Journal of Parallel and Distributed Computing 180 (2023) 104715

Contents lists available at ScienceDirect

Journal of Parallel and Distributed Computing

journal homepage: www.elsevier.com/locate/jpdc

Tile low-rank approximations of non-Gaussian space and space-time Tukey g-and-h random field likelihoods and predictions on large-scale systems

Sagnik Mondal[a,b], Sameh Abdulah[a,c,*], Hatem Ltaief[a,c], Ying Sun[a,b,c], Marc G. Genton[a,b,c], David E. Keyes[a,c]

[a] Computer, Electrical, and Mathematical Sciences and Engineering Division, King Abdullah University of Science and Technology, Thuwal, 23955-6900, Saudi Arabia
[b] Statistics Program, King Abdullah University of Science and Technology, Thuwal, 23955-6900, Saudi Arabia
[c] Extreme Computing Research Center, King Abdullah University of Science and Technology, Thuwal, 23955-6900, Saudi Arabia

ARTICLE INFO

Article history:
Received 7 October 2022
Received in revised form 30 March 2023
Accepted 9 May 2023
Available online 19 May 2023

Keywords:
Climate/weather data
Large-scale geostatistical modeling
Likelihood approximation
Non-Gaussian random field
Tile Low-Rank (TLR) approximation

ABSTRACT

Large-scale statistical modeling has become necessary with the vast flood of geospace data coming from various sources. In space statistics, the Maximum Likelihood Estimation (MLE) is widely considered for modeling geospace data by estimating a set of statistical parameters related to a predefined covariance function. This covariance function describes the correlation between a set of geospace locations where the main goal is to model given data samples and impute missing data. Climate/weather modeling is a prevalent application for the MLE operation where data interpolation and forecasting are highly required. In the literature, the Gaussian random field is often used to describe geospace data as one of the most popular models for MLE. However, real-life datasets are often skewed and/or have extreme values, and non-Gaussian random field models are more appropriate for capturing such features. In this work, we provide an exact and approximate parallel implementation of the well-known Tukey g-and-h (TGH) non-Gaussian random field in the context of climate/weather applications. The proposed implementation alleviates the computation complexity of the log-likelihood function, which requires $\mathcal{O}(n^2)$ storage and $\mathcal{O}(n^3)$ operations, where N is the number of geospace locations, M is the number of time slots, and $n = N \times M$. Based on tile low-rank (TLR) approximations, our implementation of the TGH model can tackle large-scale problems. Furthermore, we rely on task-based programming models and dynamic runtime systems to provide fast execution for the MLE operation in space and space-time cases. We assess the performance and accuracy of the proposed implementations using synthetic space and space-time datasets up to $800K$. We also consider a 12-month precipitation dataset in Germany to demonstrate the advantage of using non-Gaussian over Gaussian random field models. We evaluate the prediction accuracy of the TGH model on the precipitation dataset using the Probability Integral Transformation (PIT) tool showing that the TGH model outperforms the Gaussian modeling in the real dataset. Moreover, our performance assessment indicates that TLR computations allow solving larger matrix sizes while preserving the required accuracy for prediction. The TLR-based approximation shows a speedup up to $7.29X$ and $2.96X$ over the exact solution.

1. Introduction

Geostatistical space and space-time modeling techniques from the statistics literature are becoming very popular for analyzing climate/weather data over space and space-time because of their wide range of applications. The Gaussian model is among the most broadly used space and space-time statistics models. Indeed, it is more straightforward to fit the Gaussian model to geospace data, draw inferences, and make predictions compared to the physics-based modeling of space and space-time processes that solve multiple discretized partial differential equations. We fit the Gaussian model by finding the Maximum Likelihood Estimate (MLE) of the model parameters obtained by maximizing a predefined likelihood function. To compute this likelihood function for Gaussian models for a univariate dataset with N space locations and M time slots,

[*] Corresponding author.
E-mail address: sameh.abdulah@kaust.edu.sa (S. Abdulah).

https://doi.org/10.1016/j.jpdc.2023.104715
0743-7315/

with $n = N \times M$, we need to invert an $n \times n$ symmetric, positive definite covariance matrix by using the Cholesky factorization. The latter requires $\mathcal{O}(n^3)$ computation complexity and $\mathcal{O}(n^2)$ memory space complexity. Naturally, if n is large, which is most often the case for climate and environmental datasets nowadays, this computation becomes very daunting. This fact emphasizes the necessity of parallel computation for these applications where compute-intensive kernels dominate the execution. Another solution is to provide an algebraic method to approximate the calculation of the covariance function inversion. In particular, low-rank approximation of the covariance matrix has been shown to be effective in the space statistics framework by Stein et al. [43], Abdulah et al. [2], and Salvaña et al. [40] and in the space-time statistics framework by Salvaña et al. [41]. With the low-rank approximation of the covariance matrix, one can counter the memory space and the computation complexity for problems with a very high number of space-time locations. In this work, we combine parallel processing and algebraic approximation method to settle the complexity of computing the likelihood function.

The assumptions of the Gaussian model for real geospace data are relatively strong. The Gaussian model assumes the process has a symmetric distribution at each space location for each time point. Furthermore, the Gaussian model also fails to capture tail-thickness, which often is a feature of real-life environmental and climate data. Hence, we need more general space-time models for handling skewed and tail-heavy random fields. In space statistics, one popular approach for capturing non-Gaussian features is to apply a nonlinear transformation on a space Gaussian random field. The square-root transformation by Berrocal et al. [13] and Yan and Genton [50], the Box-Cox transformation by De Oliveira et al. [22], and the log-normal transformation by Rios and Tobar [38] are a few examples of such non-Gaussian space models obtained from this transformation approach. However, these transformations are applicable only to all-positive datasets and are ineffective when we have negative values in some locations for any particular dataset. Moreover, such transformations for arbitrary datasets are not always easy to find. The authors in [49] proposed a transformation-based space non-Gaussian random field for which the transformation itself is parameterized. This random field is known as the Tukey g-and-h (TGH) random field. Consequently, one can avoid foreguessing the transformation for an arbitrary dataset and directly fit this model. The parameters of the transformation will adapt themselves according to the dataset. Xu and Genton [49] have shown that many well-known non-Gaussian models obtained by this transformation approach are indeed exceptional cases of the Tukey g-and-h random field.

This work proposes a parallel implementation for the Tukey g-and-h random field in exact and approximate forms. The implementation involves non-Gaussian space and space-time modeling for large climate and environmental applications datasets. We rely on the task-based programming model and runtime systems to provide a robust implementation that can easily be ported to different parallel hardware architectures. Precisely, we rely on the Chameleon [17] library for the dense linear algebra operations required to compute the likelihood function. In addition, we rely on the StarPU runtime system [11] to orchestrate computational tasks. We also provide an approximate modeling framework based on Tile Low-Rank (TLR) approximation. For TLR, we rely on the HiCMA library [3] for matrix compression and linear algebra operations. The TLR-based implementation speeds the execution up to 7.29X and 2.96X on shared-memory and distributed-memory systems, respectively, compared to the exact implementation.

Our main contributions can be summarized as follows: 1) we propose a parallel implementation of the exact computation of the TGH likelihoods and predictions to model a wide range of space and real space-time datasets on leading-edge parallel systems; 2) we provide parallel TLR-based approximations to the exact predictive space and space-time models to allow better compression to the covariance matrix and faster computation to its inverse; 3) we provide a qualitative comparison between Gaussian and non-Gaussian modeling on large synthetic and real datasets to highlight the advantages of using non-Gaussian modeling on geospace applications; 4) we assess the performance of the provided exact and TLR approximation using Chameleon/StarPU implementations on shared-memory and distributed-memory systems; 5) we demonstrate the quality of our implementation using space and space-time precipitation datasets in Germany with a covariance matrix of dimensions up to 300K. The results show the effectiveness of the non-Gaussian modeling and TLR approximation when considering real geospace datasets.

The remainder of this paper is organized as follows. Section 2 covers related work. Section 3 gives an overview and background of our problem. Section 4 describes the TGH algorithm and our parallel implementation for modeling and prediction. Section 5 analyzes accuracy and performance using synthetic and real datasets in the context of climate/weather applications. Finally, we conclude in Section 6.

2. Related work

This section briefly reviews some existing popular space and space-time non-Gaussian models in the statistical literature. In space statistics, many non-Gaussian distributions have been used for constructing various non-Gaussian random fields. For example, the skew-Gaussian distribution [32], Student's t-distribution [39], skew-t distribution [14], log-skew-elliptical distribution [35]. Palacios and Steel [37] built non-Gaussian models using a scale mixture of the Gaussian random field. The theory of copulas has also been used to construct non-Gaussian random fields [34]. Using stochastic partial differential equations, [48] created a class of non-Gaussian space Matérn fields. The idea of constructing non-Gaussian models by taking a nonlinear transformation of a Gaussian process has been discussed before. A few other related works where the transformation approach has been used to construct non-Gaussian models are the logarithm transformation by De Oliveira [21], the square-root transformation by Johns et al. [30], and the power transformations by Allcroft and Glasbey [7].

Some ideas from space non-Gaussian models can be extended to the space-time regime. For example, Fonseca and Steel [24] proposed a non-Gaussian space-time model using the scale mixture of the Gaussian space-time process. The non-Gaussian space models based on copulas can also be extended to non-Gaussian space-time processes. Krupskii and Genton [33] extended the factor copula model to the space-time setting. Tang et al. [46] proposed a copula-based semiparametric space-time model for analyzing non-Gaussian space-time data. The idea of the trans-Gaussian random field can also be extended to the space-time setting by applying nonlinear transformations to a Gaussian space-time random process. Hence, all the afore-discussed transformations can be applied to constructing space-time non-Gaussian processes. In recent years, other proposals for constructing non-Gaussian space-time processes have been proposed. For instance, [44] introduced quantile function estimators for space and temporal data with a fused adaptive Lasso penalty for modeling the dependence in space and time. Hazra et al. [29] proposed a Dirichlet process mixture of space skew-t processes for modeling space-time extremes. Barzegar et al. [12] developed a class of non-Gaussian space-time process models based on the closed skew-normal distribution. Tagle et al. [45] provided a space-time non-Gaussian model that assumes an unconditional skew-t data distribution with a convenient hierarchical representation. Among all the non-Gaussian space-time models, we select the space-time TGH model for large-scale implementation. The space-time TGH model is defined by taking Tukey's g-and-h transformation on a Gaussian space-time process. We select the TGH model because of its easily-interpretable parameterization and its many appealing statistical properties for large-scale implementation.

Similar to the space-time Gaussian random process, the main computational challenge for fitting the space-time TGH model is to evaluate the log-likelihood function for a given parameter vector. For both models, the computation of the log-likelihood requires dense matrix inversion. The matrix in question is a symmetric matrix of size $n \times n$, n being the number of space-time locations for a particular dataset. This inversion is almost impossible in sequential computation when n is large. Given that nearly all the datasets on climate and environment applications consist of very large space-time locations, parallel computation frameworks are needed for space-time statistical analysis. Many space statistics techniques have already been implemented in parallel computation frameworks. For example, the space prediction (or kriging) routine has been implemented for the Gaussian model in parallel computing by Goulart et al. [28] and Cheng [18] using Message Passing Interface (MPI), OpenMP, Parallel Virtual Machines (PVMs), and/or Graphics Processing Units (GPUs). The authors in [1] provided a parallel framework for computing the exact log-likelihood of the Gaussian model using dense linear algebra task-based algorithms and dynamic runtime systems. This framework has been extended to the non-Gaussian settings [36] by providing the space TGH modeling framework. Furthermore, the framework by Abdulah et al. [1] has been extended to accommodate the multivariate space Gaussian modeling by Salvaña et al. [40] and to adjust the Gaussian space-time modeling by Salvaña et al. [41].

Various matrix approximation techniques have tackled the computation and storage complexity problems. Furrer et al. [25] and Sang and Huang [42] used the covariance tapering method to approximate the large covariance matrix where the correlation between two points is very far apart is assumed to be zero to make the large covariance matrix sparse. Other approximation techniques include low-rank approximations of the covariance matrix. For example, Abdulah et al. [2], and Geoga et al. [26] used Tile Low-Rank (TLR) and Hierarchically Off-Diagonal Low-Rank (HODLR) approximations of the covariance matrix for computation of the log-likelihood of the Gaussian model. Mondal et al. [36] have compared the two low-rank approximation methods, the TLR and the HODLR, in the context of TGH space modeling.

3. Problem statement

This section overviews the Tukey g-and-h space and space-time models. We also show the log-likelihood function and prediction function for these two models. Finally, we describe the construction of the geospace covariance matrix under the Tile Low-Rank (TLR) approximation technique.

3.1. Definition of Tukey g-and-h space random fields

The Tukey g-and-h space random field is introduced by Xu and Genton [49] by applying Tukey's g-and-h transformation on a latent Gaussian random field. The Tukey's g-and-h transformation

$$\tau_{g,h}(z) = g^{-1}\{\exp(gz)-1\}\exp(hz^2/2), \tag{1}$$

is a monotonic function of z for $g \in \mathbb{R}$ and $h \geq 0$. From now on, the values of quantities involving g at $g=0$ are defined as their limit when $g \to 0$. Tukey's g-and-h transformation on a random variable following a Gaussian distribution induces skewness and tail-thickness in the distribution. The parameter g is responsible for the skewness in the distribution, the sign of skewness is the same as the sign of g, and the parameter h is responsible for the tail-thickness in the distribution.

The standard Tukey g-and-h space random field is defined as

$$T(\boldsymbol{s}) = \tau_{g,h}\{Z(\boldsymbol{s})\}, \tag{2}$$

where, $Z(\boldsymbol{s})$, $\boldsymbol{s} \in \mathbb{R}^d$, $d \geq 1$ is a standard Gaussian random field, i.e. $\mathbb{E}\{Z(\boldsymbol{s})\} = 0$ and $\mathbb{V}\mathrm{ar}\{Z(\boldsymbol{s})\} = 1$, with some correlation function $\mathbb{C}\mathrm{or}\{Z(\boldsymbol{s}_1), Z(\boldsymbol{s}_2)\} = \rho_Z(\boldsymbol{s}_1, \boldsymbol{s}_2)$. Unlike the Gaussian random field, the TGH random field can have skewed and heavy-tailed marginals. Moreover, when g and h are zero, we return the Gaussian random field. Hence, the TGH random field is more flexible than the Gaussian random field, with two additional parameters, g and h. The TGH random field includes a large family of trans-Gaussian random fields. For example, when $g > 0$ and $h = 0$, $T(\boldsymbol{s})$ becomes a shifted log-Gaussian random field, and when $g = 0$ and $h > 0$, $T(\boldsymbol{s})$ becomes a random field with a Pareto-like marginal distribution. The standard TGH model can be generalized further by introducing the location and scale parameters. The location-scale TGH space random field (from now on, will be referred to as the TGH space process) with location parameter $\xi \in \mathbb{R}$ and scale parameter $\omega > 0$ is defined as

$$T(\boldsymbol{s}) = \xi + \omega\tau_{g,h}\{Z(\boldsymbol{s})\}. \tag{3}$$

3.2. Definition of Tukey g-and-h space-time random fields

Similar to the space TGH model, we define the space-time TGH model by applying Tukey's g-and-h transformation on a latent Gaussian space-time random process. Suppose we have a zero mean and unit variance space-time Gaussian process $Z(\boldsymbol{s}, t)$, $\boldsymbol{s} \in \mathbb{R}^d$ and $t \in \mathbb{R}$, $d \geq 1$. Here, we assume that the correlation of the process $Z(\boldsymbol{s}, t)$ is obtained from a space-time correlation function, i.e., $\mathbb{C}\mathrm{or}\{Z(\boldsymbol{s}_1, t_1), Z(\boldsymbol{s}_2, t_2)\} = \rho_Z(\boldsymbol{s}_1, t_1, \boldsymbol{s}_2, t_2)$, where $\rho_Z(\cdot)$ is a space-time correlation function. The standard space-time TGH process is defined as

$$T(\boldsymbol{s}, t) = \tau_{g,h}\{Z(\boldsymbol{s}, t)\}. \tag{4}$$

This transformation imposes skewness and tail-thickness on the marginal distribution of the space-time Gaussian random field. The standard space-time TGH process is further generalized to the location-scale TGH space-time process (from now on will be referred to as the TGH space-time process) and is defined as

$$T(\boldsymbol{s}, t) = \xi + \omega\tau_{g,h}\{Z(\boldsymbol{s}, t)\}, \quad \xi \in \mathbb{R}, \quad \omega > 0. \tag{5}$$

3.3. Log-likelihood of Tukey g-and-h space random field

In the space case, let $\boldsymbol{\theta}_1 = (\xi, \omega, g, h)^\top$ and $\boldsymbol{\theta}_2$ be the parameter vector corresponding to $\rho_Z(\boldsymbol{s}_1, \boldsymbol{s}_2)$, the correlation function of $Z(\boldsymbol{s})$ in (3), and let $\boldsymbol{\theta} = (\boldsymbol{\theta}_1^\top, \boldsymbol{\theta}_2^\top)^\top$. Consider a dataset $\mathcal{D}_s = \{y(\boldsymbol{s}_1), \ldots, y(\boldsymbol{s}_N)\}$ collected from the TGH random field, $T(\boldsymbol{s})$, at locations $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_N$. The log-likelihood function of $\boldsymbol{\theta}$ given the dataset $\mathcal{D}_s$ is

$$\begin{aligned} L(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 | \mathcal{D}_s) \propto & -\frac{1}{2}\{\boldsymbol{Z}_{\boldsymbol{\theta}_1}^\top(\boldsymbol{R}_{\boldsymbol{\theta}_2}^{-1} + h\boldsymbol{I}_N)\boldsymbol{Z}_{\boldsymbol{\theta}_1} + \log|\boldsymbol{R}_{\boldsymbol{\theta}_2}|\} \\ & - \sum_{i=1}^{N} \log[\exp(gz_{\boldsymbol{\theta}_1,\boldsymbol{s}_i}) \\ & + g^{-1}\{\exp(gz_{\boldsymbol{\theta}_1,\boldsymbol{s}_i}) - 1\}hz_{\boldsymbol{\theta}_1,\boldsymbol{s}_i}] - N\log\omega, \end{aligned} \tag{6}$$

where $z_{\boldsymbol{\theta}_1,\boldsymbol{s}_i} = \tau_{g,h}^{-1}\left\{\dfrac{y(\boldsymbol{s}_i)-\xi}{\omega}\right\}$, $\boldsymbol{Z}_{\boldsymbol{\theta}_1} = (z_{\boldsymbol{\theta}_1,\boldsymbol{s}_1},\ldots,z_{\boldsymbol{\theta}_1,\boldsymbol{s}_N})^\top$ and $(\boldsymbol{R}_{\theta_2})_{i,j} = \rho_Z(\boldsymbol{s}_i,\boldsymbol{s}_j)$, $i,\ j = 1,\ldots,N$. We estimate $\boldsymbol{\theta}$ by maximizing the log-likelihood in (6).

3.4. Log-likelihood of Tukey g-and-h space-time random field

In the space-time case, let $\boldsymbol{\theta}_1 = (\xi,\omega,g,h)^\top$ and $\boldsymbol{\theta}_2$ be the parameter vector corresponding to the correlation function of $Z(\boldsymbol{s},t)$ in (5), $\rho_Z(\boldsymbol{s}_1,t_1,\boldsymbol{s}_2,t_2)$, and let $\boldsymbol{\theta} = (\boldsymbol{\theta}_1^\top,\boldsymbol{\theta}_2^\top)^\top$. Consider a dataset $\mathcal{D}_{\text{st}} = \{y(\boldsymbol{s}_1,t_1),\ldots,y(\boldsymbol{s}_n,t_n)\}$ collected from the TGH space-time random process, $T(\boldsymbol{s},t)$, at space-time locations $(\boldsymbol{s}_1,t_1),\ldots,(\boldsymbol{s}_n,t_n)$, $n = NT$. The log-likelihood function of $\boldsymbol{\theta}$ given the dataset $\mathcal{D}_{\text{st}}$ is

$$\begin{aligned}L(\boldsymbol{\theta}_1,\boldsymbol{\theta}_2|\mathcal{D}_{\text{st}}) \propto & -\frac{1}{2}\{\boldsymbol{Z}_{\boldsymbol{\theta}_1}^\top(\boldsymbol{R}_{\boldsymbol{\theta}_2}^{-1}+h\boldsymbol{I}_n)\boldsymbol{Z}_{\boldsymbol{\theta}_1}+\log|\boldsymbol{R}_{\boldsymbol{\theta}_2}|\}\\ &-\sum_{i=1}^{n}\log[\exp(gz_{\boldsymbol{\theta}_1,\boldsymbol{s}_i,t_i})\\ &+g^{-1}\{\exp(gz_{\boldsymbol{\theta}_1,\boldsymbol{s}_i,t_i})-1\}hz_{\boldsymbol{\theta}_1,\boldsymbol{s}_i,t_i}]-n\log\omega,\end{aligned} \tag{7}$$

where $z_{\boldsymbol{\theta}_1,\boldsymbol{s}_i,t_i} = \tau_{g,h}^{-1}\left\{\dfrac{y(\boldsymbol{s}_i,t_i)-\xi}{\omega}\right\}$, $\boldsymbol{Z}_{\boldsymbol{\theta}_1} = (z_{\boldsymbol{\theta}_1,\boldsymbol{s}_1,t_1},\ldots,z_{\boldsymbol{\theta}_1,\boldsymbol{s}_n,t_n})^\top$ and $(\boldsymbol{R}_{\theta_2})_{i,j} = \rho_Z(\boldsymbol{s}_i,t_i,\boldsymbol{s}_j,t_j)$, $i,\ j = 1,\ldots,n$. Like the space case, we find the maximum likelihood estimates (MLE) of $\boldsymbol{\theta}$ for the given space-time dataset $\mathcal{D}_{\text{st}}$ by maximizing the log-likelihood in (7).

3.5. Kriging with Tukey g-and-h random fields

One problem of interest is making predictions at new locations. This problem of making predictions is also known as kriging. In kriging, the objective is to find an optimal point estimator of the process under consideration at a new location $\boldsymbol{s}_0$ by minimizing some loss function. For TGH space random fields, the best kriging predictor of $T(\boldsymbol{s}_0)$ under the mean squared error loss function is

$$\begin{aligned}\widehat{T}(\boldsymbol{s}_0) = \widehat{\xi} + &\frac{\widehat{\omega}}{\widehat{g}\sqrt{1-\widehat{h}\tilde{\sigma}^2}}\exp\left\{\frac{\widehat{h}\tilde{\mu}^2}{2(1-\widehat{h}\tilde{\sigma}^2)}\right\}\\ &\times\left[\exp\left\{\frac{\widehat{g}^2\tilde{\sigma}^2+2\widehat{g}\tilde{\mu}}{2(1-\widehat{h}\tilde{\sigma}^2)}\right\}-1\right],\end{aligned} \tag{8}$$

where $\tilde{\mu} = \boldsymbol{r}_{\widehat{\boldsymbol{\theta}}_2}^\top\boldsymbol{R}_{\widehat{\boldsymbol{\theta}}_2}^{-1}\boldsymbol{Z}_{\widehat{\boldsymbol{\theta}}_1}$, $\tilde{\sigma}^2 = 1-\boldsymbol{r}_{\widehat{\boldsymbol{\theta}}_2}^\top\boldsymbol{R}_{\widehat{\boldsymbol{\theta}}_2}^{-1}\boldsymbol{r}_{\widehat{\boldsymbol{\theta}}_2}$ and $\boldsymbol{r}_{\widehat{\boldsymbol{\theta}}_2} = \{\rho_Z(\boldsymbol{s}_0,\boldsymbol{s}_1),\ldots,\rho_Z(\boldsymbol{s}_0,\boldsymbol{s}_N)\}^\top$ and $\widehat{\boldsymbol{\theta}} = (\widehat{\boldsymbol{\theta}}_1^\top,\widehat{\boldsymbol{\theta}}_2^\top)^\top$ is the MLE of $\boldsymbol{\theta}$.

For the space-time TGH model, the kriging equation for a new space-time location $(\boldsymbol{s}_0,t_0)$, obtained by minimizing the mean squared error loss function is

$$\begin{aligned}\widehat{T}(\boldsymbol{s}_0,t_0) = \widehat{\xi} + &\frac{\widehat{\omega}}{\widehat{g}\sqrt{1-\widehat{h}\tilde{\sigma}^2}}\exp\left\{\frac{\widehat{h}\tilde{\mu}^2}{2(1-\widehat{h}\tilde{\sigma}^2)}\right\}\\ &\times\left[\exp\left\{\frac{\widehat{g}^2\tilde{\sigma}^2+2\widehat{g}\tilde{\mu}}{2(1-\widehat{h}\tilde{\sigma}^2)}\right\}-1\right],\end{aligned} \tag{9}$$

where $\boldsymbol{r}_{\widehat{\boldsymbol{\theta}}_2} = \{\rho_Z(\boldsymbol{s}_0,t_0,\boldsymbol{s}_1,t_1),\ldots,\rho_Z(\boldsymbol{s}_0,t_0,\boldsymbol{s}_n,t_n)\}^\top$, $\tilde{\mu} = \boldsymbol{r}_{\widehat{\boldsymbol{\theta}}_2}^\top\boldsymbol{R}_{\widehat{\boldsymbol{\theta}}_2}^{-1}\boldsymbol{Z}_{\widehat{\boldsymbol{\theta}}_1}$, $\tilde{\sigma}^2 = 1-\boldsymbol{r}_{\widehat{\boldsymbol{\theta}}_2}^\top\boldsymbol{R}_{\widehat{\boldsymbol{\theta}}_2}^{-1}\boldsymbol{r}_{\widehat{\boldsymbol{\theta}}_2}$, and $\widehat{\boldsymbol{\theta}} = (\widehat{\boldsymbol{\theta}}_1^\top,\widehat{\boldsymbol{\theta}}_2^\top)^\top$ is the MLE of $\boldsymbol{\theta}$.

3.6. Space Matérn correlation function

For defining the space TGH model in Equation (3) we need a space correlation function for constructing the correlation matrix of the latent Gaussian random field $Z(\boldsymbol{s})$. The correlation matrix has to be a symmetric positive definite matrix. We use the Matérn correlation function in this work because of its high flexibility, which is defined as

$$\rho_Z(\boldsymbol{s}_1,\boldsymbol{s}_2) = \rho_Z(h) = \frac{1}{\Gamma(\nu)2^{\nu-1}}\left(4\sqrt{2\nu}\frac{h}{\phi}\right)^{\nu}\mathcal{K}_\nu\left(4\sqrt{2\nu}\frac{h}{\phi}\right), \tag{10}$$

where $h = \|\boldsymbol{s}_1-\boldsymbol{s}_2\|$ is the distance between locations $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$, $\nu > 0$ is the smoothness parameter, $\phi > 0$ is the range parameter, $\Gamma(\cdot)$ is the gamma function, and $\mathcal{K}_\nu(\cdot)$ is the modified Bessel function of the second kind of order ν. As the name suggests, the smoothness parameter ν dictates the smoothness of the random field, and the range parameter ϕ controls how quickly the correlation of the random field decreases with distance. Many popular correlation functions come under the Matérn correlation family. For example, when $\nu = 0.5$, the Matérn correlation function becomes the exponential correlation function $\rho(h) = \exp(-4h/\phi)$, when $\nu = 1$, it becomes the Whittle correlation function $\rho(h) = (4h/\phi)\,\mathcal{K}_1\,(4h/\phi)$.

3.7. Space-time Matérn correlation function

A proper space-time correlation function is needed in the space-time case, which yields a positive definite correlation matrix for $Z(\boldsymbol{s},t)$. For this work, we use the space-time Matérn correlation function introduced by Gneiting [27], and its functional form is

$$\begin{aligned}&\rho_Z(\boldsymbol{s}_1,t_1,\boldsymbol{s}_2,t_2) = \rho_Z(h,u)\\ &\quad = \frac{1}{2^{\nu-1}\Gamma(\nu)(u^{2\alpha}/a_t+1)}\left\{\frac{h/a_s}{(u^{2\alpha}/a_t+1)^{\beta/2}}\right\}^{\nu}\\ &\quad\times\mathcal{K}_\nu\left\{\frac{h/a_s}{(u^{2\alpha}/a_t+1)^{\beta/2}}\right\},\end{aligned} \tag{11}$$

where $h = \|\boldsymbol{s}_1-\boldsymbol{s}_2\|$ is the distance between locations $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$, $u = |t_1-t_2|$ is the absolute difference between two-time points, $\nu > 0$ and $\alpha \in (0,1]$ are the smoothness parameters in space and time, respectively, $a_s, a_t > 0$ are the range parameters in space and time, respectively, $\beta \in [0,1]$ is the space-time interaction parameter, $\Gamma(\cdot)$ is the gamma function, and $\mathcal{K}_\nu(\cdot)$ is the modified Bessel function of the second kind of order ν. The smoothness parameters ν and α dictate the smoothness of the random process in space and time, respectively. The range parameters a_s and a_t control how quickly the correlation of the random process decreases with distance and the time difference, respectively. The space-time interaction parameter β dictates to what extent the space-time components interact between themselves. When the parameter $\beta = 0$, the correlation function can be written as factors of pure space and purely temporal components. These kinds of correlation functions are known as separable correlation functions. There is no interaction between space and time components for a separable correlation function.

3.8. Assessment of fitted TGH model using PIT

The Probability Integral Transformation (PIT) can be used to check the model's adequacy. Suppose we fit the space TGH model to a dataset $\mathcal{D}_s = \{y(\boldsymbol{s}_1),\ldots,y(\boldsymbol{s}_N)\}$, assuming it is collected from a space TGH random field $T(\boldsymbol{s})$, defined in (3), at locations $\boldsymbol{s}_1,\ldots,\boldsymbol{s}_N$. Then, from Theorem 4 in [49], we can estimate the conditional distribution function of $T(\boldsymbol{s}_0)$ given $\mathcal{D}_s$ as

$$\widehat{F}_{\boldsymbol{s}_0}(u|\mathcal{D}_s) = \Phi\left(\frac{z-\tilde{\mu}}{\tilde{\sigma}}\right), \tag{12}$$

where $z = \tau^{-1}_{\widehat{g},\widehat{h}}\{(u-\widehat{\xi})/\widehat{\omega}\}$, $\Phi(\cdot)$ is the distribution function of standard Gaussian distribution, and $\tilde{\mu}$ and $\tilde{\sigma}^2$ are defined in (8). If the dataset is emulating a space TGH model, then the probability integral transform of $T(\boldsymbol{s}_0)$, i.e., $F_{\boldsymbol{s}_0}\{T(\boldsymbol{s}_0)\}$ will be an observation from a uniform distribution over $(0,1)$. The validation procedure can be performed by dividing the data into training and testing. Suppose the data are coming from a TGH random field. In that case, the probability integral transformations of the testing data, transformed by the estimated distribution function with the training data, should be approximately uniformly distributed over $(0,1)$.

In the space-time regime, suppose we fit the space-time TGH model to the dataset $\mathcal{D}_{st} = \{y(\boldsymbol{s}_1, t_1), \ldots, y(\boldsymbol{s}_n, t_n)\}$. Then the estimated conditional distribution function of $T(\boldsymbol{s}_0, t_0)$ given $\mathcal{D}_{\rm st}$ is

$$\widehat{F}_{\boldsymbol{s}_0,t_0}(u|\mathcal{D}_{\rm st}) = \Phi\left(\frac{z-\tilde{\mu}}{\tilde{\sigma}}\right), \tag{13}$$

where $z = \tau^{-1}_{\widehat{g},\widehat{h}}\{(u-\widehat{\xi})/\widehat{\omega}\}$ and $\tilde{\mu}$ and $\tilde{\sigma}^2$ are defined in (9). Similar to the space case, we can check the PIT histogram of the testing data and see if it resembles the uniform distribution over $(0,1)$ for checking the model adequacy.

3.9. Task-based parallelism

Task-based parallelism is a standard parallel programming model that defines a set of predefined tasks with inputs/outputs and their respective data directions. The algorithm may then translate into a Directed Acyclic Graph (DAG) representing the dependencies between the tasks during the execution. According to the DAG, tasks may run on existing processing units when the data dependencies are satisfied. Matrix operations are the core of the log-likelihood function in Gaussian and non-Gaussian random fields, where computing the inversion of the correlation matrix is the most time-consuming operation with cubical complexity. In the literature, parallel linear algebra solvers have two types of approaches, block-based and tile-based algorithms. The block-based algorithms decompose the target matrix into successive panels and update the trailing submatrix. LAPACK [9] and ScaLAPACK [19] are state-of-the-art software that provides block-based linear solvers for shared and distributed-memory systems. The tile-based algorithms rely on splitting the matrix into a set of tiles and defining the underlying algorithms to weaken the artifactual synchronization points during the matrix operations. PLASMA [5], DPLASMA [15] and Chameleon [17] are examples of state-of-the-art tile-based linear algebra solvers.

3.10. Dynamic runtime systems

Representing the numerical operation as a set of tasks allows better distribution of the individual tasks into available processing units. In this case, a task-based dynamic runtime system such as OmpSs [23], OpenMP [20], StarPU [10], and PaRSEC [16] can be employed to run tasks on different hardware resources while ensuring the integrity of data dependencies. In this work, we rely on the StarPU runtime system to implement the non-Gaussian log-likelihood function.

3.11. Tile low-rank approximations

Tile Low-Rank (TLR) approximation [6,8] is a flat algebraic compression approach that splits the dense matrix into tiles of similar sizes. Fig. 1 shows an example of compressing an off-diagonal tile $\boldsymbol{T}_{12}$ to two matrices $\boldsymbol{U}_{12}$ and $\boldsymbol{V}_{12}$, where the Singular Value Decomposition (SVD) is used to compress the dense matrix. The most significant k singular values are captured with their associated singular vectors, which correspond to the rank of the tile. Once each tile is compressed, the tile-based algorithm is expressed in terms of tasks interconnected by their data dependencies. The original algorithm can then be translated into a Directed Acyclic Graph (DAG), where nodes are fine-grained computational tasks, and edges express their data dependencies. As integrated into the HiCMA library, the runtime orchestrates the asynchronous scheduling of tasks with their data dependencies onto processing units [3]. The task-based programming model creates opportunities for look-ahead, which enables to maintain high hardware occupancy.

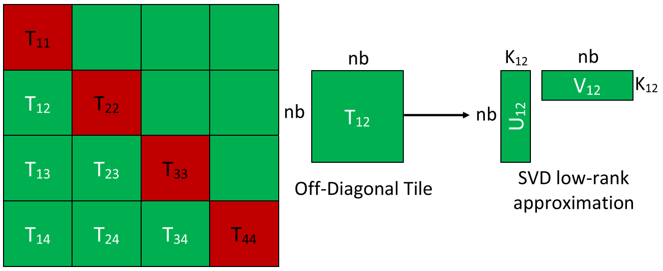


Fig. 1. An example of TLR approximation tile: diagonal tiles (in red) are dense. Off-diagonal tiles (in green) are represented as low-rank approximation.

4. Parallel TGH modeling and prediction

This section explains our proposed parallel implementation of the TGH likelihoods and predictions in exact and TLR approximation. We describe in detail the modeling and prediction algorithms. We also show the distribution of the ranks in TLR matrices associated with space and space-time TGH random fields.

4.1. Non-Gaussian space and space-time log-likelihood estimation

Recalling the TGH random field log-likelihood functions for space in (6) and for space-time in (7), the log-likelihood estimation process involves generating a covariance matrix $\boldsymbol{R}_{\boldsymbol{\theta}_2}$ where $\boldsymbol{\theta}_2$ is the parameters of a given correlation function. Given a set of N geospace locations (and M time slots), a selected covariance function can be used to build an $N \times N$ (or $NM \times NM$) covariance matrix. In this work, we use the parametrizable Matérn correlation function with two parameters: ϕ, the space range parameter, and ν, the random field smoothness parameter. The other input parameter vector $\boldsymbol{\theta}_1 = (\xi, \omega, g, h)^\top$ is used to transform the non-Gaussian measurement vector $\boldsymbol{Z}$ to a Gaussian vector $\boldsymbol{Z}_{\boldsymbol{\theta}_1}$.

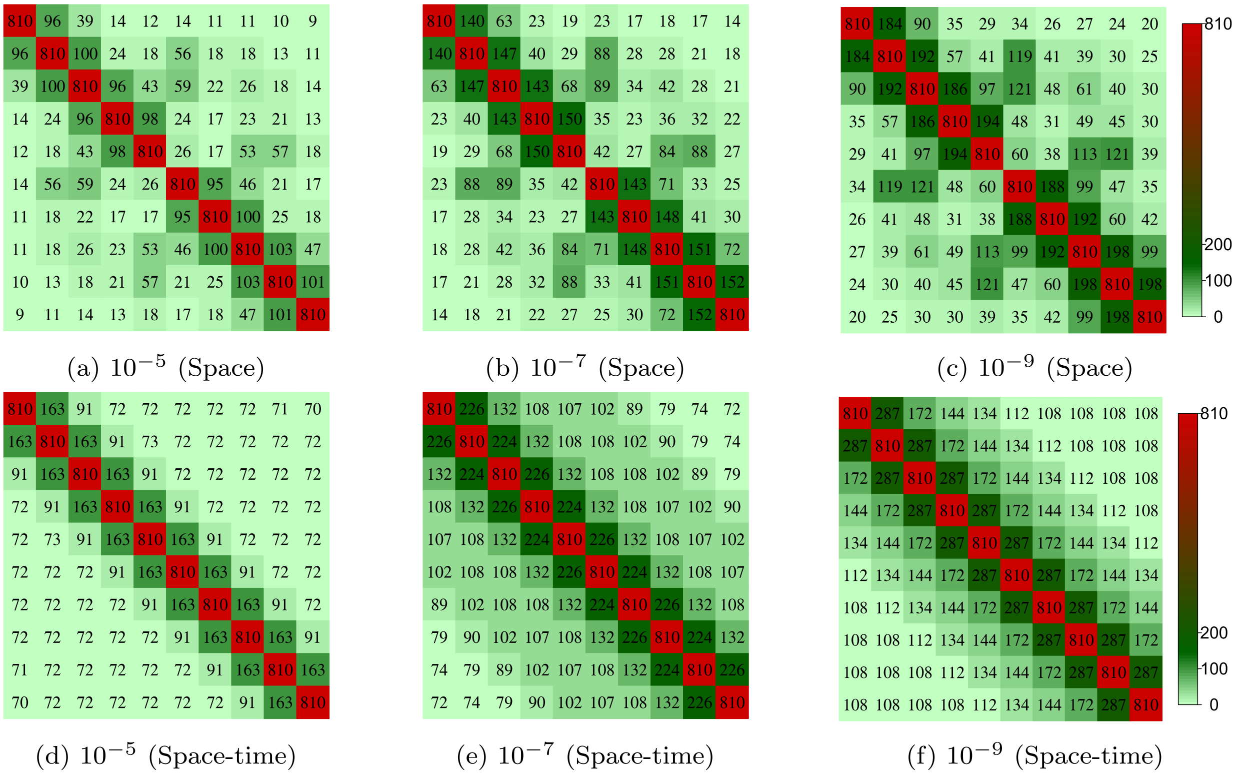
Algorithm 1 presents the TGH log-likelihood estimation algorithm in space and space-time cases. The $\boldsymbol{Z}_{\boldsymbol{\theta}_1}$ vector transformation step is performed first, as shown in line 4 with a given $\boldsymbol{\theta}_1$ parameter vector.

Since the $\tau^{-1}_{g,h}(\cdot)$ has no closed form, we use the Newton-Raphson method to approximate the function for $\boldsymbol{Z}_{\boldsymbol{\theta}_1}$ as follows:

$$\tau^{-1}_{g,h}(z_{\boldsymbol{\theta}_1,\boldsymbol{s}_i}) = \frac{y(\boldsymbol{s}_i)-\xi}{\omega}, \tag{14}$$

$$\tau^{-1}_{g,h}(z_{\boldsymbol{\theta}_1,\boldsymbol{s}_i,t_i}) = \frac{y(\boldsymbol{s}_i,t_i)-\xi}{\omega}. \tag{15}$$

In line 4, the covariance matrix is generated based on Equation (10) or Equation (11). The most time-consuming step in Algorithm 1 is the Cholesky factorization of the covariance matrix $\boldsymbol{R}_{\boldsymbol{\theta}_2}$ in line (3). In line 6 we compute the determinant of the covariance matrix based on Cholesky factorization in line 5. In line 7, the non-Gaussian component S from the log-likelihood function in eq (6) and (7) is computed. In line 8, a triangular solve operation is performed between the factorized matrix and $\boldsymbol{Z}_{\boldsymbol{\theta}_1}$. In line 11, the final value of the likelihood function is computed based on the result of the dot product operation in line 9.



810	96	39	14	12	14	11	11	10	9
96	810	100	24	18	56	18	18	13	11
39	100	810	96	43	59	22	26	18	14
14	24	96	810	98	24	17	23	21	13
12	18	43	98	810	26	17	53	57	18
14	56	59	24	26	810	95	46	21	17
11	18	22	17	17	95	810	100	25	18
11	18	26	23	53	46	100	810	103	47
10	13	18	21	57	21	25	103	810	101
9	11	14	13	18	17	18	47	101	810

(a) 10^{-5} (Space)

810	140	63	23	19	23	17	18	17	14
140	810	147	40	29	88	28	28	21	18
63	147	810	143	68	89	34	42	28	21
23	40	143	810	150	35	23	36	32	22
19	29	68	150	810	42	27	84	88	27
23	88	89	35	42	810	143	71	33	25
17	28	34	23	27	143	810	148	41	30
18	28	42	36	84	71	148	810	151	72
17	21	28	32	88	33	41	151	810	152
14	18	21	22	27	25	30	72	152	810

(b) 10^{-7} (Space)

810	184	90	35	29	34	26	27	24	20
184	810	192	57	41	119	41	39	30	25
90	192	810	186	97	121	48	61	40	30
35	57	186	810	194	48	31	49	45	30
29	41	97	194	810	60	38	113	121	39
34	119	121	48	60	810	188	99	47	35
26	41	48	31	38	188	810	192	60	42
27	39	61	49	113	99	192	810	198	99
24	30	40	45	121	47	60	198	810	198
20	25	30	30	39	35	42	99	198	810

(c) 10^{-9} (Space)

810	163	91	72	72	72	72	72	71	70
163	810	163	91	73	72	72	72	72	72
91	163	810	163	91	72	72	72	72	72
72	91	163	810	163	91	72	72	72	72
72	73	91	163	810	163	91	72	72	72
72	72	72	91	163	810	163	91	72	72
72	72	72	72	91	163	810	163	91	72
72	72	72	72	72	91	163	810	163	91
71	72	72	72	72	72	91	163	810	163
70	72	72	72	72	72	72	91	163	810

(d) 10^{-5} (Space-time)

810	226	132	108	107	102	89	79	74	72
226	810	224	132	108	108	102	90	79	74
132	224	810	226	132	108	108	102	89	79
108	132	226	810	224	132	108	107	102	90
107	108	132	224	810	226	132	108	107	102
102	108	108	132	226	810	224	132	108	107
89	102	108	108	132	224	810	226	132	108
79	90	102	107	108	132	226	810	224	132
74	79	89	102	107	108	132	224	810	226
72	74	79	90	102	107	108	132	226	810

(e) 10^{-7} (Space-time)

810	287	172	144	134	112	108	108	108	108
287	810	287	172	144	134	112	108	108	108
172	287	810	287	172	144	134	112	108	108
144	172	287	810	287	172	144	134	112	108
134	144	172	287	810	287	172	144	134	112
112	134	144	172	287	810	287	172	144	134
108	112	134	144	172	287	810	287	172	144
108	108	112	134	144	172	287	810	287	172
108	108	108	112	134	144	172	287	810	287
108	108	108	108	112	134	144	172	287	810

(f) 10^{-9} (Space-time)

Fig. 2. Subfigures (a), (b), and (c) represent rank distributions of a 8,100 × 8,100 space covariance TLR matrix using $nb = 810$ with Matérn parameters $\boldsymbol{\theta}_2 = (0.96, 0.5)^\top$ under different accuracy levels. Subfigures (d), (e), and (f) represent rank distributions of a 8,100 × 8,100 space-time covariance TLR matrix, i.e., 36 locations and 225 timeslots, using $nb = 810$ with Matérn parameters $\boldsymbol{\theta}_2 = (0.1055, 0.5, 0.0485, 0.1, 0.5)^\top$ under different accuracy levels.

Algorithm 1 TGH log-likelihood algorithm for space/space-time random fields.

```
1: Input:
   Space Case: a set of locations $\boldsymbol{S} = \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_N\}$, associated measurements $y(\boldsymbol{s}_1), \ldots, y(\boldsymbol{s}_N)$, and current parameter vector $\boldsymbol{\theta}_1 = (\xi, \omega, g, h)^\top$
   Space-Time Case: a set of space-time locations $(\boldsymbol{S}, T) = \{(\boldsymbol{s}_1, t_1), \ldots, (\boldsymbol{s}_n, t_n)\}$, associated measurements $y(\boldsymbol{s}_1, t_1), \ldots, y(\boldsymbol{s}_n, t_n)$, and current parameter vector $\boldsymbol{\theta}_1 = (\xi, \omega, g, h)^\top$
2: Output: the log-likelihood estimation for the current parameter vector $\boldsymbol{\theta}_1$
3: Space Case: Inverse TGH transformation for $z_{\theta_1, \boldsymbol{s}_i} \leftarrow \tau_{g,h}^{-1}\left\{\frac{y(\boldsymbol{s}_i) - \xi}{\omega}\right\}$, and $\boldsymbol{Z}_{\theta_1} = (z_{\theta_1,\boldsymbol{s}_1}, \ldots, z_{\theta_1,\boldsymbol{s}_N})^\top$
   Space-time Case: Inverse TGH transformation for $z_{\theta_1,\boldsymbol{s}_i,t_i} \leftarrow \tau_{g,h}^{-1}\left\{\frac{y(\boldsymbol{s}_i, t_i) - \xi}{\omega}\right\}$, and $\boldsymbol{Z}_{\theta_1} = (z_{\theta_1,\boldsymbol{s}_1,t_1}, \ldots, z_{\theta_1,\boldsymbol{s}_n,t_n})^\top$.
4: Space Case: Generate covariance matrix $\boldsymbol{R}_{\theta_2}$
   Space-time Case: Generate covariance matrix $\boldsymbol{R}_{\theta_2}$ → eq (11)
5: POTRF($\boldsymbol{R}_{\theta_2}$) → Cholesky factorization
6: determinant = Det($\boldsymbol{R}_{\theta_2}$)
7: Space Case: $S = \sum_{i=1}^{N}(\log[\exp(gz_{\theta_1,\boldsymbol{s}_i}) + g^{-1}\{\exp(gz_{\theta_1,\boldsymbol{s}_i}) - 1\}hz_{\theta_1,\boldsymbol{s}_i}] + hz_{\theta_1,\boldsymbol{s}_i}^2) + N\log\omega$
   Space-time Case: $S = \sum_{i=1}^{n}(\log[\exp(gz_{\theta_1,\boldsymbol{s}_i,t_i}) + g^{-1}\{\exp(gz_{\theta_1,\boldsymbol{s}_i,t_i}) - 1\}hz_{\theta_1,\boldsymbol{s}_i,t_i}] + hz_{\theta_1,\boldsymbol{s}_i,t_i}^2) + n\log\omega$
8: TRSM ($\boldsymbol{R}_{\theta_2}$, $\boldsymbol{Z}_{\theta_1}$) → Triangular solve
9: dotproduct = ($\boldsymbol{Z}_{\theta_1} \times \boldsymbol{Z}_{\theta_1}$)
10: llh = $-\frac{1}{2}\{dotproduct + \log(determinant)\} - S$.
```

We rely on the state-of-the-art dense task-based libraries, i.e., Chameleon, to perform the linear algebra operations that appear in Algorithm 1, including the Cholesky factorization of the covariance matrix and the triangular solve in line (6). Furthermore, we exploit the data sparsity structure of the covariance matrix and apply low-rank approximation methods to reduce the overall complexity of the underlying linear algebra operations. In particular, we use TLR matrix approximations to speed up the computation of the Cholesky factorization heavyweight operation while preserving the accuracy requirement of the application.

4.1.1. TLR approximation of $\boldsymbol{R}_{\theta_2}$

The TLR approximation in the HiCMA library compresses the individual tiles using the Singular Value Decomposition (SVD) algorithm [47], where the ranks of the tiles represent the most significant singular values and vectors in each off-diagonal tile [6]. Therefore, the effectiveness of the TLR mechanism depends on the ranks of the off-diagonal tiles after compression, which in turn depends on the application's accuracy requirements. Thus, we initially validate the potency of the TLR approximation with the space and space-time TGH modeling by estimating the ranks corresponding to different accuracy levels, namely, TLR-5 (10^{-5}), TLR-7 (10^{-7}), and TLR-9 (10^{-9}). The required accuracy level of our application and the corresponding performance assessment is shown in detail in the performance section. Herein, we validate the effectiveness of using TLR with our two models. SubFigs. 2 (a), (b), and (c) depict the rank distribution of an 8, 100 × 8, 100 space covariance matrix generated by the Matérn covariance function shown in (10). In comparison, SubFigs. 2 (d), (e), and (f) depict the rank distribution of an 8,100 × 8,100 space-time covariance matrix generated by the Matérn covariance function shown in (11). As shown, the ranks of the off-diagonal tiles grow as the tiles get closer to the diagonal with different TLR accuracy levels, with a monotonic increase of the ranks with tighter tolerance. However, even with TLR-9, the ranks are still smaller than the full dense tiles in the diagonal. The given two examples are drawn from two synthetic sets of non-Gaussian space and space-time Matérn parameters, i.e., $\boldsymbol{\theta}_2 = (0.96, 0.5)^\top$ and $\boldsymbol{\theta}_2 = (0.1055, 0.5, 0.0485, 0.1, 0.5)^\top$, respectively. We also examine other space and space-time correlation strengths, which all show almost the same ranks with different accuracy levels. We only observe a change in the ranks with varying values of smoothness. Another observation from both figures is that the space-time covariance function produces higher ranks compared to the space covariance function.

4.2. Non-Gaussian space and space-time prediction

The optimization process of the likelihood function aims at tuning the TGH parameter vectors, i.e., $\widehat{\boldsymbol{\theta}}_1$ and $\widehat{\boldsymbol{\theta}}_2$. These parameters are used to predict missing measurements at a set of new locations using Equation (8). Algorithm 2 shows in detail the numerical steps to predict missing values using the two tuned parameter vectors. Line 4 accounts for most of the algorithmic complexity as it involves the Cholesky-based solver of the covariance matrix. This algorithm predicts the missing data at a certain location (i.e., space case) or a certain location and a certain time slot (i.e., space-time case).

Algorithm 2 TGH prediction algorithm for space and space-time random fields.

```
1: **Input:**
   *Space Case*: a set of locations $\boldsymbol{S} = \{\boldsymbol{s}_1, \dots, \boldsymbol{s}_N\}$ and associated measurements $y(\boldsymbol{s}_1), \dots, y(\boldsymbol{s}_N)$, and estimated parameter vector $\widehat{\boldsymbol{\theta}}$
   *Space-Time Case*: a set of space-time locations $(\boldsymbol{S}, T) = \{(\boldsymbol{s}_1, t_1), \dots, (\boldsymbol{s}_n, t_n)\}$ and associated measurements $y(\boldsymbol{s}_1, t_1), \dots, y(\boldsymbol{s}_n, t_n)$, and estimated parameter vector $\widehat{\boldsymbol{\theta}}$
2: **Output:**
   *Space Case*: the predicted values $\widehat{T}(\boldsymbol{s}_{01}), \dots, \widehat{T}(\boldsymbol{s}_{0N_{new}})$ at new locations $\boldsymbol{s}_{01}, \dots, \boldsymbol{s}_{0N_{new}}$
   *Space-Time Case*: the predicted values $\widehat{T}(\boldsymbol{s}_{01}, t_{01}), \dots, \widehat{T}(\boldsymbol{s}_{0n_{new}}, t_{0n_{new}})$ at new space-time locations $(\boldsymbol{s}_{01}, t_{01}), \dots, (\boldsymbol{s}_{0n_{new}}, t_{0n_{new}})$
3: *Space Case*: Inverse TGH transformation for $z_{\widehat{\boldsymbol{\theta}}_1, \boldsymbol{s}_i} \leftarrow \tau_{g,h}^{-1}\left\{\frac{y(\boldsymbol{s}_i) - \widehat{\xi}}{\widehat{\omega}}\right\}$ and $\boldsymbol{Z}_{\widehat{\boldsymbol{\theta}}_1} = (z_{\widehat{\boldsymbol{\theta}}_1, \boldsymbol{s}_1}, \dots, z_{\widehat{\boldsymbol{\theta}}_1, \boldsymbol{s}_N})^\top$
   *Space-Time Case*: Inverse TGH transformation for $z_{\widehat{\boldsymbol{\theta}}_1, \boldsymbol{s}_i, t_i} \leftarrow \tau_{g,h}^{-1}\left\{\frac{y(\boldsymbol{s}_i, t_i) - \widehat{\xi}}{\widehat{\omega}}\right\}$ and $\boldsymbol{Z}_{\widehat{\boldsymbol{\theta}}_1} = (z_{\widehat{\boldsymbol{\theta}}_1, \boldsymbol{s}_1, t_1}, \dots, z_{\widehat{\boldsymbol{\theta}}_1, \boldsymbol{s}_n, t_n})^\top$
4: *Space Case*: Generate covariance matrix $\boldsymbol{R}_{\boldsymbol{\theta}_2}$ → eq (10)
   *Space-Time Case*: Generate covariance matrix $\boldsymbol{R}_{\boldsymbol{\theta}_2}$ → eq (11)
5: *Space Case*: Generate covariance vector $\boldsymbol{r}_{\widehat{\boldsymbol{\theta}}_2}$ → eq (8)
   *Space-Time Case*: Generate covariance vector $\boldsymbol{r}_{\widehat{\boldsymbol{\theta}}_2}$ → eq (9)
6: POSV $(\boldsymbol{R}_{\widehat{\boldsymbol{\theta}}_2}, \boldsymbol{Z}_{\widehat{\boldsymbol{\theta}}_1})$ → System of linear equations solver
7: CPY $(\boldsymbol{r}_{\widehat{\boldsymbol{\theta}}_2}, \boldsymbol{rcpy}_{\widehat{\boldsymbol{\theta}}_2})$
8: TRSM $(\boldsymbol{R}^\top_{\widehat{\boldsymbol{\theta}}_2}, \boldsymbol{r}_{\widehat{\boldsymbol{\theta}}_2})$ → Triangular solve
9: $\boldsymbol{tmp}_1$ = GEMV$(\boldsymbol{rcpy}_{\widehat{\boldsymbol{\theta}}_2}, \boldsymbol{Z}_{\widehat{\boldsymbol{\theta}}_1})$ → Matrix-vector multiplication
10: $\boldsymbol{tmp}_2$ = GEMM$(\boldsymbol{rcpy}_{\widehat{\boldsymbol{\theta}}_2}, \boldsymbol{r}_{\widehat{\boldsymbol{\theta}}_2})$ → Matrix-matrix multiplication
11: **for** $k = 1$ to $N_{new}$ (or $n_{new}$) **do**
12:     $\tilde{\mu}$ = $\boldsymbol{tmp}_1[k]$
13:     $\tilde{\sigma}^2$ = $1 - \boldsymbol{tmp}_2[k]$
14:     *Space Case*: $\widehat{T}(\boldsymbol{s}_{0k}) = \widehat{\xi} + \frac{\widehat{\omega}}{\widehat{g}\sqrt{1-\widehat{h}\tilde{\sigma}^2}} \exp\{\frac{\widehat{h}\tilde{\mu}^2}{2(1-\widehat{h}\tilde{\sigma}^2)}\} \times [\exp\{\frac{\widehat{g}^2\tilde{\sigma}^2+2\widehat{g}\tilde{\mu}}{2(1-\widehat{h}\tilde{\sigma}^2)}\} - 1]$
        *Space-Time Case*: $\widehat{T}(\boldsymbol{s}_{0k}, t_{0k}) = \widehat{\xi} + \frac{\widehat{\omega}}{\widehat{g}\sqrt{1-\widehat{h}\tilde{\sigma}^2}} \exp\{\frac{\widehat{h}\tilde{\mu}^2}{2(1-\widehat{h}\tilde{\sigma}^2)}\} \times [\exp\{\frac{\widehat{g}^2\tilde{\sigma}^2+2\widehat{g}\tilde{\mu}}{2(1-\widehat{h}\tilde{\sigma}^2)}\} - 1]$
15: **end for**
```

Using low-rank approximation in the form of TLR for both $\boldsymbol{R}_{\widehat{\boldsymbol{\theta}}_2}$ and $\boldsymbol{r}_{\widehat{\boldsymbol{\theta}}_2}$ in Algorithm 2 can help reduce the complexity of the TGH prediction algorithm similar to the likelihood estimation operation. The following section gives a detailed performance assessment of different proposed TGH implementations.

5. Simulation study and performance assessment

This section provides a detailed evaluation of the accuracy of the proposed TGH implementation when considering big geospace datasets. The evaluation aims to assess the space and space-time modeling and prediction accuracy of the TGH model compared to the Gaussian model. We also aim to show the accuracy of the TLR approximation compared to the exact solution on both synthetic and real datasets. Finally, we aim to assess the performance of the proposed TGH implementations using Chameleon/StarPU for both exact and TLR approximation. The performance assessment covers both shared-memory and distributed-memory systems.

5.1. Testbed and methodology

The assessment of the exact and approximate implementations of TGH modeling and prediction has been conducted on Intel and AMD chips to highlight our software portability: a 28-core dual-socket Intel Xeon IceLake Gold 6330 CPU running at 2.00 GHz, and a 64-core dual-socket AMD EPYC Milan 7713 CPU running at 2.00 GHz. For the distributed-memory experiments, we use KAUST's Shaheen-II, a Cray XC40 system with 6,174 dual-socket compute nodes based on 16-core Intel Haswell processors running at 2.3 GHz. Each node has 128 GB of DDR4 memory. The system has a total of 197,568 processor cores and 790 TB of aggregate memory.

Our implementation enables performance portability as long as an optimized BLAS/LAPACK library is available on the target system. We rely on Intel MKL v2020.0.166 as the optimized BLAS/LAPACK library to link against the necessary optimized numerical routines for our implementations on our testbed systems. We compile the proposed exact and TLR code using Chameleon/StarPU and HiCMA linear algebra libraries with GCC V11.1, HWLOC v2.5.0, StarPU v1.3.8, GSL v2.7, and NLopt v2.6.2 optimization libraries.

5.2. Modeling and prediction accuracy assessment

In this section, we validate the proposed computational framework for fitting the space TGH model by means of parameter estimation accuracy. We perform the validation using Monte Carlo simulations for the space and space-time models. The Monte Carlo simulations process starts with generating many synthetic datasets using a pre-selected set of initial values to the parameter vector of the selected model. Then, these datasets are modeled using the proposed implementation to estimate a set of parameters that should be close to the initial ones.

5.2.1. Monte Carlo space TGH modeling simulations

For this simulation experiment, we generate synthetic random observations on 20,164 random locations on the $[0, 1] \times [0, 1]$ square, from the space TGH model (one time slot) defined in Equation (3) with the following parameter settings:

(a) $\xi = 0$, $\omega = 2$, $g = 0.2$, $h = 0.2$, $\nu = 0.5$, and $\phi = 0.42$, with effective range 0.3;
(b) $\xi = 0$, $\omega = 2$, $g = 0.2$, $h = 0.2$, $\nu = 0.5$, and $\phi = 0.70$, with effective range 0.5;
(c) $\xi = 0$, $\omega = 2$, $g = 0.5$, $h = 0.3$, $\nu = 0.5$, and $\phi = 0.96$, with effective range 0.5;
(d) $\xi = 0$, $\omega = 2$, $g = 0.2$, $h = 0.2$, $\nu = 0.5$, and $\phi = 0.98$, with effective range 0.7.

The effective range is the distance at which the correlation between two points becomes less than 0.05. We evaluate ϕ for different setups by changing the effective range, given the other parameters. To find ϕ in different setups, we use the functional form of the correlation function of the TGH random field, as given by Xu and Genton [49]. For each of these four setups, we find the MLE of the parameters by maximizing the log-likelihood function in Equation (6). Moreover, by maximizing the TLR approximated log-likelihood for different accuracies we obtain the MLE of the parameters. We repeat the procedure 100 times for both exact and TLR log-likelihood with different accuracies. We summarize our findings in the boxplots of the estimated parameter values for different setups and for different computation techniques in Fig. 3. We used three accuracy levels for TLR (i.e., TLR-5, TLR-7, and TLR-9) compared to the exact estimation (i.e., Exact). From the boxplots, we can say that our framework is well able to estimate the TGH model parameters under different circumstances. More-

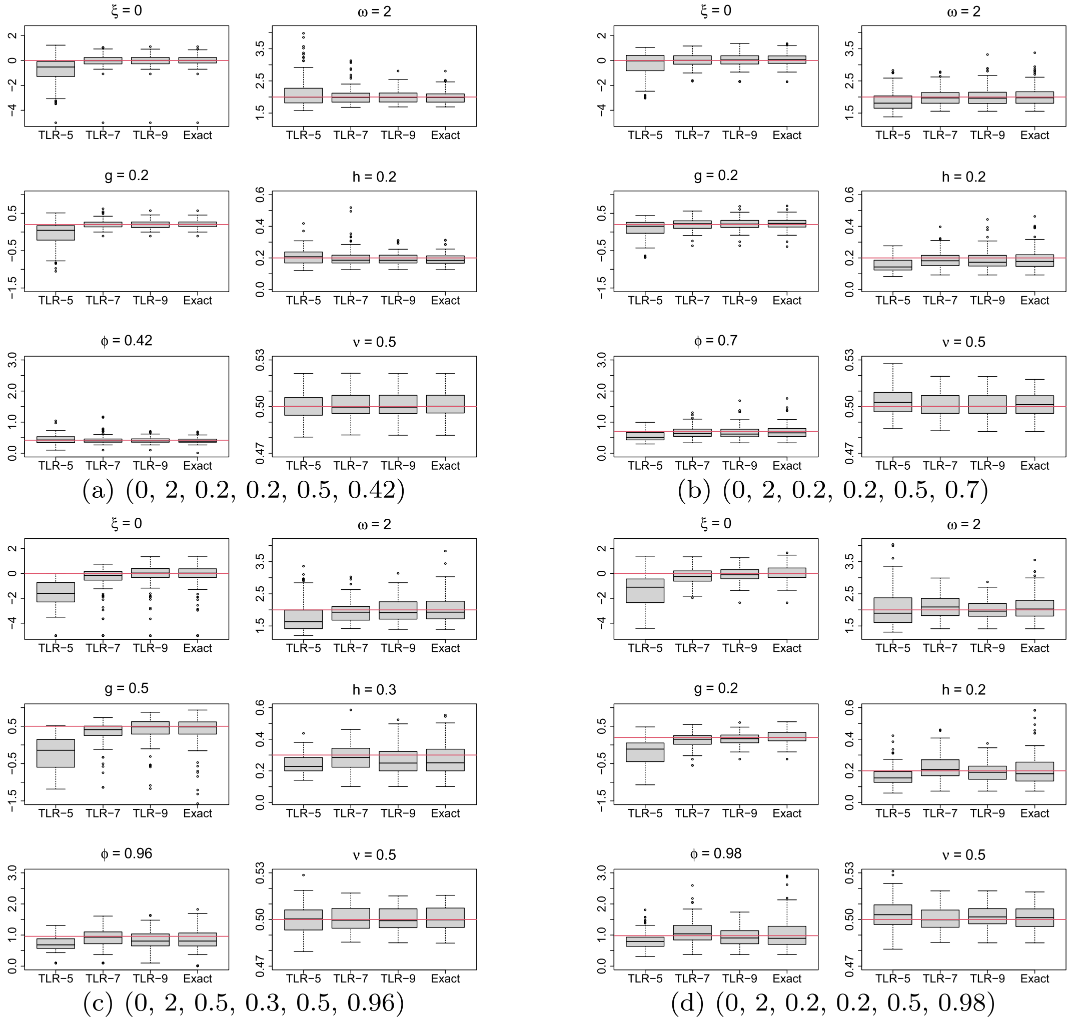


Fig. 3. Boxplots of parameter estimates of the TGH non-Gaussian space only case (ξ, ω, g, h, ϕ, ν) under the TLR and exact log-likelihood computation. The true parameters are indicated by the red lines.

over, the TLR approximation of the log-likelihood with an accuracy of 10^{-7} is giving us comparable results to the exact log-likelihood.

5.2.2. Evaluation of prediction with the space TGH random fields

Here, we provide a justification of the PIT assessment tool, discussed in Section 3.8, with a simulation study. In this simulation study, we generate 20,164 random observations on random locations on $[0, 1] \times [0, 1]$ square from the space TGH models with parameter setup (a) from the previous section. We fit the Gaussian space model and the space TGH model to a 90% subsample of the generated data (training data). Using the method discussed in Section 3.8 using the remaining data points (10%, testing data), we create the PIT histogram using the method mentioned in Section 3.8. Separate histograms are obtained based on the exact and TLR-7 parameter estimates. Fig. 4 suggests that the PIT histogram obtained by fitting the TGH model looks closer to a uniform distribution than the PIT histogram obtained from the Gaussian model. This simulation study shows that the model-adequacy judgment technique discussed in Section 3.8 is reasonable.

5.2.3. Monte Carlo space-time TGH modeling simulations

In this simulation experiment, we generate random observations on 100 space locations ×400 time locations from the space-time TGH model defined in Equation (5) and estimate the space-time TGH model parameters by maximizing the log-likelihood function, given in Equation (7). We use two parameter settings $(a_s, a_t) = (0.105, 0.048)$ and $(0.176, 0.054)$ for the space-time Matérn correlation function in Equation (11), along with $\nu = 0.5$, $\alpha = 0.1$, and $\beta = 0.5$ to generate two different space-time correlation matrices. Moreover, we use $\xi = 0$, $\omega = 2$, $g = 0.2$, and $h = 0.2$ to generate the space-time TGH observations for each space-time correlation matrix. We use the specific range parameters for creating weakly and moderately correlated observations in space and time, respectively. We find the MLE of the parameters from the synthetic dataset for each parameter set using the exact and TLR computation under different accuracy levels and repeat the process 100 times. The parameter estimates are summarized in the boxplots given in Fig. 5. The boxplots show that the estimation is accurate under different correlation settings suggesting the computation framework works properly under both the exact and TLR cases. Moreover, from the boxplots, we conclude that the TLR with an accuracy of 10^{-7} produces results comparable to the exact computation.

5.2.4. Evaluation of prediction with the space-time TGH random fields

Similar to the space case, we justify the PIT assessment tool, discussed in Section 3.8, with a simulation study in the space-time setting. In this simulation study, we generate random observations on 400 space locations ×100 time locations from the space-time TGH model with parameter setup $\xi = 0$, $\omega = 2$, $g = 0.2$, $h = 0.2$, $a_s = 0.08$, $\nu = 1$, $a_t = 1.08$, $\alpha = 0.5$, and $\beta = 0.5$. On 90% of the

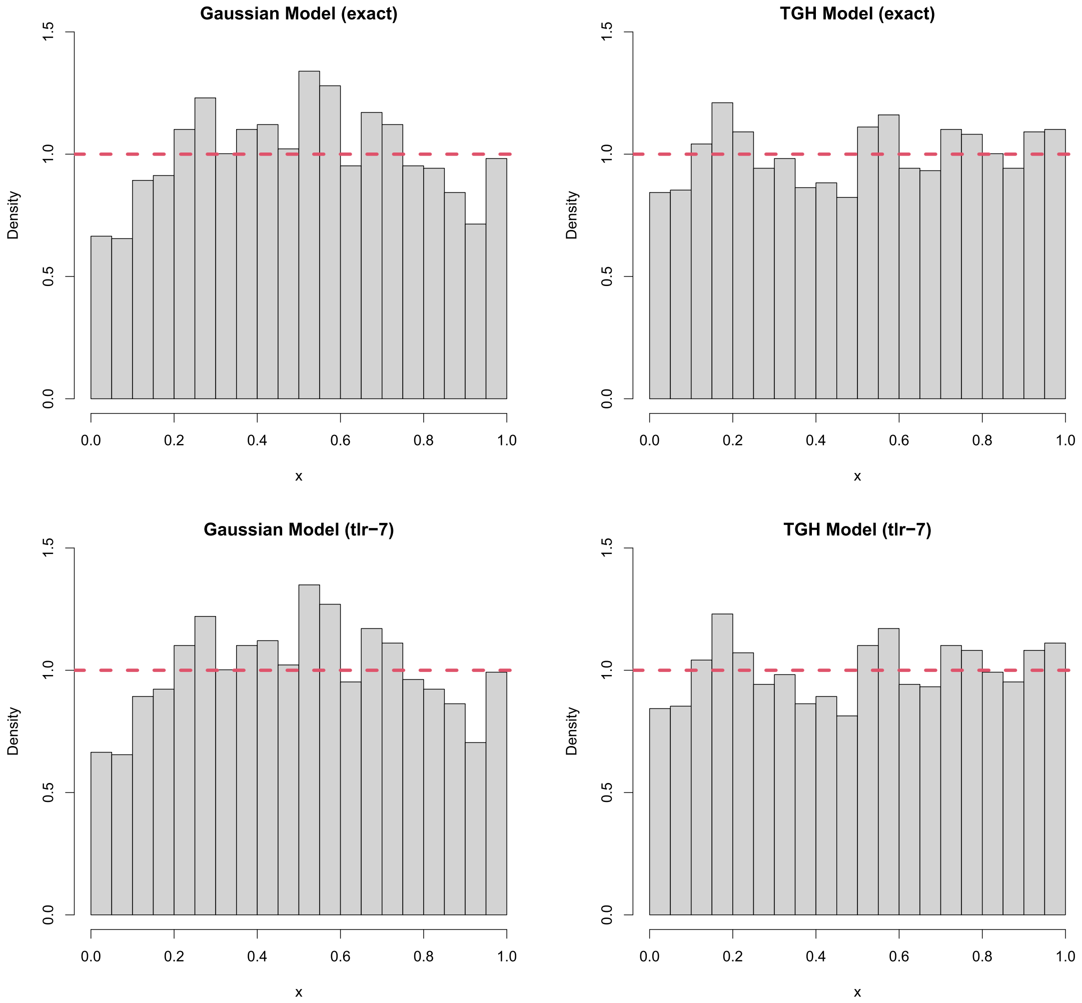
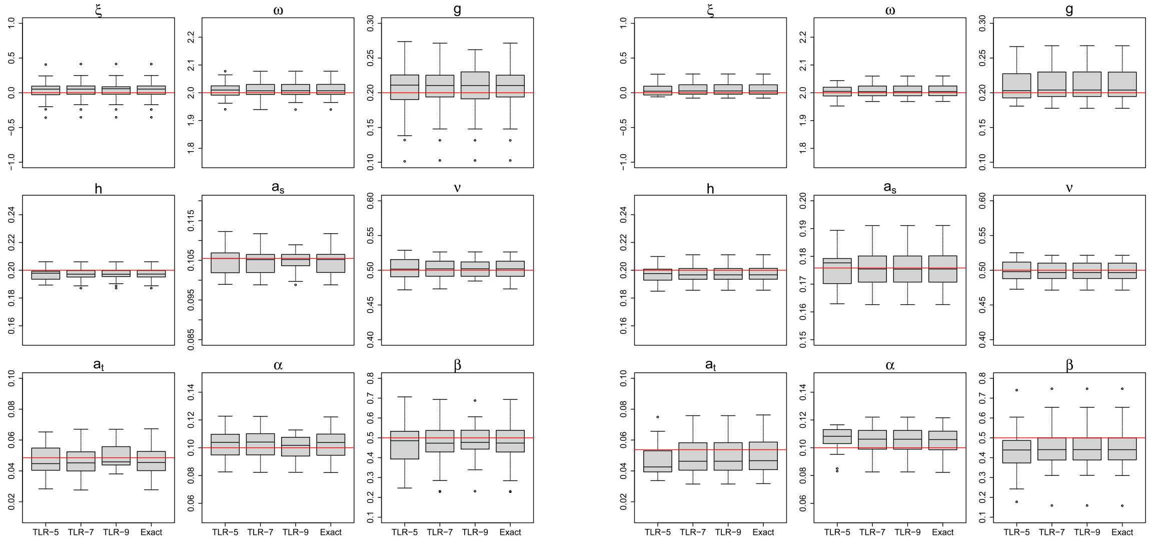


Fig. 4. PIT histograms for predictive distribution with the same TGH synthetic space dataset with exact and TLR-7 estimates.



(a) (0, 2, 0.2, 0.2, 0.105, 0.5, 0.048, 0.1, 0.5)

(b) (0, 2, 0.2, 0.2, 0.176, 0.5, 0.054, 0.1, 0.5)

Fig. 5. Boxplots of parameter estimates of the TGH non-Gaussian space-time case (ξ, ω, g, h, a_s, ν, a_t, α, β) under the TLR and exact log-likelihood computation. The true parameters are indicated by the red lines.

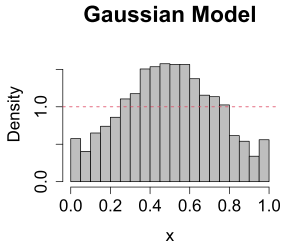


Fig. 6. PIT histograms for predictive distribution with the same TGH synthetic space-time dataset.

Table 1
Parameter estimates for Gaussian and TGH models on 300K space locations of the precipitation data in January 2021.

Model	ξ	ω	g	h	ϕ	ν
Gaussian	0.26	1.04	-	-	10.36	0.57
TGH	−0.37	0.85	−1.69	0.65	10.78	0.64

simulated data, we fit the space-time TGH model and the space-time Gaussian model. The PIT histogram from both models based on the remaining 10% testing dataset is given in Fig. 6. The PIT histogram from the space-time TGH model looks closer to a uniform distribution over $(0, 1)$ compared to the PIT histogram from the Gaussian model. This simulation study shows that the model-adequacy judgment technique discussed in Section 3.8 can be extended to the space-time scenario as well.

5.2.5. Real dataset application

We use the month-wise daily average precipitation of Germany in the year 2021. The data is collected by Kaspar et al. [31] and covers the whole of Germany with a space resolution of 1 km × 1 km. The daily average precipitation is given in mm. Because of this high space resolution, the total number of locations is $n = 358{,}303$. To make the data stationary, we remove the mean of the daily average precipitation of each month over the year 2000 to 2020 from the data. So, the data can be interpreted as the monthly excess daily average precipitation of 2021. The space image of the data is given in Fig. 7.

5.2.5.1. Monthly-based space modeling We fit the Gaussian model and TGH model to a subsample of size 300K locations from the data for the month of January. The estimated parameter values are presented in Table 1. The estimates of the parameters g and h from the space TGH model show that the data for the month of January are far from a Gaussian random field. This claim is justified by the proposed PIT assessment tool. We compute the PIT based on the estimated distribution function for the space Gaussian model and the space TGH model on the remaining testing data, i.e., 58,803 samples. The PIT histograms based on both models are given in Fig. 8. We can see from Fig. 8 that the PIT histogram obtained from the TGH model emulates the uniform distribution better over $(0, 1)$ compared to the Gaussian model. This observation leads us to conclude that the fitted TGH model is more appropriate to explain the variability of the data compared to the Gaussian model.

We also analyze the monthly data at 20,000 locations separately for each month. The locations have been selected randomly. For each month, we use the data on the chosen locations to fit the space Gaussian and TGH models. The parameter estimates from both models are summarized in Table 2 for all months. The estimates of g and h based on the space TGH model are very far from 0 for all months, suggesting the TGH model is more appropriate for this dataset, compared to the Gaussian model. This claim has been validated by the PIT assessment as well. We used 2000 random locations (separately drawn from the initial 20,000 locations) as our testing set, and based on that we draw the PIT for each month for two models, given in Fig. 9. The figure shows how the PIT histogram from the TGH models emulates the uniform distribution over $(0, 1)$ better than the PIT histogram for the Gaussian model for different months. This makes the TGH model more appropriate to this given real dataset than the Gaussian model.

5.2.5.2. Monthly space-time data experiment So far, we have analyzed only the space data for different months separately. In this section, we analyze the same monthly data from the previous section, collected over 20,000 locations, with the help of the Gaussian space-time model and the TGH space-time model. Moreover, with the month-wise estimates of $\widehat{\xi}_t$, $\widehat{\omega}_t$, $\widehat{g}_t$, and $\widehat{h}_t$ from the space TGH model, we transform the data to the Gaussian scale by $\tau^{-1}_{\widehat{g}_t,\widehat{h}_t}\{\frac{y(\mathbf{s}_j,t)-\widehat{\xi}_t}{\widehat{\omega}_t}\}$, where $j \in \{1, \ldots, 20{,}000\}$ and $t \in \{1, \ldots, 12\}$. This transformed data can be interpreted as observations from a zero mean unit variance Gaussian space-time random field. We fit the space-time TGH model to this transformed data by fixing the parameters $\xi = 0$, $\omega = 1$, $g = 0$, and $h = 0$. The difference between this model and the space TGH model described in Equation (5) is that the space TGH model assumes the parameter ξ, ω, g, and h are constant over time. However, the space-time TGH model on the Gaussian scale assumes the parameter ξ, ω, g, and h are different for different times, and their estimation is performed by fitting the space TGH model to the space data at each time point. The estimated parameters for all three space-time models based on this data are given in Table 3. Based on the testing dataset, the PIT histogram for all three models is presented in Fig. 10. The PIT histograms show that the Gaussian model is not the best option for modeling this space-time data. This is because the data has non-Gaussian behavior, which requires a non-Gaussian model to capture it. Thus, the TGH model outperforms the Gaussian model. However, since different months have different estimates for ξ, ω, g and h, as reported in Table 2, the Gaussian scale model outperforms both modeling techniques.

5.3. Performance assessment

In this section, we aim to assess the performance of our implementation on shared-memory and distributed memory systems.

5.3.1. TLR matrices performance assessment

We provide the TGH space and space-time modeling and prediction operations in low-rank structures using TLR approximations. The baseline TLR software is the HiCMA library, which runs on shared and distributed-memory architectures with the aid of the StarPU runtime system. We compare the TGH model using TLR implementation on two different shared-memory architectures from two vendors, i.e., Intel and AMD. We assess the performance of both implementations of the TGH likelihood function.

Performance on Shared-Memory Systems: Fig. 11 shows the performance of the TLR approximation compared to the exact implementation of the TGH likelihood function on our target two shared-memory systems and using different data sizes. With different accuracy, TLR outperforms the exact version of the likelihood estimation with a different number of locations. TLR shows better performance than the exact implementation, reaching up to 7.29X and 5.74X on Intel IceLake and AMD Milan, respectively. Furthermore, Fig. 12 shows the performance of the prediction operation using exact and TLR. The figure shows the TLR approximation of accuracy 10^{-9} outperforms the exact computation with up to 4.29X and 4.88X on the given machines.

Performance on Distributed-Memory Systems: We assess the exact and TLR-based approximation performance on the Shaheen-II Cray XC40 system. Fig. 13a shows the scalability of the exact non-Gaussian MLE on 64, 128, 256, and 512 nodes. Furthermore, the

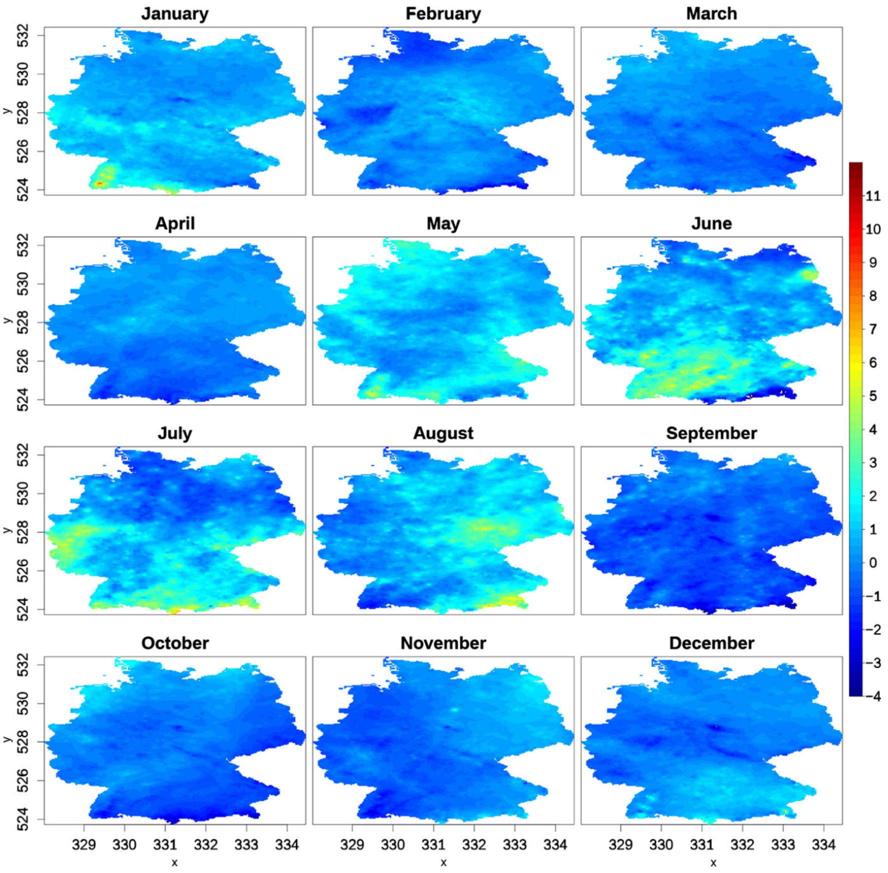
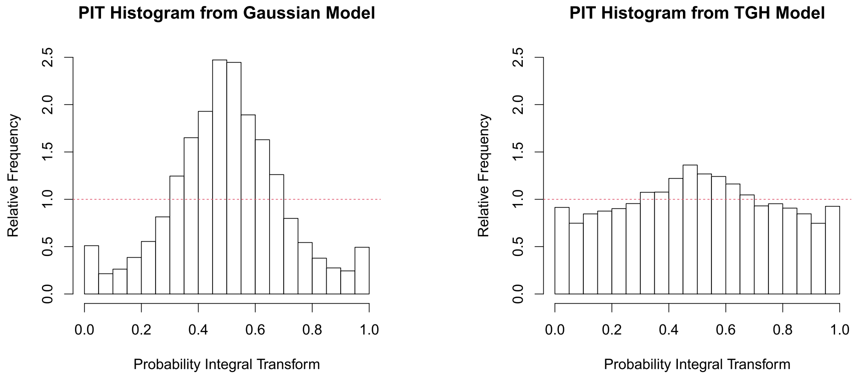


Fig. 7. Monthly excess daily average precipitation data over Germany in 2021.



(a) Gaussian Model.

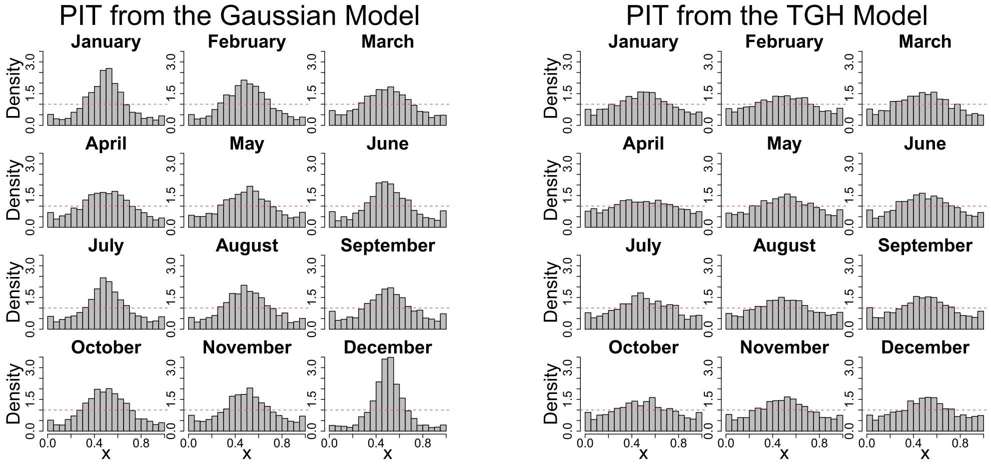
(b) TGH model.

Fig. 8. PIT histograms for both Gaussian and TGH models with the real dataset.

figure shows that the exact MLE implementation scales very well with the same matrix size and a different number of nodes. For instance, with 562, 500 space locations, the total execution time for a single MLE iteration is 629.44, 380.78, 255.97 seconds using 128, 256, and 512 nodes. We also assess the performance of the TLR-based implementation using 512 nodes on Shaheen-II compared to the exact computation. Fig. 13b shows the execution time of the exact and TLR approximation for MLE with different accuracy levels. As shown, the TLR approximation performs better than exact computation with a different number of locations up to 800 K locations using 512 nodes. The figure shows that the TLR approximation outperforms the exact calculation by up to 2.96X. We believe the performance of TLR MLE can be further improved by using a runtime system that provides rank-aware data distribution to mitigate the load imbalance [4]. In the two subfigures of Fig. 13b, the exact computation curve shows some performance difference with matrix size less than or equal to 455,625. Since the workload is small for the given number of nodes, the performance with these matrix sizes can vary due to different system loads and network congestion. Larger matrix sizes show more stable performance with the same number of nodes since all the nodes were involved in computation more than data movements.

6. Conclusion

This paper introduced parallel non-Gaussian modeling and prediction implementation based on the Tukey g-and-h (TGH) random fields in the context of climate/weather applications. We pro-



(a) Gaussian model.

(b) TGH Model.

Fig. 9. PIT histograms for both Gaussian and TGH model with the real dataset.

Table 2
Parameter estimates for the Gaussian and the TGH models on $20K$ space locations of the monthly precipitation data of the year 2021.

Month	Gaussian Model Estimates						TGH Model Estimates					
	ξ	ω	g	h	ϕ	ν	ξ	ω	g	h	ϕ	ν
Jan	0.69	0.69	-	-	28.42	0.49	-0.71	1.78	-2.51	0.48	46.19	0.80
Feb	0.61	0.57	-	-	49.98	-0.64	-0.24	0.39	-0.24	0.22	30.13	0.79
Mar	0.65	1.98	-	-	43.26	0.23	-6.23	20.90	-3.68	0.11	170.86	0.72
Apr	0.62	0.83	-	-	159.81	-0.51	0.49	1.02	1.78	0.52	104.68	0.71
May	0.87	0.59	-	-	16.11	0.86	-0.68	1.74	-1.40	0.39	44.22	0.89
Jun	0.90	1.06	-	-	16.52	0.54	-2.09	4.12	-2.12	0.33	30.23	0.99
Jul	0.77	1.33	-	-	33.49	1.11	0.64	0.95	0.44	0.08	20.80	0.92
Aug	0.86	0.92	-	-	21.09	0.69	0.60	0.69	0.30	0.10	17.42	0.99
Sep	0.66	0.74	-	-	45.14	-1.13	-0.95	0.47	-0.40	0.13	23.45	0.74
Oct	0.55	1.30	-	-	352.91	-1.09	-12.54	58.36	-4.97	0.10	467.64	0.68
Nov	0.62	0.68	-	-	75.38	-0.11	-0.25	0.55	-0.29	0.15	60.05	0.70
Dec	0.70	0.45	-	-	11.94	-0.15	-0.52	0.65	-1.15	0.33	43.03	0.76

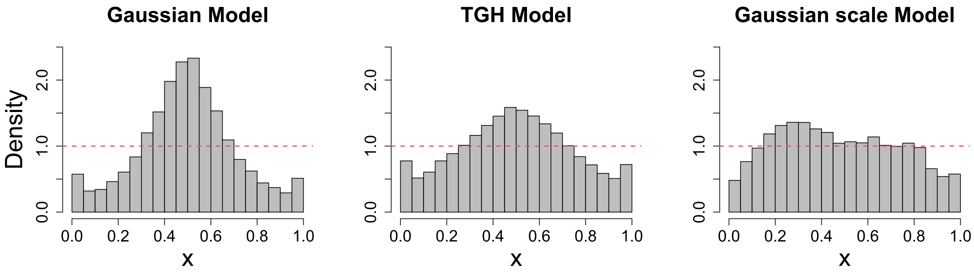
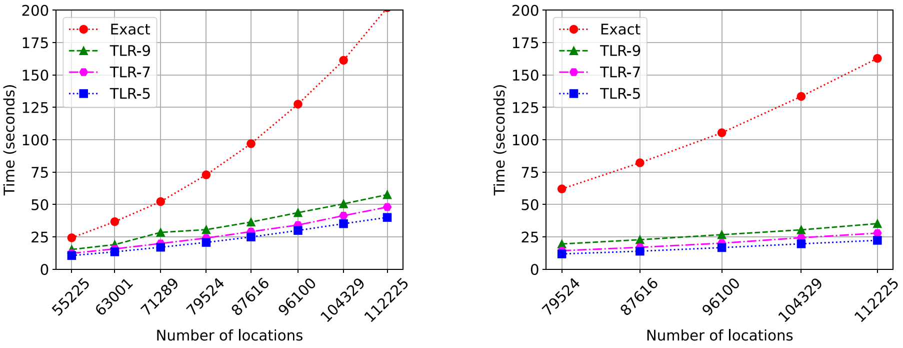


Fig. 10. PIT histograms of different space-time models for the space-time real data.

Table 3
Parameter estimates for the space-time Gaussian, TGH, and the space-time model in the Gaussian scale for the monthly precipitation data of the year 2021 over $20K \times 12$ space-time locations.

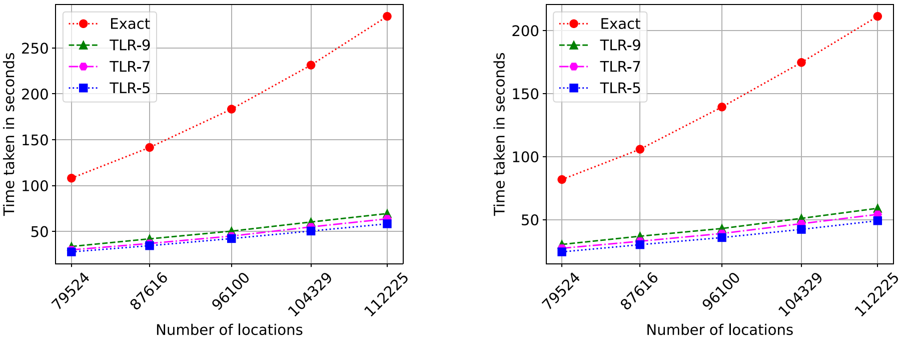
Model	ξ	ω	g	h	a_s	ν	a_t	α	β
Space-Time Gaussian	0.03	0.92	-	-	72.28	0.75	0.34	1	0.71
Space-Time TGH	-0.55	0.6	-0.45	0.19	56.26	0.84	0.32	0.73	0.40
Space-Time Gaussian in Gaussian scale	0 (fixed)	1 (fixed)	-	-	72.32	0.81	0.13	0.45	0.17



(a) 56-core Intel IceLake

(b) 128-core AMD Milan

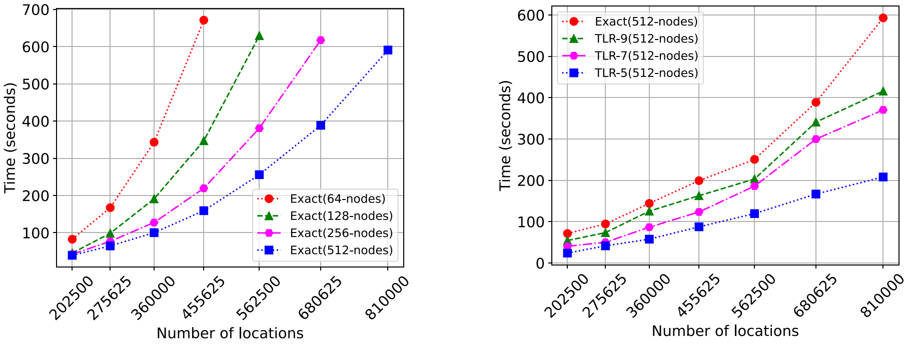
Fig. 11. Performance of a single TGH space MLE iteration on shared-memory architectures using exact and TLR matrices.



(a) 56-core Intel IceLake.

(b) 128-core AMD Milan.

Fig. 12. Performance of a single TGH space prediction operation on shared-memory architectures using exact and TLR matrices.



(a) Scalability up to 512 nodes.

(b) Performance on 512 nodes.

Fig. 13. (a) Scalability of a single non-Gaussian MLE iteration on Shaheen-II with up to 512 nodes using exact matrices. (b) Performance of a single non-Gaussian MLE iteration on Shaheen-II nodes using exact and TLR matrices.

posed an exact and approximate parallel solution for modeling and prediction operations with the aid of existing dense linear algebra libraries to tackle the underlying matrix operations with large problem sizes. Furthermore, we rely on the Tile Low-Rank (TLR) approximation to alleviate the complexity of the modeling process in the TGH modeling. The TLR approximation was performed for modeling and prediction operations. Using up to $800K$ geospace locations, it also showed a speedup of up to $7.29X$ and $2.96X$ compared to the exact implementation on shared and distributed-memory systems. For future work, we plan to consider mixed-precision arithmetics with GPU hardware accelerators to boost the performance of dense linear algebra kernels. We would also like to investigate randomized algorithms to directly generate the compressed matrix and improve the matrix assembly phase.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgment

This work is funded and supported by King Abdullah University of Science and Technology (KAUST) through the Office of Sponsored Research (OSR). This research used the resources of the

Extreme Computing Research Center (ECRC) and the KAUST Supercomputing Laboratory, including Cray XC40, Shaheen II supercomputer.

References

[1] Sameh Abdulah, Hatem Ltaief, Ying Sun, Marc G. Genton, David E. Keyes, ExaGeoStat: a high performance unified software for geostatistics on manycore systems, IEEE Trans. Parallel Distrib. Syst. 29 (12) (2018) 2771–2784.

[2] Sameh Abdulah, Hatem Ltaief, Ying Sun, Marc G. Genton, David E. Keyes, Parallel approximation of the maximum likelihood estimation for the prediction of large-scale geostatistics simulations, in: IEEE International Conference on Cluster Computing (CLUSTER), 2018, pp. 98–108.

[3] Sameh Abdulah, Kadir Akbudak, Wajih Boukaram, Ali Charara, David Keyes, Hatem Ltaief, Aleksander Mikhalev, Dalal Sukkari, George Turkiyyah, Hierarchical computations on manycore architectures (HiCMA), See http://github.com/ecrc/hicma, 2022.

[4] Sameh Abdulah, Qinglei Cao, Yu Pei, George Bosilca, Jack Dongarra, Marc G. Genton, David Keyes, Hatem Ltaief, Ying Sun, Accelerating geostatistical modeling and prediction with mixed-precision computations: a high-productivity approach with PaRSEC, IEEE Trans. Parallel Distrib. Syst. 33 (4) (2022) 964–976, https://doi.org/10.1109/TPDS.2021.3084071.

[5] Emmanuel Agullo, Jim Demmel, Jack Dongarra, Bilel Hadri, Jakub Kurzak, Julien Langou, Hatem Ltaief, Piotr Luszczek, Stanimire Tomov, Numerical linear algebra on emerging architectures: the plasma and magma projects, J. Phys. Conf. Ser. 180 (2009) 012037, IOP Publishing.

[6] Kadir Akbudak, Hatem Ltaief, Aleksandr Mikhalev, David Keyes, Tile low rank Cholesky factorization for climate/weather modeling applications on manycore architectures, in: International Supercomputing Conference, Springer, 2017, pp. 22–40.

[7] David J. Allcroft, Chris A. Glasbey, A latent Gaussian Markov random-field model for spatiotemporal rainfall disaggregation, J. R. Stat. Soc., Ser. C, Appl. Stat. 52 (4) (2003) 487–498.

[8] Patrick Amestoy, Cleve Ashcraft, Olivier Boiteau, Alfredo Buttari, Jean-Yves L'Excellent, Clément Weisbecker, Improving multifrontal methods by means of block low-rank representations, SIAM J. Sci. Comput. (2015).

[9] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. DuCroz, A. Greenbaum, S. Hammarling, A. McKenney, D. Sorensen, Lapack: A Portable Linear Algebra Library for High-Performance Computers, 1990.

[10] Cédric Augonnet, Samuel Thibault, Raymond Namyst, Pierre-André Wacrenier, Starpu: a unified platform for task scheduling on heterogeneous multicore architectures, in: European Conference on Parallel Processing, Springer, 2009, pp. 863–874.

[11] Cédric Augonnet, Samuel Thibault, Raymond Namyst, Pierre-André Wacrenier, StarPU: a unified platform for task scheduling on heterogeneous multicore architectures, Concurr. Comput. 23 (2) (2011) 187–198.

[12] Zahra Barzegar, Firoozeh Rivaz, Jafari Khaledi, A skew-Gaussian spatio-temporal process with non-stationary correlation structure, J. Iran. Stat. Soc. 18 (2) (2019) 63–85.

[13] Veronica J. Berrocal, Alan E. Gelfand, David M. Holland, A bivariate space-time downscaler under space and time misalignment, Ann. Appl. Stat. 4 (4) (2010) 1942–1975.

[14] Moreno Bevilacqua, Christian Caamaño-Carrillo, Reinaldo B. Arellano-Valle, Víctor Morales-Oñate, Non-Gaussian geostatistical modeling using (skew) t processes, Scand. J. Stat. 48 (1) (2021) 212–245.

[15] George Bosilca, Aurelien Bouteiller, Anthony Danalis, Mathieu Faverge, Azzam Haidar, Thomas Herault, Jakub Kurzak, Julien Langou, Pierre Lemarinier, Hatem Ltaief, et al., Flexible development of dense linear algebra algorithms on massively parallel architectures with dplasma, in: 2011 IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum, IEEE, 2011, pp. 1432–1441.

[16] George Bosilca, Aurelien Bouteiller, Anthony Danalis, Mathieu Faverge, Thomas Hérault, Jack J. Dongarra, Parsec: exploiting heterogeneity to enhance scalability, Comput. Sci. Eng. 15 (6) (2013) 36–45.

[17] Chameleon Software, The Chameleon project, https://project.inria.fr/chameleon/, October 2022.

[18] Tangpei Cheng, Accelerating universal Kriging interpolation algorithm using CUDA-enabled GPU, Comput. Geosci. 54 (2013) 178–183.

[19] Jaeyoung Choi, Jack J. Dongarra, Roldan Pozo, David W. Walker, Scalapack: a scalable linear algebra library for distributed memory concurrent computers, in: The Fourth Symposium on the Frontiers of Massively Parallel Computation, IEEE Computer Society, 1992, pp. 120–121.

[20] Leonardo Dagum, Ramesh Menon, Openmp: an industry standard api for shared-memory programming, IEEE Comput. Sci. Eng. 5 (1) (1998) 46–55.

[21] Victor De Oliveira, On optimal point and block prediction in log-Gaussian random fields, Scand. J. Stat. 33 (3) (2006) 523–540.

[22] Victor De Oliveira, Benjamin Kedem, David A. Short, Bayesian prediction of transformed Gaussian random fields, J. Am. Stat. Assoc. 92 (440) (1997) 1422–1433.

[23] Alejandro Duran, Eduard Ayguadé, Rosa M. Badia, Jesús Labarta, Luis Martinell, Xavier Martorell, Judit Planas, Ompss: a proposal for programming heterogeneous multi-core architectures, Parallel Process. Lett. 21 (02) (2011) 173–193.

[24] Thaís C.O. Fonseca, Mark F.J. Steel, Non-Gaussian spatiotemporal modelling through scale mixing, Biometrika 98 (4) (2011) 761–774.

[25] Reinhard Furrer, Marc G. Genton, Douglas Nychka, Covariance tapering for interpolation of large spatial datasets, J. Comput. Graph. Stat. 15 (3) (2006) 502–523.

[26] Christopher J. Geoga, Mihai Anitescu, Michael L. Stein, Scalable Gaussian process computations using hierarchical matrices, J. Comput. Graph. Stat. 29 (2) (2020) 227–237.

[27] Tilmann Gneiting, Nonseparable, stationary covariance functions for space–time data, J. Am. Stat. Assoc. 97 (458) (2002) 590–600.

[28] TLRSG Goulart, Leonardo Goliatta Tavaresa, Marcelo Loboscoa Rodrigo, Weber dos Santosa, Filipe de Oliveira Chavesb, A parallel implementation of the ordinary Kriging algorithm for heterogeneous computing environments, J. Comput. Interdiscip. Sci. 8 (3) (2017) 143–152.

[29] Arnab Hazra, Brian J. Reich, Benjamin A. Shaby, Ana-Maria Staicu, A semiparametric Bayesian model for spatiotemporal extremes, arXiv preprint, arXiv:1812.11699, 2018.

[30] Craig J. Johns, Douglas Nychka, Timothy G.F. Kittel, Chris Daly, Infilling sparse records of spatial fields, J. Am. Stat. Assoc. 98 (464) (2003) 796–806.

[31] F. Kaspar, G. Müller-Westermeier, E. Penda, H. Mächel, K. Zimmermann, A. Kaiser-Weiss, T. Deutschländer, Monitoring of climate change in Germany–data, products and services of Germany's National Climate Data Centre, Adv. Appl. Sci. Res. 10 (1) (2013) 99–106.

[32] Hyoung-Moon Kim, Bani K. Mallick, A Bayesian prediction using the skew Gaussian distribution, J. Stat. Plan. Inference 120 (1–2) (2004) 85–101.

[33] Pavel Krupskii, Marc G. Genton, Factor copula models for data with spatio-temporal dependence, Spat. Stat. 22 (2017) 180–195.

[34] Pavel Krupskii, Raphaël Huser, Marc G. Genton, Factor copula models for replicated spatial data, J. Am. Stat. Assoc. 113 (521) (2018) 467–479.

[35] Yulia V. Marchenko, Marc G. Genton, Multivariate log-skew-elliptical distributions with applications to precipitation data, Environmetrics 21 (3–4) (2010) 318–340.

[36] Sagnik Mondal, Sameh Abdulah, Hatem Ltaief, Ying Sun, Marc G. Genton, David E. Keyes, Parallel approximations of the Tukey g-and-h likelihoods and predictions for non-Gaussian geostatistics, in: 2022 IEEE International Parallel and Distributed Processing Symposium (IPDPS), 2022, pp. 379–389.

[37] M. Blanca Palacios, Mark F.J. Steel, Non-Gaussian Bayesian geostatistical modeling, J. Am. Stat. Assoc. 101 (474) (2006) 604–618.

[38] Gonzalo Rios, Felipe Tobar, Learning non-Gaussian time series using the Box-Cox Gaussian process, in: 2018 International Joint Conference on Neural Networks (IJCNN), IEEE, 2018, pp. 1–8.

[39] Jo Røislien, Henning Omre, T-distributed random fields: a parametric model for heavy-tailed well-log data, Math. Geol. 38 (7) (2006) 821–849.

[40] Mary Lai O. Salvaña, Sameh Abdulah, Huang Huang, Hatem Ltaief, Ying Sun, Marc G. Genton, David E. Keyes, High performance multivariate geospatial statistics on manycore systems, IEEE Trans. Parallel Distrib. Syst. 32 (11) (2021) 2719–2733.

[41] Mary Lai O. Salvaña, Sameh Abdulah, Hatem Ltaief, Ying Sun, Marc G. Genton, David E. Keyes, Parallel space-time likelihood optimization for air pollution prediction on large-scale systems, in: Proceedings of the Platform for Advanced Scientific Computing Conference, PASC '22, New York, NY, USA, Association for Computing Machinery, ISBN 9781450394109, 2022.

[42] Huiyan Sang, Jianhua Z. Huang, A full scale approximation of covariance functions for large spatial data sets, J. R. Stat. Soc., Ser. B, Stat. Methodol. 74 (1) (2012) 111–132.

[43] Michael L. Stein, Jie Chen, Mihai Anitescu, Stochastic approximation of score functions for Gaussian processes, Ann. Appl. Stat. (2013) 1162–1191.

[44] Ying Sun, Huixia J. Wang, Montserrat Fuentes, Fused adaptive lasso for spatial and temporal quantile function estimation, Technometrics 58 (1) (2016) 127–137.

[45] Felipe Tagle, Stefano Castruccio, Marc G. Genton, A hierarchical bi-resolution spatial skew-t model, Spat. Stat. 35 (2020) 100398.

[46] Yanlin Tang, Huixia J. Wang, Ying Sun, Amanda S. Hering, Copula-based semiparametric models for spatiotemporal data, Biometrics 75 (4) (2019) 1156–1167.

[47] Michael E. Wall, Andreas Rechtsteiner, Luis M. Rocha, Singular value decomposition and principal component analysis, in: A Practical Approach to Microarray Data Analysis, Springer, 2003, pp. 91–109.

[48] Jonas Wallin, David Bolin, Geostatistical modelling using non-Gaussian Matérn fields, Scand. J. Stat. 42 (3) (2015) 872–890.

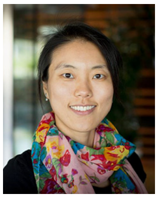
[49] Ganggang Xu, Marc G. Genton, Tukey g-and-h random fields, J. Am. Stat. Assoc. 112 (519) (2017) 1236–1249.

[50] Yuan Yan, Marc G. Genton, Non-Gaussian autoregressive processes with Tukey g-and-h transformations, Environmetrics 30 (2) (2019).

Sagnik Mondal is currently a PhD student in the Statistics program at KAUST, in Spatio-Temporal Statistics & Data Science group led by Dr Marc G. Genton. Before joining KAUST, Sagnik completed a Master of Science degree in Statistics from the Indian Institute of Technology (IIT), Kanpur, India. His research interest is multivariate non-gaussian distribution and non-gaussian random field.

Sameh Abdulah received the MS and PhD degrees from Ohio State University, Columbus, USA, in 2014 and 2016, respectively. He is currently a research scientist with the Extreme Computing Research Center (ECRC), King Abdullah University of Science and Technology, Saudi Arabia. His research interests include high performance computing applications, bitmap indexing in big data, large spatial datasets, parallel statistical applications, algorithm-based fault tolerance, and machine learning and data mining algorithms.

Hatem Ltaief is the Principal Research Scientist of the Extreme Computing Research Center, King Abdullah University of Science and Technology (KAUST), Saudi Arabia. His research interests include parallel numerical algorithms, parallel programming models, and performance optimizations for multicore architectures and hardware accelerators.



Ying Sun is an Associate Professor of Statistics at King Abdullah University of Science and Technology (KAUST). She received her Ph.D. degree in statistics from Texas A&M University in 2011. At KAUST, she established and leads the Environmental Statistics Research Group, developing statistical models and methods for complex data to address important environmental problems. She has made original contributions to environmental statistics, in particular in the areas of spatiotemporal statistics, functional data analysis, visualization, and computational statistics, with an exceptionally broad array of applications.

Marc G. Genton is a Distinguished Professor of Statistics at the King Abdullah University of Science and Technology (KAUST) in Saudi Arabia. He received the Ph.D. degree in Statistics (1996) from the Swiss Federal Institute of Technology (EPFL), Lausanne. He is a fellow of the American Statistical Association (ASA), of the Institute of Mathematical Statistics (IMS), and the American Association for the Advancement of Science (AAAS), and is an elected member of the International Statistical Institute (ISI). In 2010, he received the El-Shaarawi award for excellence from the International Environmetrics Society (TIES) and the Distinguished Achievement award from the Section on Statistics and the Environment (ENVR) of the American Statistical Association (ASA). He received an ISI Service award in 2019 and the Georges Matheron Lectureship award in 2020 from the International Association for Mathematical Geosciences (IAMG). He led a Gordon Bell Prize finalist team with the ExaGeoStat software for Super Computing 2022. He received the Royal Statistical Society (RSS) 2023 Barnett Award for his outstanding research in environmental statistics. His research interests include statistical analysis, flexible modeling, prediction, and uncertainty quantification of spatio-temporal data, with applications in environmental and climate science, as well as renewable energies.

David Keyes is a professor of applied mathematics, computer science, and mechanical engineering at the King Abdullah University of Science and Technology (KAUST), where he directs the Extreme Computing Research Center, and where he was founding Dean in 2009. He is also Adjunct Professor of applied mathematics at Columbia and an affiliate of several US national labs. He earned a BSE in aerospace and mechanical sciences from Princeton in 1978 and a PhD in applied mathematics from Harvard in 1984. He works at the interfaces between parallel computing and the numerical analysis of PDEs and spatial statistics, with a focus on scalable implicit solvers and exploiting data sparsity. He helped develop and popularize the Newton-Krylov-Schwarz (NKS) and Additive Schwarz Preconditioned Inexact Newton (ASPIN) methods. He has been awarded the ACM Gordon Bell Prize and the IEEE Sidney Fernbach Prize and is a fellow of the SIAM, AMS, and AAAS.