

A Fast Algorithm for Computing Multivariate Normal Densities using Matérn-like Precision Structures with Unit Marginal Variances

Brynjólfur Gauti Guðrúnar Jónsson*

University of Iceland,

E-mail: brynjolfur@hi.is

Introduction

Let $\mathbf{X} = (X_1, X_2, \dots, X_n)$ be a multivariate random vector with marginal distribution functions F_i for $i = 1, 2, \dots, n$. The joint distribution function of \mathbf{X} can be written as:

$$F_{\mathbf{X}}(\mathbf{x}) = C(F_1(x_1), F_2(x_2), \dots, F_n(x_n)),$$

where C is the Gaussian copula defined by the GMRF precision matrix Q .

The Gaussian copula C is given by:

$$C(u_1, u_2, \dots, u_n) = \Phi_Q(\Phi^{-1}(u_1), \Phi^{-1}(u_2), \dots, \Phi^{-1}(u_n)),$$

where Φ_Q is the joint cumulative distribution function of a multivariate normal distribution with mean vector $\mathbf{0}$ and precision matrix Q , and Φ^{-1} is the inverse of the standard normal cumulative distribution function.

It is imperative that the precision matrix Q governing the GMRF Copula, C , has marginal variance equal to 1 so that is it on the same scale as the transformed data, $\Phi^{-1}(u_i)$. This can be troublesome because GMRFs are defined in terms of their precision matrices, Q , which more often than not have marginal variances that are different from 1.

This paper presents an fast and efficient algorithm for creating a Matérn-like precision matrix, Q , with unit marginal variance, and computing the multivariate Gaussian copula density of $Z = \Phi^{-1}(u)$ where $u \sim \text{Uniform}(0, 1)$.

The matrix, Q , is defined as

$$Q = Q_1 \otimes I + I \otimes Q_1,$$

where Q_1 is the precision matrix of a standardized one-dimensional AR(1) process and \otimes is the kronecker product.

The method leverages the special structure of Q_1 and the use of a kronecker sum in the definition to avoid explicit formation and inversion of the larger matrix, Q , making it particularly suitable for high-dimensional spatial data.

Methods

Theory

One-Dimensional AR(1) Matrix

The core of our approach is based on the eigendecomposition of an AR(1) precision matrix, which forms the building block of our Matérn-like precision structure. For a one-dimensional AR(1) process with parameter ρ , the precision matrix Q_1 has a tridiagonal structure:

$$Q = \frac{1}{1 - \rho^2} \begin{bmatrix} 1 & -\rho & 0 & \cdots & 0 & 0 \\ -\rho & 1 + \rho^2 & -\rho & \cdots & 0 & 0 \\ 0 & -\rho & 1 + \rho^2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 + \rho^2 & -\rho \\ 0 & 0 & 0 & \cdots & -\rho & 1 \end{bmatrix}$$

Matérn-like Precision Matrix

For two-dimensional spatial fields, we construct a Matérn-like precision matrix Q using Kronecker products:

$$Q = Q_1 \otimes I + I \otimes Q_1$$

where I is the identity matrix and \otimes denotes the Kronecker product. The eigenvalues (λ_k) and eigenvectors (v_k) of Q_1 can be calculated numerically and used to evaluate the multivariate Gaussian density, letting us skip out on forming Q entirely. This is due to the following theorem:

Eigendecomposition of Kronecker Sums

Theorem

Let $A \in \mathbb{R}^{n \times n}$ have eigenvalues λ_i , $i \in \{1, \dots, n\}$, and let $B \in \mathbb{R}^{m \times m}$ have eigenvalues μ_j , $j \in \{1, \dots, m\}$. Then the Kronecker sum $A \oplus B = (I_m \otimes A) + (B \otimes I_n)$ has eigenvalues $\lambda_i + \mu_j$, $i \in \{1, \dots, n\}, j \in \{1, \dots, m\}$.

Moreover, if x_1, \dots, x_p are linearly independent right eigenvectors of A corresponding to $\lambda_1, \dots, \lambda_p$ ($p \leq n$), and z_1, \dots, z_q are linearly independent right eigenvectors of B corresponding to μ_1, \dots, μ_q ($q \leq m$), then $z_j \otimes x_i \in \mathbb{R}^{mn}$ are linearly independent right eigenvectors of

$A \oplus B$ corresponding to $\lambda_i + \mu_j$, $i \in \{1, \dots, p\}, j \in \{1, \dots, q\}$.

Discussion

This theorem provides a crucial insight that allows us to efficiently construct the eigendecomposition of the full precision matrix Q for two-dimensional spatial fields by leveraging the eigendecomposition of the one-dimensional precision matrix Q_1 , avoiding the computationally intensive process of explicitly forming and inverting the large matrix Q .

Furthermore, we can compute the log-density of an observation x directly by using just the eigenvalues and eigenvectors of Q_1 , thereby enabling efficient computation of the log-density of the multivariate normal distribution even for large spatial fields.

Example: Calculating Eigenvalues and Eigenvectors for Q Using Q_1

To illustrate the application of Theorem 13.16, consider the one-dimensional precision matrix Q_1 with known eigendecomposition. Suppose Q_1 has eigenvalues λ_i and corresponding eigenvectors v_i . For a two-dimensional spatial field, we construct the Matérn-like precision matrix Q using Kronecker products as follows:

$$Q = Q_1 \otimes I + I \otimes Q_1,$$

where I is the identity matrix and \otimes denotes the Kronecker product.

Eigenvalues

The eigenvalues of Q can be determined from the eigenvalues of Q_1 . If Q_1 has eigenvalues λ_i for $i = 1, 2, \dots, n$, then the eigenvalues of Q are given by:

$$\lambda_{ij} = \lambda_i + \lambda_j \quad \text{for } i, j = 1, 2, \dots, n.$$

Eigenvectors

Similarly, the eigenvectors of Q can be constructed from the eigenvectors of Q_1 . If v_i and v_j are eigenvectors of Q_1 corresponding to eigenvalues λ_i and λ_j respectively, then the eigenvectors of Q are given by:

$$v_{ij} = v_i \otimes v_j \quad \text{for } i, j = 1, 2, \dots, n.$$

Here, v_{ij} is the Kronecker product of v_i and v_j .

These relationships allow us to efficiently compute the eigendecomposition of the full precision matrix Q using the eigendecomposition of the smaller matrix Q_1 , significantly reducing the computational complexity.

Efficient Density Calculation Using Eigendecomposition

Given the eigendecomposition of Q_1 , we can efficiently compute the multivariate normal density with respect to the precision matrix Q . Let λ_i and v_i be the eigenvalues and eigenvectors of Q_1 , respectively. The log-density of a multivariate normal distribution with precision matrix Q is given by:

$$\log p(x) = -\frac{1}{2}(n \log(2\pi) + \log |Q| + x^T Q x)$$

where n is the dimension of x , $|Q|$ is the determinant of Q , and $x^T Q x$ is the quadratic form.

Log-Determinant Calculation

The log-determinant of Q can be computed efficiently using the eigenvalues of Q_1 :

$$\log |Q| = \sum_{i=1}^d \sum_{j=1}^d \log(\lambda_i + \lambda_j)$$

where d is the dimension of Q_1 .

Quadratic Form Calculation

First, we define the product of the eigenvector and the data vector x as:

$$y_{ij} = (v_i \otimes v_j)^T x,$$

where v_i and v_j are the eigenvectors of Q_1 and \otimes denotes the Kronecker product.

Next, we define the eigenvalue sum as:

$$\mu_{ij} = \lambda_i + \lambda_j,$$

where λ_i and λ_j are the eigenvalues of Q_1 . Using these definitions, the quadratic form can be expressed as:

$$x^T Q x = \sum_{i=1}^d \sum_{j=1}^d \mu_{ij} y_{ij}^2.$$

In this way, we calculate the quadratic form without having to form the matrix Q or its full set of eigenvectors or values.

Scaling the Input x

The input vector $x = \Phi^{-1}(u)$ has zero mean and unit marginal variance. Instead of scaling the precision matrix to have $(Q^{-1})_{ii} = 1$, we calculate the marginal standard deviations, σ_k , implied by Q using the eigenstructure of Q_1 , then scale the input vector x using those

standard deviations. First we compute the marginal standard deviations:

$$\sigma_k = \sqrt{\sum_{i=1}^d \sum_{j=1}^d \frac{(v_i \otimes v_j)_k^2}{\lambda_i + \lambda_j}}$$

where $(v_i \otimes v_j)_k$ is the k -th element of the Kronecker product $v_i \otimes v_j$.

Algorithm Implementation

The density calculation is implemented as follows:

- Compute the eigendecomposition of Q_1 .
- Calculate the marginal standard deviations σ_k .
- For each observation x :
 - a. Standardize x by element-wise multiplication with σ_k .
 - b. Compute the log-determinant and quadratic form using the formulas above.
 - c. Combine the terms to get the log-density.

This approach avoids explicit formation and inversion of the full precision matrix Q , allowing for efficient computation even for large spatial fields. This method has been implemented in C++ code that can perform a full set of computations for a 200×200 spatial grid (i.e. Q would be 40.000×40.000) in just over one second.