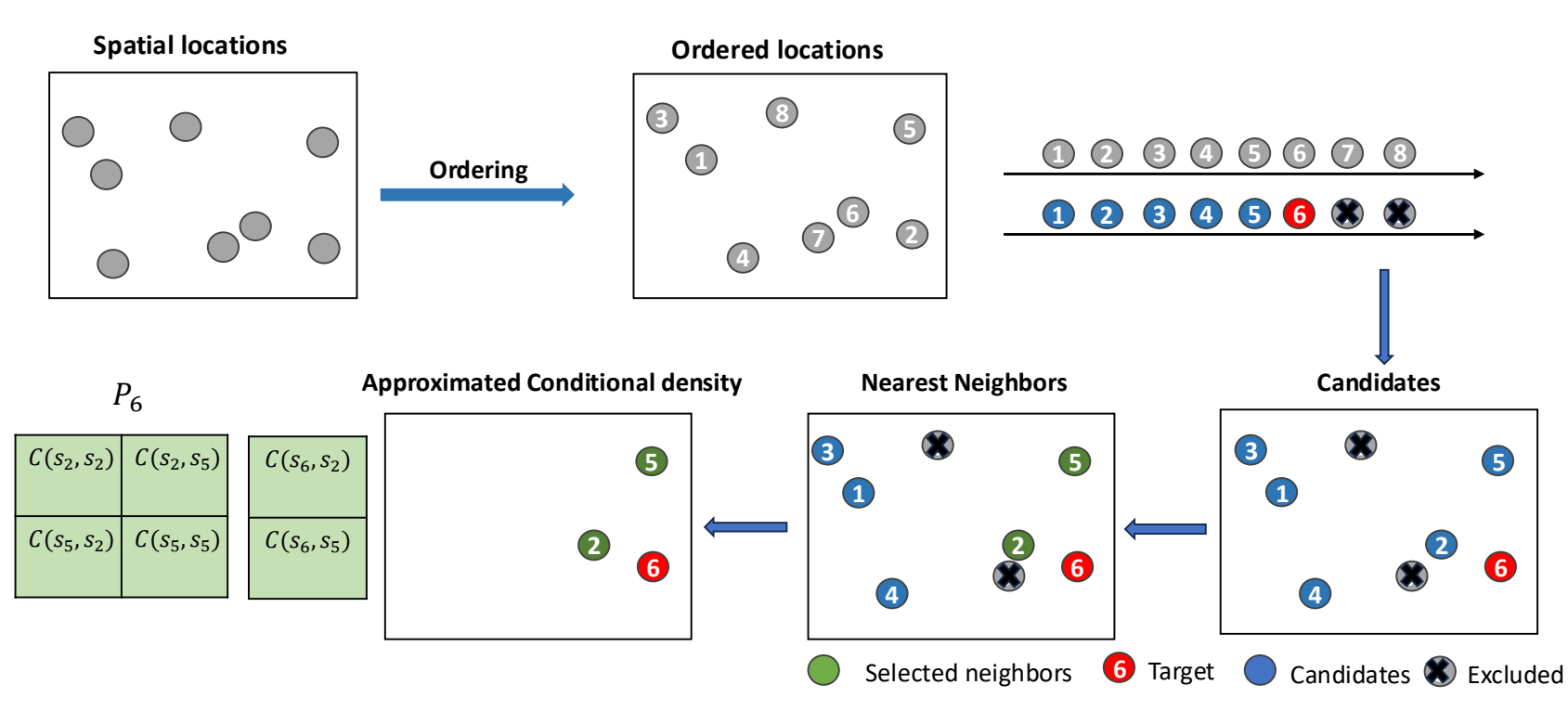
$$\ell_{\boldsymbol{\theta}}(\boldsymbol{y}) = \log p_{\boldsymbol{\theta}}\left(y_1^{\tau}\right) + \sum_{i=2}^{n} \log p_{\boldsymbol{\theta}}(y_i^{\tau} \mid \underbrace{y_1^{\tau}, \ldots, y_{i-1}^{\tau}}_{Complete})$$

2. Vecchia log-likelihood

$$\ell_{\boldsymbol{\theta}}(\boldsymbol{y}) \approx \log p_{\boldsymbol{\theta}}\left(y_1^{\tau}\right) + \sum_{i=2}^{n} \log p_{\boldsymbol{\theta}}(y_i^{\tau} \mid \underbrace{y_{j_{i1}}^{\tau}, \ldots, y_{j_{im_i}}^{\tau}}_{Subvector})$$

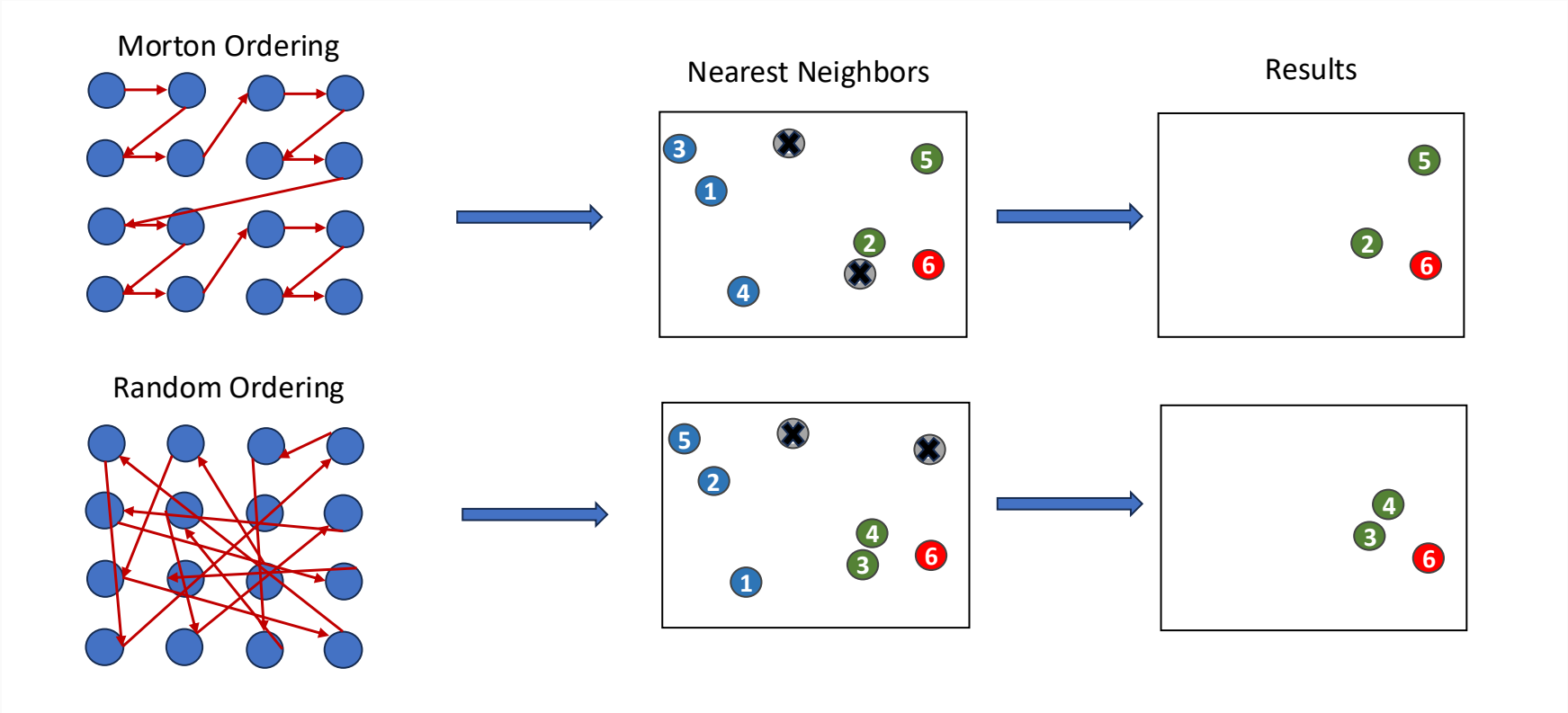
The subvector, $(y_{j_{i1}}^{\tau}, \ldots, y_{j_{im_i}}^{\tau}), m_i \leq i-1$ is defined as conditioning set. Moreover, the τ is invariant to exact likelihood, but not Vecchia likelihood

An illustrative example of Vecchia algorithm



Note: Different ordering algorithms play an important role in providing candidates for the conditioning set

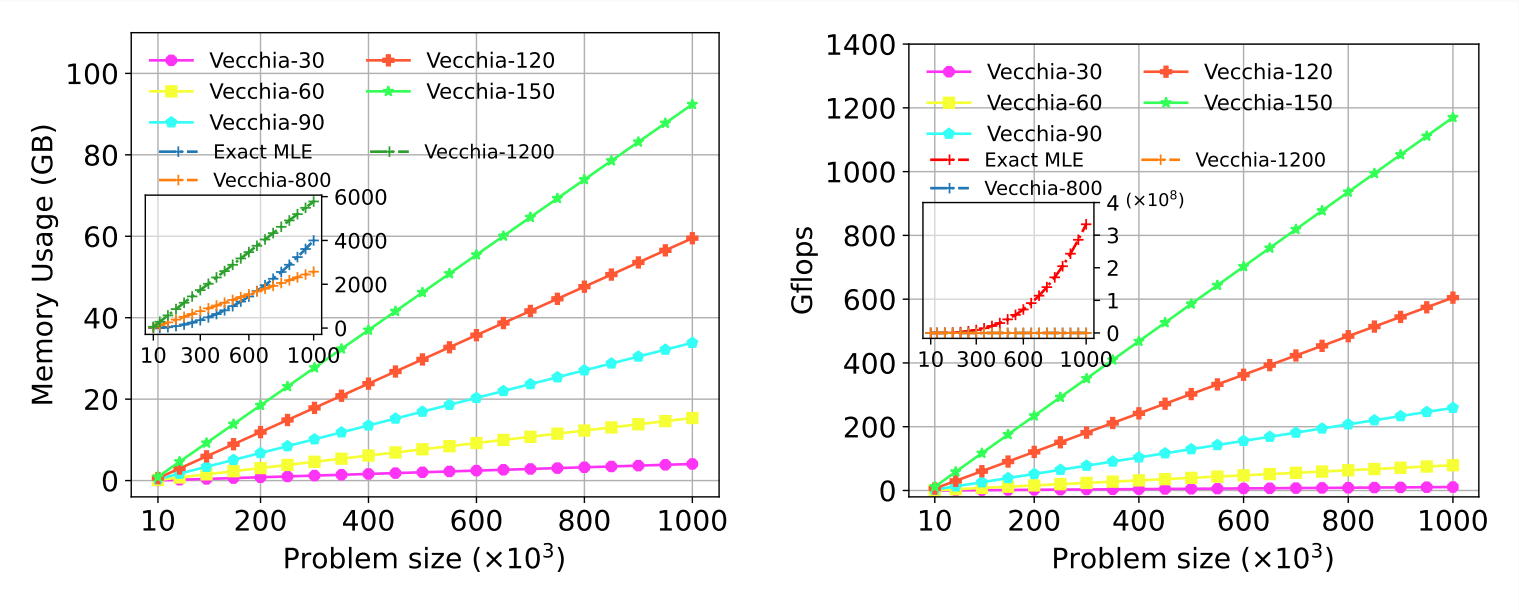
Impact of orderings on neighbors selection



Different ordering leads to different nearest neighbors, thus affecting the approximation quality (KL-divergence)

Comparing to exact MLE algorithms

1. Memory, $\mathcal{O}(n^2) \to \mathcal{O}(nm^2)$
2. Computation, $\mathcal{O}(n^3) \to \mathcal{O}(nm^3)$



Batched operations

1. Vecchia log-likelihood: independent computation tasks

$$\log p_{\boldsymbol{\theta}}\left(y_1^{\tau}\right) + \sum_{i=2}^{n} \log p_{\boldsymbol{\theta}}\left(y_i^{\tau} \mid \mathbf{y}_{J_i}^{\tau}\right)$$

2. Abundant and light-weight computation tasks,

$$\#\boldsymbol{J}_i = m \ll n,$$

the conditioning sizes $\#\boldsymbol{J}_i$, e.g., $m = 30$, and the number of tasks, e.g., 1M

- This inspires us to use the batched operations to parallel these lightweight tasks on GPU
- KBLAS routines and CUDA are used for the batched kernel computations

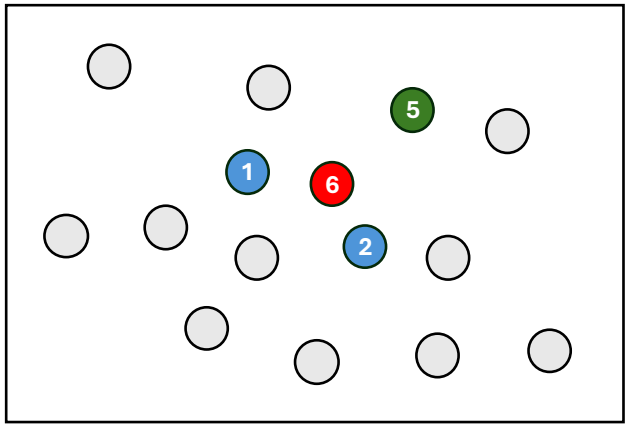
1. Innovative software: A batched implementation of the Vecchia algorithm on GPUs

   https://github.com/kaust-es/ParallelVecchiaGP

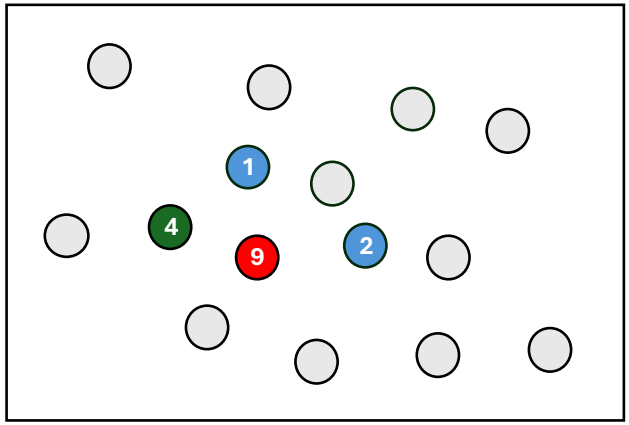
2. Computational advantages: The speedup of the proposed framework achieves 800X to 1380X compared to the exact solution on single A100 and H100 GPUs
3. Scalability: Larger problem sizes, e.g., up to 1 million geospatial locations on a single GPU while maintaining accuracy

Redundancy in Classic Vecchia (CV)

$P(y_6|y_1, y_2, y_5)$



$P(y_9|y_1, y_2, y_4)$



Target

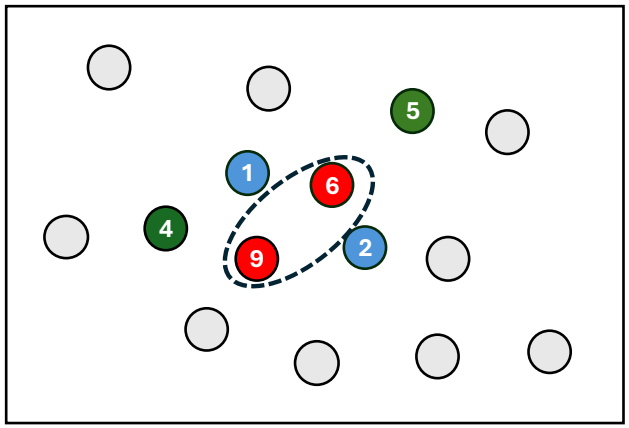
Nearest Neighbors

Not selected neighbors

Shared neighbors

The shared neighbors cause the redundancy in Cholesky decomposition, triangular linear solver.

Solution: Block Vecchia (BV) Approximation



Target

Nearest Neighbors

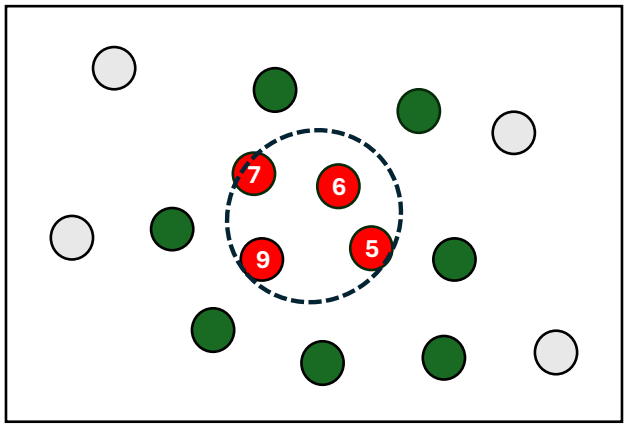
Not selected neighbors

Shared neighbors

Merge:

$$P(y_6, y_9 | y_1, y_2, y_4, y_5)$$

$$P(y_6 | y_1, y_2, y_5) \qquad P(y_9 | y_1, y_2, y_4)$$



1. BV approximation is more computationally efficient than CV, and the software is available.

   https://github.com/kaust-es/ParallelBlockVecchiaGP

2. Larger conditioning size and block count improve BV's approximation accuracy for large problem size N
3. The BV method demonstrates an approximately 80X speedup and 40X larger problem size compared to the CV algorithm

Distributed Scaled Block Vecchia (SBV) Approximation

- GP emulators are widely used in various research areas, however, the BV approximation cannot scale to even large problems, e.g., billion-level and high-dimensional problems
- Two challenges and solutions:

  ❶ High-dimensional input space in GPs $\Rightarrow$ Modified covariance functions (distances)

  ❷ Large problem size $\Rightarrow$ Distribute the BV algorithm (multiple GPUs)

Scaled Covariance Function

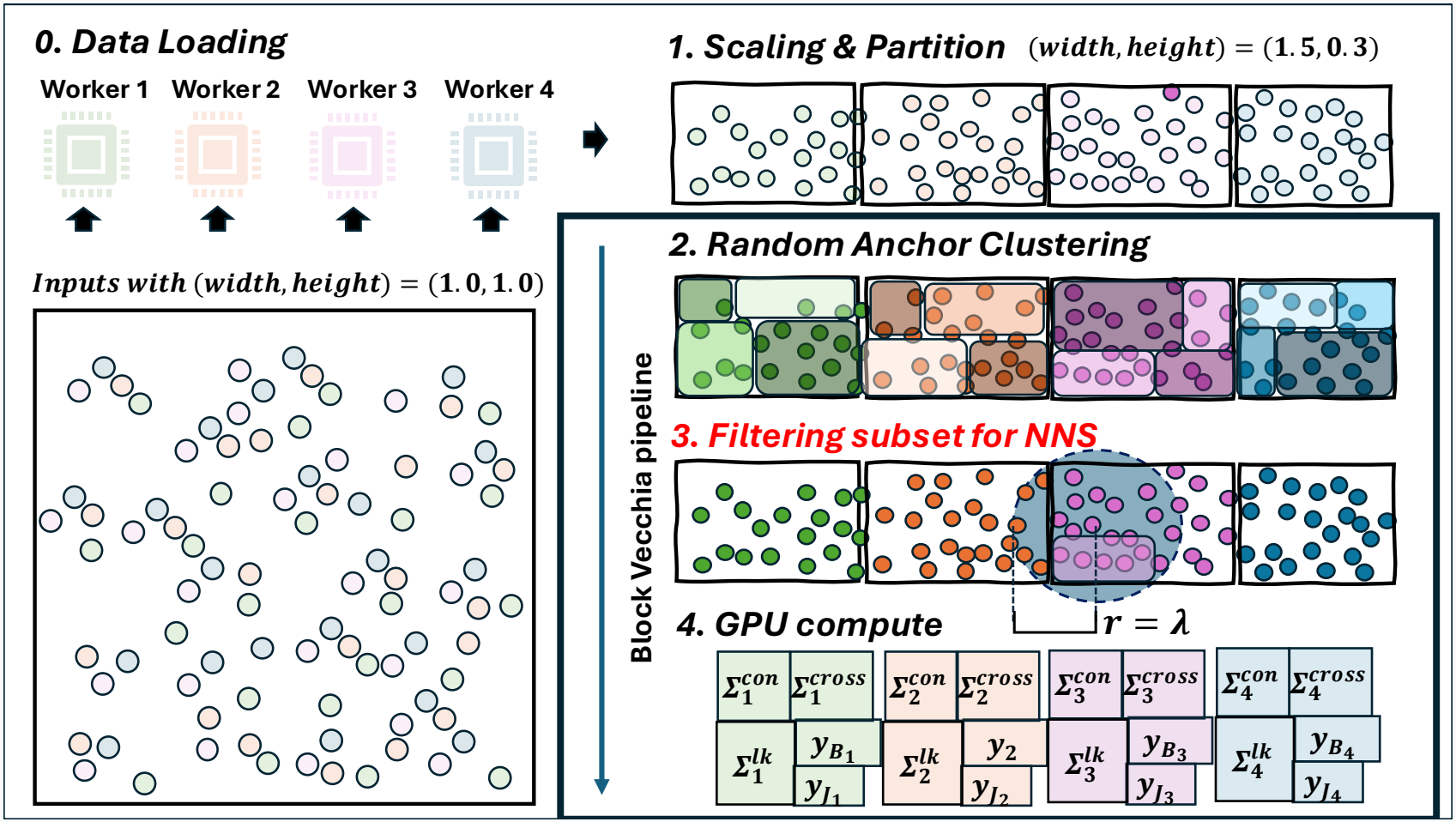
Definition

The scaled covariance function (kernel) is defined as,

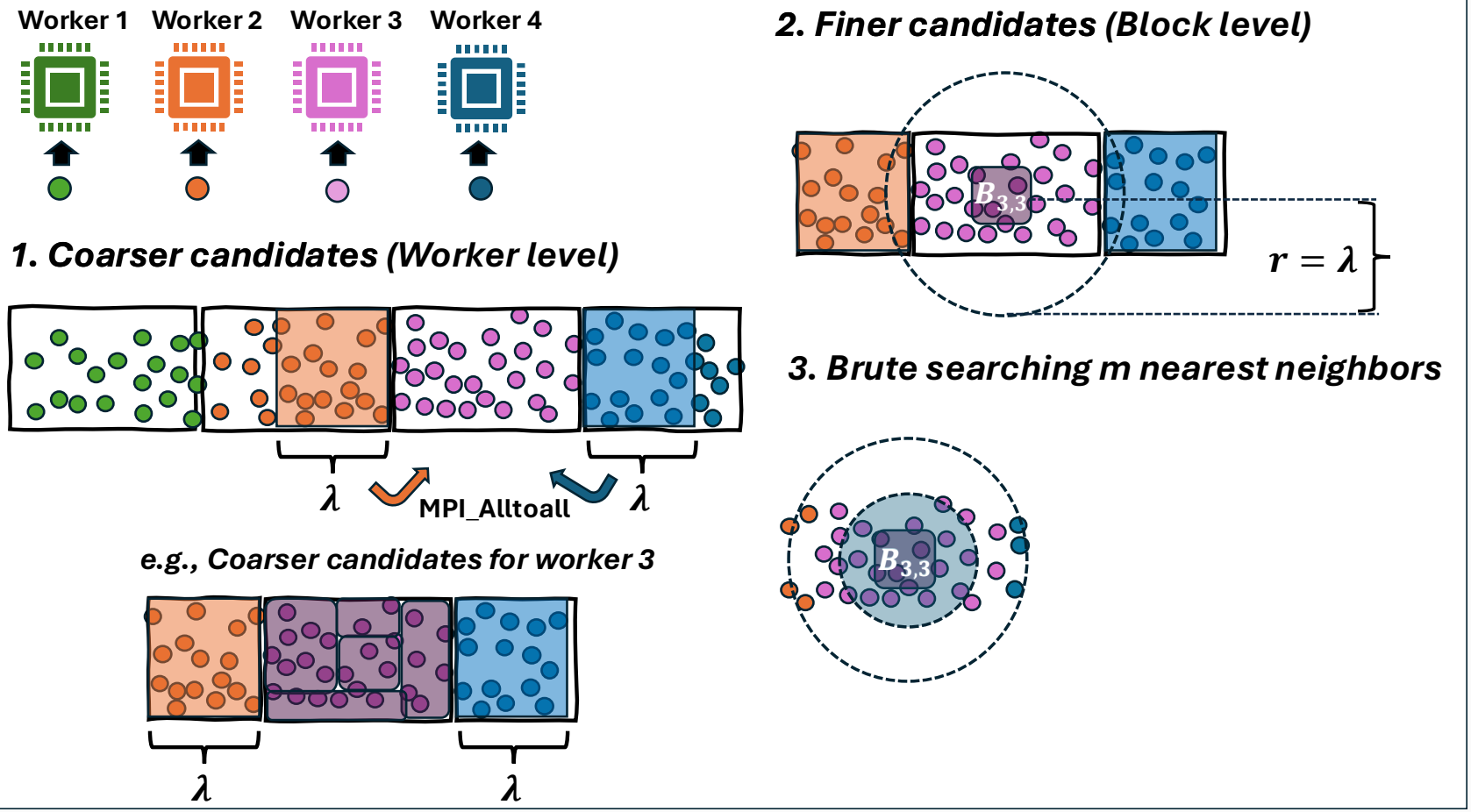
$$K_{\boldsymbol{\theta}}(\boldsymbol{x}_k, \boldsymbol{x}_{k'}) = f\left(\left(\sum_{i=1}^{d} \frac{(x_{ki} - x_{k'i})^2}{\beta_i^2}\right)^{\frac{1}{2}}\right), \tag{1}$$

where x_{ki} and $x_{k'i}$ denote the i-th components of $\boldsymbol{x}_k, \boldsymbol{x}_{k'} \in \mathbb{R}^d$, and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_d)^\top$ contains the dimension-specific range (scaling) parameters. $f(\cdot)$ is a covariance function, such as the Matérn covaraince function. If $\boldsymbol{\beta} = \mathbf{1}$, then the scaled covariance function is reduced to the isotropic covariance function. Additionally, $\frac{1}{\beta_i}$ is defined as the relevance of ith dimension.

Distributed Scaled Block Vecchia (SBV) Approximation



Filtering Subset for Nearest Neighbor Search

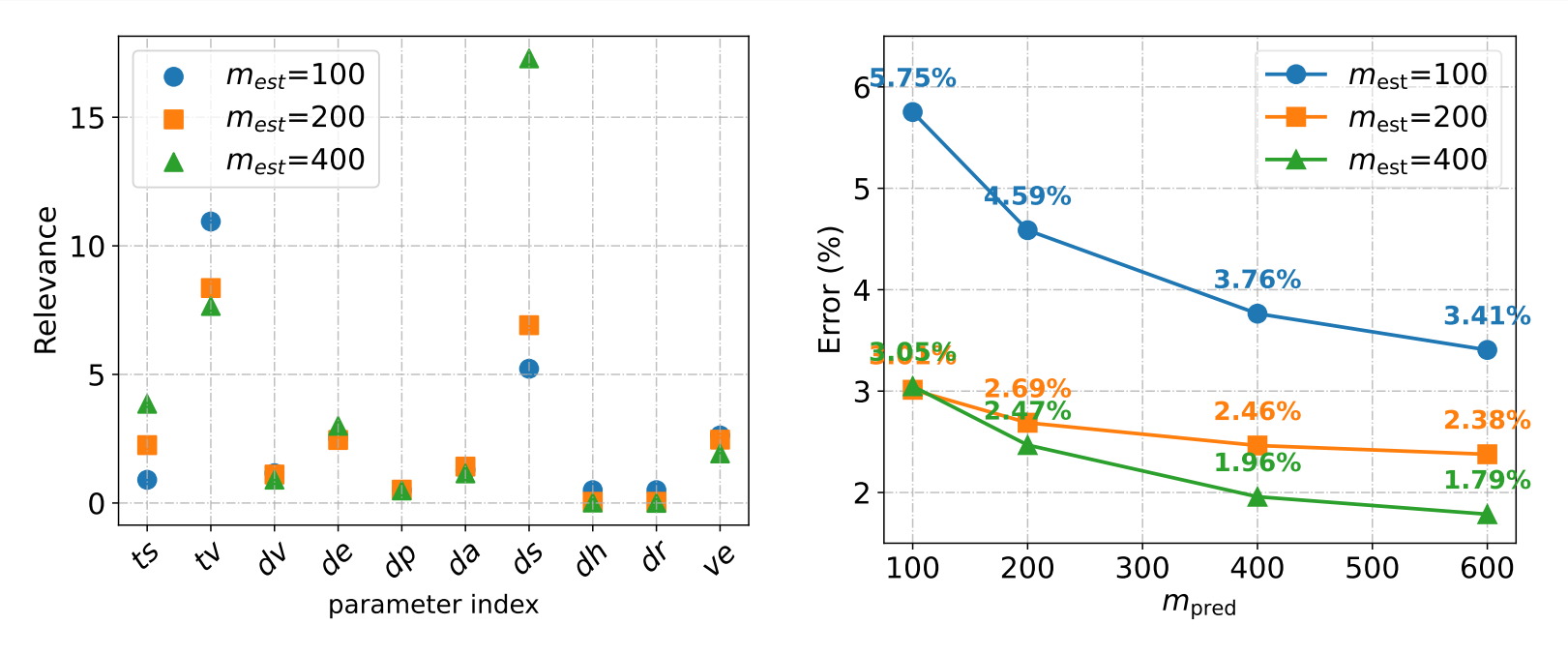


Real Application

MetaRVM is a simulator that models the spread of respiratory viruses with input variables summarized in the table. 50M sets of randomly generated input values are used to obtain the accumulated number of hospitalizations over 100 days as the output

Input	Meaning	Bound
ts	transmissibility for susceptible	(0.1, 0.9)
tv	transmissibility for vaccinated	(0.1, 0.9)
dv	mean duration in vaccinated state	(30, 90)
de	mean duration in exposed state	(1, 5)
dp	mean duration in infectious presymptomatic state	(1, 3)
da	mean duration in infectious asymptomatic state	(1, 9)
ds	mean duration in infectious symptomatic state	(1, 9)
dh	mean duration in hospitalized state	(1, 5)
dr	mean duration in recovered state	(30, 90)
ve	vaccine efficacy	(0.3, 0.8)

Estimation and Prediction

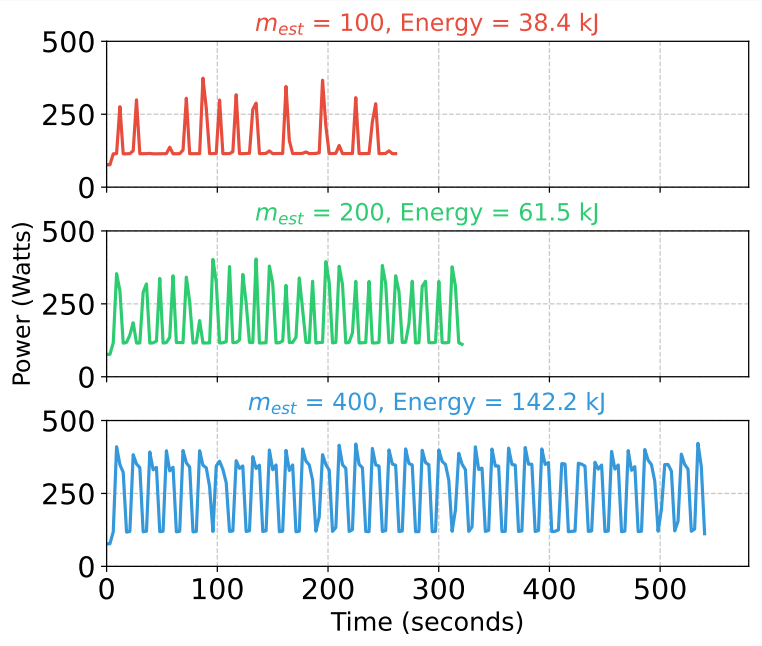


(e) Estimated parameters. **(f)** RMSPE.

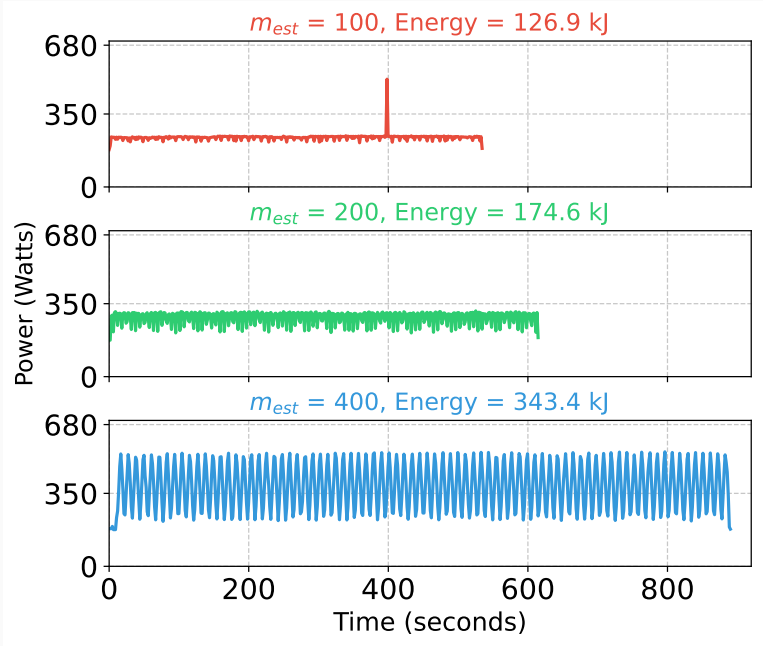
Remark

1) Relevance dh and dr are close to 0, as these input variables are not involved in the accumulated number of hospitalizations in the `MetaRVM` simulator; 2) The two most relevant input variables are ds and tv.

Power Consumption



(g) NVIDIA A100 GPU (2M points).



(h) NVIDIA Hopper GPU (5M points).

Figure 1: Power consumption/energy (kJ) on a single GPU for two NVIDIA GPUs over 500 iterations per m_{est}. (A full charge of the iPhone 16 Pro Max 64kJ)

Remark

SBV consumes around 0.5% the energy compared to exact GPs for a single iteration on those systems, while handling 16X-40X problems.

1. Scale Block Vecchia (SBV) approximation is suitable and efficient approximation of high-dim GP

   https://github.com/kaust-es/ParallelScaledBlockVecchiaGP
2. SBV improves accuracy with larger conditioning size
3. SBV exhibits approximately linear scalability
4. SBV is energy consumption friendly

2.3 Spatial and space-time DeepKriging

In geostatistics, Kriging is commonly used for spatial prediction

Features of Kriging prediction:

- it provides the best linear unbiased predictor (BLUP)
- it involves modeling the mean and the covariance function of a spatial process

Drawbacks of Kriging prediction:

- it is typically not optimal for non-Gaussian data, e.g., skewed, heavy-tailed, count, or categorical data
- it is computationally expensive for massive datasets (for both estimation and prediction)

Deep learning provides a computationally scalable methodology for a variety of data types and non-linear prediction

Challenges in spatial prediction

Why classical deep learning methods cannot be directly used for spatial prediction? In real applications:

1. observations are often spatially sparse with inadequate spatial coverage
2. only one (few) replicated spatial data can be collected
3. only few covariates (features) are observed at the location to be predicted
4. processes typically exhibit spatial dependence

First research goal: Develop deep learning-based methodology suitable for spatial prediction

Challenges in forecasting

Probabilistic forecasting

1. is beyond single-valued or point forecasts
2. is about distributional forecasts
3. extrapolation requires special treatment

Second research goal: Develop deep learning-based methodology suitable for probabilistic forecasting

Deep learning in spatial prediction

- Suppose the spatial process X is observed at n locations, $\mathbf{X} = (X(\mathbf{s}_1), \ldots, X(\mathbf{s}_n))^T$ is the data vector, and $\mathbf{x}_f(\mathbf{s}_i)$ are observed covariates/features. The predictor of DNN at $\mathbf{s}_0$ is $\widehat{X}_{DNN}(\mathbf{s}_0, \mathbf{X}) = f^{DNN}(\mathbf{x}_f(\mathbf{s}_0), \widehat{\boldsymbol{\theta}})$ where

$$\widehat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} L\{f^{DNN}(\mathbf{x}_f(\mathbf{s}_i), \boldsymbol{\theta}), X(\mathbf{s}_i)\}$$

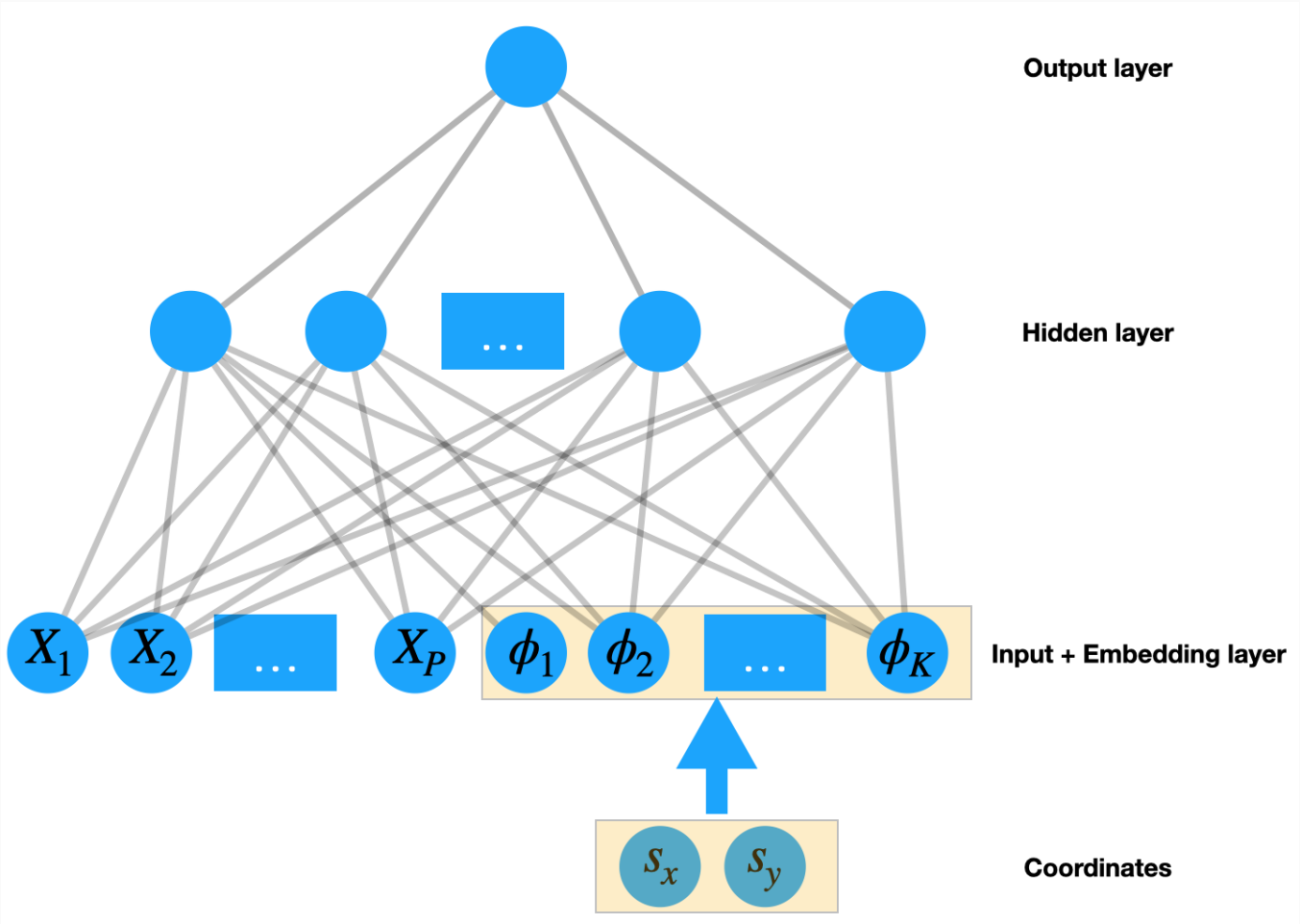
  where $L(\cdot, \cdot)$ is the loss function, and $\widehat{\boldsymbol{\theta}}$ is obtained by training
- How to account for the spatial information in the features? A natural way is adding coordinates (e.g., longitude and latitude) to the features
  - The final predictor can be a very complex function over $\mathbf{s}_0$
  - It may take a huge effort to achieve a good approximation

Motivation of DeepKriging

$X(\mathbf{s}) = \mathbf{x}(\mathbf{s})^T\boldsymbol{\beta} + \nu(\mathbf{s})$, where $\nu(\mathbf{s})$ is a zero-mean spatial process

- Karhunen–Loève (KL) theorem (empirical version): the spatial process $\nu(\mathbf{s})$ can be approximated by a finite weighted sum of basis functions, i.e., $\widehat{\nu}(\mathbf{s}) = \sum_{k=1}^{K} \beta_k \phi_k(\mathbf{s})$
  - **The idea of DeepKriging**: rather than only including the spatial coordinates $\mathbf{s}$ in the features, we transform the n coordinates to K basis functions
  - In deep learning, this corresponds to a n to K embedding layer in the network before we pass the data to the hidden layers

Structure of DeepKriging



Visualization of the DeepKriging structure in 2D spatial prediction based on a three-layer DNN

Space-time DeepKriging

- Spatio-temporal interpolation to obtain gridded data
  - It is a direct extension of DeepKriging
  - The embedding layer consists of spatio-temporal basis functions
- Spatio-temporal forecasting
  - We choose the long short-term memory (LSTM) network: LSTM is a variety of recurrent neural networks (RNNs) that are capable of learning long-term temporal dependencies, especially in sequence prediction problems
  - Limitation: Although it is highly effective for capturing temporal dependence, it does not use information from other locations
  - Convolutional LSTM: includes data from neighboring locations by passing the CNN layer as the input to the LSTM layer

Probabilistic forecasting

QLSTM and QConvLSTM (*Nag et al., Spatial Statistics, 2023*)

- We propose a quantile-based loss for training the LSTM layers

$$\rho_\tau(v) = v(\tau - I(v < 0))$$

where τ is quantile level

- We also propose an activation function for the output layer to avoid quantile crossing

$$\Psi(\tau, x) = \begin{cases} x & \text{for } \tau = 0.5, \\ f_{Constant} + \frac{\lambda(\tau - 0.5)}{1+e^{-x}} & \text{for } \tau > 0.5, \\ f_{Constant} - \frac{\lambda(0.5 - \tau)}{1+e^{-x}} & \text{for } \tau < 0.5, \end{cases}$$

where $f_{Constant}$ is the estimated median. One can choose λ proportional to the range of the observations

2022 Competition on spatial statistics for large datasets

- Estimation and prediction for nonstationary, space-time, and multivariate Gaussian processes
- Launched March 1, 2022; Ended May 1, 2022
- Hosted the competition on the Kaggle machine learning and data science platform
- 20 research teams worldwide registered

Competition results

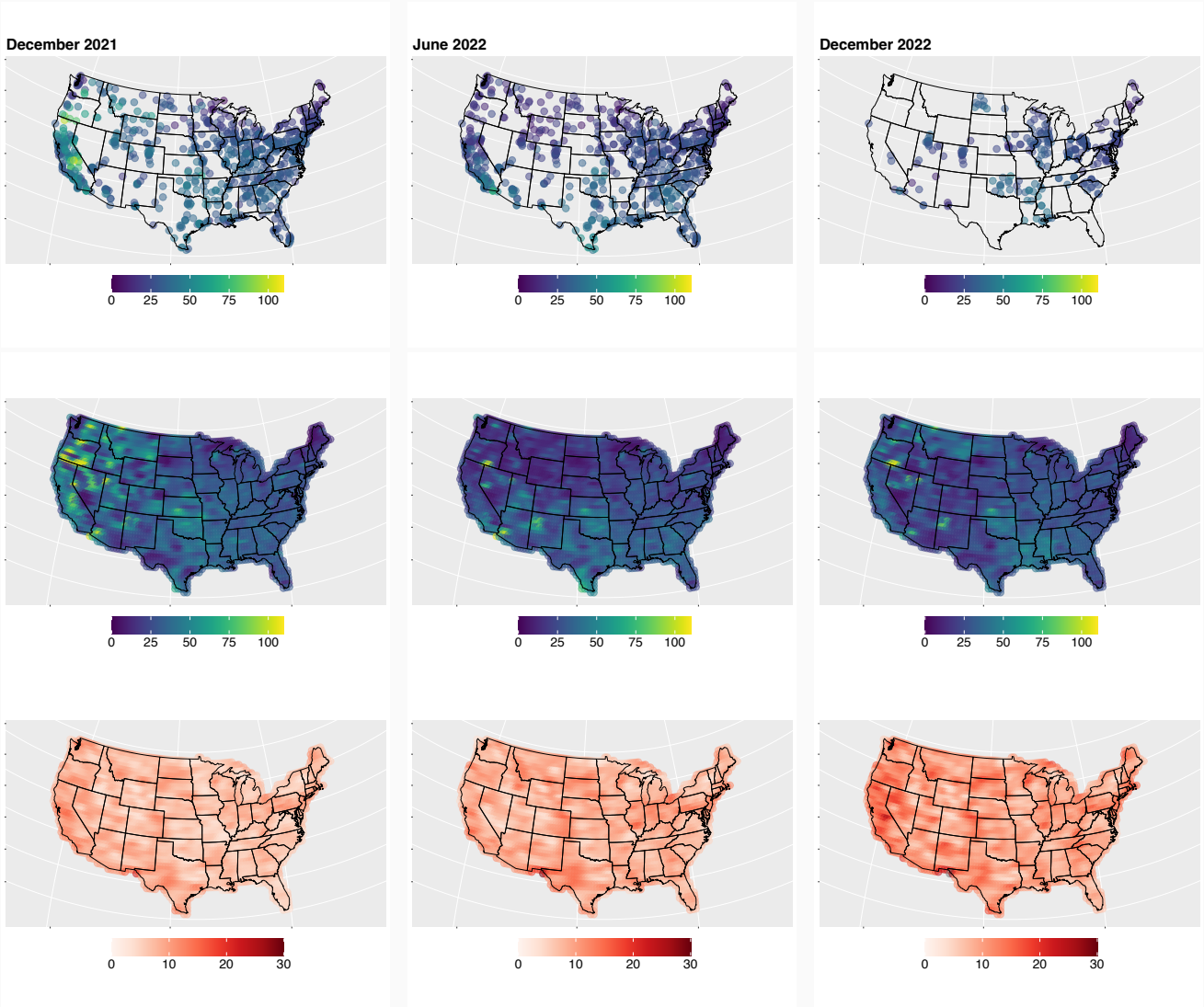
- The proposed Space-time DeepKriging won the competition for 100K and 1M space-time locations
- 22% and 9% improvement over the second best, respectively
- Other competing approaches include Vecchia's approximation by using the GpGp package (Guinness et al., 2021, ranked second) and block-composite likelihood (Eidsvik et al., 2014, ranked third)

Space-time: Application to $PM_{2.5}$ in the US

- Time period: from January 1998 to December 2022
- Time resolution: monthly, in total 286 months
- Spatial dimension: There are on average 1900 spatial observations per month

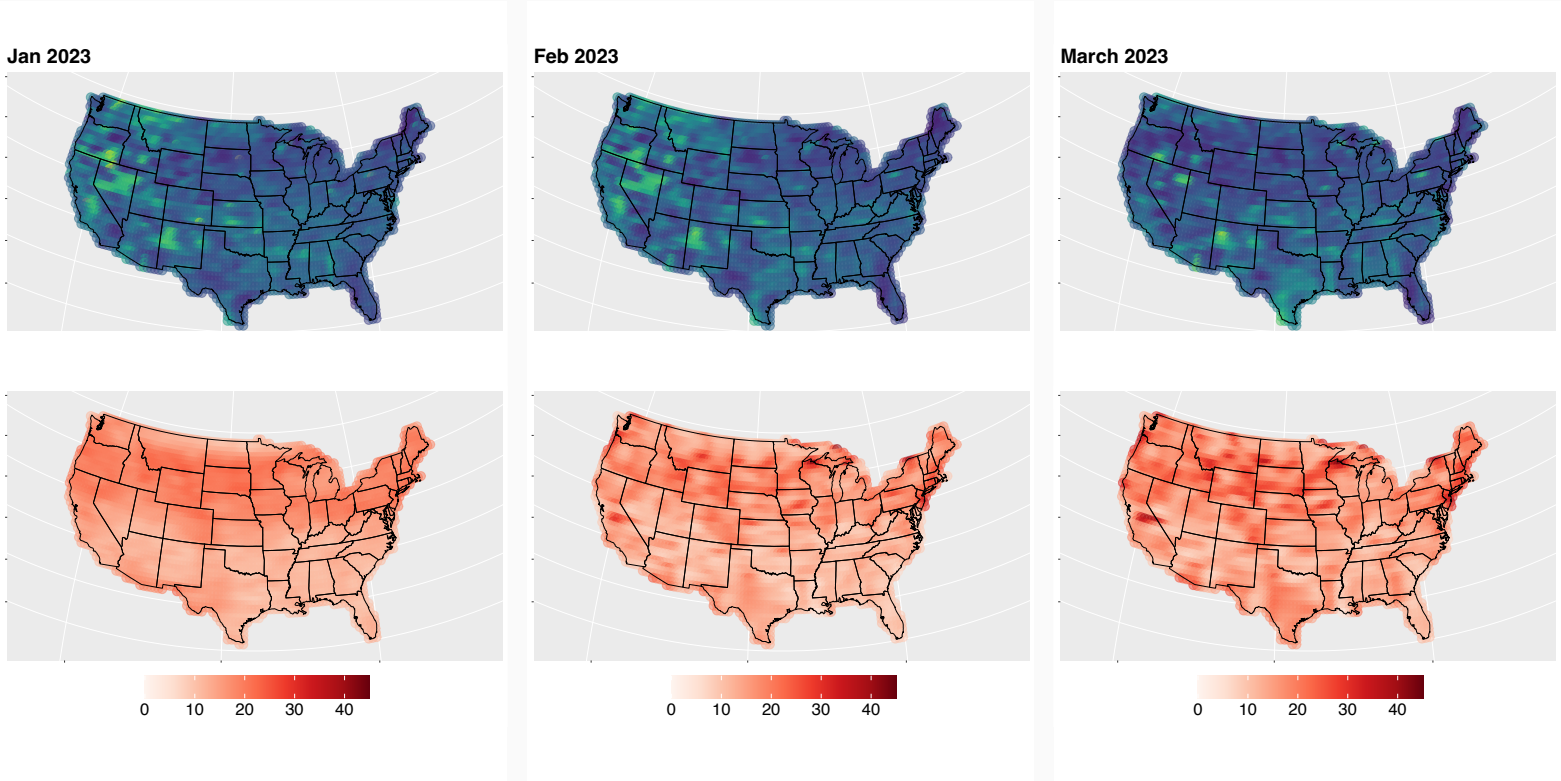
Interpolation of $PM_{2.5}$

For three selected months with the length of the 90% prediction interval



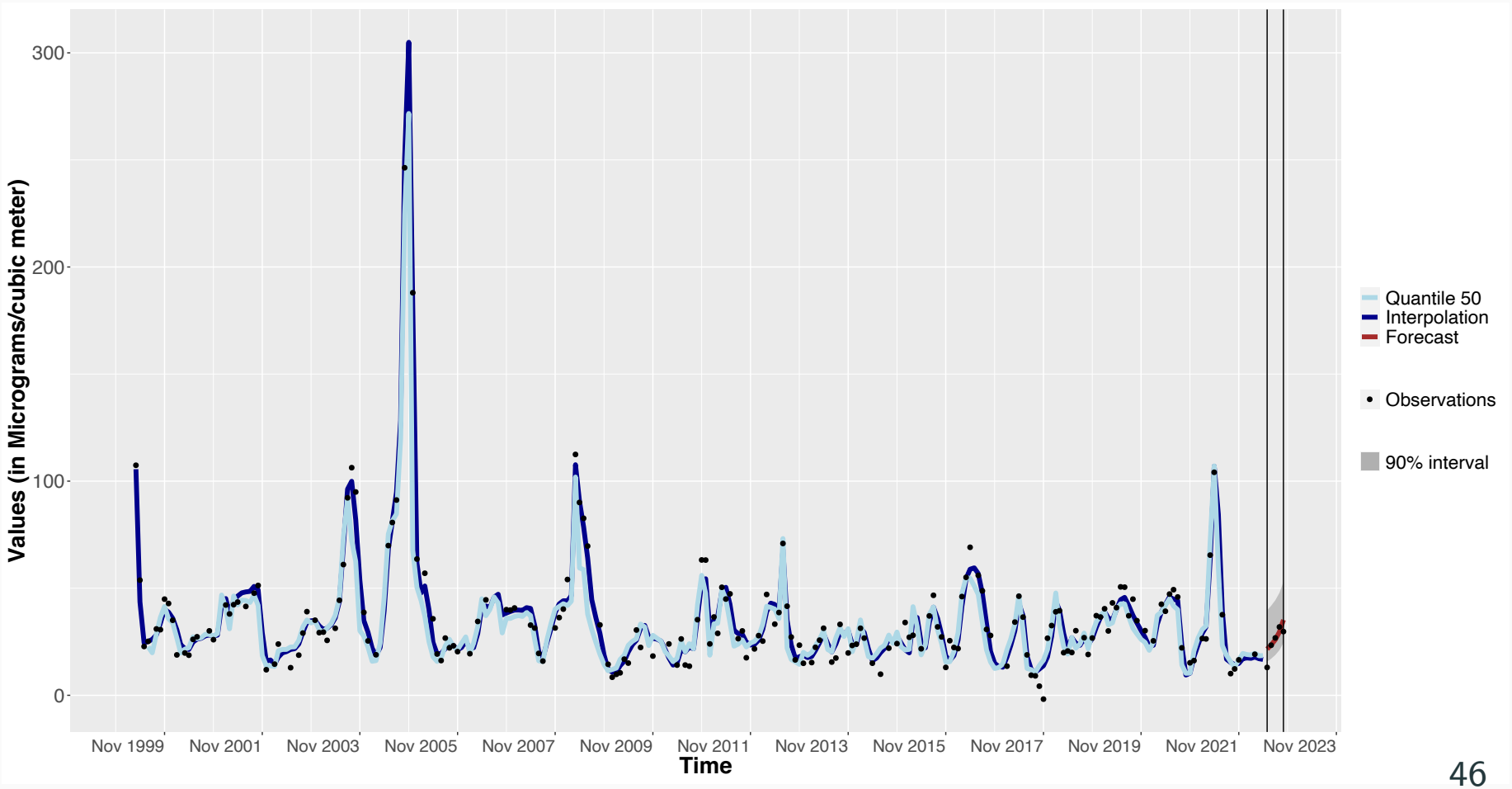
Forecasting $PM_{2.5}$

Forecasting for the first three months of 2023 with the length of the 90% forecast interval



Forecast using QConvLSTM of $PM_{2.5}$ for San Fransisco

Probabilistic forecasting with 90% forecast interval for the final five observed months of 2022



Key References:

- Q. Pan, S. Abdulah, M. G. Genton, D. E. Keyes, H. Ltaief, and Y. Sun. GPU-Accelerated Vecchia Approximations of Gaussian Processes for Geospatial Data using Batched Matrix Computations. In *ISC High Performance 2024 Research Paper Proceedings (39th International Conference)* (pp. 1-12). May 2024.
- S. Abdulah, H. Ltaief, Y. Sun, M. G. Genton, and D. E. Keyes. ExaGeoStat: A high performance unified framework for geostatistics on manycore architectures. *IEEE Transactions on Parallel and Distributed Systems*, 29:2771– 2784, 2018.
- Q. Pan, S. Abdulah, M. G. Genton and Y. Sun. (2025). Block Vecchia Approximation for Scalable and Efficient Gaussian Process Computations. *Technometrics*, to appear.
- Q. Pan, S. Abdulah, M. Abduljabbar, H. Ltaief, A. Herten, M. Bode, M. Pratola , A. Fadikar, M. G. Genton, D. E. Keyes and Y. Sun. (2025). Scaled Block Vecchia Approximation for High-Dimensional Gaussian Process Emulation on GPUs. *arXiv preprint* arXiv:2504.12004.
- W. Chen, Y. Li, B. Reich, and Y. Sun. DeepKriging: Spatially dependent deep neural networks for spatial prediction. *Statistica Sinica*, 34:291–311, 2024.
- P. Nag, Y. Sun, and B. Reich. Spatio-temporal DeepKriging for interpolation and probabilistic forecasting. *Spatial Statistics*, 57:100773, 2023.

Summary