LIKELIHOOD OPTIMISATION FOR GAUSSIAN PROCESSES

Anant Mathur

Supervisor: Dr Zdravko Botev

School of Mathematics and Statistics
UNSW Sydney

December 2019

Submitted in partial fulfillment of the requirements of the degree of
Bachelor of Science with Honours

Plagiarism statement

Acknowledgements

I would like to express my sincere gratitude to to my supervisor, Zdravko, for his continued support and guidance throughout the year.

Thanks must also go to my friends and family for offering constant encouragement.

I would also like thank Askhay Shanker for proof-editing and helping me structure the introduction.

Abstract

Gaussian processes (GPs) are a flexible class of methods with a wide range of applications including curve fitting, time series, and spatial statistics. However, naive implementations of GPs are restricted by their $O(n^3)$ run time requirements, making their application infeasible for any problem of reasonable size.

This paper will describe improved Monte Carlo estimates for current state-of-the-art stochastic optimization techniques. These Monte Carlo methods promise to offer $O(n\log n)$ likelihood optimization evaluation times for stationary GPs through the circulant embedding of toeplitz matrices. The accuracy gains in our adjusted Monte Carlo estimates are achieved through variance reduction techniques, and, allow one to employ the same procedures with a smaller Monte Carlo size. To test the validity of these adjusted estimators, the performance gain is theoretically and numerically analysed in the context of these stochastic procedures.

Additionally, a Monte Carlo free optimisation approach is described for the same problem. This new method allows for similarly fast computations without the inherent stochastic error, and, the requirement of sub-iterative conjugate methods that may be ill-conditioned.

Keywords

Gaussian Processes, circulant embedding, Toeplitz linear systems, Monte Carlo Expectation-Maximization, score functions, variance reduction, maximum Likelihood estimation.

Contents

Notation

$\mathcal{CN}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	multivariate complex normal with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	multivariate normal with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$
$\hat{z}$	estimate or approximation
$\mathbb{E}$	expectation
$\mathbb{V}\text{ar}$	variance
$\mid \boldsymbol{A} \mid$	determinant of matrix $\boldsymbol{A}$
$\operatorname{tr}(\boldsymbol{A})$	trace of matrix $\boldsymbol{A}$
$\boldsymbol{A}^\top$, $\boldsymbol{z}^\top$	transpose of matrix $\boldsymbol{A}$ or vector $\boldsymbol{z}$
$\boldsymbol{A}^H$, $\boldsymbol{z}^H$	conjugate transpose of matrix $\boldsymbol{A}$ or vector $\boldsymbol{z}$
$\boldsymbol{C_{oo}}$	upper left block of matrix $\boldsymbol{C}$
$\boldsymbol{C_{ou}}$	upper right block of matrix $\boldsymbol{C}$
$\boldsymbol{C_{uo}}$	lower left block of matrix $\boldsymbol{C}$
$\boldsymbol{C_{uu}}$	lower right block of matrix $\boldsymbol{C}$
$\boldsymbol{F}$	DFT matrix
$\boldsymbol{F}^{-1}$	inverse DFT matrix
$\boldsymbol{I}_n$	$n \times n$ identity matrix
$\boldsymbol{Z}$	random vector
$\boldsymbol{z}$	vector
z	scalar
EM	expectaction maximisation
GP	Gaussian process
KL	Kullback-Liebler
ln	(natural) log
MLE	maximum likelihood estimator/estimate

Chapter 1

Introduction

In today's age of big data it has become pertinent to employ statistical models that can learn rich representations of data. In doing so, one can achieve both a mechanism for inference on large data sets and improve performance in predictive tasks. Gaussian processes (GPs) promise to be one such class of models; they allow for flexible non-parametric curve fitting whilst inheriting many of the advantageous properties of the Gaussian probability distribution [30].

Applications of Gaussian processes to large data sets requires likelihood inference. However, likelihood inference is generally infeasible for large data sets due to high computational costs. The cost arises due to the calculation of a matrix inverse, which in turn requires $O(n^3)$ computational operations, where n is size of the data. To deal with the problem of the large number of computational operations, a number of Monte Carlo likelihood approximations have been utilised by the literature to estimate parameters of stationary Gaussian processes. These methods employ circulant embedding techniques, allowing the implementation of the fast Fourier transform (FFT) to compute matrix operations in $O(n\log n)$ time. Intuitively, these approaches embed smaller dimensional optimization problems in larger dimensions, subsequently allowing the structure of the larger matrices to be exploited.

However, the Monte Carlo approximation methods currently used by the literature, namely, the Monte Carlo Expectation Maximisation (EM) algorithm [35] and the stochastic approximation of the score function [34], rely on crude Monte Carlo estimates. In particular, these methods face two short-comings. First, in order to gain precise parameter estimates, these estimates may incur a computational cost that may counteract the efficiencies gained by circulant embedding. Second, these algorithms rely on sub-iterative simulation and conjugate methods that are not guaranteed to be fast or well-conditioned. The purpose of this paper is to address these two challenges. First, we propose variance reduction techniques that theoretically increase the accuracy of the Monte Carlo estimates. The increased accuracy of the Monte Carlo estimates allow us to reduce the necessary Monte Carlo size which subsequently reduces computational cost. Second, to avoid the sub-iterative procedures, we present a deterministic approach that simplifies the maximum likelihood estimation to a computationally feasible global optimization problem. These improved procedures can be applied to larger data sets where existing likelihood techniques cannot yield accurate parameter estimates.

The rest of this paper proceeds as follows, in Chapter 1, we provide an introduction to Gaussian processes and describe the likelihood problem. Certain motivating examples will be given, illustrating the wide applicability of Gaussian Processes. In Chapter 2, we will critically analyse pre-exisitng Monte Carlo techniques and describe numerous variance reduction improvements. Specifically, we will inspect the advantages and limitations of these variance reduction improvements to the Monte Carlo Expectation Maximisation (EM) algorithm and the stochastic approach to estimating the score. In Chapter 3, we will propose a deterministic method that alleviates numerous iterative shortcomings apparent in stochastic methods. Finally in Chapter 4, we provide numerical simulations of these algorithms that provide a comparison of effective performance.

Chapter 2

Gaussian Processes

In essence GPs can be thought of as a generalization of the Gaussian distribution; from a distribution over vectors to a distribution over functions. Therefore, we see a rich history of GP implementation tracing back to over a century ago, with the discovery and use of important stochastic processes such as the Weiner process, Ornstein-Uhlenbeck process and the Brownian bridge.

Since the 1970's Gaussian processes have been heavily implemented in the field of geostatistics and meteorology, with the development of Kriging, often referred to as Gaussian process regression by the French mathematician Georges Matheron and South African mining engineer Daniel G. Krige. Since spatial statistics is a core source of GP research it is common to find research applied to processes existing in two or three space dimensions.

2.1 Definition and Likelihood

Formally we define a Gaussian process as the following.

Definition 2.1.1. *A collection of random variables $\{Z_{\boldsymbol{x}} : \boldsymbol{x} \in \mathcal{X}\}$ is defined to be a Gaussin process (GP) on the space $\mathcal{X}$ if, for any choice of indices $\boldsymbol{x_1}, \ldots, \boldsymbol{x_n} \in \mathcal{X}$ the vector $[Z_{x_1}, \ldots, Z_{x_n}]^\top$ has a multivariate Gaussian distribution.*

A GP is fully specified by a mean function $\mu : \mathcal{X} \to \mathbb{R}$ and a covariance function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, which we define for any $\boldsymbol{x}$, $\boldsymbol{x'} \in \mathcal{X}$ as

$$\begin{aligned} \mu(\boldsymbol{x}) &= \mathbb{E}[Z_{\boldsymbol{x}}], \\ k(\boldsymbol{x}, \boldsymbol{x'}) &= \mathbb{E}\left[(Z_{\boldsymbol{x}} - \mu(\boldsymbol{x}))(Z_{\boldsymbol{x'}} - \mu(\boldsymbol{x'}))\right]. \end{aligned} \tag{2.1.1}$$

The function $k(\cdot, \cdot)$ is referred to as the covariance function or kernel, often governed by a set of *hyperparameters* denoted collectively as the vector $\boldsymbol{\theta}$. In section 1.3 we shall provide examples of commonly-used covariance functions and examine their properties in more detail, but for now we are free in our choice of covariance function, as long as the kernel satisfies Mercer's condition - that is, for all square-integrable functions $f(x) : \mathcal{X} \to \mathbb{R}$,

$$\int f(\boldsymbol{x})k(\boldsymbol{x}, \boldsymbol{x'})f(\boldsymbol{x'})\, d\boldsymbol{x}\, d\boldsymbol{x'} \geq 0. \tag{2.1.2}$$

Let $\boldsymbol{Z} = [Z_{\boldsymbol{x_1}}, \dots, Z_{\boldsymbol{x_n}}]^\top$ be taken from a GP with a mean function with constant value μ and covariance function $k(\cdot, \cdot; \boldsymbol{\theta})$, letting $\boldsymbol{\Theta} = \{\mu, \boldsymbol{\theta}\}$ denote the unknown parameters. It follows from definition, that $\boldsymbol{Z}$ will simply collapse to a drawn vector from the multivariate distribution $\boldsymbol{Z} \sim \mathcal{N}(\mu\mathbf{1}, \boldsymbol{C}(\boldsymbol{\theta}))$, where the covariance matrix $\boldsymbol{C}(\boldsymbol{\theta})$ is a *Gram matrix*, specified by,

$$\boldsymbol{C}(\boldsymbol{\theta}) = \begin{bmatrix} k(\boldsymbol{x_1}, \boldsymbol{x_1}) & k(\boldsymbol{x_1}, \boldsymbol{x_2}) & \dots & k(\boldsymbol{x_1}, \boldsymbol{x_n}) \\ k(\boldsymbol{x_2}, \boldsymbol{x_1}) & k(\boldsymbol{x_2}, \boldsymbol{x_2}) & \dots & k(\boldsymbol{x_2}, \boldsymbol{x_n}) \\ \vdots & \vdots & \ddots & \vdots \\ k(\boldsymbol{x_n}, \boldsymbol{x_1}) & k(\boldsymbol{x_n}, \boldsymbol{x_2}) & \dots & k(\boldsymbol{x_n}, \boldsymbol{x_n}) \end{bmatrix}.$$

An equivalent condition for (2.1.2) is to say a kernel function must always give rise to a *positive semidefinite* Gram matrix. That is, the $n \times n$ matrix $\boldsymbol{C}(\boldsymbol{\theta})$ must satisfy, $\mathbf{v}^\top \boldsymbol{C}(\boldsymbol{\theta})\mathbf{v} \geq 0$ for all vectors $\mathbf{v} \in \mathbb{R}^n$.

Maximum Likelihood Estimation

Suppose we have the observed data $\boldsymbol{z}$, treated as a realisation of the multivariate Gaussian vector $\boldsymbol{Z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{C}(\boldsymbol{\theta}))$, the likelihood function can then be defined as,

$$\begin{aligned} L(\boldsymbol{\Theta}; \boldsymbol{z}) &:= p(\boldsymbol{z}|\boldsymbol{\Theta}), \\ &= (2\pi)^{-\frac{n}{2}} |\boldsymbol{C}(\boldsymbol{\theta})|^{-\frac{1}{2}} \exp\left\{ \frac{1}{2}(\boldsymbol{z} - \boldsymbol{\mu})^\top \boldsymbol{C}(\boldsymbol{\theta})^{-1}(\boldsymbol{z} - \boldsymbol{\mu}) \right\}. \end{aligned}$$

We can also consider the log-likelihood given by,

$$\begin{aligned} \ell(\boldsymbol{\Theta}; \boldsymbol{z}) &:= \ln p(\boldsymbol{z}|\boldsymbol{\Theta}), \\ &= -\frac{n}{2} \ln 2\pi - \frac{1}{2} \ln |\boldsymbol{C}(\boldsymbol{\theta})| - \frac{1}{2}(\boldsymbol{z} - \boldsymbol{\mu})^\top \boldsymbol{C}(\boldsymbol{\theta})^{-1}(\boldsymbol{z} - \boldsymbol{\mu}). \end{aligned} \tag{2.1.3}$$

The likelihood function gives a criterion to evaluate how likely particular values of $\boldsymbol{\Theta}$ are for a given set of data $\boldsymbol{z}$. Whereby $\ell(\boldsymbol{\Theta}_1; \boldsymbol{z}) > \ell(\boldsymbol{\Theta}_2; \boldsymbol{z})$ indicates that the data $\boldsymbol{z}$ is more likely under a GP with parameter $\boldsymbol{\Theta}_1$ than $\boldsymbol{\Theta}_2$. With the aim to choose the parameter that "best" represents the data, we refer to the parameter that which maximizes $\ell(\boldsymbol{\Theta}; \boldsymbol{z})$ over the whole parameter space as the *maximum likelihood estimator* (MLE),

$$\hat{\boldsymbol{\Theta}}_{\text{MLE}} = \underset{\boldsymbol{\Theta}}{\operatorname{argmax}}\, \ell(\boldsymbol{\Theta}; \boldsymbol{z})$$

Obtaining this estimate therefore becomes the primary goal when fitting Gaussian process models. Traditional methods are constricted to problems with at most only a few thousand training points due to the $O(n^3)$, $O(n^2)$ computation time and memory required when storing, inverting and calculating the log-determinant of any specified gram matrix $C(\boldsymbol{\theta})$.

One can use the Cholesky decomposition [26]; a decomposition of a positive definite matrix into the product of a lower triangular matrix and its conjugate transpose,

$$\boldsymbol{C}(\boldsymbol{\theta}) = \boldsymbol{L}\boldsymbol{L}^\top,$$

to evaluate the likelihood in $O(n^3)$ time, we would employ the following algorithm.

Algorithm 2.1: $O(n^3)$ Evaluation of the Gaussian Likelihood

```
    Input  : Observed data $\boldsymbol{z}$; Covaraince matrix $\boldsymbol{C}(\boldsymbol{\theta})$
    Output: $\ln p(\boldsymbol{z}|\boldsymbol{\theta})$
1   Compute $\boldsymbol{L}$, the Cholesky decomposition of $\boldsymbol{C}(\boldsymbol{\theta})$
2   $\boldsymbol{\xi} \leftarrow \boldsymbol{L}^{-1}(\boldsymbol{z} - \boldsymbol{\mu})$
3   $l \leftarrow -\sum_{i=1}^{n} \ln \boldsymbol{L}_{ii} - \frac{1}{2}\boldsymbol{\xi}^\top\boldsymbol{\xi}$
4   return $l$
```

Score

If we were to approach this problem through gradient-based optimization procedures we require first derivatives of (2.1.3) respect to the parameter vector $\boldsymbol{\theta}$.

Using the fact that the derivative of a matrix inverse with respect to the scalar a can be expressed as

$$\frac{\partial}{\partial a}\left(\mathbf{A}^{-1}\right) = -\mathbf{A}^{-1}\frac{\partial \mathbf{A}}{\partial a}\mathbf{A}^{-1},$$

and additionally, the derivative of the log determinant of a matrix can be given by

$$\frac{\partial}{\partial a}\ln|\mathbf{A}| = \mathbf{Tr}\left(A^{-1}\frac{\partial \mathbf{A}}{\partial a}\right),$$

we obtain the partial log-likelihood gradient,

$$\frac{\partial}{\partial \theta_i}\ell(\boldsymbol{\Theta};\boldsymbol{z}) = -\frac{1}{2}\mathbf{Tr}\left(\boldsymbol{C}(\boldsymbol{\theta})^{-1}\frac{\partial \boldsymbol{C}(\boldsymbol{\theta})}{\partial \theta_i}\right) + \frac{1}{2}(\boldsymbol{z}-\boldsymbol{\mu})^\top \boldsymbol{C}(\boldsymbol{\theta})^{-1}\frac{\partial \boldsymbol{C}(\boldsymbol{\theta})}{\partial \theta_i}\boldsymbol{C}(\boldsymbol{\theta})^{-1}(\boldsymbol{z}-\boldsymbol{\mu}).$$

Once again naive methods for computing the gradient have a cost of $O(n^3)$. We explore stochastic score estimation methods that aim to alleviate this burden in Section 2.5. In order to optimize (2.1.3) without the explicit evaluation of the above gradient function, we employ Quasi-Newton methods. This class of methods can be used to find the local maxima of functions as an alternative to Newton's method, without the need of the gradient function or Hessian matrix. For the sake of simplicity, from this group of algorithms, we only rely on the Broyden-Fletcher-Goldfarb-Shannon (BFGS) algorithm [19], as $\boldsymbol{\theta}$ is generally low-dimensional. The BFGS method can be implemented through the `fminuc` Matlab function.

2.2 Covariance Functions

A critical part of model fitting with GP's is the choice of covariance function as it allows the encoding of prior assumptions about the process from which we wish to infer from. This section aims to provide examples of common covariance functions and to examine their properties.

Stationarity

Throughout this thesis we deal with *stationary* kernels, covariance functions that are dependent on $\boldsymbol{x} - \boldsymbol{x'}$ only and therefore invariant to translations in input space. Furthermore kernel functions dependent on $||\boldsymbol{x} - \boldsymbol{x'}||$ are called *isotropic*, and are invariant to rotations in input space. In the case of a stationary GP the covariance function can be simplified to,

$$k(\boldsymbol{h}) = k(\boldsymbol{0}, \boldsymbol{h}) = k(\boldsymbol{x}, \boldsymbol{x} + \boldsymbol{h}).$$

Matérn Covariance function

The Matérn class of covariance functions [1] can given by,

$$k(h) = \frac{2^{1-\nu}a}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu h}}{l} \right)^{\nu} K_{\nu} \left(\frac{\sqrt{2\nu h}}{l} \right), \tag{2.2.1}$$

where K_ν is a modified Bessel function, Γ is the gamma function, and l, ν and a are positive hyperparameters.

The GP with covariance function belonging the Matérn class is said to be k-times mean-square differentiable if and only if $\nu > k$.

Squared Exponential Covariance function

In the special case of the Matérn covariance, where $\nu \to \infty$ we obtain the squared exponential covariance function,

$$k(h) = a \exp\left(\frac{-h^2}{2l^2} \right), \tag{2.2.2}$$

with l defined as the characteristic length scale parameter. The squared exponential covariance function is infinitely differentiable in the mean square sense, and consequently, appears very smooth. Figures 1.1-1.3 show GP sample paths generated for varying l, and fixed a.

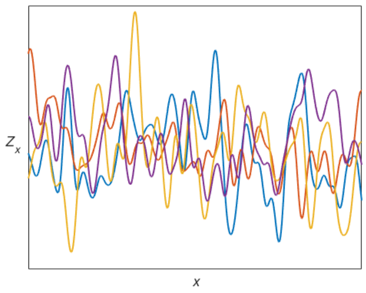


Figure 2.1: $l = 0.025$

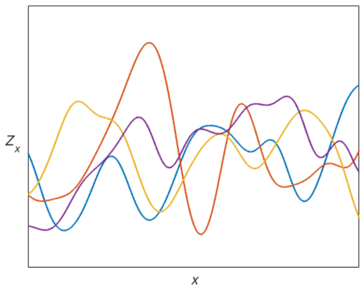


Figure 2.2: $l = 0.01$

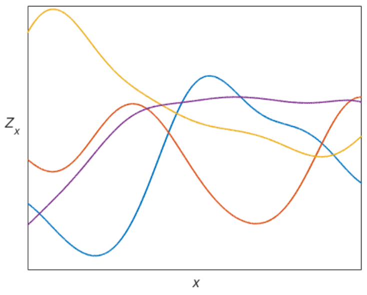


Figure 2.3: $l = 0.25$

Ornstein-Uhlenbeck Process

Similarly if we set $\nu = \frac{1}{2}$ for the Matérn covariance function we obtain the exponential covariance function or in the one-dimensional case the Ornstein-Uhlenbeck process[36],

$$k(h) = a \exp\left(\frac{-h}{2l^2}\right). \tag{2.2.3}$$

The corresponding GP is nowhere mean square differentiable however is everywhere mean square continuous, hence its apparent roughness. Figures 1.4-1.6 show Ornstein-Uhlenbeck process for varying l and fixed a.

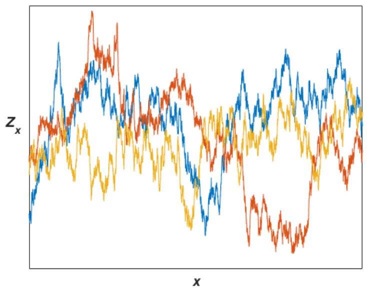


Figure 2.4: $l = 0.1$

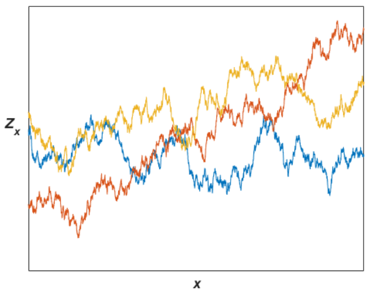


Figure 2.5: $l = 1$

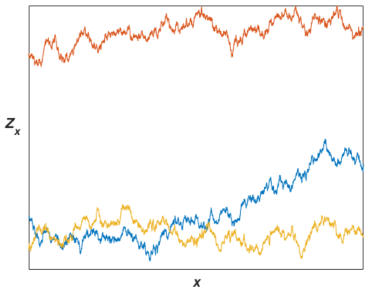


Figure 2.6: $l = 8$

2.3 Conditional Distribution and Regression

Suppose we can observe values from the GP $\{Z_x : x \in \mathbb{R}\}$ with mean function: $\mu(\cdot)$ and covariance function: $k(\cdot,\cdot)$ at inputs $D_o = \{x_0, \ldots, x_{n_1}\}$; in order to make predictions for the GP evaluated at a new set of inputs $D_u = \{\tilde{x}_0, \ldots, \tilde{x}_{n_2}\}$ we need to utilise joint and conditional properties of the multivariate Gaussian distribution.

Define $\boldsymbol{Z_o} := \{Z_x : x \in D_o\}$ as the observed data, $\boldsymbol{Z_u} := \{Z_x : x \in D_u\}$ as the unobserved data and $\boldsymbol{Z} := \{Z_x : x \in D_o \cup D_u\}$ as the complete data. It follows, $\boldsymbol{Z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{C})$[1], where $\boldsymbol{Z}$ and $\boldsymbol{C}$ can be partitioned as,

$$\boldsymbol{Z} = \begin{bmatrix} \boldsymbol{Z}_o \\ \boldsymbol{Z}_u \end{bmatrix}, \quad \boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu_o} \\ \boldsymbol{\mu_u} \end{bmatrix}, \quad \boldsymbol{C} = \begin{bmatrix} \boldsymbol{C_{oo}} & \boldsymbol{C_{ou}} \\ \boldsymbol{C_{uo}} & \boldsymbol{C_{uu}} \end{bmatrix}, \tag{2.3.1}$$

where $\boldsymbol{\mu_o} := \mathbb{E}(\boldsymbol{Z_o})$ and $\boldsymbol{\mu_u} := \mathbb{E}(\boldsymbol{Z_u})$. Moreover, the block matrix $\boldsymbol{C}$ contains the $(n_1+1) \times (n_1+1)$ matrix $\boldsymbol{C_{oo}} := \mathbb{V}\text{ar}(\boldsymbol{Z_o})$, the $(n_2+1) \times (n_2+1)$ matrix $\boldsymbol{C_{uu}} := \mathbb{V}\text{ar}(\boldsymbol{Z_u})$ and the $(n_1+1) \times (n_2+1)$ matrix $\boldsymbol{C_{ou}} = \boldsymbol{C_{uo}^\top} := \mathbb{C}\text{ov}(\boldsymbol{Z_o}, \boldsymbol{Z_u})$.

Theorem 2.3.1 (Conditional Distribution of Normal Random Vectors)**.** *Let $\boldsymbol{Z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{C})$ be an n-dimensional normal random vector with $|\boldsymbol{C}| > 0$. Decompose $\boldsymbol{Z}, \boldsymbol{\mu}$ and $\boldsymbol{C}$ as in (2.3.1), then*

$$\boldsymbol{Z_{u|o}} := (\boldsymbol{Z_u} | \boldsymbol{Z_o} = \boldsymbol{z_o}) \sim \mathcal{N}(\boldsymbol{\mu_{u|o}}, \boldsymbol{C_{u|o}}),$$

with, the conditional mean and covariance:

$$\boldsymbol{\mu_{u|o}} = \boldsymbol{\mu}_u + \boldsymbol{C_{uo}} \boldsymbol{C_{oo}^{-1}} (\boldsymbol{z_o} - \boldsymbol{\mu}_o), \tag{2.3.2}$$

$$\boldsymbol{C_{u|o}} = \boldsymbol{C_{uu}} - \boldsymbol{C_{uo}} \boldsymbol{C_{oo}^{-1}} \boldsymbol{C_{ou}}. \tag{2.3.3}$$

Thus, evaluating $\boldsymbol{\mu_{u|o_i}}$ and $\boldsymbol{C_{u|o_{ii}}}$ provides a predictive mean and variance for the process at the unseen inputs $\tilde{\boldsymbol{x}}_i$ for $i \in \{0, \ldots, n_2\}$.

Example 1.4.2 - Gaussian Process Regression

Suppose we would like to describe noisy observations expected in a regression model, with a GP; we could employ the following altered squared exponential covariance function

$$k(\boldsymbol{x_i}, \boldsymbol{x_j}) = a \exp\left(\frac{-(\boldsymbol{x_i} - \boldsymbol{x_j})^2}{2l^2} \right) + \sigma^2 \delta_{ij},$$

where $\mathbb{V}\text{ar}(Z_x) = \sigma^2$ is known.

[1]For notational simplicity, we suppress the explicit dependency of $\boldsymbol{C}$ on the hyperparameters $\boldsymbol{\theta}$.

In order for valid predictions to be made for unseen data, a suitable choice of hyperparameters $\boldsymbol{\theta} = \{a, l\}$ must be determined. Suppose $\boldsymbol{z_o}$ is known, then conventionally, the hyperparameters are determined by letting $\boldsymbol{\theta} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \ell(\boldsymbol{\theta}; \boldsymbol{z_o})$ i.e., maximising the likelihood as described in equation (2.1.3).

Figures 1.7-1.9 show predictive mean and confidence bands for differing l, highlighting the need to perform model selection by optimising the likelihood in order to avoid over and under fitting.

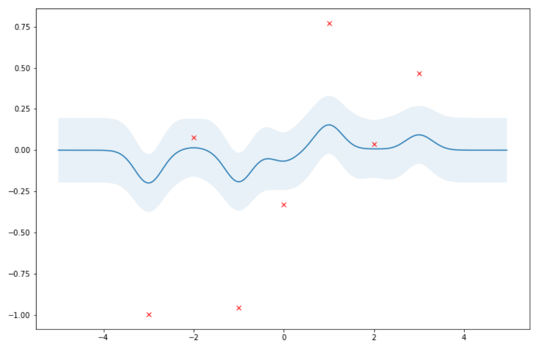


Figure 2.7: $l = 0.1$

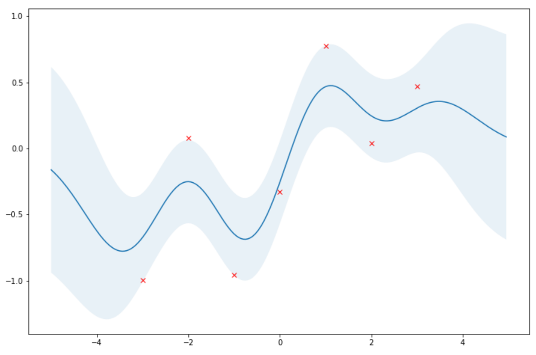


Figure 2.8: $l = 2$

Chapter 3

Efficient Likelihood Methods

The following chapter concerns itself with optimization procedures that utilise covariance matrix embedding in larger *circulant* matrices. Circulant matrices inherit computationally useful properties that allow for fast matrix multiplication and inversion in $O(nlogn)$ time, thus promise to deliver the ability to scale GP problems to orders of magnitude larger than what is restricted by the previously stated $O(n^2)$ and $O(n^3)$ techniques.

3.1 Circulant Embedding and Simulation

Circulant Matrices

Definition 3.1.1. *A $N \times N$ matrix $\boldsymbol{C}$ is circulant if and only if for a vector $\boldsymbol{c} = [c_o, c_1, \ldots, c_{N-1}]$, the matrix $\boldsymbol{C} = (c_{ij},\, i, j \in \{0, \ldots, N-1\})$ with elements $c_{ij} = c_{(i-j)\, mod\, N}$,*

$$\boldsymbol{C} = \begin{bmatrix} c_0 & c_{N-1} & \cdots & c_2 & c_1 \\ c_1 & c_0 & c_{N-1} & & c_2 \\ \vdots & c_1 & c_0 & \ddots & \vdots \\ c_{N-2} & & \ddots & \ddots & c_{N-1} \\ c_{N-1} & c_{N-2} & \cdots & c_1 & c_0 \end{bmatrix}.$$

The columns of the matrix $\boldsymbol{C}$ are cyclic permutations of the first column with an offset equal to the column index, or put simply, each column vector is rotated one element below relative to the preceding column vector.

It is well known that $\boldsymbol{C}$ can be diagonalized by the discrete Fourier matrix F [24].

$$F = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{2(N-1)} \\ 1 & \omega^3 & \omega^6 & \cdots & \omega^{3(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{N-1} & \omega^{2(N-1)} & \cdots & \omega^{(N-1)(N-1)} \end{bmatrix},$$

where $\boldsymbol{\omega} = \exp(\frac{-2\pi i}{N})$. Multiplying an arbitrary vector $\boldsymbol{v} \in \mathbb{C}^N$ by $\boldsymbol{F}$, computes its discrete Fourier transform $\hat{\boldsymbol{v}}$, that is,

$$\hat{\boldsymbol{v}}_t = (\boldsymbol{F}\boldsymbol{v})_t = \sum_{s=0}^{N-1} e^{\frac{-2\pi i}{N} st} v_s \quad t = 0, \dots, N-1.$$

Crucially the eigenvalues of the matrix $\boldsymbol{C}$ are given by,

$$\lambda = [\lambda_0, \dots, \lambda_{N-1}]^\top = \boldsymbol{F}\boldsymbol{c},$$

with the eigenvector corresponding to λ_i equal to the ith column of $\boldsymbol{F}$. Therefore, as previously alluded to, $\boldsymbol{C}$ has the eigendecomposition,

$$\boldsymbol{C} = \boldsymbol{F}\boldsymbol{\Lambda}\boldsymbol{F}^{-1}, \tag{3.1.1}$$

where $\boldsymbol{\Lambda} = \mathbf{diag}(\boldsymbol{\lambda})$. Operations involving the matrices F and F^{-1}, can be computed using the fast Fourier transform and inverse fast Fourier transform [9], both of which can be executed using $O(N\log N)$ operations. Moreover, $N^{-\frac{1}{2}}\boldsymbol{F}$ is unitary matrix, which implies $\boldsymbol{F}^{-1} = N^{-1}\overline{\boldsymbol{F}}$. Consequently, we summarize the following $O(N\log N)$ operations on C,

1. Eigenvalues: $\boldsymbol{\lambda} = \boldsymbol{F}\boldsymbol{c}$.
2. Matrix inverse: $\boldsymbol{C^{-1}} = \boldsymbol{F}\boldsymbol{\Lambda}^{-1}\boldsymbol{F}^{-1}$.
3. Matrix-vector multiplication: $\boldsymbol{C}\boldsymbol{x} = \boldsymbol{F}\boldsymbol{\Lambda}\boldsymbol{F}^{-1}\boldsymbol{x}$.
4. Quadratic form[1]: $\boldsymbol{x}^\top \boldsymbol{C}\boldsymbol{x} = N(\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{F^{-1}x})^H(\boldsymbol{\Lambda}^{\frac{1}{2}}\boldsymbol{F^{-1}x}) = N\left\|\boldsymbol{w}\right\|^2$
5. Inverse quadratic form[2]: $\boldsymbol{x}^\top \boldsymbol{C^{-1}}\boldsymbol{x} = N(\boldsymbol{\Lambda}^{-\frac{1}{2}}\boldsymbol{F^{-1}x})^H(\boldsymbol{\Lambda}^{-\frac{1}{2}}\boldsymbol{F^{-1}x}) = N\left\|\boldsymbol{z}\right\|^2$

Minimal Circulant Embedding

Let $\{Z_x : x \in \mathbb{R}\}$ be a zero-mean stationary GP. Suppose we narrow our focus to the regularly spaced locations $\boldsymbol{D_o} = \{x_o, \dots, x_n\}$ and the points $\boldsymbol{Z_o} := \{Z_x : x \in D_o\}$; the covariance matrix $\boldsymbol{C_{oo}} = \mathbb{V}\text{ar}(\boldsymbol{Z_o})$ will then take the form of a symmetric *toeplitz* matrix, with constant descending diagonals and can be specified by its first row $[a_o, \dots, a_n]$,

$$\boldsymbol{C_{oo}} = \begin{bmatrix} a_0 & a_1 & a_2 & \cdots & \cdots & a_n \\ a_1 & a_0 & a_1 & \ddots & & \vdots \\ a_2 & a_1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & a_1 & a_2 \\ \vdots & & \ddots & a_1 & a_0 & a_1 \\ a_n & \cdots & \cdots & a_2 & a_1 & a_0 \end{bmatrix}.$$

[1]C must be positive semi-definite to ensure square root factorization

It is now guaranteed that there exists $n-1$ unobserved locations, say $D_u = \{\tilde{x}_0, \ldots, \tilde{x}_{n-2}\}$, such that the covariance matrix $\mathbb{V}\text{ar}(\boldsymbol{Z})$ is circulant, where $\boldsymbol{Z} := \{Z_x : x \in D_o \cup D_u\}$.

Note the explicit locations D_u do not have to be known, rather the circulant matrix $\boldsymbol{C} = \mathbb{V}\text{ar}(\boldsymbol{Z})$ can be constructed by embedding the matrix C_{oo} in the $2n \times 2n$ symmetric toeplitz matrix whose first row $\boldsymbol{c}$ consists of the entries,

$$\boldsymbol{c} = [a_0, a_1, \ldots, a_n, a_{n-1}, \ldots, a_1]. \tag{3.1.2}$$

The $(n+1) \times (n+1)$ upper block along the main diagonal of $\boldsymbol{C}$ will a copy of $\boldsymbol{C_{oo}}$. The ciruclant matrix $\boldsymbol{C}$, can therefore be presented as a block matrix, partitioned identically to the covariance matrix in result (2.3.1), where $\boldsymbol{C_{uu}}$ has dimensions $(n-1) \times (n-1)$.

$$\boldsymbol{C} = \left(\begin{array}{ccccc|ccccc}
a_0 & a_1 & \cdots & a_{n-1} & a_n & a_{n-1} & a_n & \cdots & a_2 & a_1 \\
a_1 & a_0 & \cdots & a_{n-2} & a_{n-1} & a_{n-2} & a_{n-1} & \cdots & a_3 & a_2 \\
\vdots & \ddots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\
a_{n-1} & a_{n-2} & \cdots & a_0 & a_1 & a_2 & a_3 & \cdots & a_{n-1} & a_{n-2} \\
a_n & a_{n-1} & \cdots & a_1 & a_0 & a_1 & a_2 & \cdots & a_n & a_{n-1} \\
\hline
a_{n-1} & a_n & \cdots & a_2 & a_1 & a_0 & a_1 & \cdots & a_{n-3} & a_{n-2} \\
a_{n-2} & a_{n-1} & \cdots & a_3 & a_2 & a_1 & a_0 & \cdots & a_{n-4} & a_{n-3} \\
\vdots & \ddots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\
a_2 & a_3 & \cdots & a_{n-1} & a_{n-2} & a_{n-3} & a_{n-4} & \cdots & a_0 & a_1 \\
a_1 & a_2 & \cdots & a_n & a_{n-1} & a_{n-2} & a_{n-3} & \cdots & a_1 & a_0
\end{array}\right).$$

However, when using this embedding technique, the circulant matrix $\boldsymbol{C}$ is not guaranteed to be positive semidefinite, which is necessary for $\boldsymbol{C}$ to represent a valid covariance matrix. Numerous approaches prescribed in [14] aim to ensure positive semidefinite embeddings; the main being to embed $\boldsymbol{C_{oo}}$ in a lager $2M \times 2M$ matrix, with $M > n$, where $\boldsymbol{c}$ is constructed by

$$\begin{aligned}
\boldsymbol{c}_k &= a_k, & k &= 0, \ldots, n, \\
\boldsymbol{c}_{2M-k} &= a_k, & k &= 0, \ldots, n, .
\end{aligned}$$

The entries $\boldsymbol{c_{n+1}}, \ldots, \boldsymbol{c_{2M-n}}$ can be arbitrarily chosen or alternatively set by a secondary covariance function $\psi(h)$ satisfying certain conditions, (e.g., see [21] [33]). In general, it is guaranteed for a strictly positive definite toeplitz matrix to have a positive definite embedding [12], however this often requires a very large value of M, which subsequently result in prohibitive computations.

Therefore it's beneficial to seek conditions under which the minimal circulant embedding prescribed in (3.1.2), with $M = n$, is positive semi-definite. These conditions are explored in [14] and more recently [23]; where theoretical properties of the Fourier transform of the covariance function $k(h)$ are considered. However for the sake of the following analysis and experimentation, these conditions are assumed to hold true, meaning the minimal embedding technique can always be implemented.

Unconditional Simulation

Standard methods for simulating the Gaussian vector $\boldsymbol{Z_o} \sim \mathcal{N}(\boldsymbol{\mu_o}, \boldsymbol{C_{oo}})$ once again require $O(n^3)$ flops. Namely, we can apply the following algorithm[2] involving the Cholesky decomposition.

Algorithm 3.1: $O(n^3)$ Gaussian Simulation

```
    Input  : Mean vector $\boldsymbol{\mu_o}$; Covariance matrix $\boldsymbol{C_{oo}}$
    Output: $\boldsymbol{Z_o} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{C_{oo}})$
1   Compute $\boldsymbol{L}$, the Cholesky decomposition of $\boldsymbol{C_{oo}}$
2   Generate $W_1, \dots, W_n \overset{\text{i.i.d}}{\sim} \mathcal{N}(0,1)$. Let $\boldsymbol{W} = [W_1, \dots, W_n]^\top$.
3   $\boldsymbol{Z_o} \leftarrow \boldsymbol{\mu_o} + \boldsymbol{LW}$
4   return $\boldsymbol{Z_o}$
```

Circulant embedding was proposed by Wood and Chan[37], and, Dietrich and Newsam [14] as a fast method for simulating stationary GPs. Suppose we have embedded the $(n+1) \times (n+1)$ matrix $\boldsymbol{C_{oo}}$ in the $N \times N$ positive semidefinite circulant matrix $\boldsymbol{C}$, where $N = 2n$. Using the eigendecomposition (3.1.1) and the fact $\boldsymbol{F}^{-1} = N^{-1}\overline{\boldsymbol{F}}$ we can decompose $\boldsymbol{C}$ as

$$\boldsymbol{C} = N^{-1}\boldsymbol{F}\boldsymbol{\Lambda}\overline{\boldsymbol{F}} = \boldsymbol{D}\boldsymbol{D}^H,$$

where $\boldsymbol{D} = N^{\frac{1}{2}}F\boldsymbol{\Lambda}^{-\frac{1}{2}}$. Consequently, we can obtain two simulations of $\boldsymbol{Z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{C})$ by collecting the real and imaginary parts of the random vector $\boldsymbol{V} = \boldsymbol{DW} + \boldsymbol{\mu}$, where $\boldsymbol{W} \sim \mathcal{CN}(\boldsymbol{0}, \boldsymbol{I_N})$. To obtain the realisations of $\boldsymbol{Z_o}$ we simply take the first $n+1$ components of $\Re(\boldsymbol{V})$ and $\Im(\boldsymbol{V})$. Hence, we obtain our first practical utility from the circulant embedding technique; an O(NlogN) process for simulating $\boldsymbol{Z_o}$.

[2]To verify this method, one can simply check $\mathbb{E}(\boldsymbol{Z_o})$ and $\mathbb{V}\text{ar}(\boldsymbol{Z_o})$.

Algorithm 3.2: O(NlogN) Stationary Gaussian Simulation

```
Input  : Mean vector $\boldsymbol{\mu}$; Circulant matrix $\boldsymbol{C}$
Output: $\boldsymbol{Z_o} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{C_{oo}})$; $\boldsymbol{Z_u} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{C_{uu}})$
1 $\boldsymbol{\lambda} \leftarrow \boldsymbol{Fc}$ via the FFT
2 $\boldsymbol{\eta} \leftarrow \frac{1}{\sqrt{N}}\left[\sqrt{\lambda_1}, \dots, \sqrt{\lambda_N}\right]^\top$
3 Generate $\boldsymbol{W} \sim \mathcal{CN}(\boldsymbol{0}, \boldsymbol{I_N})$
4 $\boldsymbol{\zeta} \leftarrow [W_1\eta_1, \dots, W_N\eta_N]^\top$
5 $\boldsymbol{V} \leftarrow \boldsymbol{F\zeta}$ via the FFT
6 Let $\boldsymbol{A}$ be the first $n+1$ components of $\boldsymbol{V}$
7 Let $\boldsymbol{B}$ be the last $n-1$ components of $\boldsymbol{V}$
8 return $\boldsymbol{Z_o} \leftarrow \Re(\boldsymbol{A})$; $\boldsymbol{Z_u} \leftarrow \Re(\boldsymbol{B})$
```

Conditional Simulation

Certain likelihood estimation approaches require conditional simulations from the random vector $\boldsymbol{Z_{u|o}}$ defined in theorem (2.3.1). Direct simulation from $\mathcal{N}(\boldsymbol{\mu_{u|o}}, \boldsymbol{C_{u|o}})$ is infeasible when N is large as $\boldsymbol{C_{u|o}}$ is not guaranteed to be toeplitz. This enforces an $O(N^3)$ cost when storing and evaluating the Cholesky decomposition of $\boldsymbol{C_{u|o}}$. To avoid such cumbersome computations, we aim to obtain the conditional simulation $\boldsymbol{Z_{u|o}}$ by defining the following linear transformation.

Lemma 3.1.2 (Construction of $\boldsymbol{Z_{u|o}}$)**.**

$$\boldsymbol{Z}_u + \boldsymbol{C_{uo}C_{oo}^{-1}}(\boldsymbol{z_o} - \boldsymbol{Z_o}) \overset{d}{=} (\boldsymbol{Z_u}|\boldsymbol{Z_o} = \boldsymbol{z_o}).$$

Proof. The following proof is straightforward, however useful to follow in order to familiarise oneself with block matrix algebra. It suffices to show that the first and second moment coincide with the conditional mean and variance expressions stated in theorem (2.3.1).

$$\mathbb{E}\left(\boldsymbol{Z}_u + \boldsymbol{C_{uo}C_{oo}^{-1}}(\boldsymbol{z_o} - Z_o)\right) = \boldsymbol{\mu}_u + \boldsymbol{C_{uo}C_{oo}^{-1}}(\boldsymbol{z_o} - \boldsymbol{\mu}_o) = \boldsymbol{\mu_{u|o}}. \tag{3.1.3}$$

Now, considering the variance,

$$\begin{aligned}\mathbb{V}\text{ar}\left(\boldsymbol{Z}_u + \boldsymbol{C_{uo}C_{oo}^{-1}}(\boldsymbol{z_o} - \boldsymbol{Z_o})\right) &= \mathbb{V}\text{ar}(\boldsymbol{Z}_u - \boldsymbol{C_{uo}C_{oo}^{-1}Z_o}) \\ &= \mathbb{V}\text{ar}\left(\begin{bmatrix}-\boldsymbol{C_{uo}C_{oo}^{-1}} & \boldsymbol{I_{n-1}}\end{bmatrix}\boldsymbol{Z}\right) \\ &= \begin{bmatrix}-\boldsymbol{C_{uo}C_{oo}^{-1}} & \boldsymbol{I_{n-1}}\end{bmatrix}\mathbb{V}\text{ar}(\boldsymbol{Z})\begin{bmatrix}\left(-\boldsymbol{C_{uo}C_{oo}^{-1}}\right)^\top \\ \boldsymbol{I_{n-1}}\end{bmatrix}.\end{aligned}$$

Substituting $\boldsymbol{C} = \mathbb{V}\mathrm{ar}(\boldsymbol{Z})$ in terms of its block constituents,

$$\begin{aligned}\text{RHS} &= \begin{bmatrix}-\boldsymbol{C_{uo}C_{oo}^{-1}} & \boldsymbol{I_{n-1}}\end{bmatrix}\begin{bmatrix}\boldsymbol{C_{oo}} & \boldsymbol{C_{ou}} \\ \boldsymbol{C_{uo}} & \boldsymbol{C_{uu}}\end{bmatrix}\begin{bmatrix}\left(-\boldsymbol{C_{uo}C_{oo}^{-1}}\right)^\top \\ \boldsymbol{I_{n-1}}\end{bmatrix} \\ &= \begin{bmatrix}-\boldsymbol{C_{uo}C_{oo}^{-1}} & \boldsymbol{I_{n-1}}\end{bmatrix}\begin{bmatrix}\boldsymbol{0} \\ -\boldsymbol{C_{uo}C_{oo}^{-1}C_{ou}} + \boldsymbol{C_{uu}}\end{bmatrix} \\ &= \boldsymbol{C_{uu}} - \boldsymbol{C_{uo}C_{oo}^{-1}C_{ou}} = \boldsymbol{C_{u|o}}.\end{aligned}$$

□

In order to generate conditional samples, we simulate $\boldsymbol{Z_o}$ and $\boldsymbol{Z_u}$ using Algorithm (3.2), and then proceed to evaluate $\boldsymbol{Z}_u + \boldsymbol{C_{uo}C_{oo}^{-1}}(\boldsymbol{z_o} - \boldsymbol{Z_o})$. On first inspection, we are again restrained by the $O(n^3)$ inversion of $\boldsymbol{C}_{oo}^{-1}$ and the $O(n^2)$ multiplication of the matrix $\boldsymbol{C_{\mu|o}}$.

Firstly, to circumvent the $O(n^2)$ matrix multiplication, we exploit the form of $\boldsymbol{C}$ by noticing for a vector $\boldsymbol{v} \in \mathbb{R}^{n+1}$,

$$\boldsymbol{C}\begin{bmatrix}\boldsymbol{v} \\ \boldsymbol{0}\end{bmatrix} = \begin{bmatrix}\boldsymbol{C_{oo}} & \boldsymbol{C_{ou}} \\ \boldsymbol{C_{uo}} & \boldsymbol{C_{uu}}\end{bmatrix}\begin{bmatrix}\boldsymbol{v} \\ \boldsymbol{0}\end{bmatrix} = \begin{bmatrix}\boldsymbol{C_{oo}v} \\ \boldsymbol{C_{uo}v}\end{bmatrix} \tag{3.1.4}$$

Hence, we obtain $\boldsymbol{C_{uo}v}$ by padding the vector $\boldsymbol{v}$ with $n-1$ zeroes to obtain $\boldsymbol{v}^*$. Then, we extract the last $n-1$ entries of $\boldsymbol{Cv}^*$, which can be computed with $O(N\log N)$ flops.

Moreover, to solve the linear system

$$\boldsymbol{C_{oo}v} = \boldsymbol{\tau}, \quad \text{where } \boldsymbol{\tau} = \boldsymbol{z_o} - \boldsymbol{Z_o} \tag{3.1.5}$$

we apply the *conjugate gradient method* (CG); an iterative algorithm for numerically solving linear systems. Further details of the CG method and its variants are discussed below, but assuming we have an efficient implementation at our disposal, one can follow Algorithm (3.3) to simulate $\boldsymbol{Z_{u|o}}$.

Algorithm 3.3: Conditional Gaussian Simulation

```
    Input  : $\boldsymbol{z_o} \in \mathbb{R}^{n+1}$; Circulant matrix $\boldsymbol{C}$
    Output: $\boldsymbol{Z_{u|o}}$
1   Generate $\boldsymbol{Z_o}$ and $\boldsymbol{Z_u}$ via algorithm 2.2
2   Solve $\boldsymbol{C_{oo}v} = \boldsymbol{z_o} - \boldsymbol{Z_o}$ for $\boldsymbol{v}$, via CG
3   $\boldsymbol{v}^* \leftarrow [\boldsymbol{v}, 0, \ldots, 0]^\top$, where $\dim(\boldsymbol{v}^*) = N$
4   $\boldsymbol{\lambda} \leftarrow \boldsymbol{Fc}$ via the FFT; $\boldsymbol{\eta} \leftarrow \boldsymbol{F^{-1}v^*}$ via the IFFT
5   $\boldsymbol{\eta}^* \leftarrow [\lambda_1\eta_1, \ldots, \lambda_N\eta_N]^\top$
6   $\boldsymbol{\zeta} \leftarrow \boldsymbol{F\eta}^*$ via the FFT
7   Let $\boldsymbol{\zeta}^*$ be the last $n-1$ components of $\boldsymbol{\zeta}$
8   return $\boldsymbol{Z_{u|o}} \leftarrow \boldsymbol{Z_u} + \boldsymbol{\zeta}^*$
```

Conjugate Gradient Method

Linear systems, of the form $\boldsymbol{Ax} = \boldsymbol{b}$, where $\boldsymbol{A}$ is a real and positive-definite matrix, can be iteratively solved using the conjugate gradient method [25]. The CG method holds its advantages by being a theoretically direct method that produces the exact solution $\boldsymbol{x}^*$ after a finite number of iterations. In fact the number of iterations is guaranteed to not greater that the dimension of $\boldsymbol{x}$.

However, in relation to our objective, the implementation of the CG can be viewed as a sub-component of a larger procedure. Thus, when n is large, the $O(n)$ iterations required is an unacceptable amount of work. In light of this, we treat the algorithm with termination upon a maximum number of iterations T and convergence threshold ϵ. The CG proceeds as follows.

Algorithm 3.4: The Conjugate Gradient Algorithm (CG)

```
Input: Positive-definite matrix $\boldsymbol{A}$; vector $\boldsymbol{b}$; convergence threshold $\epsilon$;
       initial vector $\boldsymbol{x}_0$; maximum no. iterations $T$
1  $\boldsymbol{r}_0 \leftarrow \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}_0$
2  $\boldsymbol{p}_0 \leftarrow \boldsymbol{r}_0$
3  for $k \leftarrow 0$ to $T$ do
4      $\alpha_k \leftarrow \frac{\boldsymbol{r}_k^\top \boldsymbol{r}_k}{\boldsymbol{p}_k^\top \boldsymbol{A}\boldsymbol{p}_k}$
5      $\boldsymbol{x}_{k+1} \leftarrow \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k$
6      $\boldsymbol{r}_{k+1} \leftarrow \boldsymbol{r}_k - \alpha_k \boldsymbol{A}_k \boldsymbol{p}_k$
7      if $\|\boldsymbol{r}_{k+1}\| \leq \epsilon$ then
8          return $\boldsymbol{x}_{\boldsymbol{k}+\boldsymbol{1}}$
9      end
10     $\beta_k \leftarrow \frac{\boldsymbol{r}_{k+1}^\top \boldsymbol{r}_{k+1}}{\boldsymbol{r}_k^\top \boldsymbol{r}_k}$
11     $\boldsymbol{p}_{k+1} \leftarrow \boldsymbol{r}_{k+1} + \beta_k \boldsymbol{p}_k$
12 end
13 return $\boldsymbol{x}_{\boldsymbol{k}+\boldsymbol{1}}$
```

Notice that when $\boldsymbol{A} = \boldsymbol{C_{oo}}$ one can implement procedures similar to (3.1.4) to compute fast $O(N\log N)$ matrix multiplications. Moreover, in a similar vain we can compute quadratic products of the kind $\boldsymbol{v}^\top \boldsymbol{C_{oo}}\boldsymbol{v}$, by noticing that

$$\begin{bmatrix}\boldsymbol{v} & 0\end{bmatrix}^\top \boldsymbol{C} \begin{bmatrix}\boldsymbol{v} \\ 0\end{bmatrix} = \begin{bmatrix}\boldsymbol{v} & 0\end{bmatrix} \begin{bmatrix}\boldsymbol{C_{oo}} & \boldsymbol{C_{ou}} \\ \boldsymbol{C_{uo}} & \boldsymbol{C_{uu}}\end{bmatrix} \begin{bmatrix}\boldsymbol{v} \\ 0\end{bmatrix} = \boldsymbol{v}^\top \boldsymbol{C_{oo}} \boldsymbol{v} \qquad (3.1.6)$$

Hence, we obtain an $O(N\log N)$ method for computing steps 4 and 6 in Algorithm (3.4).

The convergence of the CG method can be shown to be dependent on the condition number $k(\boldsymbol{A}) = \lambda_{max}/\lambda_{min}$ i.e. the ratio of largest to smallest eigenvalues of $\boldsymbol{A}$.

Lemma 3.1.3 (CG error bound [22]). *Suppose $\boldsymbol{x}^*$ solves the equation $\boldsymbol{Ax} = \boldsymbol{b}$. If Algorithm (3.4) produces iterates $\{\boldsymbol{x}_t\}$ and $k = k(\boldsymbol{A})$ then*

$$\|\boldsymbol{x}^* - \boldsymbol{x_k}\|_{\boldsymbol{A}} \leq 2\,\|\boldsymbol{x}^* - \boldsymbol{x_o}\|_{\boldsymbol{A}} \left(\frac{\sqrt{k}-1}{\sqrt{k}+1}\right)^t,$$

where,

$$\|\boldsymbol{v}\|_{\boldsymbol{A}} = \sqrt{\boldsymbol{v}^\top \boldsymbol{Av}}.$$

Improvements in $\{\boldsymbol{x}_k\}$ are typically linear and are curtailed by large $k(\boldsymbol{A})$. To fasten the speed of convergence one can introduce a preconditioning matrix, $\boldsymbol{P}$, that augments the original system,

$$\boldsymbol{Ax} - \boldsymbol{b} = 0 \quad \rightarrow \quad \boldsymbol{P}^{-1}\boldsymbol{Ax} - \boldsymbol{P}^{-1}\boldsymbol{b} = 0.$$

The choice of $\boldsymbol{P}$ is made such that, $k(\boldsymbol{P}^{-1}\boldsymbol{A})$ is smaller than $k(\boldsymbol{A})$.

There has been a considerable amount research regarding suitable choices of preconditioning matrices for covariance matrices, (see, e.g., [7], [11], [8]). Ideally, $\boldsymbol{P}$ should be chosen such that $\boldsymbol{P}^{-1}\boldsymbol{A} \approx \boldsymbol{I}$. However we are also required to solve linear systems involving $\boldsymbol{P}$, hence $\boldsymbol{P}^{-1}$ should also be easily inverted. In our implementation we have chosen the $(n+1)\times(n+1)$ upper block of the complete precision matrix, i.e., $\boldsymbol{P}^{-1} = (\boldsymbol{C}^{-1})_{oo}$ as the most suitable candidate for preconditioner, recommended in [35]. Implementing $(\boldsymbol{C}^{-1})_{oo}$ is advantageous by being a good approximation of $\boldsymbol{C}_{oo}^{-1}$ and a matrix where $O(N\log N)$ matrix operations can be performed.

When preconditioning we obtain the *preconditioned conjugate gradient algorithm* (PCG), which necessitates the following alterations to Algorithm (3.4),

- $\boldsymbol{p}_0 := \boldsymbol{P}^{-1}\boldsymbol{r}_0$
- $\boldsymbol{p}_{k+1} := \boldsymbol{P}^{-1}\boldsymbol{r}_{k+1} + \beta_k \boldsymbol{p}_k$
- $\boldsymbol{\alpha}_k := \frac{\boldsymbol{r}_k^\top \boldsymbol{P}^{-1}\boldsymbol{r}_k}{\boldsymbol{p}_k^\top \boldsymbol{A}^{-1}\boldsymbol{p}_k}$
- $\boldsymbol{\beta}_k := \frac{\boldsymbol{r}_k^\top \boldsymbol{P}^{-1}\boldsymbol{r}_{k+1}}{\boldsymbol{p}_k^\top \boldsymbol{M}^{-1}\boldsymbol{p}_{k+1}}$

Notice, when $\boldsymbol{P}^{-1} = (\boldsymbol{C}^{-1})_{oo}$ the new values for $\boldsymbol{p}_0$ and $\boldsymbol{p}_{k+1}$ can be computed with similar calculations to that of stated in (3.1.4), however, now with the inverse complete circulant matrix $\boldsymbol{C}^{-1}$. Likewise for calculations of α_k and β_k we implement operations of the kind (3.1.6). Hence, we have obtained a PCG algorithm where each iteration can be computed in $O(N\log N)$ operations.

3.2 Monte-Carlo Expectation-Maximization Algorithm

Recall, that our objective is to evaluate the MLE, $\underset{\boldsymbol{\Theta}}{\operatorname{argmax}} \ln p(\boldsymbol{Z}_o|\boldsymbol{\Theta})$ for the observed data $\boldsymbol{Z}_o$. Where the loglikelihood, ignoring constants, is

$$\ln p(\boldsymbol{z}_o|\boldsymbol{\Theta}) = -\frac{1}{2}\ln|\boldsymbol{C}_{oo}| - \frac{1}{2}(\boldsymbol{z}_o - \boldsymbol{\mu}_o)^\top \boldsymbol{C}_{oo}^{-1}(\boldsymbol{z}_o - \boldsymbol{\mu}_o).$$

As previously emphasised, this likelihood is difficult to optimise, even with gradient or hessian based numerical optimization techniques. One can simplify the problem by introducing the unobserved variables $\boldsymbol{Z}_{\boldsymbol{u}}$, that were implicitly created through the construction of the circulant matrix $\boldsymbol{C}$. We then aim to maximise

$$\mathrm{p}(\boldsymbol{z}_o|\boldsymbol{\Theta}) = \int p(\boldsymbol{z}_o, \boldsymbol{z}_u|\boldsymbol{\Theta})\, d\boldsymbol{z}_{\boldsymbol{u}} = \int p(\boldsymbol{z}|\boldsymbol{\Theta})\, d\boldsymbol{z}_{\boldsymbol{u}}. \tag{3.2.1}$$

In its closed form this integral is difficult to optimise over $\boldsymbol{\Theta}$. To obtain a general method for solving such optimization problems, we can refer to the popular *EM Algorithm* [13].

EM Algorithm

Stein Et al. [35] proposed the EM method for GPs in the context of spatial lattice date, with the aim to exploit the circulant machinery developed through Section 3.1. To understand their implementation we shall first lay out the steps of the generic EM algorithm. For further details and a proof of the algorithm refer to [13].

To begin the EM algorithm we provide initial guesses for the parameters, which we denote as $\boldsymbol{\Theta}^0$. At iteration $t+1$ we estimate new values of the parameters $\boldsymbol{\Theta}^{t+1}$ by the following two steps.

Expectation step (E-step): Find the expectation of the complete-data loglikelihood under the distribution of $\boldsymbol{Z}_u$ given the previous parameter estimates $\boldsymbol{\Theta}^t$, and the observations $\boldsymbol{Z}_o$.

$$Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t) = \mathbb{E}_{\boldsymbol{Z}_{\boldsymbol{u}}|\boldsymbol{Z}_o,\boldsymbol{\Theta}^t}\left[\ln p(\boldsymbol{Z}|\boldsymbol{\Theta})\right]$$

Maximization step (M-step): Then, maximize this expected complete-data loglikelihood with respect to the parameters $\boldsymbol{\Theta}$ to obtain the new estimates

$$\boldsymbol{\Theta}^{(t+1)} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$$

It can be shown can that the likelihood $p(\boldsymbol{z}_o|\boldsymbol{\Theta}^t)$ does not decrease with every iteration of the algorithm. Therefore, we generally continue iterating between these two steps until a stopping criteria is met. A potential criterion is to stop once,

$$\left|\frac{p(\boldsymbol{Z}_o|\boldsymbol{\Theta}^{t+1}) - p(\boldsymbol{z}_o|\boldsymbol{\Theta}^t)}{p(\boldsymbol{z}_o|\boldsymbol{\Theta}^t)}\right| \leq \epsilon$$

for some small tolerance $\epsilon \geq 0$.

Under the model specified in (2.3.1), the distribution of the complete data is $\boldsymbol{Z}|\boldsymbol{\Theta} \sim \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{\Theta}), \boldsymbol{C}(\boldsymbol{\Theta}))$. Therefore, ignoring constants and suppressing explicit dependency on $\boldsymbol{\Theta}$, the E-step can be expressed as

$$Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t) = -\frac{1}{2}\ln|\boldsymbol{C}| - \frac{1}{2}\mathbb{E}\left[(\boldsymbol{Z}-\boldsymbol{\mu})^\top \boldsymbol{C}^{-1}(\boldsymbol{Z}-\boldsymbol{\mu})|\boldsymbol{Z}_o, \boldsymbol{\Theta}^t\right]. \tag{3.2.2}$$

To simplify this expectation, we first notice that the random vector $\boldsymbol{Z}|\boldsymbol{Z}_o, \boldsymbol{\Theta}^t$ can be thought of as a multivariate Gaussian distribution with mean and covariance matrix given respectively by

$$\widetilde{\boldsymbol{\mu}} = \begin{bmatrix} \boldsymbol{z_o} \\ \boldsymbol{\mu}_{u|o} \end{bmatrix}, \qquad \widetilde{\boldsymbol{C}} = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{C}_{u|o} \end{bmatrix}. \tag{3.2.3}$$

To utilise these known facts to evaluate the exact q-step we employ the following result.

Lemma 3.2.1 (Expectation of quadratic product)**.** *Suppose* $\boldsymbol{X}$ *is a vector of* n *random variables, with* $\mathbb{E}[\boldsymbol{X}] = \boldsymbol{\mu}$ *and* $\mathbb{V}ar\,[\boldsymbol{X}] = \boldsymbol{\Sigma}$ *. Then, for a* $n \times n$ *matrix* $\boldsymbol{A}$*,*

$$\mathbb{E}\left[\boldsymbol{X}^\top \boldsymbol{A}\boldsymbol{X}\right] = \mathrm{tr}\left[\boldsymbol{A}\boldsymbol{\Sigma}\right] + \boldsymbol{\mu}^\top \boldsymbol{A}\boldsymbol{\mu}$$

Using the mean and covariance matrix defined in Theorem (2.3.1), and, the previously stated Lemma, we can obtain the exact E-step,

$$Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t) = -\frac{1}{2}\ln|\boldsymbol{C}| - \frac{1}{2}\left\{\mathrm{tr}\left(\boldsymbol{C}^{-1}\widetilde{\boldsymbol{C}}\right) + (\widetilde{\boldsymbol{\mu}}-\boldsymbol{\mu})^\top \boldsymbol{C}^{-1}(\widetilde{\boldsymbol{\mu}}-\boldsymbol{\mu})\right\}. \tag{3.2.4}$$

To avoid confusion, we reiterate that when maximising the above expression, $\widetilde{\boldsymbol{\mu}}$ and $\widetilde{\boldsymbol{C}}$ are treated as fixed terms dependent on $\boldsymbol{\Theta}^t$ while $\boldsymbol{C}$ and $\boldsymbol{\mu}$ are free terms parameterised by $\boldsymbol{\Theta}$.

Computing the exact q-step (3.2.4) fast is hindered by the trace term involving $\widetilde{\boldsymbol{C}}$, which is not circulant. Rather than attempting to evaluate (3.2.4), Stein Et al. propose a Monte Carlo approach to approximate the expectation in (3.2.2),

$$\hat{Q}(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t) = -\frac{1}{2}\ln|\boldsymbol{C}| - \frac{1}{2}\left\{\frac{1}{M}\sum_{i=1}^{M}(\boldsymbol{Z}^{(i)}-\boldsymbol{\mu})^\top \boldsymbol{C}^{-1}(\boldsymbol{Z}^{(i)}-\boldsymbol{\mu})\right\}. \tag{3.2.5}$$

In this case, $\boldsymbol{Z}^{(1)}, \ldots, \boldsymbol{Z}^{(M)}$ are conditional simulations from $\boldsymbol{Z}|\boldsymbol{Z}_o, \boldsymbol{\Theta}^t$. To construct $\boldsymbol{Z}^{(i)}$, we firstly simulate $\boldsymbol{Z}^{(i)}_{\mu|o}$ using Algorithm (3.3) and then set $\boldsymbol{Z}^{(i)} = [\boldsymbol{Z}_o, \boldsymbol{Z}^{(i)}_{\mu|o}]^\top$. Assuming that the GP is characterised by a constant mean function, in other words, $\boldsymbol{\mu} = \mu\boldsymbol{1}$, the evaluation of $\hat{Q}(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ can be computed in $O(MN\mathrm{log}N)$ operations and, proceeds as follows.

Algorithm 3.5: Evaluating $\hat{Q}(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$

```
Input  : Monte Carlo size $M$; Parameter $\boldsymbol{\Theta} = \{\mu, \boldsymbol{\theta}\}$; Conditional
         samples $\boldsymbol{Z}^{(1)}_{u|o}, \dots, \boldsymbol{Z}^{(M)}_{u|o} \sim p(\boldsymbol{Z}_u|\boldsymbol{Z}_o, \boldsymbol{\Theta}^t)$; Observed data $\boldsymbol{z}_o$
Output: $\hat{Q}(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$
1  Generate $\boldsymbol{c}$, the first row of $\boldsymbol{C}(\boldsymbol{\theta})$
2  $\boldsymbol{\lambda} \leftarrow \boldsymbol{Fc}$ via the FFT
3  $\xi \leftarrow 0$
4  for $i \leftarrow 1$ to $M$ do
5      $\boldsymbol{Z}^{(i)} \leftarrow [\boldsymbol{z}_o, \boldsymbol{Z}^{(i)}_{\mu|o}]^\top$
6      $\boldsymbol{v} \leftarrow \boldsymbol{F}^{-1}[\boldsymbol{Z}^{(i)} - \mu\boldsymbol{1}]^\top$ via the IFFT
7      $\boldsymbol{v}^* \leftarrow [v_1/\sqrt{\lambda_1}, \dots, v_N/\sqrt{\lambda_N}]^\top$
8      $\xi \leftarrow \xi + N\,\|\boldsymbol{v}^*\|$
9  end
10 return $\hat{Q} \leftarrow -\frac{1}{2}\sum_{i=1}^{N} ln\lambda_i - \frac{1}{2M}\xi$
```

Given that with each iteration of the Monte Carlo EM we obtain a stochastic estimation of the E-step, we hesitate to implement a stopping criteria, and instead terminate upon a specified number of maximum iterations T. To implement the Monte Carlo EM, we proceed as follows.

Algorithm 3.6: Monte-Carlo EM

```
Input  : $M$; Initial parameter $\boldsymbol{\Theta}^0 = \{\mu^0, \boldsymbol{\theta}^0\}$; Observed data $\boldsymbol{z}_o$
Output: Approximation to $\underset{\boldsymbol{\Theta}}{\text{argmax}} \ln p(\boldsymbol{z}_o|\boldsymbol{\Theta})$
1  for $t \leftarrow 0$ to $T$ do
2      Generate $\boldsymbol{c}$, the first row of $\boldsymbol{C}(\boldsymbol{\theta}^t)$
3      Generate $\boldsymbol{Z}^{(1)}_{u|o}, \dots, \boldsymbol{Z}^{(M)}_{u|o} \sim p(\boldsymbol{Z}_u|\boldsymbol{Z}_o, \boldsymbol{\Theta}^t)$, via Algorithm 3.3
4      $\boldsymbol{\Theta}^{t+1} \leftarrow \arg\underset{\boldsymbol{\theta}}{\max} Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ via Algorithm 2.5/Quasi-Newton method
5  end
6  return $\boldsymbol{\Theta}^{T+1}$
```

Foreknowledge of a suitable choice for T may be difficult to ascertain, thus, it is recommended to initially inspect iterates of the EM visually to determine T.

In summary, the Monte-Carlo EM is an efficient algorithm in the sense that no sub-calculation requires more than $O(N\log N)$ operations. However, as explained, the algorithm relies on two sub-iterative procedures. Namely, the PCG algorithm necessary to simulate conditional samples and the Quasi Newton numerical optimisation procedure used to evaluate the M-step. The optimality of the PCG is highly dependent on the precarious choice of preconditoner matrix $\boldsymbol{P}$. In fact, the preconditioner $(\boldsymbol{C}^{-1})_{\boldsymbol{oo}}$, may not be a one-size-fits-all solution for every choice of covariance function. In such a scenario, the PCG will lose its $O(N\log N)$ run-time, and, the EM-Algorithm will be forced to converge slowly. Moreover, since the PCG is implemented M times for each iteration, for a large value of M, the algorithm will lose its computational efficiency.

In terms of evaluating the M-step, it is unknown how well the Quasi-Newton numerical optimisation will handle problems with numerous parameters, i.e, when $\boldsymbol{\Theta}$ is high-dimensional. Especially considering that for every iteration of the EM we must call upon the Quasi-Newton process to obtain the next parameter update. These factors, and the general optimality of the Monte Carlo EM will be tested numerically and discussed further in Chapter 3.

3.3 Variance Reduction for the Monte-Carlo EM

The effectiveness of the Monte-Carlo EM is partly reliant on the precision of the estimate (3.2.5). Since $\hat{Q}(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ is an unbiased estimator, one can attempt to improve the Monte Carlo performance by reducing the variance. If we let,

$$H(\boldsymbol{Z}) = (\boldsymbol{Z}-\boldsymbol{\mu})^\top \boldsymbol{C}^{-1}(\boldsymbol{Z}-\boldsymbol{\mu})$$

we can notice that the Monte Carlo error variance for $\hat{Q}(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ will take the form

$$\mathbb{V}\mathrm{ar}\left[\hat{Q}(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)\right] = \frac{\mathbb{V}\mathrm{ar}\left[\sum_{i=1}^M H(\boldsymbol{Z}^{(i)})\right]}{4M^2} = \frac{\mathbb{V}\mathrm{ar}\left[H(\boldsymbol{Z})\right]}{4M}, \tag{3.3.1}$$

where $\boldsymbol{Z}^{(1)},\ldots,\boldsymbol{Z}^{(M)} \overset{i.i.d}{\sim} \boldsymbol{Z}|\boldsymbol{Z}_o, \boldsymbol{\Theta}^t$.

The simplest, most direct method to reduce the variance is to increase the Monte Carlo size M; however, this will incur a computational cost and may counteract the efficiencies gained by circulant embedding. Instead one can attempt to find a new estimator $G(\boldsymbol{Z})$, such that $\mathbb{E}[G(\boldsymbol{Z})] = \mathbb{E}[H(\boldsymbol{Z})]$ and $\mathbb{V}\mathrm{ar}[G(\boldsymbol{Z})] \leq \mathbb{V}\mathrm{ar}[H(\boldsymbol{Z})]$. For $G(\boldsymbol{Z})$ to be an effective replacement it is necessary that the cost to compute $G(\boldsymbol{Z})$ must be equivalent to computing $H(\boldsymbol{Z})$.

In the following section, we propose two such novel estimators that successfully reduce the variance of our estimate of $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$. We will then proceed to inspect certain asymptotic properties, and discuss their limitations in the context of the EM Algorithm.

The key insight is made by noticing that one can attempt to implement a Monte Carlo estimate for the exact of the expectation in (3.2.4), rather than the initial form of the conditional expectation (3.2.2). To this end, we propose the following unbiased estimator for the problematic trace term,

$$\operatorname{tr}\left(\boldsymbol{C}^{-1}\widetilde{\boldsymbol{C}}\right) \approx \frac{1}{M}\sum_{i=1}^{M}(\boldsymbol{Z}^{(i)} - \widetilde{\boldsymbol{\mu}})^\top \boldsymbol{C}^{-1}(\boldsymbol{Z}^{(i)} - \widetilde{\boldsymbol{\mu}}). \tag{3.3.2}$$

Verifying that this an unbiased estimator of the trace term is straightforward, by firstly noting the following unbiased estimator of $\widetilde{\boldsymbol{C}}$,

$$\mathbb{E}\left[(\boldsymbol{Z}^{(i)} - \widetilde{\boldsymbol{\mu}})(\boldsymbol{Z}^{(i)} - \widetilde{\boldsymbol{\mu}})^\top\right] = \widetilde{\boldsymbol{C}}. \tag{3.3.3}$$

Knowing that the trace is a linear mapping and invariant to cyclic permutations, we can then manipulate the trace expression to obtain

$$\begin{aligned}
\operatorname{tr}\left(\boldsymbol{C}^{-1}\widetilde{\boldsymbol{C}}\right) &\approx \operatorname{tr}\left(\boldsymbol{C}^{-1}\frac{1}{M}\sum_{i=1}^{M}(\boldsymbol{Z}^{(i)} - \widetilde{\boldsymbol{\mu}})(\boldsymbol{Z}^{(i)} - \widetilde{\boldsymbol{\mu}})^\top\right) \\
&= \frac{1}{M}\sum_{i=1}^{M}\operatorname{tr}\left((\boldsymbol{Z}^{(i)} - \widetilde{\boldsymbol{\mu}})^\top \boldsymbol{C}^{-1}(\boldsymbol{Z}^{(i)} - \widetilde{\boldsymbol{\mu}})\right), \\
&= \frac{1}{M}\sum_{i=1}^{M}(\boldsymbol{Z}^{(i)} - \widetilde{\boldsymbol{\mu}})^\top \boldsymbol{C}^{-1}(\boldsymbol{Z}^{(i)} - \widetilde{\boldsymbol{\mu}}).
\end{aligned}$$

To complete the estimation of $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$, we substitute the new trace estimator into (3.2.4) and obtain the new Monte Carlo estimate,

$$\hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t) := -\frac{1}{2}\ln|\boldsymbol{C}| - \frac{1}{2}\left\{\frac{1}{M}\sum_{i=1}^{M}(\boldsymbol{Z}^{(i)} - \widetilde{\boldsymbol{\mu}})^\top \boldsymbol{C}^{-1}(\boldsymbol{Z}^{(i)} - \widetilde{\boldsymbol{\mu}}) + (\widetilde{\boldsymbol{\mu}} - \boldsymbol{\mu})^\top \boldsymbol{C}^{-1}(\widetilde{\boldsymbol{\mu}} - \boldsymbol{\mu})\right\}.$$

From a computational standpoint, $\hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ can be evaluated just as easily as $\hat{Q}(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$, since we are only dealing with quadratic products terms involving the circulant matrix $\boldsymbol{C}^{-1}$. The complete conditional mean $\widetilde{\boldsymbol{\mu}}$ can be constructed by appending the vector $\boldsymbol{z_0}$ with the conditional mean $\boldsymbol{\mu}_{u|o}$. As stated in Theorem (2.3.1), the conditional mean requires the evaluation of $\boldsymbol{\mu}_u + \boldsymbol{C_{uo}C_{oo}^{-1}}(\boldsymbol{z_o} - \boldsymbol{\mu}_o)$, which can be completed using the PCG method and the techniques required to simulate conditional simulations. In fact, with a few simple changes to Algorithm (3.3), the conditional mean $\boldsymbol{\mu_{u|o}}$ can be returned as secondary output.

Intuitively we can interpret $\hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ as the estimator adjusted for when knowledge about the distribution $p(\boldsymbol{Z}|\boldsymbol{Z}_o, \boldsymbol{\Theta}^t)$ is known, namely, the complete conditional mean $\widetilde{\boldsymbol{\mu}} = \mathbb{E}[\boldsymbol{Z}|\boldsymbol{Z}_o, \boldsymbol{\Theta}^t]$. To prove this additional information of the exact E-step results in a reduced variance, we must show $\mathbb{V}\mathrm{ar}[G(\boldsymbol{Z})] \leq \mathbb{V}\mathrm{ar}[H(\boldsymbol{Z})]$, where G is given by,

$$G(\boldsymbol{Z}) = (\boldsymbol{Z} - \widetilde{\boldsymbol{\mu}})^\top \boldsymbol{C}^{-1}(\boldsymbol{Z} - \widetilde{\boldsymbol{\mu}}).$$

Lemma 3.3.1 (Variance of Quadratic Product). *Suppose $\boldsymbol{Y} \sim \mathcal{N}(\boldsymbol{\nu}, \boldsymbol{\Sigma})$, and $\boldsymbol{A}$ is a symmetric matrix, then*

$$\mathbb{V}ar\left[\boldsymbol{Y}^\top \boldsymbol{A} \boldsymbol{Y}\right] = 2\operatorname{tr}\left[(\boldsymbol{A}\boldsymbol{\Sigma})^2\right] + 4\boldsymbol{\nu}^\top \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}\boldsymbol{\nu}$$

Proof. We refer the reader to Rencher [31], page 105. □

Lemma 3.3.2 (Variance Reduction via Centering). *Suppose $\boldsymbol{Y} \sim \mathcal{N}(\boldsymbol{\nu}, \boldsymbol{\Sigma})$, then*

$$\mathbb{V}ar\left[(\boldsymbol{Y} - \boldsymbol{\nu})^\top \boldsymbol{A}(\boldsymbol{Y} - \boldsymbol{\nu})\right] \leq \mathbb{V}ar\left[\boldsymbol{Y}^\top \boldsymbol{A}\boldsymbol{Y}\right],$$

and

$$\mathbb{V}ar\left[(\boldsymbol{Y} - \boldsymbol{\nu})^\top \boldsymbol{A}(\boldsymbol{Y} - \boldsymbol{\nu})\right] - \mathbb{V}ar\left[\boldsymbol{Y}^\top \boldsymbol{A}\boldsymbol{Y}\right] = 4\boldsymbol{\nu}^\top \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}\boldsymbol{\nu}.$$

Proof. Firstly, we notice $(\boldsymbol{Y} - \boldsymbol{\nu}) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma})$. Then, according to Lemma (3.3.1), the corresponding quadratic product will have a variance

$$\begin{aligned}\mathbb{V}\mathrm{ar}\left[(\boldsymbol{Y} - \boldsymbol{\nu})^\top \boldsymbol{A}(\boldsymbol{Y} - \boldsymbol{\nu})\right] &= 2\operatorname{tr}\left[(\boldsymbol{A}\boldsymbol{\Sigma})^2\right] \\ &= \mathbb{V}\mathrm{ar}\left[\boldsymbol{Y}^\top \boldsymbol{A}\boldsymbol{Y}\right] - 4\boldsymbol{\nu}^\top \boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}\boldsymbol{\nu} \\ &\leq \mathbb{V}\mathrm{ar}\left[\boldsymbol{Y}^\top \boldsymbol{A}\boldsymbol{Y}\right],\end{aligned}$$

where the last inequality holds true since the covariance matrix $\boldsymbol{\Sigma}$ is a non-negative definite matrix. □

Applying the above lemmas to the estimators $G(\boldsymbol{Z})$ and $H(\boldsymbol{Z})$, we can show that $\mathbb{V}\mathrm{ar}[G(\boldsymbol{Z})] \leq \mathbb{V}\mathrm{ar}[H(\boldsymbol{Z})]$.

$$\begin{aligned}\mathbb{V}\mathrm{ar}[H(\boldsymbol{Z})] - \mathbb{V}\mathrm{ar}[G(\boldsymbol{Z})] &= 4(\widetilde{\boldsymbol{\mu}} - \boldsymbol{\mu})^\top \boldsymbol{C}^{-1}\widetilde{\boldsymbol{C}}\boldsymbol{C}^{-1}(\widetilde{\boldsymbol{\mu}} - \boldsymbol{\mu}), &(3.3.4)\\ &\geq 0. &(3.3.5)\end{aligned}$$

When the complete mean $\boldsymbol{\mu}$ is assumed to be $\boldsymbol{0}$, the difference is simplified to

$$\mathbb{V}\mathrm{ar}[\boldsymbol{Z}^\top \boldsymbol{C}^{-1}\boldsymbol{Z}] - \mathbb{V}\mathrm{ar}[(\boldsymbol{Z} - \widetilde{\boldsymbol{\mu}})^\top \boldsymbol{C}^{-1}(\boldsymbol{Z} - \widetilde{\boldsymbol{\mu}})] = 4\widetilde{\boldsymbol{\mu}}^\top \boldsymbol{C}^{-1}\widetilde{\boldsymbol{C}}\boldsymbol{C}^{-1}\widetilde{\boldsymbol{\mu}}. \qquad (3.3.6)$$

Hence, our new estimator $\hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ is guaranteed to exhibit a variance equal to or less than the original crude Monte Carlo estimate . At a superficial level the improvement of $\hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ is amplified when a large deviation exists between $\widetilde{\boldsymbol{\mu}}$ and the zero vector.

However, when implemented in the EM-Algorithm, it was noted that when updates in parameters were small, i.e., $\boldsymbol{\Theta}^{t+1} \approx \boldsymbol{\Theta}^t$, the variance reduction was not significant. To understand this, we must be able to evaluate Equation (3.3.6) exactly. To this end, we employ two general results for block matrices.

Lemma 3.3.3 (Aitken block-diagonalization formula [2]). *Suppose the covariance matrix $\boldsymbol{C}$ has the block matrix form,*

$$\boldsymbol{C} = \begin{bmatrix} \boldsymbol{C_{oo}} & \boldsymbol{C_{ou}} \\ \boldsymbol{C_{uo}} & \boldsymbol{C_{uu}} \end{bmatrix},$$

then,

$$\boldsymbol{C} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{C_{uo}C_{oo}^{-1}} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{C_{oo}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{C_{u|o}} \end{bmatrix} \begin{bmatrix} \boldsymbol{I} & \boldsymbol{C_{oo}^{-1}C_{ou}} \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix}. \tag{3.3.7}$$

The resulting block-diagonalization, can be easily verified by making the substitution $\boldsymbol{C_{u|o}} = \boldsymbol{C_{uu}} - \boldsymbol{C_{uo}C_{oo}^{-1}C_{ou}}$.

Lemma 3.3.4 (Banachiewicz inversion formula [20]). *Suppose the covariance matrix $\boldsymbol{C}$ has the block matrix form stated in Lemma (3.3.3), and $\boldsymbol{C_{u|o}}$ is non-singular, then*

$$\boldsymbol{C}^{-1} = \begin{bmatrix} \boldsymbol{C_{oo}^{-1}(C_{oo} + C_{ou}C_{u|o}^{-1}C_{uo})C_{oo}^{-1}} & \boldsymbol{-C_{oo}^{-1}C_{ou}C_{u|o}^{-1}} \\ \boldsymbol{C_{u|o}^{-1}C_{uo}C_{oo}^{-1}} & \boldsymbol{C_{u|o}^{-1}} \end{bmatrix}.$$

The resulting formula for the inverse preceded the Aitken block-diagonalization formula, however, can be immediately yielded by inverting both sides of the factorisation (3.3.7).

Proposition 3.3.5. *When trying to estimate $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$, if $\boldsymbol{\Theta} = \boldsymbol{\Theta}^t$ i.e., $\widetilde{\boldsymbol{C}}$ and $\boldsymbol{C}$ are built from the same parameters, then*

$$\mathbb{V}ar[\hat{Q}(\boldsymbol{\Theta}^t|\boldsymbol{\Theta}^t)] = \mathbb{V}ar[\hat{Q}_2(\boldsymbol{\Theta}^t|\boldsymbol{\Theta}^t)]$$

Proof. The prove this claim, we must show the difference term described in Equation (3.3.6) is equal to zero, when $\widetilde{\boldsymbol{C}}$ and $\boldsymbol{C}$ are constructed from the same parameters. Substituting $\boldsymbol{C}^{-1}$ from Lemma (3.3.3) and assuming $\boldsymbol{\mu} = 0$ without loss of generality, we obtain

$$\begin{aligned} 4\widetilde{\boldsymbol{\mu}}^\top \boldsymbol{C}^{-1}\widetilde{\boldsymbol{C}}\boldsymbol{C}^{-1}\widetilde{\boldsymbol{\mu}} &= 4\widetilde{\boldsymbol{\mu}}^\top \begin{bmatrix} * & \boldsymbol{-C_{oo}^{-1}C_{ou}C_{u|o}^{-1}} \\ * & \boldsymbol{C_{u|o}^{-1}} \end{bmatrix} \begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{C}_{u|o} \end{bmatrix} \boldsymbol{C}^{-1}\widetilde{\boldsymbol{\mu}} \\ &= 4\begin{bmatrix} \boldsymbol{z}_o^\top & \boldsymbol{\mu}_{u|o}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{0} & \boldsymbol{-C_{oo}^{-1}C_{ou}} \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} \boldsymbol{C}^{-1}\widetilde{\boldsymbol{\mu}} \end{aligned}$$

Notice that when $\boldsymbol{\mu} = \boldsymbol{0}$, the conditional mean is given by $\boldsymbol{\mu}_{u|o}^\top = \boldsymbol{z_o}^\top \boldsymbol{C_{oo}^{-1}C_{ou}}$. Henceforth,

$$\begin{aligned} 4\begin{bmatrix} \boldsymbol{z}_o^\top & \boldsymbol{\mu}_{u|o}^\top \end{bmatrix} \begin{bmatrix} \boldsymbol{0} & \boldsymbol{-C_{oo}^{-1}C_{ou}} \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix} \boldsymbol{C}^{-1}\widetilde{\boldsymbol{\mu}} &= 4\begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \boldsymbol{C}^{-1}\widetilde{\boldsymbol{\mu}}. \\ &= 0. \end{aligned}$$

□

Therefore, the performance improvement of our new estimator $\hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ diminishes when $\boldsymbol{\Theta} \to \boldsymbol{\Theta^t}$. That is to say, when the EM Algorithm approaches the true parameter and smaller updates between successive $\boldsymbol{\Theta}^t$ are made, there is no benefit obtained in our new estimate of $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$. To avoid this impediment, we aim to seek further knowledge of $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ when $\boldsymbol{\Theta} = \boldsymbol{\Theta}^t$. In fact, when $\widetilde{\boldsymbol{C}}$ and $\boldsymbol{C}$ are constructed from the same parameters the trace term in the exact q-step (3.2.4) can be simplified to

$$\begin{aligned}
\operatorname{tr}\left(\boldsymbol{C}^{-1}\widetilde{\boldsymbol{C}}\right) &= \operatorname{tr}\left(\begin{bmatrix} * & -\boldsymbol{C_{oo}^{-1}C_{ou}C_{u|o}^{-1}} \\ * & \boldsymbol{C_{u|o}^{-1}} \end{bmatrix}\begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{C}_{u|o} \end{bmatrix}\right) \\
&= \operatorname{tr}\left(\begin{bmatrix} \boldsymbol{0} & -\boldsymbol{C_{oo}^{-1}C_{ou}} \\ \boldsymbol{0} & \boldsymbol{I_{n-1}} \end{bmatrix}\right) \\
&= n-1
\end{aligned}$$

Therefore, peculiarly, at the point where there is no gain to be made by using $\hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ is in fact where the quantity we are trying to estimate is known exactly. To exploit this fact we propose the new estimator which takes the form a control variate estimator [6]. We define the new estimator as,

$$\hat{Q}_3(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t) := \hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t) + \frac{1}{2M}\left[\sum_{i=1}^{M} J(\boldsymbol{Z}^{(i)}) - \tau\right]. \tag{3.3.8}$$

The random variable $J(\boldsymbol{Z})$ is called the control variate if its expectation, $\tau = \mathbb{E}[J(\boldsymbol{Z})]$ is known or readily available. The advantage of such an estimator is that it remains unbiased i.e, $\mathbb{E}[\hat{Q}_3(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)] = \mathbb{E}[\hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)]$, and, under certain circumstances will reduce the variance of the estimator $\hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$. Let the function J be defined as

$$J(\boldsymbol{Z}) = (\boldsymbol{Z} - \widetilde{\boldsymbol{\mu}})^\top \widetilde{\boldsymbol{C}}^*(\boldsymbol{Z} - \widetilde{\boldsymbol{\mu}}), \quad \text{where } \widetilde{\boldsymbol{C}}^* := \begin{bmatrix} \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{C}_{u|o}^{-1} \end{bmatrix}. \tag{3.3.9}$$

It follows that the expectation τ can be evaluated as,

$$\begin{aligned}
\mathbb{E}[J(\boldsymbol{Z})] &= \mathbb{E}\left[(\boldsymbol{Z} - \widetilde{\boldsymbol{\mu}})^\top \widetilde{\boldsymbol{C}}^*(\boldsymbol{Z} - \widetilde{\boldsymbol{\mu}})\right] \\
&= \mathbb{E}\left[\operatorname{tr}\left((\boldsymbol{Z} - \widetilde{\boldsymbol{\mu}})^\top \widetilde{\boldsymbol{C}}^*(\boldsymbol{Z} - \widetilde{\boldsymbol{\mu}})\right)\right] \\
&= \operatorname{tr}\left(\mathbb{E}\left[(\boldsymbol{Z} - \widetilde{\boldsymbol{\mu}})(\boldsymbol{Z} - \widetilde{\boldsymbol{\mu}})^\top\right]\widetilde{\boldsymbol{C}}^*\right)
\end{aligned}$$

Substituting in the known expectation (3.3.3), and subsequently performing the block matrix multiplication, we obtain

$$
\begin{aligned}
\text{RHS} &= \operatorname{tr}\left(\widetilde{\boldsymbol{C}}\widetilde{\boldsymbol{C}}^{*}\right), \\
&= \operatorname{tr}\left(\begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & \boldsymbol{C}_{u|o}\end{bmatrix}\begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & \boldsymbol{C}_{u|o}^{-1}\end{bmatrix}\right), \\
&= \operatorname{tr}\left(\begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & \boldsymbol{I_{n-1}}\end{bmatrix}\right), \\
&= n-1.
\end{aligned}
$$

Hence, the value for τ is known to be $n-1$. However, for $\hat{Q}_3(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ to be an implementable estimator, $J(\boldsymbol{Z})$ must be easily computed. Initially, this may seem intractable as the quadratic product (3.3.9) does not involve a circulant matrix. However, upon further inspection,

$$
\begin{aligned}
(\boldsymbol{Z}-\widetilde{\boldsymbol{\mu}})^{\top}\widetilde{\boldsymbol{C}}^{*}(\boldsymbol{Z}-\widetilde{\boldsymbol{\mu}}) &= \begin{bmatrix}(\boldsymbol{Z_o}-\boldsymbol{Z_o})^{\top} & (\boldsymbol{Z_u}-\boldsymbol{\mu}_{u|o})^{\top}\end{bmatrix}\widetilde{\boldsymbol{C}}^{*}\begin{bmatrix}\boldsymbol{Z_o}-\boldsymbol{Z_o}\\ \boldsymbol{Z_u}-\boldsymbol{\mu}_{u|o}\end{bmatrix}, \\
&= \begin{bmatrix}\mathbf{0} & (\boldsymbol{Z_u}-\boldsymbol{\mu}_{u|o})^{\top}\end{bmatrix}\begin{bmatrix}\mathbf{0} & \mathbf{0}\\ \mathbf{0} & \boldsymbol{C}_{u|o}^{-1}\end{bmatrix}\begin{bmatrix}\mathbf{0}\\ \boldsymbol{Z_u}-\boldsymbol{\mu}_{u|o}\end{bmatrix}, \\
&= (\boldsymbol{Z_u}-\boldsymbol{\mu}_{u|o})^{\top}\boldsymbol{C}_{u|o}^{-1}(\boldsymbol{Z}_u-\boldsymbol{\mu}_{u|o}).
\end{aligned}
$$

Since the first $n+1$ entries of the vector $\boldsymbol{Z}-\widetilde{\boldsymbol{\mu}}$ are zero, the quadratic term $J(\boldsymbol{Z})$ only 'hits' the bottom right block $\boldsymbol{C}_{u|o}^{-1}$. This implies that the matrix $\widetilde{\boldsymbol{C}}^{*}$ can be replaced by any matrix that contains the lower right block $\boldsymbol{C}_{u|o}^{-1}$. In fact, according to the Banachiewicz inversion formula (3.3.4), the circulant matrix $\boldsymbol{C}^{-1}$ is such a matrix. That is to say,

$$
\begin{aligned}
(\boldsymbol{Z}-\widetilde{\boldsymbol{\mu}})^{\top}\boldsymbol{C}^{-1}(\boldsymbol{Z}-\widetilde{\boldsymbol{\mu}}) &= \begin{bmatrix}\mathbf{0} & (\boldsymbol{Z_u}-\boldsymbol{\mu}_{u|o})^{\top}\end{bmatrix}\begin{bmatrix}* & *\\ * & \boldsymbol{C}_{u|o}^{-1}\end{bmatrix}\begin{bmatrix}\mathbf{0}\\ \boldsymbol{Z_u}-\boldsymbol{\mu}_{u|o}\end{bmatrix}, \\
&= (\boldsymbol{Z_u}-\boldsymbol{\mu}_{u|o})^{\top}\boldsymbol{C}_{u|o}^{-1}(\boldsymbol{Z}_u-\boldsymbol{\mu}_{u|o}).
\end{aligned}
$$

Replacing $\widetilde{\boldsymbol{C}}^{*}$ with the circulant matrix $\boldsymbol{C}^{-1}$ implies that we can employ previously stated FFT procedures to obtain $J(\boldsymbol{Z})$ in $O(N\log N)$ operations. Hence, the new estimate $\hat{Q}_3(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ can be computed efficiently by evaluating,

$$
\hat{Q}_3(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t) = \hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t) + \frac{1}{2M}\left\{\sum_{i=1}^{M}(\boldsymbol{Z^{(i)}}-\widetilde{\boldsymbol{\mu}})^{\top}\boldsymbol{C}_t^{-1}(\boldsymbol{Z^{(i)}}-\widetilde{\boldsymbol{\mu}}) - (n-1)\right\}.
$$

We have employed the subscript t in $\boldsymbol{C}_t^{-1}$ to stress that the respective complete precision matrix is fixed and dependent on $\boldsymbol{\Theta}^t$. Notice, that if $\boldsymbol{\Theta}=\boldsymbol{\Theta}^t$, i.e., when $\boldsymbol{C}^{-1}=\boldsymbol{C}_t^{-1}$ the new estimate agrees with the exact value of $Q(\boldsymbol{\Theta}^t|\boldsymbol{\Theta}^t)$, where the trace term is equal to $n-1$. Further analysis of the estimator can be achieved through inspection of the variance.

Proposition 3.3.6. *When trying to estimate $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$, if $\boldsymbol{\Theta} = \boldsymbol{\Theta}^t$ i.e., $\widetilde{\boldsymbol{C}}$ and $\boldsymbol{C}$ are built from the same parameters, then*

$$\mathbb{V}ar\left[\hat{Q}_3(\boldsymbol{\Theta}^t|\boldsymbol{\Theta}^t)\right] = 0.$$

Proof. The prove this claim, we notice that the variance of $\mathbb{V}\mathrm{ar}[\hat{Q}_3(\boldsymbol{\Theta}^t|\boldsymbol{\Theta}^t)]$ can be expressed as,

$$\begin{aligned}\mathbb{V}\mathrm{ar}\left[\hat{Q}_3(\boldsymbol{\Theta}^t|\boldsymbol{\Theta}^t)\right] &= \frac{1}{4M}\mathbb{V}\mathrm{ar}\left[(\boldsymbol{Z}-\widetilde{\boldsymbol{\mu}})^\top \boldsymbol{C}^{-1}(\boldsymbol{Z}-\widetilde{\boldsymbol{\mu}}) - (\boldsymbol{Z}-\widetilde{\boldsymbol{\mu}})^\top \boldsymbol{C}_t^{-1}(\boldsymbol{Z}-\widetilde{\boldsymbol{\mu}})\right] \\ &= \frac{1}{4M}\mathbb{V}\mathrm{ar}\left[(\boldsymbol{Z}-\widetilde{\boldsymbol{\mu}})^\top \left[\boldsymbol{C}^{-1} - \boldsymbol{C}_t^{-1}\right](\boldsymbol{Z}-\widetilde{\boldsymbol{\mu}})\right].\end{aligned}$$

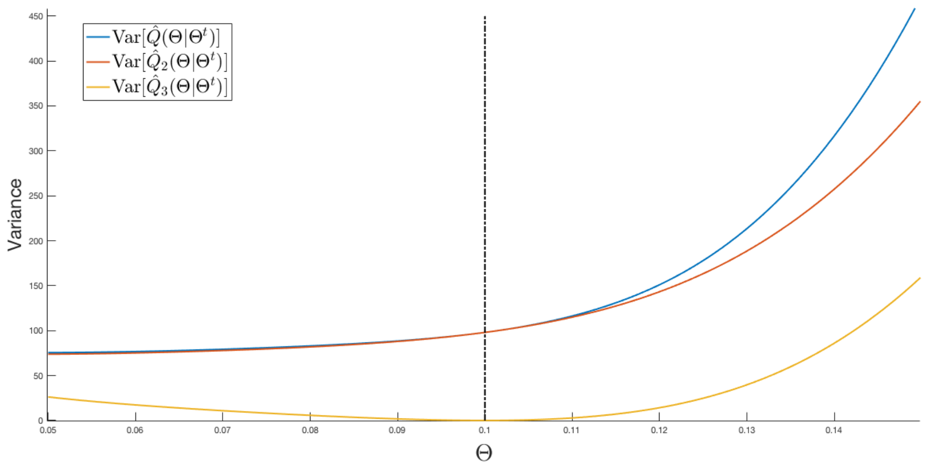
By implementing Lemma (3.3.1), we can obtain the exact variance of the quadratic form,

$$\mathbb{V}\mathrm{ar}\left[\hat{Q}_3(\boldsymbol{\Theta}^t|\boldsymbol{\Theta}^t)\right] = \frac{1}{2M}\,\mathrm{tr}\left[\left(\boldsymbol{C}^{-1} - \boldsymbol{C}_t^{-1}\right)\widetilde{\boldsymbol{C}}\left(\boldsymbol{C}^{-1} - \boldsymbol{C}_t^{-1}\right)\widetilde{\boldsymbol{C}}\right].$$

Hence, when $\boldsymbol{C}^{-1} = \boldsymbol{C}_t^{-1}$, the variance reduces to zero, and $Q(\boldsymbol{\Theta}^t|\boldsymbol{\Theta}^t)$ can be computed exactly. □

Consequently, we have solved the predicament that arose from $\hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$, and, have obtained an estimator that increasing outperforms $\hat{Q}(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ as $\boldsymbol{\Theta} \to \boldsymbol{\Theta}^t$.

Figure 3.1: $\Theta^t = 0.1$



For a comparison of the three Monte Carlo estimates refer to Figure 2.1, which displays the theoretical variance obtained at a single iteration when $\boldsymbol{\Theta}$ is restricted to a single kernel hyper parameter. Figure 2.1 visually conveys the previously proven propositions and the superior performance of our new estimator $\hat{Q}_3(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$.

The Gradient Problem

These considerable improvements made in estimating $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ originated with the intention to achieve a faster convergence towards the MLE estimates. By reducing the variance, one can theoretically decrease the Monte Carlo size M to obtain an equivalently precise estimate and, therefore, fasten each iteration of the EM algorithm. However, when implemented only a slight improvement was observed from using $\hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ over $\hat{Q}(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ and, no difference observed between $\hat{Q}_3(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ and $\hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$.

To understand this, it is necessary to inspect the operations of the M-step. Suppose we have implemented the estimator $\hat{Q}_3(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$; then, the M-step requires the evaluation of

$$\boldsymbol{\Theta}^{(t+1)} = \arg\max_{\boldsymbol{\theta}} \hat{Q}_3(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t).$$

Which in turn, is equivalent to numerically solving

$$\frac{\partial}{\partial \boldsymbol{\Theta}} \hat{Q}_3(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t) = \mathbf{0}.$$

However, as defined in Equation (3.3.8), the estimate $\hat{Q}_3(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ is constructed by adjusting the estimate $\hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ with terms that are not dependent on the parameter $\boldsymbol{\Theta}$. Therefore, when differentiating both estimators with respect to $\boldsymbol{\Theta}$, we obtain equivalent gradient functions i.e,

$$\frac{\partial}{\partial \boldsymbol{\Theta}} \hat{Q}_3(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t) = \frac{\partial}{\partial \boldsymbol{\Theta}} \hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t).$$

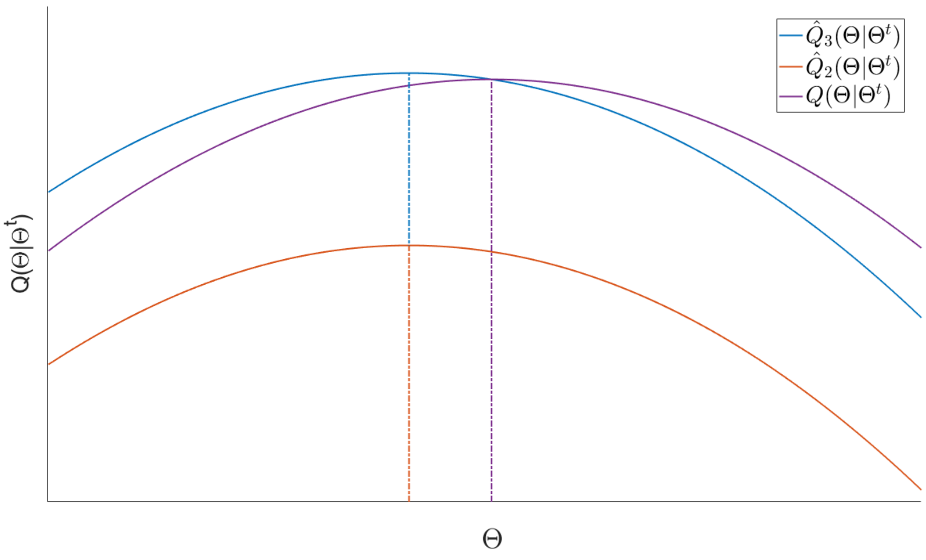
As a result, when implementing the M-step, we obtain no change in our updated estimates,

$$\boldsymbol{\Theta}^{(t+1)} = \arg\max_{\boldsymbol{\theta}} \hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t) = \arg\max_{\boldsymbol{\theta}} \hat{Q}_3(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t).$$

Figure 2.2 visually conveys this difficulty. The same value for $\boldsymbol{\Theta}$ is shown to maximise both $\hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ and $\hat{Q}_3(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$, despite $\hat{Q}_3(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ being the closer estimate.

Therefore, in the context of this Monte-Carlo EM approach, it is more important to obtain accurate estimates for the gradient rather than the function value of $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$. We can, however, prove that the estimate $Q_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ obtains a better approximation to the true gradient, compared to the crude Monte Carlo estimate. To this end, we inspect the partial derivatives of the functions $H(\boldsymbol{Z})$, and $G(\boldsymbol{Z})$ respect to the kernel hyperparamter θ_i. Employing the matrix derivative identities described in Chapter 1, we obtain the two derivatives

Figure 3.2: M=2



- $$\frac{\partial}{\partial \theta_i} H(\boldsymbol{Z}) = (\boldsymbol{Z}-\boldsymbol{\mu})^\top \boldsymbol{C}^{-1} \frac{\partial \boldsymbol{C}}{\partial \theta_i} \boldsymbol{C}^{-1} (\boldsymbol{Z}-\boldsymbol{\mu}), \tag{3.3.10}$$

- $$\frac{\partial}{\partial \theta_i} G(\boldsymbol{Z}) = (\boldsymbol{Z}-\widetilde{\boldsymbol{\mu}})^\top \boldsymbol{C}^{-1} \frac{\partial \boldsymbol{C}}{\partial \theta_i} \boldsymbol{C}^{-1} (\boldsymbol{Z}-\widetilde{\boldsymbol{\mu}}). \tag{3.3.11}$$

Notice, that the quadratic forms stay intact, allowing the variances to be compared by the formula stated in Lemma (3.3.1),

$$\mathbb{V}\text{ar}\left[\frac{\partial}{\partial \theta_i} H(\boldsymbol{Z})\right] - \mathbb{V}\text{ar}\left[\frac{\partial}{\partial \theta_i} G(\boldsymbol{Z})\right] = 4(\widetilde{\boldsymbol{\mu}}-\boldsymbol{\mu})^\top \boldsymbol{C'} \widetilde{\boldsymbol{C}} \boldsymbol{C'} (\widetilde{\boldsymbol{\mu}}-\boldsymbol{\mu}), \tag{3.3.12}$$

where $\boldsymbol{C'} = \boldsymbol{C}^{-1} \frac{\partial \boldsymbol{C}}{\partial \theta_i} \boldsymbol{C}^{-1}$. Since the covariance function is assumed to be stationary, the derivative with respect to kernel hyperparamter θ_i, will also be stationary. This implies that the matrix $\boldsymbol{C'}$, a product of symmetric matrices, will also be symmetric. As a result, the difference described in (3.3.12) is guaranteed to be non-negative. However, as the derivative matrix is dependent on the choice of covariance function, exact asymptotic properties of this term are difficult to obtain.

The variance of the derivative functions for the estimators in questions will follow a similar form to Equation (3.3.1). That is to say,

- $$\mathbb{V}\text{ar}\left[\frac{\partial}{\partial\theta_i}\hat{Q}(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)\right] = \frac{\mathbb{V}\text{ar}\left[\frac{\partial}{\partial\theta_i}H(\boldsymbol{Z})\right]}{4M}, \tag{3.3.13}$$

- $$\mathbb{V}\text{ar}\left[\frac{\partial}{\partial\theta_i}\hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)\right] = \frac{\mathbb{V}\text{ar}\left[\frac{\partial}{\partial\theta_i}G(\boldsymbol{Z})\right]}{4M}. \tag{3.3.14}$$

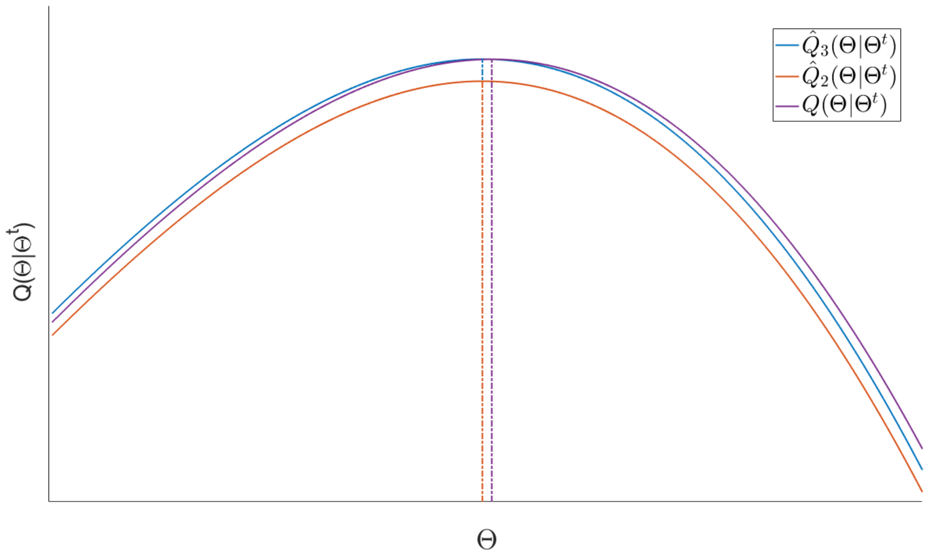
Therefore, since (3.3.12) is guaranteed to be non-negative, we conclude,

$$\mathbb{V}\text{ar}\left[\frac{\partial}{\partial\theta_i}\hat{Q}(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)\right] \leq \mathbb{V}\text{ar}\left[\frac{\partial}{\partial\theta_i}\hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)\right].$$

The new estimator $\hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$, is therefore, guaranteed to exhibit lower variance when estimating both the value and gradient of $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$.

Moreover, as the derivative of the estimators can be verified to be unbiased, increasing M will not only increase the accuracy of value of $Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$, but the gradient of this function. This is evident in Figure 2.3, where compared to Figure 2.2, both the curvature and value of $\hat{Q}_2(\boldsymbol{\Theta}|\boldsymbol{\Theta}^t)$ closely approximate the true curve.

Figure 3.3: M=50



Notwithstanding the difficulties arising from the gradient problem, investing time in methods of variance reduction has provided neat circulant formalisms that can be implemented in different optimisation settings.

3.4 Stochastic Score Estimation

Rather than exhausting further energy attempting to improve the involved Monte Carlo EM; we shift our focus to a more proximate method of likelihood optimisation. In fact, with the notion of estimating the trace in Section 2.3, together with the PCG method, which, essentially solves systems of the kind $\boldsymbol{C}_{\boldsymbol{oo}}^{\boldsymbol{-1}}\boldsymbol{z}$, we can attempt to estimate the score and subsequently employ gradient methods without the need to evaluate expensive log determinants in $p(\boldsymbol{z}|\boldsymbol{\theta})$. Recall, the MLE can be generally obtained by setting the score equal to zero. Denoting $\frac{\partial \boldsymbol{C}_{oo}}{\partial \theta_k} = \boldsymbol{C_k}$ and assuming $\boldsymbol{\mu} = \boldsymbol{0}$ without loss of generality, the score for the kth parameter is given by,

$$s(\boldsymbol{\theta})_k = -\frac{1}{2}\,\mathbf{tr}\left(\boldsymbol{C}_{oo}^{-1}\boldsymbol{C_k}\right) + \frac{1}{2}\,\boldsymbol{z_o}^\top \boldsymbol{C}_{oo}^{-1}\boldsymbol{C_k}\boldsymbol{C}_{oo}^{-1}\boldsymbol{z_o}.$$

Various papers ([11], [34], [17]), have followed similar intentions and have proposed methods to efficiently evaluate the stated score function, including [3] and [10], which implement circulant methods for fast matrix-vector multiplication. However, they do not employ variance reduction techniques of the kind specified in Section 2.3. Firstly, in order to evaluate the score, we solve the linear system

$$\boldsymbol{\eta} = \boldsymbol{C}_{\boldsymbol{oo}}^{\boldsymbol{-1}}\boldsymbol{z},$$

using the PCG method. Then, the quadratic term can be evaluated efficiently through the circulant embedding of the toeplitz matrix $\boldsymbol{C_k}$ and $O(N\log N)$ calculations of the kind stated in (3.1.4).

$$\boldsymbol{z_o}^\top \boldsymbol{C}^{-1}\boldsymbol{C_k}\boldsymbol{C}^{-1}\boldsymbol{z_o} = \boldsymbol{\eta}^\top \boldsymbol{C_k}\boldsymbol{\eta}.$$

Trace Estimators

To evaluate the trace term, we introduce the following unbiased stochastic estimator,

$$\mathrm{tr}\left(\boldsymbol{C}_{\boldsymbol{oo}}^{\boldsymbol{-1}}\boldsymbol{C_k}\right) \approx \frac{1}{M}\sum_{i=1}^{M} \boldsymbol{Y}^{\top(i)}\boldsymbol{C_k}\boldsymbol{Y}^{(i)}. \tag{3.4.1}$$

where $\boldsymbol{Y} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{C}_{\boldsymbol{oo}}^{\boldsymbol{-1}})$. To simulate $\boldsymbol{Y}$, we use Algorithm (3.2) to simulate $\boldsymbol{Z_o} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{C_{oo}})$. Then, we obtain $\boldsymbol{Y}$ by solving the equation $\boldsymbol{C_{oo}}\boldsymbol{Y} = \boldsymbol{Z}$ using the PCG method. To verify the construction of $\boldsymbol{Y}$, we notice,

$$\begin{aligned}\mathbb{V}\mathrm{ar}\left[\boldsymbol{Y}\right] &= \mathbb{V}\mathrm{ar}\left[\boldsymbol{C}_{\boldsymbol{oo}}^{\boldsymbol{-1}}\boldsymbol{Z_o}\right] \\ &= \boldsymbol{C}_{\boldsymbol{oo}}^{\boldsymbol{-1}}\boldsymbol{C_{oo}}\left(\boldsymbol{C}_{\boldsymbol{oo}}^{\boldsymbol{-1}}\right)^\top \\ &= \boldsymbol{C}_{\boldsymbol{oo}}^{\boldsymbol{-1}}.\end{aligned}$$

The resulting estimate for the score simplifies to,

$$\hat{s}(\boldsymbol{\theta})_k = -\frac{1}{2}\frac{1}{M}\sum_{i=1}^{M} \boldsymbol{Y}^{(i)\top}\boldsymbol{C_k}\boldsymbol{Y}^{(i)} + \frac{1}{2}\boldsymbol{\eta}^{\top}\boldsymbol{C_k}\boldsymbol{\eta}. \tag{3.4.2}$$

Our estimation of the trace term is not unique, as others, notably [11], [34], [18] propose alternative estimators in the same setting. Essentially, a stochastic estimate of the trace can be constructed by letting

$$\operatorname{tr}\left(\boldsymbol{C_{oo}^{-1}}\boldsymbol{C_k}\right) \approx \frac{1}{M}\sum_{i=1}^{M} \boldsymbol{Y}^{(i)\top}\boldsymbol{C_{oo}^{-1}}\boldsymbol{C_k}\boldsymbol{Y}^{(i)}, \tag{3.4.3}$$

with the random vector $\boldsymbol{Y}$ following any distribution that satisfies $\mathbb{E}\left[\boldsymbol{Y}\boldsymbol{Y}^{\top}\right] = \boldsymbol{I}$. Two commonly implemented strategies of this kind are as follows,

- **Gaussian** [32]

  Randomly choose the elements of $\boldsymbol{Y}$ independently and identically distributed from zero mean unit variance Gaussian distribution. The corresponding variance V, when $M = 1$, will be,

  $$V = \operatorname{tr}\left(\left[\boldsymbol{C_{oo}^{-1}}\boldsymbol{C_k}\right]^2\right)$$

- **Hutchinson** [27]

  Randomly choose the elements of $\boldsymbol{Y}$ independently and identically distributed from from a Rademacher distribution. It follows, the vector $\boldsymbol{Y}$ will have components drawn from $\{-1, 1\}$ with probability $1/2$. The corresponding variance V, when $M = 1$, will be,

  $$V = 2\operatorname{tr}\left(\left[\boldsymbol{C_{oo}^{-1}}\boldsymbol{C_k}\right]^2\right) - 2\sum_{i=1}^{n+1}\left(\boldsymbol{C_{oo}^{-1}}\boldsymbol{C_k}\right)_{ii}^2$$

Compared to our proposed estimator (3.4.1), simulating $\boldsymbol{Y}$ is simpler and more efficient for the above cases. However, in order to evaluate the quadratic product (3.4.3), we would require PCG calculations of the kind $\boldsymbol{C_{oo}}\boldsymbol{\eta} = \boldsymbol{Y}$ and subsequent circulant matrix-vector multiplications of the kind $\boldsymbol{C_k}\boldsymbol{Y}$. In light of this, we deem our proposed estimator to be computationally equivalent to the conventionally used estimators.

Variance reduction

The benefit of our proposed estimator is that we can exploit knowledge of Lemma (3.3.1), and perform variance reduction in a similar vain to Section 2.3. Firstly, we evaluate the variance of the unaltered estimator (3.4.1). Let,

$$K(\boldsymbol{Y}) := \boldsymbol{Y}^{\top}\boldsymbol{C_k}\boldsymbol{Y}.$$

Then, as $\boldsymbol{Y} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{C_{oo}^{-1}})$, the exact variance by Lemma (3.3.6), is given by,

$$\mathbb{V}\text{ar}\left[K(\boldsymbol{Y})\right] = 2\,\text{tr}\left(\left[\boldsymbol{C_{oo}^{-1}C_k}\right]^2\right).$$

The variance of our proposed estimator is thus no better than the Gaussian estimator, and, less accurate compared to Hutchison's estimator. To improve our estimate we propose a control variable estimate similar to that of Equation (3.3.8).

Firstly, we notice if $\boldsymbol{Y} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{C_{oo}^{-1}})$, then,

$$\begin{aligned}\mathbb{E}\left[\boldsymbol{Y}^\top \boldsymbol{C}_{oo}\boldsymbol{Y}\right] &= \text{tr}\left(\boldsymbol{C_{oo}^{-1}C_{oo}}\right),\\ &= n+1.\end{aligned}$$

Using this known expectation, we propose the following control variable estimator $L(\boldsymbol{Y})$, which remains unbiased i.e., $\mathbb{E}L(\boldsymbol{Y}) = \mathbb{E}K(\boldsymbol{Y})$.

$$L(\boldsymbol{Y}) := \boldsymbol{Y}^\top \boldsymbol{C_k Y} - \gamma\left(\left(\boldsymbol{Y}^\top \boldsymbol{C}_{oo}\boldsymbol{Y}\right) - (n+1)\right)$$

To increase flexibility, we have included the scaling term γ, which can be determined accordingly in order to minimise the variance. It follows from Lemma (3.3.1),

$$\begin{aligned}\mathbb{V}\text{ar}\left[L(\boldsymbol{Y})\right] &= 2\,\text{tr}\left([(\boldsymbol{C_k} - \gamma\boldsymbol{C_{oo}})\boldsymbol{C_{oo}^{-1}}]^2\right)\\ &= 2\,\text{tr}\left([\boldsymbol{C_k C_{oo}^{-1}} - \gamma\boldsymbol{I}]^2\right)\end{aligned} \tag{3.4.4}$$

To calculate the best value for γ, we differentiate (3.4.4) with respect to γ, and obtain

$$\frac{\partial L(\boldsymbol{Y})}{\partial \gamma} = 2\,\text{tr}\left(-2\boldsymbol{C_k C_{oo}^{-1}} + 2\gamma \boldsymbol{I_{n+1}}\right) \tag{3.4.5}$$

Solving the derivative for zero, the best choice of γ simplifies to,

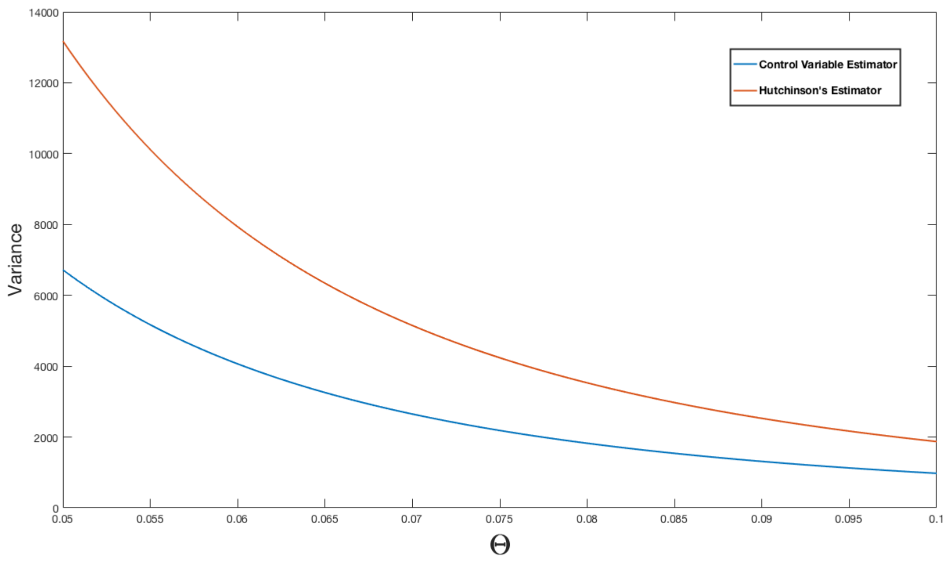
$$\gamma = \frac{\text{tr}\left(\boldsymbol{C_k C_{oo}^{-1}}\right)}{n+1} \tag{3.4.6}$$

Substituting the optimal value of γ into Equation (3.4.4) yields the minimum variance,

$$V = \operatorname{tr}\left(\left[\boldsymbol{C_k}\boldsymbol{C_{oo}^{-1}}\right]^2\right) - \frac{\operatorname{tr}\left(\boldsymbol{C_k}\boldsymbol{C_{oo}^{-1}}\right)^2}{n+1} \tag{3.4.7}$$

In Figure 2.4 we plot the theoretical variance of $L(\boldsymbol{Y})$ and Hutchinson's estimator as a function of the length scale l in the Ornstein-Uhlenbeck kernel. As illustrated, the control variable estimator theoretically exhibits superior performance when estimating the trace.

Figure 3.4: Variance Reduction



The exact evaluation of γ requires the unknown quantity, $\operatorname{tr}\left(\boldsymbol{C_k}\boldsymbol{C_{oo}^{-1}}\right)$. As this exact form is infeasible to evaluate, we substitute the crude Monte Carlo estimate (3.4.1), to obtain an approximation to γ,

$$\gamma \approx \frac{\sum_{i=1}^{M} K(\boldsymbol{Y}^{(i)})}{M(n+1)}$$

In practise, this method still out-performs Hutchison's estimator for the covariance functions described in this thesis. Since the evaluation of $L(\boldsymbol{Y})$ can be computed efficiently through the circulant embedding of $\boldsymbol{C_k}$ and $\boldsymbol{C_{oo}}$, we have successfully constructed an efficient control variable estimator of the score, which has the form,

$$\hat{s}(\boldsymbol{\theta})_k := -\frac{1}{2}\frac{1}{M}\sum_{i=1}^{M} L(\boldsymbol{Y}^{(i)}) + \frac{1}{2}\boldsymbol{\eta}^\top \boldsymbol{C_k}\boldsymbol{\eta}. \tag{3.4.8}$$

Gradient Descent

In order to solve the score equations,

$$\hat{s}(\boldsymbol{\theta})_k = \mathbf{0},$$

we employ the popular gradient descent method [5]. Initially, a starting guess $\boldsymbol{\theta}_0$ is made for the local minimum of $\ell(\boldsymbol{\theta}; \boldsymbol{z_o})$. Then the subsequent updates $\{\boldsymbol{\theta}_t\}$ are made by iterating the following steps until convergence,

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \nu_t s(\boldsymbol{\theta_t}), \ n \geq 0.$$

The resulting update can be shown to guarantee $\ell(\boldsymbol{\theta}_{t+1}; \boldsymbol{z_o}) \leq \ell(\boldsymbol{\theta}_t; \boldsymbol{z_o})$, hence the sequence $\{\boldsymbol{\theta}_t\}$ should converge to the desired local minimum. Note we define ν as the step size or learning rate, which is usually chosen trough a line search or, the Barzilai-Borwein formula [4].

3.5 MM Algorithm

Thus far, the discussed optimisation methods have all relied on stochastic Monte Carlo estimates to evaluate otherwise intractable quantities. These algorithms rely on sub-iterative conjugate procedures, that are not guaranteed to be fast or well-conditioned. Furthermore, their error is bounded by the Monte Carlo size M, and may be intractable for certain values of the parameter $\boldsymbol{\Theta}$. To alleviate these burdens, we present a novel approach that simplifies the problem to a single, computationally feasible optimization problem. This approach relies on a parametric lower bound for the likelihood (2.1.3), which is then globally maximised.

Recall, if we have observed data from the zero-mean Gaussian $\boldsymbol{Z}_o \sim \mathcal{N}(\mathbf{0}, \boldsymbol{C_{oo}})$, then, to obtain the MLE parameters $\boldsymbol{\theta}$, we maximise,

$$\ell(\boldsymbol{\theta}; \boldsymbol{z_o}) = -\frac{n}{2}\ln 2\pi - \frac{1}{2}\ln|\boldsymbol{C_{oo}}| - \frac{1}{2}\boldsymbol{z}_o^\top \boldsymbol{C_{oo}^{-1}} \boldsymbol{z_o}.$$

Ignoring constants and alternating signs, the MLE is equivalent to,

$$\hat{\boldsymbol{\theta}}_{\text{MLE}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left\{ \ln|\boldsymbol{C_{oo}}| + \boldsymbol{z_o}^\top \boldsymbol{C_{oo}^{-1}} \boldsymbol{z_o} \right\}. \tag{3.5.1}$$

The following approach is an example of the iterative MM method [29], and, is staged by the construction of two independent upper bounds for the determinant and quadratic form. Firstly, to obtain an upper bound for the log-determinant term we employ the Kullback-Leibler divergence (KL-divergence); a metric for measuring the difference between two probability densities [28].

Definition 3.5.1 (KL-Divergence). *Let p and q denote the probability densities of continuous random variables P and Q. Then the Kullback-Leibler divergence is defined to be the following integral,*

$$D_{KL}(P \parallel Q) = \int_{-\infty}^{\infty} p(x) \log \left(\frac{p(x)}{q(x)} \right) dx.$$

The following two results allow us to manipulate properties of the Normal KL-divergence and acquire an upper bound for the log-determinant.

Lemma 3.5.2 (Non-Negativeity). *The KL-divergence is always non-negative,*

$$D_{KL}(P \parallel Q) \geq 0$$

Lemma 3.5.3 (KL). *Suppose $P \sim \mathcal{N}(\boldsymbol{\mu_0}, \boldsymbol{\Sigma_0})$ and $Q \sim \mathcal{N}(\boldsymbol{\mu_1}, \boldsymbol{\Sigma_1})$, then,*

$$D_{KL}(\boldsymbol{P} \parallel \boldsymbol{Q}) = \frac{1}{2} \left\{ \operatorname{tr}\left(\boldsymbol{\Sigma}_1^{-1} \boldsymbol{\Sigma}_0 \right) + (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^{\mathsf{T}} \boldsymbol{\Sigma}_1^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) - k + \ln \left(\frac{|\boldsymbol{\Sigma}_1|}{|\boldsymbol{\Sigma}_0|} \right) \right\}$$

For a proof, see [15].

Suppose, we have the unknown $n+1$ - dimensional random vector $\boldsymbol{Y} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{A}^{-1})$. Then, the KL-divergence between $\boldsymbol{Z_o} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{C_{oo}})$ and, the normal distribution $\boldsymbol{X}$, is given by,

$$D_{\mathrm{KL}}(\boldsymbol{Y} \parallel \boldsymbol{Z_o}) = \frac{1}{2} \left\{ \operatorname{tr}(\boldsymbol{A}\boldsymbol{C_{oo}}) - (n+1) + \ln \left(\frac{|\boldsymbol{A}^{-1}|}{|\boldsymbol{C_{oo}}|} \right) \right\} \geq 0. \tag{3.5.2}$$

Simplifying the above expression, we obtain an upper bound for the log-determinant,

$$\ln |\boldsymbol{C_{oo}}| \leq \operatorname{tr}(\boldsymbol{A}\boldsymbol{C_{oo}}) - \ln |\boldsymbol{A}| - (n+1). \tag{3.5.3}$$

Notice, we obtain equality when $\boldsymbol{A} = \boldsymbol{C_{oo}^{-1}}$, or in other words, when $\boldsymbol{Y} \overset{d}{=} \boldsymbol{Z_o}$. This is in agreement with the notion of the KL-divergence being a distance metric for probability densities. Furthermore, this bound has the nice interpretation of being a distance metric for matrices. Therefore, we can re-frame the problem into estimating the matrix $\boldsymbol{A}$, such that the bound (3.5.3) is minimized.

To obtain an upper bound for the quadratic term, we fallback to the useful block matrix properties that arise from embedding $\boldsymbol{C_{oo}}$ in the circulant matrix $\boldsymbol{C}$.

Lemma 3.5.4 (Inverse Block Quadratic Product). *Suppose we have the positive definite matrix block matrix $\boldsymbol{C}$ in the form of (2.3.1), and the vector $\boldsymbol{w} \in \mathbb{R}^{n+1}$. If we define the function g as,*

$$g(\boldsymbol{w}) := \begin{bmatrix} \boldsymbol{z_o}^{\top} & \boldsymbol{w}^{\top} \end{bmatrix} \boldsymbol{C}^{-1} \begin{bmatrix} \boldsymbol{z_o} \\ \boldsymbol{w} \end{bmatrix},$$

it follows,

1. $g(\boldsymbol{w}) = \boldsymbol{z_o}^\top \boldsymbol{C_{oo}^{-1}} \boldsymbol{z_o} + (\boldsymbol{w} - \boldsymbol{C_{uo}} \boldsymbol{C_{oo}^{-1}} \boldsymbol{z_o})^\top \boldsymbol{C_{u|o}^{-1}} (\boldsymbol{w} - \boldsymbol{C}_{uo} \boldsymbol{C_{oo}^{-1}} \boldsymbol{z_o})$.
2. $\min\limits_{\boldsymbol{w}} g(\boldsymbol{w}) = \boldsymbol{z_o}^\top \boldsymbol{C_{oo}^{-1}} \boldsymbol{z_o}$.
3. $\arg\min\limits_{\boldsymbol{w}} g(\boldsymbol{w}) = \boldsymbol{\mu_{u|o}}$.

Proof. Firstly, we obtain a factorised expression for $\boldsymbol{C^{-1}}$ by inverting the Aitken formula given in Lemma (3.3.3),

$$\boldsymbol{C}^{-1} = \begin{bmatrix} \boldsymbol{I} & 0 \\ -\boldsymbol{C_{uo}} \boldsymbol{C_{oo}^{-1}} & \boldsymbol{I} \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{C_{oo}^{-1}} & 0 \\ 0 & \boldsymbol{C_{u|o}^{-1}} \end{bmatrix} \begin{bmatrix} \boldsymbol{I} & 0 \\ -\boldsymbol{C_{uo}} \boldsymbol{C_{oo}^{-1}} & \boldsymbol{I} \end{bmatrix}.$$

Then considering,

$$\begin{bmatrix} \boldsymbol{I} & 0 \\ -\boldsymbol{C_{uo}} \boldsymbol{C_{oo}^{-1}} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{z_o} \\ \boldsymbol{w} \end{bmatrix} = \begin{bmatrix} \boldsymbol{z_o} \\ \boldsymbol{C_{uo}} \boldsymbol{C_{oo}^{-1}} \boldsymbol{z_o} - \boldsymbol{w} \end{bmatrix},$$

it follows,

$$\begin{bmatrix} \boldsymbol{z_o}^\top & \boldsymbol{w}^\top \end{bmatrix} \boldsymbol{C}^{-1} \begin{bmatrix} \boldsymbol{z_o} \\ \boldsymbol{w} \end{bmatrix} = \begin{bmatrix} \boldsymbol{z_o} \\ \boldsymbol{C_{uo}} \boldsymbol{C_{oo}^{-1}} \boldsymbol{z_o} - \boldsymbol{w} \end{bmatrix}^\top \begin{bmatrix} \boldsymbol{C_{oo}^{-1}} & 0 \\ 0 & \boldsymbol{C_{u|o}^{-1}} \end{bmatrix} \begin{bmatrix} \boldsymbol{z_o} \\ \boldsymbol{C_{uo}} \boldsymbol{C_{oo}^{-1}} \boldsymbol{z_o} - \boldsymbol{w} \end{bmatrix},$$

$$= \boldsymbol{z_o}^\top \boldsymbol{C_{oo}^{-1}} \boldsymbol{z_o} + (\boldsymbol{C_{uo}} \boldsymbol{C_{oo}^{-1}} \boldsymbol{z_o} - \boldsymbol{w})^\top \boldsymbol{C_{u|o}^{-1}} (\boldsymbol{C_{uo}} \boldsymbol{C_{oo}^{-1}} \boldsymbol{z_o} - \boldsymbol{w}).$$

Hence, we have formulated the simplified expression for $g(\boldsymbol{w})$. To verify the global minimum and the respective argument, simply note as $\boldsymbol{C_{u|o}^{-1}}$ is a positive definite matrix,

$$(\boldsymbol{C_{uo}} \boldsymbol{C_{oo}^{-1}} \boldsymbol{z_o} - \boldsymbol{w})^\top \boldsymbol{C_{u|o}^{-1}} (\boldsymbol{C_{uo}} \boldsymbol{C_{oo}^{-1}} \boldsymbol{z_o} - \boldsymbol{w}) \geq 0,$$

with equality achieved when $\boldsymbol{w} = \boldsymbol{C_{uo}} \boldsymbol{C_{oo}^{-1}} \boldsymbol{z_o}$, which by definition, is the conditional mean $\boldsymbol{\mu_{u|o}}$. It proceeds to follow,

$$\boldsymbol{z_o}^\top \boldsymbol{C_{oo}^{-1}} \boldsymbol{z_o} \leq \begin{bmatrix} \boldsymbol{z_o}^\top & \boldsymbol{w}^\top \end{bmatrix} \boldsymbol{C}^{-1} \begin{bmatrix} \boldsymbol{z_o} \\ \boldsymbol{w} \end{bmatrix} \tag{3.5.4}$$

□

Conjoining the two bounds (3.5.3) and (3.5.4), we obtain, $\ell(\boldsymbol{\theta}; \boldsymbol{z_o}) \leq \ell_1(\boldsymbol{\theta}, \boldsymbol{A}, \boldsymbol{w}; \boldsymbol{z_o})$, where

$$\ell_1(\boldsymbol{\theta}, \boldsymbol{A}, \boldsymbol{w}; \boldsymbol{z_o}) := g(\boldsymbol{w}) + \operatorname{tr}(\boldsymbol{A}\boldsymbol{C_{oo}}) - \ln|\boldsymbol{A}| - (n+1). \tag{3.5.5}$$

Additionally, as this global upper bound can achieve equality by minimizing over the parameters $\boldsymbol{A}$ and $\boldsymbol{w}$, the MLE can be determined by choosing $\boldsymbol{\theta}$, from the global minimization,

$$\underset{\boldsymbol{\theta},\boldsymbol{A},\boldsymbol{w}}{\operatorname{argmin}}\, \ell_1(\boldsymbol{\theta}, \boldsymbol{A}, \boldsymbol{w}; \boldsymbol{z_o}). \tag{3.5.6}$$

This expression requires the evaluation of the unspecified positive definite matrix $\boldsymbol{A}$, and the observed covariance matrix $\boldsymbol{C_{oo}}$. A second expression, which instead involves the matrix $\boldsymbol{C^{-1}_{u|o}}$, can be obtained by the implementing the following lemma.

Lemma 3.5.5 (Block Determinant Formula)**.** *Suppose the* $2n \times 2n$ *matrix* $\boldsymbol{C}$ *has the block matrix form stated in Lemma 2.3.3. Then,*

$$ln|\boldsymbol{C}| = ln|\boldsymbol{C_{oo}}| + ln|\boldsymbol{C_{u|o}}|.$$

Notice, this result can be yielded by taking the determinant of both sides in the Aitken block-diagonalization formula (3.3.7).

Substituting the above result and the respective KL-divergence bound for $|\boldsymbol{C_{u|o}}|$ into 3.5.1, we obtain,

$$\ell_2(\boldsymbol{\theta}, \boldsymbol{A}, \boldsymbol{w}; \boldsymbol{z_o}) := g(\boldsymbol{w}) + \ln|\boldsymbol{C}| + \operatorname{tr}\left(\boldsymbol{A}\boldsymbol{C^{-1}_{u|o}}\right) - \ln|\boldsymbol{A}| - (n+1).$$

This new expression may provide better conditioned matrix operations, however, it is recommended to verify the two equivalent bounds numerically.

Optimizing over all positive matrices $\boldsymbol{A}$ is not realistic, therefore, in a crude manner we restrict $\boldsymbol{A}$ to the parametric form of a rank-one matrix plus a constant diagonal matrix,

$$\alpha\boldsymbol{I} + \boldsymbol{v}\boldsymbol{v}^\top,$$

where $\boldsymbol{v}$ is an (n-1)-dimensional vector and α is a scalar. To simplify ℓ_2, after making the aforementioned substitution for $\boldsymbol{A}$, we utilise the following linear algebra result.

Lemma 3.5.6 (Matrix Determinant Lemma)**.** *Suppose* $\boldsymbol{A}$ *is an invertible square matrix and* $\boldsymbol{u}$*,* $\boldsymbol{v}$ *are column vectors. Then,*

$$\left|\mathbf{A} + \mathbf{u}\mathbf{v}^T\right| = \left(1 + \mathbf{v}^T\mathbf{A}^{-1}\mathbf{u}\right)|\mathbf{A}|\ .$$

It follows, if $\boldsymbol{A} = \alpha \boldsymbol{I} + \boldsymbol{v}\boldsymbol{v}^\top$,

$$\ln |\boldsymbol{A}| = \ln \left(1 + \frac{\|v\|^2}{\alpha}\right) + (n+1)\ln \alpha. \tag{3.5.7}$$

Moreover, the trace term can be simplified to,

$$\operatorname{tr}\left(\boldsymbol{C}^{-1}_{\boldsymbol{u}|\boldsymbol{o}}\left(\alpha \boldsymbol{I} + \boldsymbol{v}\boldsymbol{v}^\top\right)\right) = \alpha \operatorname{tr}\left(\boldsymbol{C}^{-1}_{\boldsymbol{u}|\boldsymbol{o}}\right) + \boldsymbol{v}^\top \boldsymbol{C}^{-1}_{\boldsymbol{u}|\boldsymbol{o}} \boldsymbol{v}, \tag{3.5.8}$$

Substituting (3.5.8) + (3.5.7) into ℓ_2, and suppressing constants, we obtain the new computationally feasible expression,

$$\ell_2(\boldsymbol{\theta}, \boldsymbol{v}, \boldsymbol{w}; \boldsymbol{z_o}) := g(\boldsymbol{w}) + \ln|\boldsymbol{C}| + \alpha \operatorname{tr}\left(\boldsymbol{C}^{-1}_{\boldsymbol{u}|\boldsymbol{o}}\right) + \boldsymbol{v}^\top \boldsymbol{C}^{-1}_{\boldsymbol{u}|\boldsymbol{o}} \boldsymbol{v} - \ln\left(1 + \frac{\|v\|^2}{\alpha}\right) - (n+1)\ln\alpha.$$

Given that the lower block of the circulant matrix $\boldsymbol{C^{-1}}$ is equal to $\boldsymbol{C}^{-1}_{\boldsymbol{u}|\boldsymbol{o}}$, operations involving this term can be evaluated in $O(N\log N)$ time. Moreover, as circulant matrices are also are toeplitz we can obtain the trace by evaluating,

$$\operatorname{tr}\left(\boldsymbol{C}^{-1}_{\boldsymbol{u}|\boldsymbol{o}}\right) = (n+1)\, \boldsymbol{v*}^\top\, \boldsymbol{C}^{-1}_{\boldsymbol{u}|\boldsymbol{o}} \boldsymbol{v*},$$

where $\boldsymbol{v*} = [\boldsymbol{0} \quad 1]^\top$. Obtaining the quadratic product with this vector will allow us to evaluate to constant diagonal term which is then scaled by the dimension of the unobserved data.

Hence, we can evaluate $\ell_2(\boldsymbol{\theta}, \boldsymbol{v}, \boldsymbol{w}; \boldsymbol{z_o})$ efficiently in $O(N\log N)$ time. Minimizing this simplified expression respect to α, $\boldsymbol{v}$, $\boldsymbol{w}$ and $\boldsymbol{\theta}$, will obtain the maximum likelihood without calling upon secondary PCG and simulation algorithms. Furthermore, this approach does not require a Monte Carlo estimate, which implies, if an error exists, it is solely dependent on the parametric assumption of $\boldsymbol{A}$. Therefore the potential scope for this technique is large, since, in this simple case, we've restricted our self with the crude assumption, $\boldsymbol{A} = \alpha\boldsymbol{I} + \boldsymbol{v}\boldsymbol{v}^\top$.

From a computational perspective, implementing Quasi-Newton methods would require the evaluation of numerical derivatives for all the elements of our parameters α,$\boldsymbol{v}$, $\boldsymbol{w}$ and $\boldsymbol{\theta}$. As n grows, this task becomes rather burdensome. To this end, we supply our optimization procedure with the following exact derivatives for α,$\boldsymbol{v}$ and $\boldsymbol{w}$, all of which can be computed in $O(N\log N)$ time,

- $\dfrac{\partial}{\partial \alpha}\ell(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{w}, \boldsymbol{v}; \boldsymbol{z_o}) = \operatorname{tr}\left(\boldsymbol{C^{-1}_{u|o}}\right) + \dfrac{\|\boldsymbol{v}^2\|}{\alpha^2 + \alpha\,\|\boldsymbol{v}\|^2} - \dfrac{n-1}{\alpha}.$

- $\dfrac{\partial}{\partial \boldsymbol{w}}\ell(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{w}, \boldsymbol{v}; \boldsymbol{z_o}) = \text{The last } n-1 \text{ entries of the vector } 2\boldsymbol{C^{-1}}\begin{bmatrix}\boldsymbol{z_o}\\ \boldsymbol{w}\end{bmatrix}.$

- $\dfrac{\partial}{\partial \boldsymbol{v}}\ell(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{w}, \boldsymbol{v}; \boldsymbol{z_o}) = 2\boldsymbol{C^{-1}}\boldsymbol{v} - \dfrac{2}{a}\left(1 + \dfrac{\|\boldsymbol{v}\|^2}{a}\right)^{-1}\boldsymbol{v}.$

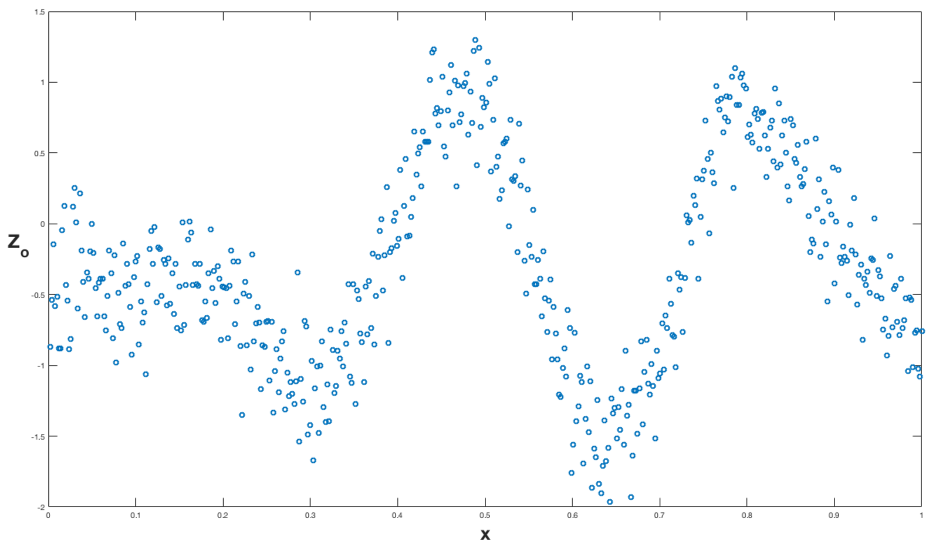
In summary, we have provided the derivation for a stochastic free likelihood approach that does not require the solution to potential ill-conditioned linear systems. Intuitively, we have reformulated the likelihood optimization into a linear algebra problem; where the aim is to find the best representation of the matrix $\boldsymbol{A}$, that minizises the KL-divergence, while, maintaining an expression that is computationally feasible.

Chapter 4

Numerical Simulations

To study the performance of the various algorithms, we carry out a simulation study using realisations of various GPs that have been synthetically simulated using Algorithm 3.2.

Figure 4.1: Simulated data drawn from the altered squared exponential GP.



We now compare the newly proposed estimator for the Monte Carlo-EM against the crude estimate proposed in [35]. Figures 4.3 and 4.3 display MLE estimate updates from the Monte-Carlo EM for identical Monte Carlo samples.

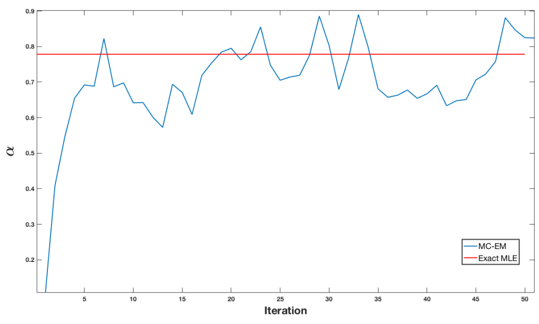


Figure 4.2: $\hat{Q}_2(\mathbf{\Theta}|\mathbf{\Theta}^t)$

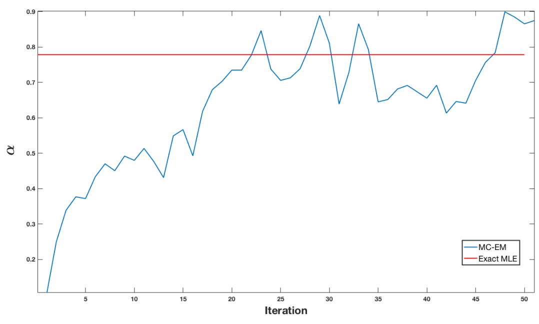


Figure 4.3: $\hat{Q}(\mathbf{\Theta}|\mathbf{\Theta}^t)$

It is evident when comparing the above estimators, that $\hat{Q}_2(\mathbf{\Theta}|\mathbf{\Theta}^t)$ outperforms the crude estimate when the estimate is far away from the true value i.e., when large updates are being made. It is observed, as the estimators approach the true value their variability more or less coincide. Theoretically, this is explained by the formula 3.3.6.

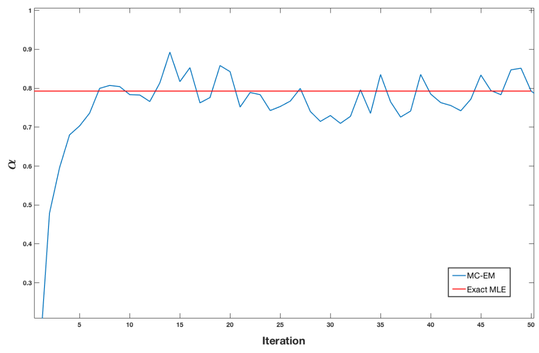


Figure 4.4: $M = 10$

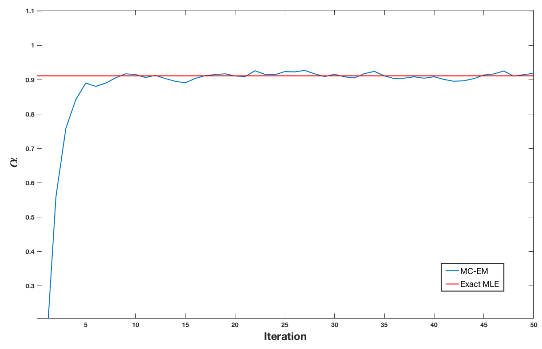


Figure 4.5: $M = 100$

Figures 4.4 and Figures 4.4 illustrate the effect of increasing the Monte Carlo size M. Increasing the Monte Carlo effectively reduces the variance at each iteration, hence, we obtain a quicker converge and lower variability in estimator updates. However, as M is equal to the number of calls to the PCG algorithm at every iteration, increasing M comes at a computational costs, therefore, there is a trade-off between speed and accuracy.

We refer to [35], for metrics to test our simulations. Specifically we use the root mean squared difference (RMSD) to measure the error in each of our MLE approximation methods. The RSMD is given by,

$$\text{RMSD} = \sqrt{\frac{1}{S}\sum_{i=1}^{S}(\hat{\theta}_{\text{MLE},\,i} - \hat{\theta}_i)^2},$$

where S is number of simulations, $\hat{\theta}_i$ denotes the approximate MLE estimate on the ith simulation and $\hat{\theta}_{\text{MLE},\,i}$ is the MLE estimate on ith simulation.

The exact MLE is computed using Levinson–Durbin recursion [16], which fastens the standard Cholesky procedure to $O(n^2)$ time by exploiting Toeplitz linear systems.

For the Monte Carlo methods we fix the size to $M = 10$ for the MC-EM, and $M = 5$ for the stochastic score. The simulation number was set to $S = 20$, for all simulations.

Recall, the Ornstein-Uhlenbeck Process is characeterised by the following covariance function,

$$k(h) = a \exp\left(\frac{-h}{2l^2}\right).$$

Table 4.1:4.3 refer to MLE approximations for simulated data from the Ornstein-Uhlenbeck Process, with known paramaters $\alpha = 1$ and $l = 0.1$. Note, run-time is given in seconds.

MC-EM			
$\boldsymbol{n}$	$\boldsymbol{\alpha -}$ **RMSD**	$\boldsymbol{l-}$**RMSD**	**Run-Time**
100	1.12e-1	1.12e-2	4.6
200	1.04e-1	9.83e-3	4.9
500	1.01e-1	9.21e-3	5.6
1000	9.42e-1	6.73e-3	7.5
5000	8.29e-1	6.23e-3	16.3
10000	8.23e-1	5.29e-3	31.2
20000	5.21e-2	2.52e-3	53.3

Table 4.1: True $\alpha = 1$, $l = 0.3$

Stochastic Score			
$\boldsymbol{n}$	$\boldsymbol{\alpha -}$ **RMSD**	$\boldsymbol{l-}$**RMSD**	**Run-Time**
100	5.84e-2	1.23e-2	0.5
200	1.21e-2	2.12e-3	0.7
500	6.23e-3	3.24e-4	1.0
1000	2.21e-3	5.63e-5	1.2
5000	8.42e-4	2.34e-5	2.0
10000	4.23e-4	1.42e-5	3.8
20000	9.23e-5	6.75e-6	7.0

Table 4.2: True $\alpha = 1$, $l = 0.3$

We see a significant performance gain between our improved stochastic score approximation technique and the Monte Carlo EM in terms of speed and accuracy. In fact, as n grows the relative error of our estimate dramatically decreases. This hints the to the fact, that the stochastic contribution of the trace diminishes as $n \to \infty$. We thus conclude that the stochastic approach to solving the score equations is the superior method to approximating the MLE.

MM Algorithm		
$\boldsymbol{n}$	$\boldsymbol{l}$**−RMSD**	**Run-Time**
100	7.95e-3	4.24
200	4.23e-3	8.5
500	2.12e-3	41
1000	1.83e-3	147
2000	1.81e-3	385

Table 4.3: True $l = 0.3$

The MM Algorithm approach developed in Section 3.5 while exhibiting impressive estimating accuracy lacks in speed, potentially due to the necessity to optimize over numerous n-dimensional vectors. Hence, it is promising that the new method is a valid means of approximating the MLE, however, more effective representations of the matrix $\boldsymbol{A}$ are required to improve the run-time performance.

Chapter 5

Conclusions

In this paper, we have provided an overview and analysis of likelihood optimization methods for stationary Gaussian processes. The analysis focused on techniques that employ circulant covariance embedding, allowing one to use FFT operations in $O(n\log n)$ time. Furthermore, we have proposed numerous Monte Carlo estimates that improve upon pre-exisitng stochastic methods that approximate the MLE. These Monte Carlo improvements were derived from variance reduction techniques and block matrix properties. These new estimators are applicable to the Monte Carlo EM algorithm proposed in [35] and the stochastic approximation of the score function. The improvements were then numerically tested, and, it was concluded that the approximation the score was the most efficient. For further testing and verification of superiority, one can implement the stochastic score method to different covariance functions that are described by higher-dimensional hyperparameters. These MLE procedures rely on sub-iterative PCG, and simulation algorithms. To avoid these procedures we proposed a deterministic expression, namely an example of the MM algorithm, that could be evaluated in $O(n\log n)$ time, and, could be minimised to obtain the MLE. This newly formulated approach is left as an open-ended research question, and can be tackled by formulating effective linear algebra techniques that attempt to minimize the aforementioned KL-divergence bound.

Please note that all code used in this thesis is available at,

$$https://github.com/anantmathur44/GP-thesis$$

Bibliography

[1] Milton Abramowitz, Irene A Stegun, and Robert H Romer. *Handbook of mathematical functions with formulas, graphs, and mathematical tables*. 1988.

[2] Alexander Craig Aitken and Tomás Rodrıguez Bachiller. *Determinants and matrices*. Oliver and Boyd Edinburgh, 1946.

[3] Mihai Anitescu, Jie Chen, and Lei Wang. “A matrix-free approach for solving the parametric Gaussian process maximum likelihood problem”. In: *SIAM Journal on Scientific Computing* 34.1 (2012), A240–A262.

[4] Jonathan Barzilai and Jonathan M Borwein. “Two-point step size gradient methods”. In: *IMA journal of numerical analysis* 8.1 (1988), pp. 141–148.

[5] Dimitri P Bertsekas. “Nonlinear programming”. In: *Journal of the Operational Research Society* 48.3 (1997), pp. 334–334.

[6] Zdravko Botev and Ad Ridder. “Variance reduction”. In: *Wiley StatsRef: Statistics Reference Online* (2014), pp. 1–6.

[7] Raymond H Chan and Michael K Ng. “Conjugate gradient methods for Toeplitz systems”. In: *SIAM review* 38.3 (1996), pp. 427–482.

[8] Raymond H Chan and Man-Chung Yeung. “Circulant preconditioners for Toeplitz matrices with positive continuous generating functions”. In: *Mathematics of computation* 58.197 (1992), pp. 233–240.

[9] James W Cooley and John W Tukey. “An algorithm for the machine calculation of complex Fourier series”. In: *Mathematics of computation* 19.90 (1965), pp. 297–301.

[10] John P Cunningham, Krishna V Shenoy, and Maneesh Sahani. “Fast Gaussian process methods for point process intensity estimation”. In: *Proceedings of the 25th international conference on Machine learning*. ACM. 2008, pp. 192–199.

[11] Kurt Cutajar et al. “Preconditioning kernel matrices”. In: *International Conference on Machine Learning*. 2016, pp. 2529–2538.

[12] Amir Dembo, Colin L Mallows, and Lawrence A Shepp. “Embedding nonnegative definite Toeplitz matrices in nonnegative definite circulant matrices, with application to covariance estimation”. In: *IEEE Transactions on Information Theory* 35.6 (1989), pp. 1206–1212.

[13] Arthur P Dempster, Nan M Laird, and Donald B Rubin. “Maximum likelihood from incomplete data via the EM algorithm”. In: *Journal of the Royal Statistical Society: Series B (Methodological)* 39.1 (1977), pp. 1–22.

[14] CR Dietrich and GN Newsam. “A fast and exact method for multidimensional Gaussian stochastic simulations”. In: *Water Resources Research* 29.8 (1993), pp. 2861–2869.

[15] John Duchi. "Derivations for linear algebra and optimization". In: *Berkeley, California* 3 (2007).

[16] James Durbin. "The fitting of time-series models". In: *Revue de l'Institut International de Statistique* (1960), pp. 233–244.

[17] Maurizio Filippone and Raphael Engler. "Enabling scalable stochastic gradient-based inference for Gaussian processes by employing the Unbiased LInear System SolvEr (ULISSE)". In: *arXiv preprint arXiv:1501.05427* (2015).

[18] Jack K Fitzsimons et al. "Improved stochastic trace estimation using mutually unbiased bases". In: *arXiv preprint arXiv:1608.00117* (2016).

[19] Roger Fletcher. *Practical methods of optimization*. John Wiley & Sons, 2013.

[20] Robert Alexander Frazer, William Jolly Duncan, Arthur Roderich Collar, et al. *Elementary matrices and some applications to dynamics and differential equations*. Vol. 1963. Cambridge University Press Cambridge, 1938.

[21] Tilmann Gneiting et al. "Fast and exact simulation of large Gaussian lattice systems in 2: exploring the limits". In: *Journal of Computational and Graphical Statistics* 15.3 (2006), pp. 483–501.

[22] GH Golub and CF Van Loan. *Matrix Computations. 3rd. edn. ed.* John Hopkins University Press, Baltimore, US, 1996.

[23] Ivan G Graham et al. "Analysis of circulant embedding methods for sampling stationary random fields". In: *SIAM Journal on Numerical Analysis* 56.3 (2018), pp. 1871–1895.

[24] Robert M Gray et al. "Toeplitz and circulant matrices: A review". In: *Foundations and Trends in Communications and Information Theory* 2.3 (2006), pp. 155–239.

[25] Magnus Rudolph Hestenes and Eduard Stiefel. *Methods of conjugate gradients for solving linear systems*. Vol. 49. 1. NBS Washington, DC, 1952.

[26] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.

[27] Michael F Hutchinson. "A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines". In: *Communications in Statistics-Simulation and Computation* 19.2 (1990), pp. 433–450.

[28] Solomon Kullback and Richard A Leibler. "On information and sufficiency". In: *The annals of mathematical statistics* 22.1 (1951), pp. 79–86.

[29] Kenneth Lange. *MM optimization algorithms*. Vol. 147. SIAM, 2016.

[30] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[31] Alvin C Rencher and G Bruce Schaalje. *Linear models in statistics*. John Wiley & Sons, 2008.

[32] RN Silver and H Röder. "Calculation of densities of states and spectral functions by Chebyshev recursion and maximum entropy". In: *Physical Review E* 56.4 (1997), p. 4822.

[33] Michael L Stein. "Fast and exact simulation of fractional Brownian surfaces". In: *Journal of Computational and Graphical Statistics* 11.3 (2002), pp. 587–599.

[34] Michael L Stein, Jie Chen, Mihai Anitescu, et al. “Stochastic approximation of score functions for Gaussian processes”. In: *The Annals of Applied Statistics* 7.2 (2013), pp. 1162–1191.

[35] Jonathan R Stroud, Michael L Stein, and Shaun Lysen. “Bayesian and maximum likelihood estimation for Gaussian processes on an incomplete lattice”. In: *Journal of computational and Graphical Statistics* 26.1 (2017), pp. 108–120.

[36] George E Uhlenbeck and Leonard S Ornstein. “On the theory of the Brownian motion”. In: *Physical review* 36.5 (1930), p. 823.

[37] Andrew TA Wood and Grace Chan. “Simulation of stationary Gaussian processes in [0, 1] d”. In: *Journal of computational and graphical statistics* 3.4 (1994), pp. 409–432.